J Computer-Aided Mater Des (2007) 14:253–308
DOI 10.1007/s10820-006-9042-9

An overview of spatial microscopic and accelerated kinetic Monte Carlo methods

Abhijit Chatterjee · Dionisios G. Vlachos

Received: 6 August 2006 / Accepted: 17 October 2006 / Published online: 28 February 2007

Abstract The microscopic spatial kinetic Monte Carlo (KMC) method has been employed extensively in materials modeling. In this review paper, we focus on different traditional and multiscale KMC algorithms, challenges associated with their implementation, and methods developed to overcome these challenges. In the first part of the paper, we compare the implementation and computational cost of the null-event and rejection-free microscopic KMC algorithms. A firmer and more general foundation of the null-event KMC algorithm is presented. Statistical equivalence between the null-event and rejection-free KMC algorithms is also demonstrated. Implementation and efficiency of various search and update algorithms, which are at the heart of all spatial KMC simulations, are outlined and compared via numerical examples. In the second half of the paper, we review various spatial and temporal multiscale KMC methods, namely, the coarse-grained Monte Carlo (CGMC), the stochastic singular perturbation approximation, and the τ-leap methods, introduced recently to overcome the disparity of length and time scales and the one-at-a time execution of events. The concepts of the CGMC and the τ-leap methods, stochastic closures, multigrid methods, error associated with coarse-graining, a posteriori error estimates for generating spatially adaptive coarse-grained lattices, and computational speed-up upon coarse-graining are illustrated through simple examples from crystal growth, defect dynamics, adsorption–desorption, surface diffusion, and phase transitions.

Keywords Review · Multiscale simulation · Coarse-graining · Mesoscopic modeling · Monte Carlo · Materials · Defects · Diffusion · Crystal growth · Phase transitions · Accelerated algorithms · Binary tree · Efficient update · Efficient search · Tau-leap · Stiff · Stochastic · Computational singular perturbation · Low-dimensional manifold

A. Chatterjee · D. G. Vlachos (✉)
Department of Chemical Engineering and Center for Catalytic Science and Technology (CCST),
University of Delaware, Newark, DE 19716, USA
e-mail: vlachos@udel.edu

1 Introduction

Spatial Monte Carlo (MC) methods have widely been employed in science and engineering since, at least, 1953 [1]. The use of MC for structure-property determination in condensed phases, such as liquids and solids, at interfaces (e.g., surfaces, defects, and nanoparticles), and in gas phases to study transport, thermophysical, magnetic, reactivity, and mechanical properties [2–11] is common in physics, chemistry, materials science, and engineering disciplines.

While over these years MC simulation has successfully been applied to predict equilibrium properties [2,12], its use in predicting transport (e.g., diffusion) and/or chemistry, in both equilibrium and non-equilibrium systems, is quite recent (a suite of modern application examples in chemical engineering appears in [13]). The MC method became popular in 1970s and made major inroads in 1980s for studying mainly transport (diffusion on surface and in materials), reaction kinetics, and crystal growth problems (for illustrative examples see [11,14–28]). Computational resources were sufficiently fast by 1980s to enable simulations, in most cases, of small to moderate size model systems (current computational resources enable simulations of upto 10^6 atoms). Even though most of the early simulations provided insights into the physics and chemistry of materials, by-and-large, they lacked predictive capabilities because of the absence of accurate molecular information, such as interaction potentials and dynamics. One of the first applications of the MC method to growth problems was the study of crystal growth modes in homoepitaxial thin films, and the transition to rough crystals (for an overview of earlier work see [29,30]). This research area continues to be [31–33] of particular interest because of its diverse applications in pharmaceuticals, separation devices, microelectronics, magnetics, catalysts, etc. The first MC simulation of surface kinetics appeared in Wicke et al. [34]. However, it was the work of Ziff and co-workers [35] in 1986 that ‘ignited’ mainly the physics community in investigating non-equilibrium phase transitions using the MC method. This work established MC as a powerful numerical molecular simulation tool for studying thermal fluctuations and spatial correlations driven phenomena in non-equilibrium systems.

Since the MC method is not viewed by many as a truly first-principles tool (see Sect. 4 on challenges), the computational power of 1990s propelled molecular dynamics (MD) as the tool of choice for molecular or atomic scale dynamic simulations at the expense of MC simulation. Further increase in computational power, in conjunction with the need to predict transport, materials properties and structure under realistic conditions, led to the realization that most systems contain multiple length and time scales [36–42]. As a result, the MC method has recently regained considerable attention in the multiscale modeling community. As an example, the importance of stochasticity in biological networks [43–47], arising in part from the small species populations, triggered intense research on accelerated stochastic (mainly MC) methods. It is not then surprising that recent research roadmaps underscore the key role of the MC method in multiscale modeling [48,49].

The MC method has been discussed in detail numerous times in articles and books. A review of the vast literature on MC is well beyond the scope of this article. Our focus herein is on the kinetic MC (KMC) method (see Sect. 3 for a classification). Even though some introductory chapters on this method have recently appeared [50,51], the lack of literature on algorithmic aspects of the KMC method have not only led to confusion regarding its implementation, but also to reinvention of the same algorithm several times. The emphasis of this article is on providing an overview of recent mathematical and algorithmic developments that can enable KMC simulations of large length and time scales.

The organization of this paper is as follows: in Sect. 2, a non-equilibrium statistical mechanics representation for KMC simulations is described followed by physical examples on crystal growth, surface defect dynamics, and equilibrium fluid-surface interactions. In Sects. 3 and 4, the KMC method and its major challenges are summarized. In Sect. 5, the null-event and rejection-free algorithms are described, with the objective of providing a firm basis for the former and showing the statistical equivalence between the two algorithms. This, to the best of our knowledge, has not been done before. Two different approaches for addressing the challenges in KMC method are addressed in rest of the paper. In Sect. 6, efficient search and update algorithms are reviewed. Numerical examples are presented in Sect. 7 to illustrate the equivalence of null-event and rejection-free algorithms and provide a comparison of CPU times of various steps of a KMC algorithm. In Sects. 8 and 9, modern non-equilibrium, statistical mechanics-based spatial and temporal acceleration methods are presented. Various examples (some from our published work and some new) are presented to highlight main algorithmic and mathematical developments. Finally, we close with some concluding remarks and possible future directions.

2 Non-equilibrium statistical mechanics description of physical systems

In this section, we define kinetic and thermodynamics variables in the context of non-equilibrium statistical mechanics. In particular, the concept of a microscopic state in a KMC simulation, and non-equilibrium theories for describing state transformation are discussed. Three physical systems are described at the end of this section to illustrate these concepts.

2.1 Microscopic state, microscopic processes, and the master equation

In most materials problems, atoms or molecules vibrate around specific locations (minima in free energy) separated by large free energy barriers. Occasionally atoms ‘jump’ from one location to a nearby one (e.g., to a vacancy). In most cases, the time scale of the jump is significantly larger than the time scales associated with thermal vibrations. As a consequence, it is assumed that after a large number of thermal vibrations, the system has attained quasi-equilibrium.

The momentum degrees of freedom of all atoms/molecules and thermal vibrations are integrated out to compute the microscopic rates used in a stochastic description of a system. The resulting (microscopic) state variable $\underline{\sigma}$ (underbars denote a vector) is a function of only spatial and time coordinates. The aforementioned free energy minima map to spatial locations that are referred to as lattice sites. Note though that a regular lattice is not necessary (see Sect. 3.2). In this paper, the total number of sites is denoted as N_{L}. Note that the dimensionality of the lattice is arbitrary. The atomic ‘jump,’ referred to as a (microscopic) process in this paper, results in a change in $\underline{\sigma}$ once the jump has occurred. The total number of processes that can occur at site i is denoted as N_{p}.

The starting point for a stochastic description of a system is the underlying master equation [52], given by

$$\frac{\mathrm{d}P(\underline{\sigma})}{\mathrm{dt}} = \sum_{\substack{\underline{\sigma}' \\ \underline{\sigma}' \neq \underline{\sigma}}} G(\underline{\sigma}' \to \underline{\sigma})P(\underline{\sigma}') - \sum_{\substack{\underline{\sigma}' \\ \underline{\sigma}' \neq \underline{\sigma}}} G(\underline{\sigma} \to \underline{\sigma}')P(\underline{\sigma}), \tag{1}$$

which gives the evolution of the probability density function $P(\underline{\sigma})$ of observing a particular state $\underline{\sigma}$. Here $G(\underline{\sigma} \to \underline{\sigma}')$ is viewed as an element of the transition matrix for the transition from state $\underline{\sigma}$ to state $\underline{\sigma}'$ [2]. Alternatively, one can write an equivalent difference-differential equation [53]

$$\mathrm{d}\sigma_i = \sum_j \Gamma_{ij}^{+}(\underline{\sigma})\mathrm{dt} - \sum_j \Gamma_{ij}^{-}(\underline{\sigma})\mathrm{dt}, \tag{2}$$

which gives the temporal evolution of σ_i in terms of the transition probability $\Gamma_{ij}^{+}(\underline{\sigma})$ ($\Gamma_{ij}^{-}(\underline{\sigma})$) of all processes j that lead to a particle addition (deletion) at site i. In this paper, the term particle implies an atom, a molecule or a vacancy. In a KMC simulation, it is assumed that j is an independent, Markovian process. In the KMC literature the terms microscopic rate, transition probability per unit time, and transition probability are used interchangeably.

The master equation is deterministic. Due to the large number of dimensions present in most physical systems, Eq. 2 cannot be solved analytically. Furthermore, the huge number of states renders a deterministic solution of Eq. 2 formidable for most practical problems of interest. MC methods become naturally the most efficient solution techniques.

While the matrix of transition probabilities $[\Gamma_{ij}(\underline{\sigma})]$ can be large, only a few elements are non-zero (a very sparse matrix). Let $\{\varepsilon_{ij}\}_{j=1}^{N_\mathrm{P}}$ be the participation indices of site i, i.e., whether the ith site can give rise to process j ($\varepsilon_{ij} = 1$) or not ($\varepsilon_{ij} = 0$)

$$\varepsilon_{ij} = \begin{cases} 1, & \text{if site } i \text{ participates in process } j, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

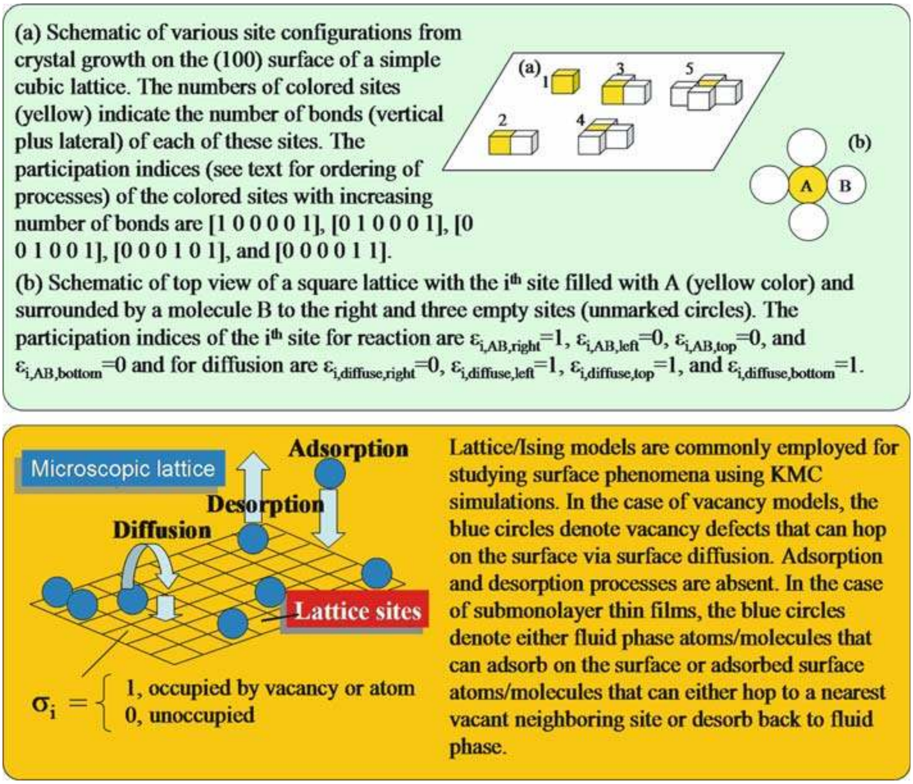
The participation matrix $[\varepsilon_{ij}]$ consists then of ones and zeros (the many zeros highlight the matrix sparsity). The participation indices of a site may depend on neighboring interacting sites, and vary with time. Examples are given in the top panel of Fig. 1.

The transition probabilities $\{\Gamma_{ij}\}_{j=1}^{N_\mathrm{P}}$ depend on the participation index of a site in a process and are given by

$$\Gamma_{ij} = \Gamma_j \varepsilon_{ij} = \begin{cases} \Gamma_j, & \text{if site } i \text{ participates in process } j, \\ 0, & \text{otherwise,} \end{cases} \quad i = 1, \ldots, N_\mathrm{L}, \quad j = 1, \ldots, N_\mathrm{P}. \tag{4}$$

The definition of a process makes it appropriate to introduce the concept of a class. Each process, j, introduces a class, and all sites participating in this process belong to this class (the terms grouping or lumping are also appropriate). The size of the jth class is given by

$$n_j = \sum_{i=1}^{N_\mathrm{L}} \varepsilon_{ij}. \tag{5}$$



(a) Schematic of various site configurations from crystal growth on the (100) surface of a simple cubic lattice. The numbers of colored sites (yellow) indicate the number of bonds (vertical plus lateral) of each of these sites. The participation indices (see text for ordering of processes) of the colored sites with increasing number of bonds are [1 0 0 0 0 1], [0 1 0 0 0 1], [0 0 1 0 0 1], [0 0 0 1 0 1], and [0 0 0 0 1 1].

(b) Schematic of top view of a square lattice with the ith site filled with A (yellow color) and surrounded by a molecule B to the right and three empty sites (unmarked circles). The participation indices of the ith site for reaction are $\varepsilon_{i,AB,right}=1$, $\varepsilon_{i,AB,left}=0$, $\varepsilon_{i,AB,top}=0$, and $\varepsilon_{i,AB,bottom}=0$ and for diffusion are $\varepsilon_{i,diffuse,right}=0$, $\varepsilon_{i,diffuse,left}=1$, $\varepsilon_{i,diffuse,top}=1$, and $\varepsilon_{i,diffuse,bottom}=1$.

$$\sigma_i = \begin{cases} 1, \text{occupied by vacancy or atom} \\ 0, \text{unoccupied} \end{cases}$$

Lattice/Ising models are commonly employed for studying surface phenomena using KMC simulations. In the case of vacancy models, the blue circles denote vacancy defects that can hop on the surface via surface diffusion. Adsorption and desorption processes are absent. In the case of submonolayer thin films, the blue circles denote either fluid phase atoms/molecules that can adsorb on the surface or adsorbed surface atoms/molecules that can either hop to a nearest vacant neighboring site or desorb back to fluid phase.

Fig. 1 The solid-on-solid model (*SOS*) (*top panel*) and lattice model (*bottom panel*) for a (100) surface are examples of KMC models used extensively in the areas of crystal growth and catalysis/materials, respectively, over the last 30 years

2.2 Illustrative physical problems and notation

In this section, we present simple models for three prototype physical systems to illustrate the notation used in Sect. 2.1. These models are employed in later sections to illustrate equivalence and discuss computational accuracy and efficiency of different KMC algorithms.

2.2.1 A crystal growth model

One of the first applications of spatial KMC has been in the area of crystal growth [29,30]. The solid-on-solid (SOS) approximation has been a prototype model for such simulations. In this model, a crystal film contains particles on top of particles and there are no overhangs or vacancies. One is able to simulate a three-dimensional (3D) system by using essentially two-dimensional (2D) simulations (nowadays these are often called 2+1-dimensional simulations). A 2D array, whose elements $\{\sigma_i\}_{i=1}^{N_{\rm L}}$ represent the number of particles or layers at each site i (local height), is used to represent the surface topology (see top panel of Fig. 1). Upon selection of a site and a process, the array element is updated accordingly.

Lets consider a specific example of crystal growth on the (100) surface of a simple cubic lattice with the SOS approximation. For illustration purposes, only adsorption and desorption processes are considered. Assuming first-nearest neighbor interactions of strength $w\,(w > 0$ implies attractive interactions), then each site can have between 1 (a vertical) and 5 (a vertical plus four lateral) interactions. As a result, there are a total of five desorption processes with distinct transition probabilities plus an adsorption process (taken as site independent), i.e., $N_\mathrm{p} = 6$ processes are present on each site.

The transition probabilities, for a single desorption ($j = 1,\ldots,5$) and adsorption ($j = 6$) processes, are [54]

$$\Gamma_j = \nu_\mathrm{d}\mathrm{e}^{-jw/kT}, \quad j = 1, \ldots 5, \tag{6}$$

$$\Gamma_6 = \nu_\mathrm{d}\mathrm{e}^{-3w/kT}\mathrm{e}^{\Delta\mu/kT}. \tag{7}$$

Here ν_d is the frequency for desorption of an atom, kT the thermal energy, T the temperature, k the Boltzmann factor, and $\Delta\mu$ is the difference in chemical potential between the gas and solid phases.

At each instant, a site can participate in either an adsorption or one of the five desorption steps. In the SOS example, the number of sites with a certain number of nearest neighbors determines the class size, n_j. In this case, there are six classes.

2.2.2 A prototype model for diffusion of surface vacancies

The KMC method has extensively been employed to study defects of varying dimensionality [55–61]. Here we study a prototype model of surface defects diffusing on a lattice. This is a simple Ising type of model (see bottom panel of Fig. 1). The simulation is carried out in a canonical ensemble with $N_\mathrm{population}$ defects on N_L lattice sites of a (100) surface (the 2D defect concentration is given by $c_0 = N_\mathrm{population}/N_\mathrm{L}$). Either an atom or a vacancy is allowed per lattice site. The occupancy at each site i is given by $\sigma_i = 1$ ($\sigma_i = 0$) when the site is occupied by a vacancy (atom). Typically, $N_\mathrm{population}$ is determined via the defect creation and annihilation rates [56,57,62], when the system is far from equilibrium, or by thermodynamics when at equilibrium [63]. For simplicity, we fix $N_\mathrm{population}$ at the beginning of a simulation by prohibiting defect creation and annihilation.

An Ising-like interaction model [55] is employed to describe the interaction potential $J(|i - i'|)$ between two neighboring vacancies at sites i and i' at a separation $r = |i - i'|$ (note $r = |i - i'|$ denotes the distance between the two sites rather than the difference between indices). The binding energy of a vacancy at site i is given by

$$U_\mathrm{binding}(i) = \sum_{i'} J(|i - i'|)\sigma_{i'}. \tag{8}$$

The vacancy–vacancy interaction potential $J(r)$ is obtained with a suitable fit from the interatomic interaction potentials. $J(r) > 0$ ($J(r) < 0$) implies attractive (repulsive) interactions.

A vacancy defect at site i can jump to a nearest neighbor site at site i' occupied by an atom. Such a jump is shown in the bottom panel of Fig. 1. The transition probability for a jump is given by

$$\Gamma(i \to i') = \frac{1}{4}\nu_m \sigma_i (1 - \sigma_{i'}) \mathrm{e}^{-U_{\mathrm{binding}}(i)/kT}. \tag{9}$$

In this paper, a hopping frequency of $\nu_m = 4 \times 10^{13}\mathrm{s}^{-1}$ is assumed. Furthermore, the activation energy is taken to depend on the interactions at the departing site only, termed as Arrhenius dynamics [64,65] (this is obviously a crude approximation of reality; other approximations are feasible and have been employed in the literature). Even though this model barely captures the complexity of realistic defects, it is adequate for demonstrating the application of KMC method to materials' defects, and for comparing the efficiency of different KMC algorithms. In this model, the participation index (in the notation of Sect. 2.1) is $\varepsilon_{ii'} = \sigma_i(1 - \sigma_{i'})$.

2.2.3 A submonolayer film of adsorbates on a substrate

Consider the case of adsorbates (up to a monolayer) on a single crystal in contact with a fluid reservoir, depicted in the bottom panel of Fig. 1. A lattice model, analogous to Sect. 2.2.2, is employed. Typical processes that may occur include adsorption and desorption, surface diffusion, and surface reactions. Such systems have extensively been used to study phase transitions in Ising type of models and non-equilibrium phase transitions in irreversible surface (catalytic) reaction systems.

In this model, at most one atom is allowed per lattice site (exclusion principle). The occupancy at each site i is given by $\sigma_i = 1$ ($\sigma_i = 0$) when the site is occupied (unoccupied) by an atom. The spatially averaged coverage over the entire lattice is denoted as $\theta = \sum_{i=1}^{N_\mathrm{L}} \sigma_i / N_\mathrm{L}$. The microscopic Hamiltonian is given by

$$H = -\sum_{i} \sum_{\substack{i' \\ i' \neq i}} J(|i - i'|)\sigma_i \sigma_{i'} + \sum_i h\sigma_i, \quad i, i' = 1, \ldots, N_\mathrm{L}, \tag{10}$$

where $J(r)$ is the interaction potential between two adsorbed atoms at separation r, and h is the external field, such as the chemical potential.

Atoms/molecules from the fluid phase adsorb on the lattice with a transition probability

$$\Gamma_\mathrm{a}(i) = \nu_\mathrm{a}(1 - \sigma_i), \tag{11}$$

and adsorbed atoms can hop or desorb with a transition probability given by Eq. 9 and

$$\Gamma_\mathrm{d}(i) = \nu_\mathrm{d}\sigma_i \mathrm{e}^{-U_{\mathrm{binding}}(i)/kT}, \tag{12}$$

respectively. Here ν_a and ν_d are the adsorption rate constant and desorption frequency, respectively. It is assumed that the desorption follows Arrhenius dynamics. Note that $(1 - \sigma_i)$ and σ_i in Eqs. 11 and 12 play the same role as ε_{ij} of Sect. 2.2.1 (the participation index notation is more general and will be used later). Examples of participation indices for (a) diffusion of species A on a square lattice with first-nearest neighboring jumps or (b) reaction between A and B are shown in the top panel of Fig. 1.

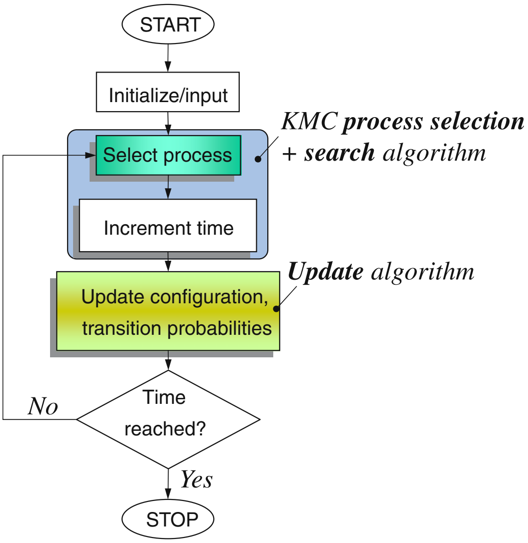

START
Initialize/input
Select process
Increment time
Update configuration, transition probabilities
Time reached?
No
Yes
STOP
KMC **process selection** + **search** algorithm
Update algorithm

Fig. 2 Generic flowchart for any KMC algorithm. Available methods for performing each step in this flowchart are explained in detail in the text. *Shaded boxes* denote computationally most expensive steps

3 The Kinetic Monte Carlo (KMC) method and its variants

In this section, we discuss basic concepts of spatial KMC simulation, as well as similarities and subtle differences between different types of KMC algorithms. Furthermore, KMC and traditional MC methods are also contrasted.

3.1 Generic flowchart of a KMC algorithm

The main steps in a KMC algorithm are outlined in Fig. 2. The underlying principle in all KMC algorithms is the random selection of a process based on the transition probabilities of all processes, execution of the selected process (i.e., appropriately modifying the configuration of the system), and updating the time clock and the transition probabilities. In mathematical terms, the KMC method follows a discrete Markov evolution of the system with continuous time increments given by an exponential distribution [66].

Each of the steps in the entire flowchart presents specific challenges that are enumerated in the sections that follow. Subtle differences between different KMC algorithms originate from (1) the selection of a process and (2) determining the time increment after a process is executed.

3.2 Off-lattice and on-lattice simulation

The probably best-known Metropolis algorithm [1] and its variants, such as the Kawasaki dynamics [67], was introduced and is mainly used (but not exclusively) for *off-lattice* simulations, where atoms or molecules do not have prescribed positions in space. Rather, their average locations are determined from free energy minimization. The transition probabilities for transitioning from one configuration to the next do not have dynamic but purely thermodynamic information. A key element in constructing the transition probabilities for such traditional MC method is that they obey detailed balance; as a result, they provide the correct thermodynamic results.

Spatial movements of atoms or molecules are typically very small, and thus, sampling of configurational space is slow. At each trial, a random number, from a uniform distribution, is compared to the instantaneously computed Metropolis probability, p, and the event is executed only when the random number is lower than p. Otherwise, nothing happens in this specific trial (a *null-event*).

As mentioned earlier in Sect. 2.1, in most physical systems there is a large separation of time scales between the thermal vibrations and microscopic processes (short vibrational time versus long jump time). Such separation of time scales is known in numerical analysis as *stiffness*. MD and off-lattice MC simulations are too slow to deal with such time scale separation, created by large barriers, and they spend most computer time sampling fast vibrations, rather than executing the slower jumps. These slow jumps are termed as *rare events* because they happen infrequently in comparison to the fast processes. The small time increments of fast processes result in minor advancement of the simulation time clock. The term 'rare events' is more general than what was used above and indicates separation of time scales between processes.

On-lattice KMC simulations attempt to overcome the aforementioned type of stiffness via integrating out the effect of thermal vibrations on microscopic rates. In these simulations, particles reside at discrete spatial coordinates on a suitable lattice and slow events, in comparison to the vibrational time, are fired. On-lattice simulation can be parameterized to capture off-lattice effects (for an example see [61,68]). KMC simulations on non-regular lattices are also possible [69].

3.3 KMC classification: null-event and rejection free algorithms

There are two broad algorithms employed for spatial simulation, namely the *null-event* and the *rejection-free* algorithms. In the former, not all MC events (trials) are successful. The Metropolis algorithm is an example of a null-event algorithm, where atomic displacements are actually adjusted to achieve a certain ratio of successful events. In this algorithm, energetically favorable moves, i.e., the ones that reduce the system energy, are selected with probability 1; moves that increase the system energy are accepted with probability < 1. In contrast, in rejection-free algorithms, all events are successful.

It is not surprising that null-event algorithms are assumed, but not necessarily correctly, to be less efficient than rejection-free algorithms. We return to this point in the following sections, where we provide a more detailed description of the two algorithms and numerical examples.

3.4 The time clock issue

Early KMC simulations reported results in MC events (MCE) or MC steps (MCS) 'time units.' One MCS is defined as the number of MC events or trials, that is, equal to the number of sites of the system N_{L}. In the case of crystal growth (and other applications), where steady state is established after some time, a MCS is proportional to real time. However, *a MCS does not generally correspond to a fixed amount of real time*. Furthermore, early KMC simulations of crystal growth and surface kinetics employed a null-event algorithm instead of the rejection-free method. This aspect further complicates mapping of MCS to real time.

It is then not surprising to read in the literature that MC methods lack real time. Aside from the earlier use of null-event algorithms, where association of real time with

MC events was not typically established, this has been in part due to the widespread use of the Metropolis and Kawasaki algorithms, which lack dynamic information in their transition probability. The issue of dynamics is immaterial as far as equilibrium is concerned. However, real dynamics is most often essential for non-equilibrium systems.

The MC simulations follow the stochastic dynamics of a master equation; with appropriate parameterization of the transition probabilities (see Sect. 4.1), they provide continuous time information as well (discrete Markov states, continuum time processes). The MC method, which employs transition probabilities, is what is usually known as *KMC or dynamic MC* (*DMC*) *method*. The former term is probably more often used. It is this method that is the subject of the remaining of the paper.

4 Challenges in KMC simulations

In this section, we outline main challenges in carrying out spatial KMC simulation. Table 1 summarizes these challenges along with methods for coping with them and provides key references. More details are given in the text of this and subsequent sections.

4.1 Input to a KMC simulation

A major challenge of on-lattice KMC simulations (Step 1 of the KMC flowchart in Fig. 2; see also Table 1) is to create a *complete* catalog of all possible processes along with their transition probabilities. This key input to any KMC simulation can be extracted from smaller length and time scale simulation tools, such as density functional theory (DFT) [18,99–102] (these are often termed first principles KMC simulations), transition state identification methods, transition state theory (TST) [103,104], and MD [3,4,61,105,106] typically via a bottom-up approach (information passage from small to large scales) [36,107–110]. Significant work in this area to address the computational requirements of DFT and MD and the approximations involved, e.g., in TST, has led to the development of a whole new research area. Important developments in potential energy surface (PES) prediction include semiempirical/empirical methods, such as the tight-binding method, and interaction potentials, such as the embedded-atom method and the effective medium theory [111–114]. Methods developed for exploring the PES, saddle points and path sampling, include double-ended search methods (initial and final states are known), such as the nudged elastic band, discrete and transition path sampling, and single-ended search methods (only initial state known), such as the Hessian-based and the minimum mode methods [115–117]. In addition, accelerated MD methods, such as the temperature accelerated dynamics, hyperdynamics, and on-the-fly KMC [69,118,119] are also useful for accessing long time scale processes. Ultimately, the basic paths, i.e., the initial and final microscopic states, are identified, and the transition probability for the process is typically postulated in terms of a rate prefactor and a Boltzmann term and fitted using the aforementioned tools. The difficulty involved in identifying all microscopic processes (especially the rare ones) may render the catalog of process incomplete. Recent on-the-fly techniques, such as the self-learning KMC, are under development to address this very issue, e.g., [69,120].

Table 1 Summary of major challenges in spatial KMC simulation, examples, and techniques (mainly focusing on spatial methods) for overcoming these challenges

Challenge in kinetic Monte Carlo simulation	Example	Technique for overcoming the challenge
Input to KMC (identification of microscopic processes and determination of their transition probabilities)	Diffusion of adatoms on a surface (types of jumps, such as first and second nearest neighbors, exchange mechanism, etc.)	Molecular dynamics; transition state search algorithms; transition state theory; density function theory (see Sect. 4.1)
Efficient search and update methods of transition probabilities	All KMC algorithms	n-level search methods, binary search methods and efficient update algorithms;[a] efficient process selection algorithms, e.g., [70, 71] and n-fold method [72]
Time scale separation among various processes (stiffness; low and large barriers)	Diffusion too fast in comparison to surface reactions; multiple diffusion paths with different barriers	Net-event MC method [73];[b] Probability-weighted MC method [75];[c] Rescaling of probabilities [76];[d] Partial equilibrium assumption [77,78];[e] Multiscale KMC technique (stochastic low-dimensional manifold/singular perturbation method) [79][f]
Length scale separation	Internal interfaces, standing and traveling waves, boundary conditions, gradients in external field (e.g., chemical potential, electric field)	Spatial coarse-grained Monte Carlo (CGMC) method with the local mean field stochastic closure [83–85]; Adaptive CGMC method [86–88]; CGMC method with various cluster expansions ([79], M. Katsoulakis et al., submitted) Two grid (mesh) CGMC method with pdf and microscopic transition probabilities from microscopic KMC passed to the coarse CGMC scale [79]; Wavelet transformation of the Hamiltonian [89,90]
Execution of one event at-a-time	All 'microscopic' algorithms	τ-leap (Poisson and binomial based) methods for microscopic KMC (D.G. Vlachos, submitted); τ-leap (Poisson and binomial based) methods for the CGMC method [91][g]

[a] See Sect. 4.2

[b] Its spatial homogeneous version was introduced in Vlachos [74]. Its noise is reduced with respect to a microscopic KMC simulation

[c] Never applied to a spatial simulation. Its noise gets amplified with respect to a microscopic KMC simulation

[d] This method is similar to the probability-weighted method introduced for well-mixed systems. Both of these methods are ad hoc in nature and lack theoretical foundation but could be useful for simulation

[e] Various closures are employed to approximate the fast network. Examples include a mean field approach, use of a few moments, etc

[f] A similar method with subtle differences has been applied to well-mixed (spatially homogeneous) systems in Liu and Eijnden, Samant and Vlachos, and Salis and Kaznessis [80–82]

[g] The τ-leap method has been applied to well-mixed (spatially homogeneous) systems in Vlachos, Gillespie, Rathinam et al., Tian and Burrage, Chatterjee et al., Auger et al., and Cao et al. [36,92–98]

As an alternative, the microscopic processes along with their corresponding transition probabilities can be obtained via experiments, such as field emission or fast scanning tunneling microscopy [67,121]. However, this is not a trivial or even a feasible task in many cases (many of these processes can be too fast to be observed even at low temperatures). The creation of a library (database/lookup table) of transition probabilities emerges as a powerful modeling approach in computational materials science [122,123]. Despite its importance, this first challenge of KMC simulation is not further discussed in this article.

4.2 Search and update algorithms and data structure for KMC implementation

The retrieval of information from libraries and the update following execution of an event can become a major computational bottleneck for systems with a large number of processes (see Table 1). We term the corresponding methods as *search and update algorithms*. When the entire library is sequentially searched and updated, we term the method as *global*. In contrast, when a portion of the library is searched and updated, we term the method as *local*. Due to their important role in computational cost, these methods are discussed in Sect. 6 and their computational efficiency is compared in Sect. 7.

The literature on data structures for generic implementation of KMC algorithms and on efficient algorithms for searching atomic processes and updating system configuration and transition probabilities is scattered. A generic KMC tool was discussed in Dooling and Broadbelt [9] that can model complex systems by representing atomistic processes as reactions, rather than representing the lattice surface via arrays. This approach enables random number generators, rate constants and lattice positions to be easily plugged/updated in a modular fashion.

4.3 Separation of length and time scales

Despite the substantial acceleration of on-lattice KMC simulation achieved by leaving out vibrations, free energy barriers of very different size introduce a different *separation of time scales*, namely separation of time scales among the group of *rare events*. Most KMC simulations get stuck via sampling the fast (low barrier) processes, whereas the long time dynamics is controlled via the slow (high barrier) processes that are rarely sampled. This is another major challenge of KMC simulations.

In some applications, spatial inhomogeneity is restricted to nanometer length scales. Consider the example of repulsive first-nearest neighbor interactions between adsorbates on the (100) surface of a crystal. The chessboard solution, resulting under suitable interaction strength with respect to thermal energy, is such an example of short-range inhomogeneity (nanopattern formation). In these situations, the simulation box of a KMC simulation with periodic boundary conditions (PBCs) is sufficient to resolve the length scales of interest. However, there are a number of problems where there is a *separation of length scales*. Examples include a propagating front, such as a crack, pattern formation (e.g., nanopatterns formed in heteroepitaxial growth of materials), and diffusion through a membrane under a gradient in chemical potential. In some of these applications, PBCs may not be suitable, and the entire material domain, spanning up to microns, millimeters, or even much larger scales, must be simulated. In others (e.g., pattern formation), PBCs are suitable but the wavelength of spatial heterogeneity is comparable to or larger than the typical simulation box size. This situation is obviously beyond the realm of KMC simulation, which in 2D is currently limited to $\sim 500 \times 500\,\mathrm{nm}^2$. In the last part of this paper (Sect. 8), we present recently introduced techniques that can cope with the separation of length and time scales.

Finally, experienced users of numerical methods realize that the conventional KMC method handles one event-at-a time. This is in contrast to most other numerical methods, such as MD simulation or integration of differential equations (DEs), where all equations or all species are simultaneously advanced. This one-at-a time aspect seriously limits computational efficiency of the KMC simulation. The recent introduction of multifiring methods, discussed in Sect. 9, enables firing of multiple events at once while maintaining accuracy in the solution.

In summary, there are five major challenges in KMC simulation, namely (1) creating the input, (2) algorithmic efficiency in search and update of databases, (3) time scale separation, (4) length scale separation, and (5) execution of one process-at-a time. This article focuses on the last four topics.

5 Microscopic algorithms for spatial KMC simulation

In this section, we provide first a brief description of the rejection-free methods, focusing mainly on the *direct* KMC method, followed by a firm foundation of the null-event KMC algorithm. Finally, the statistical equivalence of the two methods is shown.

5.1 Rejection-free KMC methods

5.1.1 Algorithm

An efficient implementation of the rejection-free method for spatially distributed systems, termed as the *n*-fold method (see Sect. 6.3.1 for more detailed discussion on the *n*-fold method), was introduced by Bortz, Kalos, and Lebowitz (nowadays is also abbreviated as the BKL method) in Bortz et al. [72]. The following year, Gillespie introduced two different rejection-free algorithms for well-mixed systems, namely, the direct and the first reaction method [66,124]. Both the direct and the first reaction methods find widespread applications in spatially distributed systems as well. The physics/mathematics and the chemistry communities refer to the former (the BKL method) and the latter (Gillespie's work), respectively, and rarely reference each other.

The best-known and most used rejection-free method is the *direct KMC method* [66,124]. It involves two steps, namely one (1) computes a priori the microscopic rates of all processes and (2) selects a process and a site using a single random number. The *first reaction method* uses one random number for each process, and as a result, it is typically expensive and not as widely spread. The most recently introduced, *next reaction method* [125] is another efficient selection method (see also Sect. 6 on fast search and update methods).

The direct KMC method uses fewer random numbers than the first reaction method, has been the most popular, and is discussed next (we return to the next reaction method below). The *j*th process at the *i*th site is selected with a probability

$$p_{ij} = \Gamma_{ij} / \sum_{j=1}^{N_\mathrm{P}} \sum_{i=1}^{N_\mathrm{L}} \Gamma_{ij} = \Gamma_{ij} \Big/ \sum_{i=1}^{N_\mathrm{L}} \Gamma_{i,\mathrm{tot}} = \Gamma_{ij} / \Gamma_{\mathrm{tot}}. \tag{13}$$

Here $\Gamma_{ij} = \Gamma_j \varepsilon_{ij}$ is the transition probability of the jth process at site i, $\Gamma_{i,\mathrm{tot}}$ the total transition probability of all processes at site i, and Γ_{tot} is the total transition probability over the entire lattice. All Γ_{ij} and $\Gamma_{i,\mathrm{tot}}$ are computed a priori, i.e., prior to selecting an event. In the direct KMC method, there are two steps.

Step 1 The (I, J) pair is selected using a uniform distribution random number $\zeta_1 \in (0, 1)$ according to

$$\sum_{j=1}^{J-1} \sum_{i=1}^{I} \Gamma_{ij} / \Gamma_{\mathrm{tot}} < \zeta_1 \le \sum_{j=1}^{J} \sum_{i=1}^{I} \Gamma_{ij} / \Gamma_{\mathrm{tot}}. \tag{14}$$

Step 2 The time is advanced via an increment given from the exponential distribution

$$\Delta \mathrm{t} = -\frac{\ln \zeta_2}{\Gamma_{\mathrm{tot}}}, \tag{15}$$

where $\zeta_2 \in (0, 1)$ is another uniform distribution random number. The average time step is the inverse of the total transition probability and can be written as (we omit an indicator of time averaging since the lack of a random number makes this notation obvious)

$$\Delta \mathrm{t} = \frac{1}{\Gamma_{\mathrm{tot}}} = \frac{1}{\sum\limits_{j=1}^{N_\mathrm{P}} \sum\limits_{i=1}^{N_\mathrm{L}} \Gamma_{ij}} = \frac{1}{\sum\limits_{j=1}^{N_\mathrm{P}} \sum\limits_{i=1}^{N_\mathrm{L}} \varepsilon_{ij} \Gamma_j} = \frac{1}{\sum\limits_{j=1}^{N_\mathrm{P}} n_j \Gamma_j} = \frac{1}{N_\mathrm{L} \sum\limits_{j=1}^{N_\mathrm{P}} \varphi_j \Gamma_j}. \tag{16}$$

Here n_j is the number of sites of the lattice that can participate in process j, i.e., the class size, and $\varphi_j = n_j / N_\mathrm{L}$ is the fraction of sites that can participate in process j.

5.1.2 CPU scaling laws of the direct rejection-free KMC method

The computational cost of the direct KMC method, in both (1) building the transition probability matrix $\{\Gamma_{ij}\}_{i=1}^{N_\mathrm{L}}{}_{j=1}^{N_\mathrm{P}}$ at each event and (2) selecting one element from the matrix, increases linearly with the total number of transition probabilities $N_\mathrm{P} * N_\mathrm{L}$ (i.e., the size of the transition probability matrix). As a result, the computational cost escalates with both the number of processes and with the lattice size. This scaling holds on *a per event basis*.

Building the entire transition probability matrix (updating of the matrix elements) and selecting one element, as done in the original Gillespie algorithm, requires scanning the entire lattice and is hereafter termed *global updating and search* (see also Sect. 4.2) Global updating and search methods limit the direct KMC method to systems of a small lattice size and a few processes. Obviously, more efficient algorithms can be used, as discussed in Sect. 6.

An important point underscored by Eq. 16 is that the time step is inversely proportional to the lattice size. As a result, as the lattice size increases, not only updating and searching per each event takes more time but also the real time reached is much smaller, and thus, additional KMC events are needed to reach the same real time.

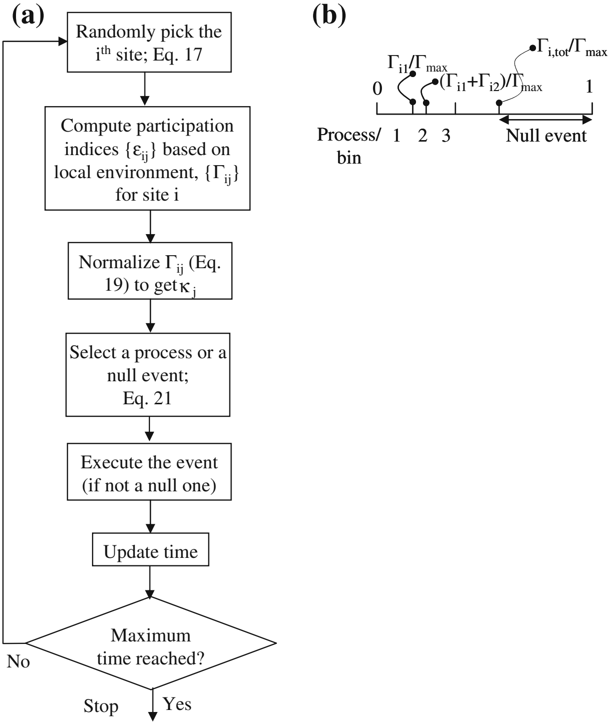

(a)
Randomly pick the i^th site; Eq. 17
Compute participation indices {ε_ij} based on local environment, {Γ_ij} for site i
Normalize Γ_ij (Eq. 19) to get κ_j
Select a process or a null event; Eq. 21
Execute the event (if not a null one)
Update time
Maximum time reached?
No
Stop
Yes
(b)
Γ_i,tot/Γ_max
Γ_i1/Γ_max
0
(Γ_i1+Γ_i2)/Γ_max
1
Process/ bin
1 2 3
Null event

Fig. 3 (**a**) Flowchart of the null-event algorithm. (**b**) Schematic for selection of a process or a null event, using linear search

Consequently, in the simplest direct KMC method, the CPU needed to reach the same real time scales as N_{L}^{2}. This scaling obviously makes simulation of large domains prohibitive.

The direct KMC method uses two random numbers at each event. This is another important point, since the CPU in some cases may be controlled from the cost of calling the random number generator.

5.2 A general null-event algorithm

5.2.1 Algorithm

Earlier lattice KMC algorithms employed a null-event approach, e.g., [35,126] and were specific to a particular application. Here we present the algorithm and discuss the time scale and null-event issues in more general terms than done previously in the literature. In null-event algorithms, there are four steps (see Fig. 3).

Step 1 A site, say *i*, is randomly chosen among all lattice sites from a uniform distribution, i.e., with an equal chance. The probability of picking the *i*th site is

$$p_i^{\mathrm{site}} = 1/N_{\mathrm{L}}. \tag{17}$$

This is accomplished using a random number $\zeta_1 \in (0,1)$ from a uniform distribution; the picked site is determined from

$$\mathrm{I} = \mathrm{floor}(\zeta_1 N_{\mathrm{L}}) + 1. \tag{18}$$

Here the command 'floor' indicates rounding of the real number $\zeta_1 N_{\mathrm{L}}$ to the closest lower integer. Obviously, all sites do not have the same sum of transition probabilities $\Gamma_{i,\mathrm{tot}} = \sum_j \Gamma_{ij}$, and thus, a correction needs to be done to account for the equal probability of picking sites (see below).

Step 2 The transition probabilities $\{\Gamma_{ij}\}_{j=1}^{N_{\mathrm{p}}}$ are computed for that site only after the site is picked (a posteriori calculation)

$$\Gamma_{ij} = \Gamma_j \varepsilon_{ij}, \quad j = 1, \ldots, N_{\mathrm{p}} \tag{19}$$

using the known process transition probabilities $\{\Gamma_j\}_{j=1}^{N_{\mathrm{p}}}$ and the participation indices. In this step, the participation indices need actually to be computed for this site by searching within the radius of interactions ($L_{\mathrm{potential}}$). Since $\{\varepsilon_{ij}\}_{j=1}^{N_{\mathrm{p}}}$ vary with time, $\{\Gamma_{ij}\}_{j=1}^{N_{\mathrm{p}}}$ are also time-dependent.

Step 3 These transition probabilities are then normalized with a suitable microscopic rate, $\Gamma_{\max}$, to provide the probability of executing the jth process at the ith site

$$\kappa_j = \frac{\Gamma_{ij}}{\Gamma_{\max}} = \frac{\Gamma_j \varepsilon_{ij}}{\Gamma_{\max}}. \tag{20}$$

The normalization rate $\Gamma_{\max}$ makes all probabilities less than 1, and should be chosen to be the same for all sites of the entire lattice rather than for the selected site. This is an important issue that is discussed below. A convenient way is to choose $\Gamma_{\max}$ as the maximum (over the entire lattice) sum of all microscopic rates at a site

$$\Gamma_{\max} = \max_{i=1,\ldots,N_{\mathrm{L}}} \sum_{j=1}^{N_{\mathrm{p}}} \Gamma_{ij} = \max_{i=1,\ldots,N_{\mathrm{L}}} \sum_{j=1}^{N_{\mathrm{p}}} \Gamma_j \varepsilon_{ij}. \tag{21}$$

This is a subtle and often overlooked issue (by new comers to the field) that ensures that different sites are picked appropriately, i.e., with a probability, that is, proportional to the overall transition probability of each site, according to Eq. 13. As a result of this normalization, typically a null bin exists, and when the random number falls in this bin, no event is executed (this explains the name of a null-event algorithm). In our calculations $\Gamma_{\max}$ is typically computed at the beginning of a simulation, depending on the participation indices of sites, according to Eq. 21 and is thereafter kept constant. By its definition, $\Gamma_{\max}$ is the maximum possible transition probability *at a single site* rather than of the entire lattice.

Step 4 A process, say J, is randomly picked using another uniform distribution random number, ζ_2 in the (0,1) interval

$$\sum_{j=1}^{J-1} \kappa_j < \zeta_2 \le \sum_{j=1}^{J} \kappa_j. \tag{22}$$

Once a process is selected and executed, the algorithm is repeated. A flowchart of the algorithm and a schematic of the selection process are shown in Fig. 3. Overall, the algorithm is fairly easy to implement and has minimum bookkeeping.

5.2.2 Null bin and success probability

The probability for a successful event in a null-event algorithm deserves more attention. The null bin at the *i*th site has a size of $\Gamma_{\max} - \sum_{j=1}^{N_{\mathrm{p}}} \Gamma_{ij}$. The probability of a null event on the *i*th site is

$$p_i^{\mathrm{null}} = \frac{\Gamma_{\max} - \sum_{j=1}^{N_{\mathrm{p}}} \Gamma_{ij}}{\Gamma_{\max}} = 1 - \frac{\sum_{j=1}^{N_{\mathrm{p}}} \Gamma_{ij}}{\Gamma_{\max}} = 1 - \frac{\Gamma_{i,\mathrm{tot}}}{\Gamma_{\max}}. \tag{23}$$

Here $\Gamma_{i,\mathrm{tot}}$ is the total transition probability at the *i*th site. Note that by design (Eq. 21), $\Gamma_{\max} \geq \Gamma_{i,\mathrm{tot}}$. Thus, $p_i^{\mathrm{null}} \leq 1$ (a well-behaved situation).

The probability of an event being successful at the *i*th site is

$$p_i^{\mathrm{success}} = 1 - p_i^{\mathrm{null}} = \frac{\sum_{j=1}^{N_{\mathrm{p}}} \Gamma_{ij}}{\Gamma_{\max}} = \frac{\Gamma_{i,\mathrm{tot}}}{\Gamma_{\max}}. \tag{24}$$

Based on Eq. 21, $p_i^{\mathrm{success}} \leq 1$ (a well-behaved situation). One can easily see that a large $\Gamma_{\max}$ results in a large null bin, i.e., the probability of a null event is high and more null events are encountered. Thus, a large null bin is detrimental to the efficiency of null-event algorithms.

The introduction of a null bin corrects the equal probability used in choosing sites, according to Eq. 17, by penalizing sites with a lower transition probability $\Gamma_{i,\mathrm{tot}}$ to likely give more null events. It should be noted that when $\Gamma_{\max}$ is equal to $\Gamma_{i,\mathrm{tot}}$, then the null bin is zero and no null events occur ($p_i^{\mathrm{success}} = 1$). This idea comes from rejection-free algorithms but obviously indicates that sites are picked completely randomly (Eq. 17 only is employed) and not based on their microscopic rates, i.e., this choice of $\Gamma_{\max}$ is incorrect. This aspect underscores the importance of proper normalization of microscopic rates in a null-event algorithm. Therefore, one should typically employ Eq. 21 or written in a different way

$$\Gamma_{\max} = \max_{i,\underline{\sigma}}(\Gamma_{i,\mathrm{tot}}) \tag{25}$$

for all possible configurations $\underline{\sigma}$.

The probability of the *j*th process happening on the *i*th site, p_{ij}, depends on the probability that the *i*th site is being picked among all lattice sites, p_i^{site}, on the probability that an event occurs at this site, p_i^{success}, and on the probability that this process is selected among all processes at this site, κ_j, (this is a sequence of conditional probabilities)

$$p_{ij} = p_i^{\mathrm{site}} p_i^{\mathrm{success}} \kappa_j = \frac{1}{N_{\mathrm{L}}} \frac{\Gamma_{i,\mathrm{tot}}}{\Gamma_{\max}} \frac{\Gamma_{ij}}{\Gamma_{i,\mathrm{tot}}} = \frac{1}{N_{\mathrm{L}}} \frac{\Gamma_{ij}}{\Gamma_{\max}} = \frac{1}{N_{\mathrm{L}}} \frac{\Gamma_j \varepsilon_{ij}}{\Gamma_{\max}}. \tag{26}$$

Note that p_{ij} from Eq. 26 differs from (and is generally smaller than) that of Eq. 13. This difference arises because of null-events.

The probability of picking the *j*th process over the entire lattice is

$$\psi_j = \sum_{i=1}^{N_{\mathrm{L}}} p_{ij} = \sum_{i=1}^{N_{\mathrm{L}}} \frac{1}{N_{\mathrm{L}}} \frac{\Gamma_{ij}}{\Gamma_{\max}} = \frac{1}{N_{\mathrm{L}}} \frac{\Gamma_{\mathrm{tot},j}}{\Gamma_{\max}}. \tag{27}$$

Here $\Gamma_{\mathrm{tot},j}$ is the transition probability of the jth process (over the entire lattice)

$$\Gamma_{\mathrm{tot},j} = \sum_{i=1}^{N_{\mathrm{L}}} \Gamma_{ij} = \sum_{i=1}^{N_{\mathrm{L}}} \Gamma_j \varepsilon_{ij} = \Gamma_j \sum_{i=1}^{N_{\mathrm{L}}} \varepsilon_{ij} = n_j \Gamma_j \tag{28}$$

and is simply the number of sites n_j which participate in the process (i.e., for which $\varepsilon_{ij} = 1$) times the transition probability Γ_j of this process. This transition probability is precisely the rate of a class defined for this process in the n-fold rejection-free algorithm of Bortz et al. [72] (see Sect. 6.3.1).

Finally, the probability of picking any process over the entire lattice is

$$\psi_{\mathrm{tot}} = \sum_{j=1}^{N_{\mathrm{p}}} \psi_j = \frac{1}{N_{\mathrm{L}}} \frac{\Gamma_{\mathrm{tot}}}{\Gamma_{\max}}. \tag{29}$$

Since $\Gamma_{\max} \geq \Gamma_{i,\mathrm{tot}}$, $\sum_{i=1}^{N_{\mathrm{L}}} \Gamma_{\max} \geq \sum_{i=1}^{N_{\mathrm{L}}} \Gamma_{i,\mathrm{tot}}$ or $N_{\mathrm{L}}\Gamma_{\max} \geq \sum_{i=1}^{N_{\mathrm{L}}} \Gamma_{i,\mathrm{tot}} = \Gamma_{\mathrm{tot}}$. The latter relation ensures that $\psi_{\mathrm{tot}} \leq 1$ in Eq. 29. Note that the summation of probabilities in Eqs. 24–28 was possible since the processes are independent of each other.

Equation 29 gives the overall probability of having a successful event in a null-event algorithm, and is a measure of the efficiency of a null-event algorithm. One could pick an arbitrarily large $\Gamma_{\max}$, but this will reduce the overall probability of having successful events. A large separation of transition probabilities (stiffness) results in $\Gamma_{\max}$, that is, much larger than the transition probabilities of the slow processes. In such a case, the probability of a rare event being successful is very low. Even though this feature is also common to the rejection-free KMC algorithms, it is more severe in the case of the null-event KMC method because too many unsuccessful events occur. This partly rationalizes the observation made in Reese et al. [127], regarding the inefficiency of the null-event algorithm for stiff systems.

Given a total of $N_{\mathrm{e,null}}$ events in a null-event algorithm, the number of successful events will be

$$N_{\mathrm{e,null}}^{\mathrm{success}} = N_{\mathrm{e,null}} \psi_{\mathrm{tot}}. \tag{30}$$

5.2.3 Time increments and time clocks

Next we turn to the time increment issue. This problem has been touched upon only sporadically for simple unimolecular processes [128–130] and rarely for non-linear and/or complex problems [47,99]. The time can be advanced based on a single process either at a site or over the entire lattice *every time this process is picked*. Below we illustrate the approach based on a single process over the entire lattice.

Consider a clock advancing time only when the jth process is picked at any site of the entire lattice. The average time elapsed each time a *successful* jth process occurs is $1/\Gamma_{\mathrm{tot},j}$. Thus, in a simulation the average time is advanced according to

$$t_j = t_j + 1/\Gamma_{\mathrm{tot},j}. \tag{31}$$

Note that when a null-event or another process is picked, the jth process clock does not advance.

Next we compare the time clocks of the direct, rejection-free and the null-event KMC methods. Since not all events are successful, the probability of success of an event is considered to theoretically compare methods. The average time elapsed depends on the probability of the process being picked up and its corresponding average time increment and is given by

$$\text{Clock Increment}^{j\text{ th process}}_{\text{null-event}} = \Delta \mathrm{t}_j = \psi_j \frac{1}{\Gamma_{\text{tot},j}}$$
$$= \frac{1}{N_{\mathrm{L}}} \frac{\Gamma_{\text{tot},j}}{\Gamma_{\max}} \frac{1}{\Gamma_{\text{tot},j}} = \frac{1}{N_{\mathrm{L}}} \frac{1}{\Gamma_{\max}}, \quad j = 1, \ldots, N_{\mathrm{p}}. \quad (32)$$

In the direct KMC method, the average time advanced per event is $\frac{1}{\Gamma_{\text{tot}}}$, i.e.,

$$\text{Clock Increment}_{\text{direct}} = \frac{1}{\Gamma_{\text{tot}}}. \quad (33)$$

Suppose the same experiment is repeated N_{e} times (events), without actually executing the event (in order to obtain statistics). In order to achieve the same number of successful events between null-event and rejection-free methods, one has $N^{\text{success}}_{\text{e,null}} = N_{\mathrm{e}}$, or using Eq. 30

$$N_{\text{e,null}} \psi_{\text{tot}} = N_{\mathrm{e}}. \quad (34)$$

The time advanced in a null-event algorithm for the same number of successful events, using the jth process as our time clock, is then (using Eqs. 32, 34, and 29)

$$t_{\text{null}} = \Delta \mathrm{t}_{j,\text{null}} N_{\mathrm{e}}/\psi_{\text{tot}} = \frac{1}{N_{\mathrm{L}}} \frac{1}{\Gamma_{\max}} N_{\mathrm{e}} N_{\mathrm{L}} \frac{\Gamma_{\max}}{\Gamma_{\text{tot}}} = \frac{N_{\mathrm{e}}}{\Gamma_{\text{tot}}} = t_{\text{direct}}. \quad (35)$$

This result indicates that the time advanced is theoretically *independent* of the process chosen to advance the time clock (see Eq. 32). Thus, any process can be picked for advancing time. In practice, it is advantageous to select a process that occurs frequently in order to have higher resolution of when various events happen. Alternatively, a group of processes can be picked for updating time. For example, one can use an average time increment of $\frac{1}{N_{\mathrm{L}}} \frac{1}{\Gamma_{\max}}$ every KMC event (independent of whether the event is successful or not) without relying on any specific process.

It is important to pick a process to easily compute $\Gamma_{\text{tot},j}$. For example, in the case of a site-independent process (say process J), with a transition probability of Γ, $\Gamma_{\text{tot},J} = N_{\mathrm{L}} \Gamma$ is a fixed number during the simulation and needs to be calculated only once. Even if a processes is picked, whose transition probability is site dependent, for advancing the time clock, one can compute $\Gamma_{\text{tot},J}$ at the beginning of a simulation by summing over the entire lattice and 'locally' update the clock increment by subtracting the old and adding the new contributions. An example was given for the case of diffusion/reaction processes in Mayawala et al. [131].

5.2.4 CPU scaling laws

Since one selects randomly a site, the entire matrix of transition probabilities is never searched and updated. As a result, the CPU per event is lattice-size independent. Once a site is selected, one has to search up to N_{p} processes, so the CPU scales at most linearly with N_{p}. In summary, the CPU per event scales as $N^0_{\mathrm{L}} N_{\mathrm{p}}$. The CPU for the same real time obviously increases linearly with lattice size (see last part of Eq. 32) as in a rejection-free method.

A major advantage of the null-event method, in comparison to the direct, rejection-free KMC algorithm (Sect. 5.1.2.), is its lattice size independence. On other hand, null events diminish, to some extent, this advantage (there is an overhead associated with execution of more events). Section 5.4 compares the CPUs of two methods in more detail and provides some tips for method selection. Numerical examples in Sect. 7 provide further guidance on method selection.

Another major advantage of the null-event algorithm is its ease of implementation. When a system does not exhibit stiffness and has a small number of possible processes that can occur, then the probability of successful events is high and the method is very efficient. Under these circumstances, the null-event algorithm should be preferred given its ease of implementation.

5.3 Statistical equivalence of null-event and rejection-free KMC methods

There are two aspects that demonstrate the equivalence of the null-event and rejection-free KMC methods. First the probability of picking processes from the two methods should be same, so the correct probability density function is computed. Second, real time after a fixed number of successful events should be statistically the same in both methods, so that the correct dynamics is predicted.

The ratio of probabilities at two sites for any two processes from Eqs. 26, for the null-event algorithm, and 13, for the rejection-free algorithm, is identical. Furthermore, Eq. 35 indicates that the time advanced by the two methods is the same once an appropriately large number of events are carried out in the null-event algorithm to account for the unsuccessful events. Numerical examples in Sect. 7 further illustrate these points.

5.4 Comparison of computational efficiency, memory, and implementation of the rejection-free and null-event KMC methods

In early spatial KMC simulations, it was typically assumed that since a process is always selected at every trial of a rejection-free KMC method, i.e., there are no null events, the rejection-free KMC method exhibits superior performance than a null-event algorithm employed e.g., in Gilmer, Ziff et al., Gilmer and Bennema and van der Eerden et al. [29,35,126,132]. While this is often the case, this is not generally true and depends heavily on the search and update methods employed. For example in Reese et al. [127], the implementation and the efficiency of rejection-free and null-event KMC methods with local and global search and update methods (see Sect. 6 for these methods) were compared for a specific example of a surface reaction. It was concluded that the rejection-free algorithm is more efficient for rare event dynamics (stiff systems) but a null-event algorithm could, under some conditions, outperform the rejection-free method due to lack of book-keeping and updating.

In comparison to the direct, rejection-free KMC method, the number of events needed in a null-event algorithm to achieve the same number of successful events is increased by a factor of $1/\psi_{\mathrm{tot}} - 1 = N_{\mathrm{L}} \frac{\Gamma_{\max}}{\Gamma_{\mathrm{tot}}} - 1$ (according to Eqs. 29 and 30) due to the presence of null events. Therefore, the CPU times of the two methods for reaching the same real time scale as

$$\frac{\mathrm{CPU}_{\mathrm{rejection-free}}}{\mathrm{CPU}_{\mathrm{null-event}}} \approx \frac{N_{\mathrm{L}} N_{\mathrm{p}}}{N_{\mathrm{L}}{}^{0}N_{\mathrm{p}}} \frac{1}{1/\psi_{\mathrm{tot}}} = \frac{\Gamma_{\mathrm{tot}}}{\Gamma_{\max}} \leq N_{\mathrm{L}}. \tag{36}$$

Since Γ_{tot} is typically larger than Γ_{max}, Eq. 36 indicates that the simple rejection-free KMC method (with global search and update methods) can in fact be much more inefficient than the null-event algorithm. Specifically, when $N_{\mathrm{L}}\Gamma_{\mathrm{max}} \approx \Gamma_{\mathrm{tot}}$, the probability of successful events in the null-event algorithm is high and the null-event algorithm is very efficient because global searches and updates are absent.

In local update rejection-free methods, the transition probabilities of all interacting sites of a central site have to be updated. As elaborated in Sect. 6, this cost scales linearly or faster with the number of interacting neighbors N_{interac} for site i and can be a considerable fraction of a calculation. In contrast, the null-event method exhibits no bookkeeping (involves minimum calculation, since only the transition probabilities at the chosen site are determined).

In summary, because of there is no overhead associated with updating classes and searching the entire lattice, null-event algorithms usually outperform rejection-free algorithms, when the latter employ global searches and updates. This may (albeit rarely) be true even for local-based rejection-free methods when the probability of successful events happens to be close to 1.

Null-event algorithms are easy to implement and do not suffer from large memory requirement that can become a bottleneck in rejection-free algorithms for large size systems involving many processes. On the other hand, simple rejection-free methods are also easy to implement but are very inefficient. Rejection-free methods, which use efficient search and update algorithms, are very efficient, but require much more complicated coding as discussed in the next section.

6 Search and update algorithms for KMC Simulation

Process selection and updates of the transition probabilities in a KMC simulation (see Fig. 3 for an example depicting the null-event algorithm) are computationally intensive. As alluded to above, the CPU requirements per KMC event increase linearly with increasing the number of processes (N_{p}) and the lattice size (N_{L}) when global search and update methods are employed. Furthermore, the computational cost escalates with an increase of the interaction potential cut-off length ($L_{\mathrm{potential}}$). Even in the case of a null-event KMC algorithm, which identifies a site using Eq. 18, process identification (via Eqs. 19–22) can become a bottleneck, when the number of possible processes N_{p} on a single site is large. Such a situation is typically encountered in multicomponent systems. For example, hundreds of diffusion and reaction processes are typically possible on each lattice site on a catalyst surface [133] and in defects in materials [56]. In microkinetic modeling and in biological systems, one frequently encounters reaction networks comprising of hundreds to thousands of elementary reaction steps [134,135]. For these reasons, efficient algorithms for search and update are critical for accessing reasonably large time scales on large lattices.

In this section, a brief overview of challenges in search and update algorithms, encountered in different areas of computational science, and approaches employed to overcome these challenges are discussed. This is followed by a detailed discussion of such algorithms.

The literature on search and update algorithms is broad and application specific. Numerous search algorithms have been developed in computer science for small as well as large databases [136]. Typically, these algorithms require as inputs a *key*, the information to be retrieved, and an algorithm-specific implementation for the database. $\zeta\Gamma_{\text{tot}}$ is an example of a key in the context of the direct KMC simulation. The information of interest is detailed information about a specific element of the database. Database implementations commonly include arrays/linked lists, binary trees, and graph structures. The CPU requirements of these search algorithms typically scale as $\mathrm{O}(\log(\log(N_{\text{db}})))$ to $\mathrm{O}(N_{\text{db}})$, where N_{db} is the size of the database.

The need for efficient update of transition probabilities in KMC algorithms parallels the need for fast update algorithms in other areas of computational science. For example, in MD and Metropolis MC simulations, the energy landscape experienced by an atom or a molecule needs to be evaluated at each time step before solving the equations of motion. Algorithms, such as the linked-list and Ewald summation methods [2, 3], enable efficient on-the-fly estimation of the potential energy surface. A similar approach may be employed for off-lattice KMC modeling. However, more efficient algorithms exist for on-lattice KMC simulation, which take advantage of the fixed lattice.

Some efficient algorithms for KMC of well-mixed systems were recently discussed in Gibson and Bruck [125]. A dependency graph was employed for locally updating processes (transition probability matrix) that are affected when a particular atomic process is executed. Furthermore, an indexed priority queue was employed that efficiently identifies the process with the least firing time using the first reaction method. A binary tree search was employed in the same paper for efficiently selecting processes using the rejection-free KMC algorithm. Besides these approaches developed specifically for KMC, other approaches developed in computational science can be implemented with minor modification. Next, we discuss some of these algorithms and their efficiency.

6.1 *n*-level linear search methods

6.1.1 Linear search

The linear search is probably the simplest and most frequently employed search technique for selecting a process in a KMC simulation. The method employs an array or a linked list, as shown in Figs. 3b and 4a. In this scheme, $\zeta\Gamma_{\text{tot}}$ is known at the beginning of the search, and the process is sequentially searched until the criterion given by Eq. 14 or 22 is satisfied. The CPU requirements for linear search scales as $O(N_{\text{list}})$, when all processes have the same order of magnitude average transition probability. Here N_{list} is the average size of the list of processes being searched. For example, $N_{\text{list}} = N_{\text{p}} * N_{\text{L}}$ for the rejection-free KMC method.

When time scale separation between processes exists, the linear search method is most efficient when fast processes (processes with large expected transition probability) are located at the beginning of the array/linked list [137].

6.1.2 Two-level linear search

A slightly modified version of the linear search, shown in Fig. 4b, involves representing processes by a 2D square matrix of size $N_2 \times N_2$, where $N_2 = \sqrt{N_{\text{list}}}$ when $\sqrt{N_{\text{list}}}$ is an integer and $N_2 = \text{floor}\left(\sqrt{N_{\text{list}}}\right) + 1$, otherwise. The sum of transition probabilities of all processes in the ith column, denoted as $\Gamma_{c,\text{tot}}$, and $\zeta\Gamma_{\text{tot}}$ are known at the beginning of the search. Two linear searches are performed. The first linear search selects the Cth column using $\left\{\Gamma_{c,\text{tot}}\right\}_{i=1}^{N_{\text{list}}}$ as weights. For example, in the case of the rejection-free KMC algorithm, this search criterion is given by

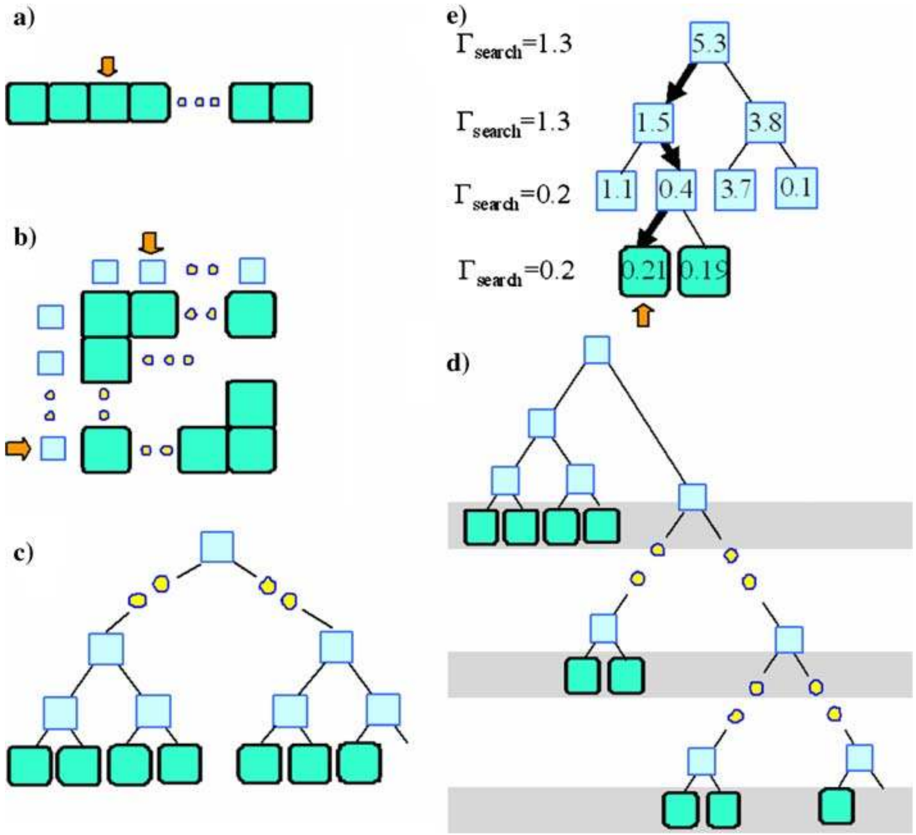


Fig. 4 (**a**) Implementation of the linear search scheme using a 1D array of size N_{list}. *The arrow* indicates the process selected based on Eq. 14 or 22. (**b**) Implementation of two-level linear search scheme with a 2D array of size $N_2 \times N_2$ (see text). A column, shown by the vertical red arrow, is selected first using Eq. 37. A process in the selected column is next picked using Eq. 14 or 22. Implementation of the binary search scheme using a (**c**) balanced and **d** unbalanced binary tree. The gray bands in (**d**) denote spectral bands, i.e., time scales of processes. In this case, all processes with the same time scale are found at the same level of the unbalanced binary tree. (**e**) Logical operations are employed to search for a process, starting from the top of the tree toward the bottom. Squares with rounded (*sharp*) corners denote single (*multiple*) processes

$$\sum_{c=1}^{C-1} \Gamma_{c,\text{tot}} / \Gamma_{\text{tot}} < \zeta \leq \sum_{c=1}^{C} \Gamma_{c,\text{tot}} / \Gamma_{\text{tot}}. \tag{37}$$

The second linear search determines the process to be executed using Eq. 14. The CPU requirement of this method scales as $O(N_2)$. As an example, when $N_{\text{list}} \sim 10^6$, the two-level linear search is about 10^3 times faster than the simple linear search method.

6.1.3 n-level linear search

Larger CPU savings are obtained with an n-dimensional matrix of size $N_n \times N_n \times \cdots \times N_n$ (n times), where $N_n{=}{N_{\text{list}}}^{1/n}$, when ${N_{\text{list}}}^{1/n}$ is an integer, and $N_n{=}\text{floor}\left({N_{\text{list}}}^{1/n}\right){+}1$, otherwise. For example, when $N_{\text{list}} \sim 10^6$, a three-level linear search is about 10^4 times faster than the linear search technique. Due to the increasing number of linear searches, the CPU for a n-level linear search scales as $nO(N_n)$.

To the best of our knowledge, the simple extension of linear search method to 2- and n-level linear search has not been reported in the KMC literature. The n-dimensional matrix is best implemented via the use of pointers. However, it is certainly not the most efficient KMC-search technique.

6.2 Binary search

A binary search algorithm employs a divide-and-conquer approach to find the location of a particular value in a list of processes [136]. The most effective way of employing binary search is via a binary tree using pointers (see Fig. 4c). An example of a binary tree search for KMC simulations was given in Gibson and Bruck and Schulze [70, 125].

In this data structure, each node has two children nodes in the level below it. There are N_{level} levels in the binary tree, where $N_{\text{level}} = \log_2(N_{\text{list}})$, when $\log_2(N_{\text{list}})$ is an integer, and $N_{\text{level}} = \text{floor}(\log_2(N_{\text{list}})) + 1$, otherwise. A node at the bottom level represents a single process and has a value given by the process transition probability. When $N_{\text{list}} < N_{\text{level}}$, $N_{\text{level}} - N_{\text{list}}$ bottom nodes have a value zero and do not represent any process. Each node at a level above the bottom level contains the sum of values of its children. The head node (topmost node) contains the sum of transition probabilities of all processes. The values in the entire binary tree and $\zeta\Gamma_{\text{tot}}$ are known at the beginning of the search.

The binary search starts from the head node, and rules out either the left or the right branch, using logic operations with the help of the search variable Γ_{search} (see Fig. 4e). Initially, $\Gamma_{\text{search}} = \zeta\Gamma_{\text{tot}}$. The left branch node is selected if it has a value less than Γ_{search}. When selected, the value of Γ_{search} is retained for the lower level. When Γ_{search} has a value greater than the value of the left branch node, the right branch node is selected and Γ_{search} has a new value given by Γ_{search} minus the value of the left branch node. This procedure is repeated until the bottom-most node is encountered. This node contains the process, that is, executed.

The CPU time for the binary search scales as $O(\log_2(N_{\text{list}}))$. For example, the binary search method is 10^4–10^5 times faster for $N_{\text{list}} = 10^6$. Despite its complexity, the large efficiency in comparison to linear search methods is a major advantage of the binary search method. The weakness of the binary search method is the large storage requirements, given by $2^{N_{\text{level}}+1}$ nodes, required for creating the binary tree.

When time scale separation between processes exists, an unbalanced tree is slightly more efficient compared to a balanced tree (Fig. 4d). In an unbalanced binary tree, the fastest (slowest) processes are situated at the highest (lowest) levels. This approach is suited when the number of processes is extremely large, such that the number of levels for a balanced tree is high.

6.3 The n-fold method

The n-fold method of Bortz et al. [72] is an efficient search spatial KMC method and is an example of what is called a Hash-table search [136]. The hashing function in this example catagorizes processes according to their transition probability. It is discussed separately due to its significance and the fact that is often overlooked.

One realizes that interactions are often short-ranged, e.g., first-nearest neighbors. Consequently, each site allows only a small number of possible processes, i.e., the number of processes for which $\varepsilon_{ij} = 1$ is often much smaller than N_{p}. All sites with $\varepsilon_{ij} = 1$ are grouped into a class (termed the jth class). Let n_j be the number of sites belonging to the jth class (class size). The class size is determined from Eq. 5. In the n-fold method, first a class is selected and then a site from this class is randomly chosen. Specifically, the jth class is selected with a probability

$$p_j^{\text{class}} = \frac{\sum_{i=1}^{N_{\mathrm{L}}} \Gamma_j \varepsilon_{ij}}{\sum_{k=1}^{N_{\mathrm{p}}} \sum_{i=1}^{N_{\mathrm{L}}} \Gamma_k \varepsilon_{ik}} = \frac{\Gamma_j \sum_{i=1}^{N_{\mathrm{L}}} \varepsilon_{ij}}{\sum_{k=1}^{N_{\mathrm{p}}} \Gamma_k \sum_{i=1}^{N_{\mathrm{L}}} \varepsilon_{ik}} = \frac{n_j \Gamma_j}{\sum_{k=1}^{N_{\mathrm{p}}} n_k \Gamma_k} = \frac{n_j \Gamma_j}{\Gamma_{\text{tot}}} \tag{38}$$

using a uniform distribution random number $\zeta_1 \in (0, 1)$

$$\sum_{i=1}^{j-1} n_i \Gamma_i / \Gamma_{\text{tot}} < \zeta_1 \leq \sum_{i=1}^{j} n_i \Gamma_i, \quad j = 1, \ldots, N_{\mathrm{p}}. \tag{39}$$

Once a class is selected, a site, say m, from this class is randomly chosen using another random number $\zeta_2 \in (0, 1)$

$$m = \text{floor}(n_{\mathrm{j}} \zeta_2) + 1. \tag{40}$$

The computational cost of the selection now scales linearly with N_{p} rather than $N_{\mathrm{p}}{}^*N_{\mathrm{L}}$, i.e., the computational savings are of order $O(N_{\mathrm{L}})$. A key ingredient of the n-fold method is that one avoids scanning the entire lattice (aside from the very first time during initialization). This is then an extremely efficient search ($O(1)$) method when a small or moderate number of classes is involved.

While fast selection of a class and a site is very important, updating of the entire transition probability matrix is also of $O(N_{\mathrm{L}})$. Therefore, fast execution of the n-fold method requires that only the modified elements of the transition probability matrix are updated. We term this approach *local updating algorithm*. Construction of lists of neighbors is essential in that regard. Once a site is selected, it is only the transition probabilities of these neighboring sites and of the central site that are accessed and updated. Local updating involves complex coding. As a result, global updating is often implemented.

The potentially low, storage requirements, given by the number of processes N_{p}, and the low, CPU requirements, given by $O(1)$, are the main advantages of the n-fold method. Even though the n-fold method is one of the most efficient search algorithms for KMC simulations, tabulation of classes becomes a major issue when the number of classes is large. For instance the number of classes increases rapidly with increasing interaction potential cut-off length, $L_{\text{potential}}$. The number of classes for the submonolayer example of Sect. 2.2.3 increases as $L_{\text{potential}}^2$ in the asymptotic limit of a long potential. Additionally, the number of classes and the way of ascertaining a class type (a pattern identification procedure) depends on the lattice and the number of chemical species. For example, the number of classes in the SOS model of the (100) surface example is 10 and 48 for 1 and 2 chemical species, respectively. These issues undermine the portability of the n-fold KMC code to complex, long-range interacting systems.

6.4 Update algorithms

In this section, different techniques for updating the system configuration, interaction energies, and transition probabilities are discussed. Upon selection of a process, the populations of all chemical species are updated most efficiently using a stoichiometric matrix. A stoichiometric matrix is commonly employed in KMC simulations of well-mixed systems [124]. The (i,j) element of the sparse matrix is an integer that denotes the net number of atoms/molecules of the ith chemical species that are consumed or generated by the jth process. The sign of the (i,j) element is positive (negative) when the chemical species is generated (consumed) and zero if the species does not participate in this reaction. The memory requirements of the stoichiometric matrix can be reduced using a 1D array of size N_{p} of linked lists. Each element j of the array gives the number of particles created/consumed by process j.

In an inefficient update method, termed as the global update method [127], one reevaluates all interparticle interactions and all process transition probabilities after a process is executed (we separate interactions from probabilities because the former may not be used for some systems but could become very expensive to compute depending on the functional form of the potential). Unless the temporal accelerated τ-leap method is used (see below), this is often computationally unnecessary since changes are typically localized around the selected site where a process has just occurred. For instance, the CPU requirements of a global update method for computing only the interactions is $O(N_{\mathrm{interac}}{}^{*}N_{\mathrm{L}})$ for two-body potentials. Additionally, the CPU requirements for updating the transition probabilities (given the interactions) is $O(N_{\mathrm{p}}{}^{*}N_{\mathrm{L}})$. Here, N_{interac} is the average number of interacting neighboring sites of a single site.

Local update methods are more efficient compared to global updates; for a comparison see [127]. Interaction energies are locally updated using a list of interacting neighbors and a connectivity matrix for the transition probabilities. The CPU requirements in a local update method for computing both interactions and the transition probabilities are roughly N_{L} times lower than those of the corresponding global update method. A list of interacting neighbors is easily implemented using arrays or linked lists. Two 1D arrays are employed with the array-based list of interacting neighbors, as shown in Fig. 5 [2]. The first array (Arr1 in Fig. 5) contains interacting neighbors on a site-by-site basis. The second array (Arr2 in Fig. 5) is $2N_{\mathrm{interac}}$ long and contains index limits for the section in Arr1 where interacting neighbors are located.

Processes are locally updated via a connectivity matrix. The connectivity matrix gives a list of process transition probabilities that are updated when a particular atom/molecule participates in a process or has a modified energy landscape due to a process executed nearby. Alternatively, dependency graphs [125] can be employed instead of linked lists to update interactions and process transition probabilities.

The binary tree used with the binary search method is updated, after a process is executed, using either global or local update methods. In a global update, all branches are updated, i.e., the computational cost is proportional to $N_{\mathrm{p}}{}^{*}N_{L}$. In contrast, the local update method is the most efficient approach since only branches containing the modified process rates are updated. The computational cost of the local update method is proportional to, at most, N_{p}.

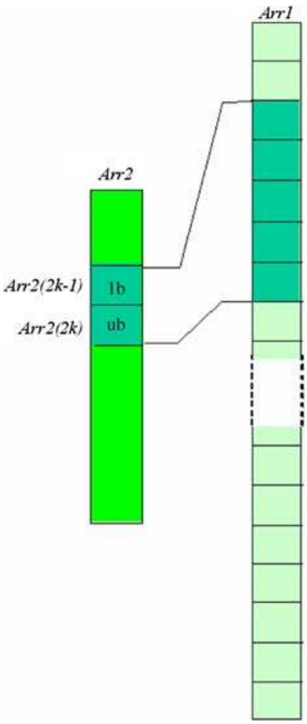


Fig. 5 Two arrays (Arr1 and Arr2) are employed to locally update interactions. Arr1 is a complete list of interacting neighbors, whereas Arr2 is a list of 'pointers.' The interacting sites of the *k*th site are stored in the dark green shaded region in Arr1. This region has indices Arr2(2k-1)–Arr2(2k). In other words, Arr2(2k-1) and Arr2(2k) are the lower and upper bounds, respectively

7 Numerical examples of equivalence and of CPU requirements of common KMC algorithms

In this section, we implement some of the search and update techniques for the KMC methods described above and we compare various methods.

7.1 Crystal growth

In order to illustrate the above issues, we employ a null-event algorithm and a rejection-free algorithm for a simple SOS crystal growth example on the (100) surface of the cubic lattice (see Sect. 2.2.1 for the model). Implementation of all algorithms was done in Matlab 7.0. The rejection-free algorithm used the *n*-fold method for efficient selection of a class and a site. Update of the transition probability matrix was done by employing the global update method; however, a vectorized code was written in Matlab to re-compute the transition probability matrix. This Matlab vectorization resulted in substantial speedup.

In this example, the time of the null-event algorithm has been advanced in two ways. First, every time a successful adsorption event happens, the time clock is incremented by an average amount of $\Delta t = 1/(\Gamma_6 N_\mathrm{L})$. Alternatively, in every MC event, the time clock is advanced by an average amount of $\Delta t = 1/(\Gamma_{\max} N_\mathrm{L})$, according to Eq. 21. In this example, $\Gamma_{\max} = \Gamma_6 + \max\{\Gamma_j\}_{j=1}^5$. For attractive interactions ($w > 0$) used in this example, $\max\{\Gamma_j\}_{j=1}^5 = \nu_\mathrm{d} \mathrm{e}^{-w/kT}$. For repulsive interactions ($w < 0$), on the other hand, one would have $\max\{\Gamma_j\}_{j=1}^5 = \nu_\mathrm{d} \mathrm{e}^{-5w/kT}$.

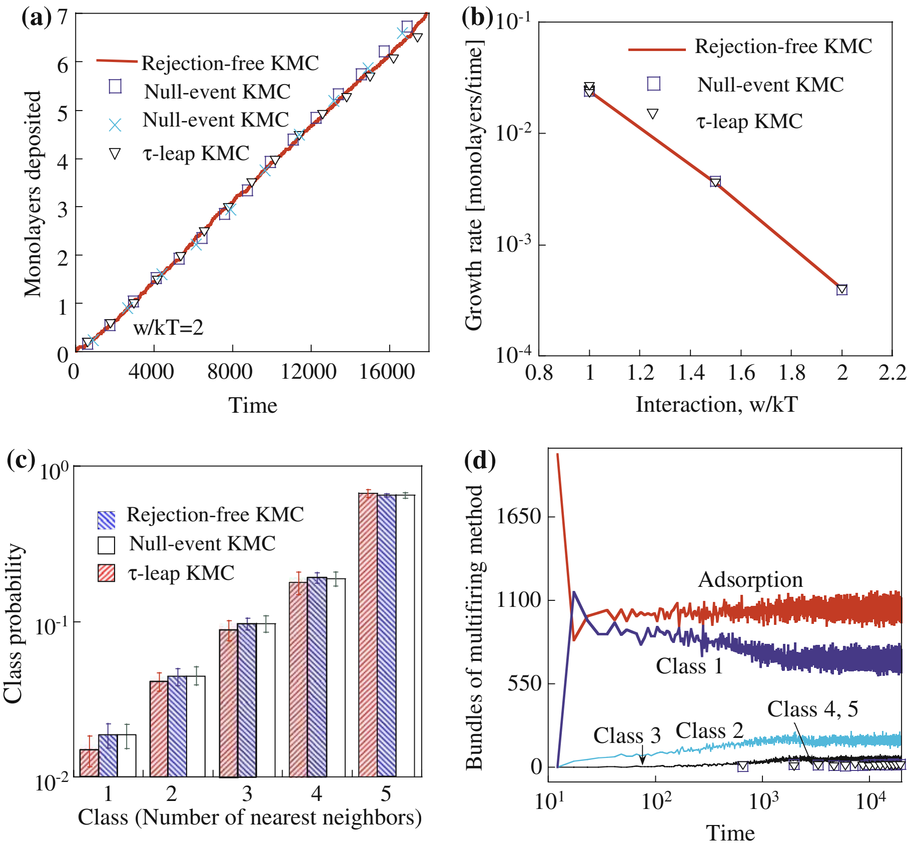


Fig. 6 Crystal growth example (Sect. 2.2.1). (**a**) Layers deposited versus time using different algorithms for $w/kT = 2$. The squares correspond to the time being calculated based on adsorption events only and the crosses based on the average time increment $\frac{1}{N_L}\frac{1}{\Gamma_{\max}}$ (see text), both for the null-event algorithm. The acceleration factor of the τ-leap method is $f = 20$. (**b**) Growth rate versus interaction strength using three algorithms. The multiple triangles at $w/kT = 1$ indicate results for seven values of the acceleration factor of the τ-leap method ranging from $f = 15$ to 1,000. Even for the largest f, the error in growth rate is small. (**c**) Probability of observing a desorption class (arithmetic mean over a long period showing in panel *a*) using the three algorithms ($f = 20$) (*error bars* denote two standard deviations). For panels **a**–**c**, the lattice size is $N_L = 40 \times 40$. (**d**) Bundles (number of events executed at each time increment) versus time from a $N_L = 200 \times 200$ lattice simulation using the τ-leap method with $f = 2{,}000$ and $w/kT = 2$. For all simulations, $\Delta\mu/kT = 1.5$ and $\nu_d = 1$

Figure 6 compares results from the different algorithms. Figure 6a shows the number of layers deposited versus time. Two sets of points are shown for the null-event algorithm corresponding to the different methods of computing time. The excellent agreement indicates the accuracy of all methods and the fact that the time clock of null-event algorithms can be advanced correctly in different ways. Figure 6b shows the growth rate versus the strength of interaction for all methods. The coincidence of points with the line underscores the equivalence of all methods. Figure 6c shows the probability of observing sites with a certain number of nearest neighbors, after quasi-equilibration has been reached, for the simulation depicted in Fig. 6a. This is a better statistic of the agreement of methods and shows that the surface topology is the same within error.

Figure 7 compares the computational efficiency of the null-event and rejection-free methods. The horizontal dotted line separates the graph into two and indicates the regime where each algorithm is more efficient. At low-interaction energies, the null-event algorithm is more efficient despite the large fraction of unsuccessful events due to the lack of updating the entire system. This is contrary to the general belief. In fact, this behavior can happen even if local updating is used (see [127] for an example). Under such conditions, the desorption transition probabilities of sites with 1 and 2 nearest neighbors are not that low in comparison to adsorption, and thus, the probability of successful desorption events is moderate. However, as the interaction strength increases, the probability of unsuccessful events goes to 1, and the null-event method becomes more inefficient. Under such conditions, the desorption transition probabilities are too low in comparison to that of adsorption, and thus, the normalization constant, $\Gamma_{\max}$, is dictated by adsorption. At the same time, the probability of having successful desorption events is too low.

7.2 Defect dynamics

The KMC method is often employed to obtain transport properties, such as diffusion coefficients, of defects. Here we focus on the model of Sect. 2.2.2. Figure 8 shows the tracer diffusion coefficient for the aforementioned surface vacancy model, which is computed as

$$D_t = \left\langle \delta^2(t) \right\rangle / t \tag{41}$$

for various defect coverages. Here $\delta^2(t)$ is the ensemble averaged displacement of the tracer particle during a time interval t. Defects are initially placed randomly on a lattice of size $N_{\mathrm{L}} = 100 \times 100$. The range of interaction potential is $L_{\mathrm{potential}} = 1$. The standard deviation in the diffusivity data is roughly 0.01 times the diffusion coefficient. Implementation was done in Fortran 90.

The results show that the diffusion coefficient decreases with increasing defect coverage because diffusion is possible only between a substrate atom and a defect, and because of frequent collisions between vacancies at large vacancy coverages. At complete defect coverage, $c_0 = 1$, $D_{\mathrm{t}} = 0\,\mathrm{m}^2\mathrm{s}^{-1}$. At low-defect coverages, the diffusion

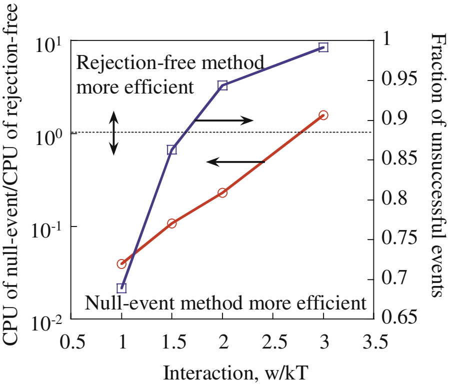


Fig. 7 Comparison of CPU of null-event and rejection-free methods (*left vertical axis*) and fraction of unsuccessful events of the null-event algorithm (*right vertical axis*)

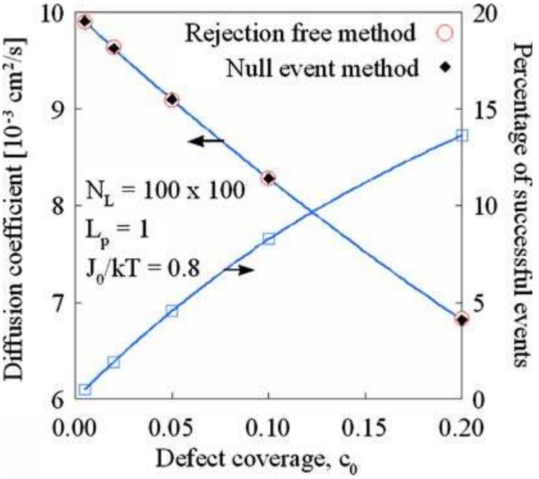


Fig. 8 Tracer diffusion coefficient (*left vertical axis*) obtained from the rejection-free and the null-event KMC methods and percentage of successful events in the null-event method (*right vertical axis*) versus defect coverage c_0

coefficient is given by $D_t \approx \nu_m a^2/4$, where a is the lattice constant. The computed tracer diffusion coefficient, in Fig. 8, matches the tracer diffusion coefficient at the asymptotic zero-defect coverage limit.

The rejection-free and the null-event KMC methods give identical results, further demonstrating the equivalence of the two methods discussed based on probability theory and shown also above with the crystal growth example. The percentage of successful events in the null-event algorithm is also shown as a function of defect coverage in Fig. 8. At low coverages, the probability of selecting a defect site, according to Eq. 18, is low, and the probability of a successful event (Eq. 29) is also low. However, the method becomes more efficient as the coverage increases.

Figures 9–11 compare the CPU requirements of various parts of a KMC simulation for a low-defect coverage of $c_0 = 0.005$. Figures 9 and 10 employ the rejection-free algorithm, whereas Fig. 11 is obtained using the null-event algorithm. Figure 9 employs a binary tree structure to represent the transition probabilities, whereas the other two figures employ array structures. Local (global) updates of the transition probabilities are employed in panels a and *c* (panels *b* and *d*) of these three figures. Local (global) interaction energy updates are employed in panels *a* and *b* (panels *c* and *d*) of the figures. The CPU for the same fixed real time, obtained with the Portland group (PG) profiler, is mentioned above the pie chart in each panel. Each pie chart gives the relative computational requirements of various KMC parts. Since similar conclusions are obtained regarding the efficiency of local versus global search and update methods from all three figures, we discuss Fig. 9 in more detail.

Figure 9a shows that when local update methods are employed, the least computational resources are needed. After each diffusion event, the interaction energy at four lattice sites, and the transition probabilities of 20 diffusion processes need to be updated. For this reason, and given the costly exponentiation involved in Eq. 9, the local update of transition probabilities is more time consuming than the local interaction update step (note though that this is problem specific). The CPU requirements for process search with the binary tree search are negligible compared to the update steps because of the small number of levels present in the binary tree (N_{level} is 11 in

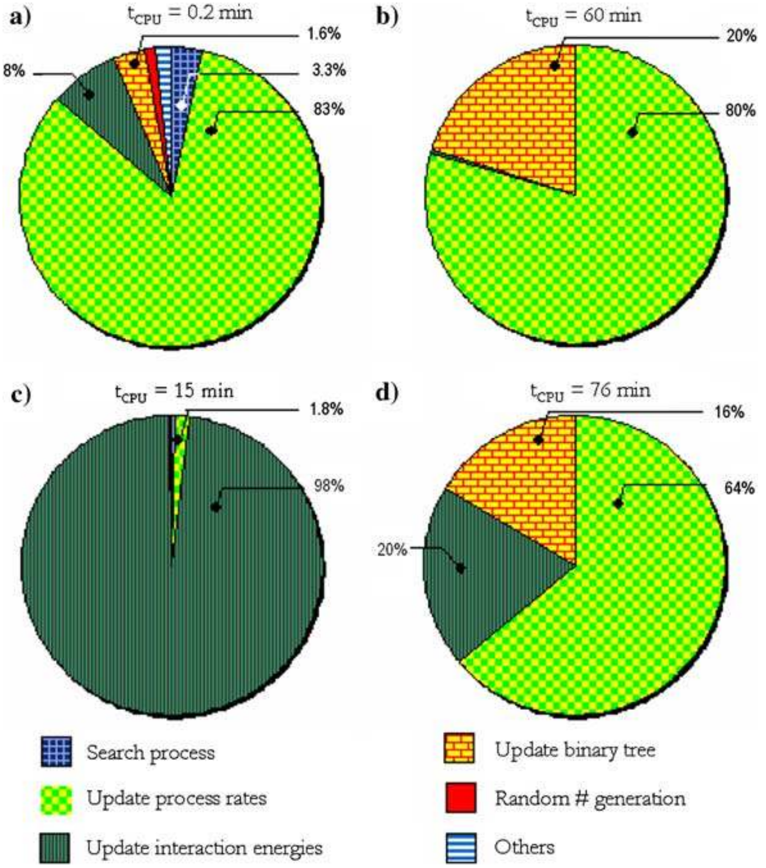


Fig. 9 Computational requirements for local and global update methods with the binary search method (using the binary tree) and the rejection-free KMC algorithm with $c_0 = 0.005$ in Fig. 8. Local (global) update of the transition probabilities is applied in panels *a* and *c* (panels *b* and *d*). Local (global) update of the interaction energy is applied in panels *a* and *b* (panels *c* and *d*)

this case) and the large CPU requirements for computing Γ_{ij}. This is not the case with the linear search method in Fig. 10.

The CPU scaling laws mentioned earlier suggest that the ratio of CPU times between global and local update methods would be of the order of $N_{\mathrm{L}} = 10^4$. In general, these predicted ratios seem to hold well [127]. However, in certain cases the predicted ratio is an upper bound. For example, after enabling all optimizations during compiling and linking of the KMC code, smaller ratios were obtained for this example. When all microscopic rates and branches of the binary tree are updated (panel *b*), 300 times more computational time is needed in comparison to panel a. In panel *c*, most of the time is spent in updating the interaction energy. The CPU requirements of panels *c* and *d* are 75 and 380 times higher, respectively, in comparison to panel *a*. Nonetheless, despite the complexities involved in their implementation, computational savings from efficient search, and local update methods are tremendous.

Significant computational resources are required for performing a linear search in Fig. 10, and in generating uniform random deviates for failed events in the null-event

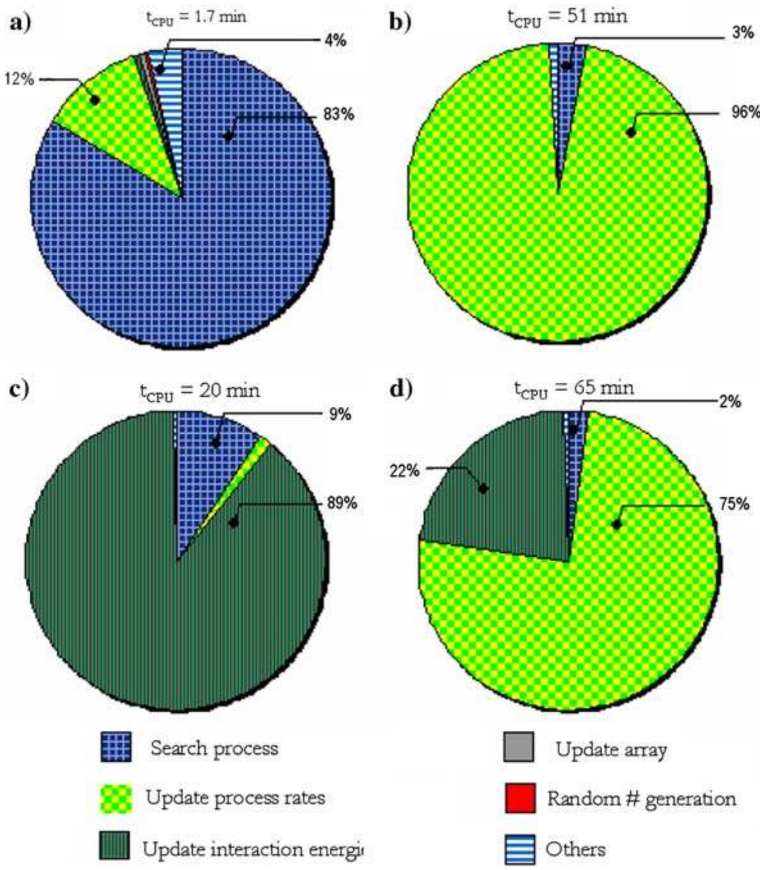


Fig. 10 Computational requirements for local and global updates using the linear search method (using an array) and the rejection-free KMC algorithm with $c_0 = 0.005$ in Fig. 8. Local (global) update for the transition probabilities is applied in panels *a* and *c* (panels *b* and *d*). Local (global) update for the interaction energy is applied in panels *a* and *b* (panels *c* and *d*)

KMC method, as shown in Fig. 11. In the case of the null-event KMC method, efficient pseudorandom number generators can reduce computational requirements [138]. Alternatively, a modified hybrid null-event KMC algorithm can be employed [131], in which only sites, where diffusion events are possible, are selected, to increase the number of successful events. Such hybrid algorithms are a compromise between the ease of implementation of the null-event method and the efficiency of 'efficient' rejection-free methods. In this example, the null-event algorithm is slightly more efficient than a rejection-free method with global search and update despite the very low coverage of defects used for these simulations. While it does not make sense to locally update the microscopic rates and not the interaction energies, these simulations re-emphasize the fact that *computational efficiency requires that all modules of an algorithm should perform local updates* (not just one of them).

The CPU requirements for local and global interaction energy updates with different interaction potential cut-off lengths, $L_{\text{potential}}$, are shown in Fig. 12. Long-ranged interactions are commonly encountered with electrostatic, steric, and elastic interac-

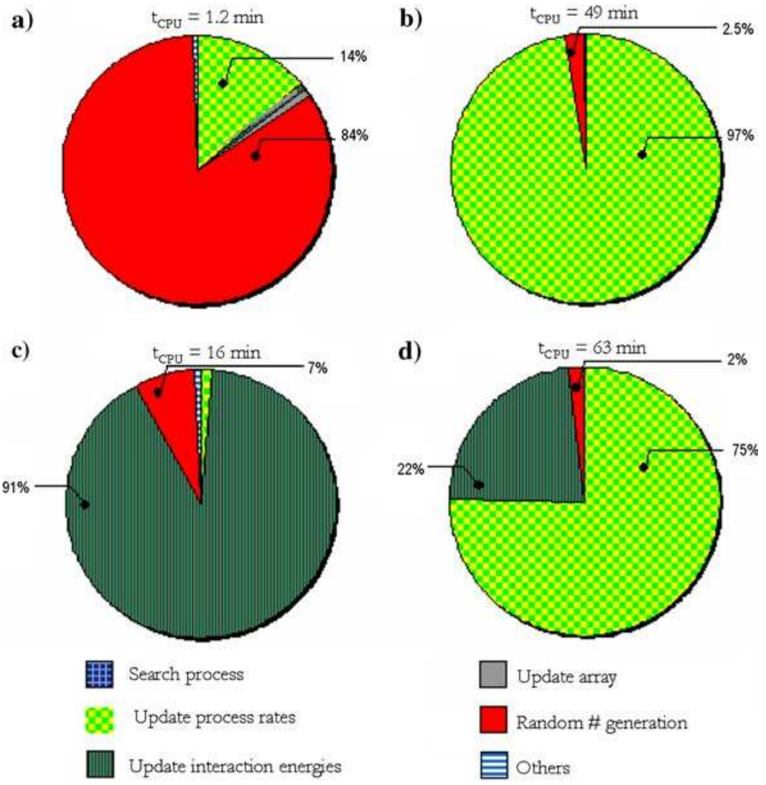
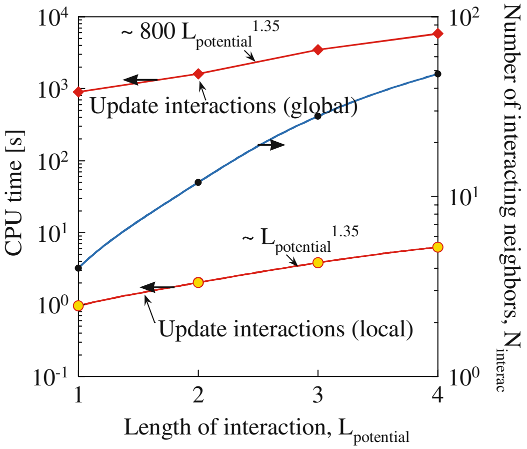


Fig. 11 Computational requirements for local and global update methods using the linear search method (using an array) in the null-event KMC algorithm with $c_0 = 0.005$ in Fig. 8. Local (global) update for the transition probabilities is applied in panels *a* and *c* (panels *b* and *d*). Local (global) update for the interaction energy is applied in panels *a* and *b* (panels *c* and *d*)

tions. As shown on the right *y*-axis of Fig. 12, the number of interacting interactions increases with increasing $L_{\text{potential}}$. For example, the number of interacting sites of each site is $N_{\text{interac}} = 4$ and 48 for $L_{\text{potential}} = 1$ and 4, respectively. At larger values of $L_{\text{potential}}$, the functional dependence is proportional to $L^2_{\text{potential}}$. Rather than identifying interacting neighbors in each KMC event (as in MD or off-lattice KMC), which is a time consuming process, a list of neighbors is employed to take advantage of the fixed lattice. This list contains information about the strength of interaction between any two chemical species at two different sites. The length of the list for a site is proportional to the number of interacting neighbors. During an update step at site *i*, all interactions are recomputed for the entire list of neighbors of *i*. It should be noted that a list of neighbors with a finer lattice representation (i.e., with a sub lattice constant spacing) and interpolation techniques could be employed for off-lattice KMC. When an array representation is employed for a periodically repeating lattice, a list of neighbors for a single site can be used as a template for all sites. The neighbors of a different site are obtained based on the relative positions between sites stored in this template. Even though this approach drastically reduces memory requirements,

Fig. 12 CPU requirements for global and local interaction energy updates (*left-handed axis*) and number of interacting neighbors (*right-handed axis*) versus potential cut-off distance, $L_{\text{potential}}$. The binary tree search method was employed for all cases



which would become necessary for very large lattices and number of processes, the computational cost increases when the actual coordinates are computed.

Figure 12 shows that the global update method is 800 times more expensive than the local update method with all compilation and linking optimizations enabled. Both global and local updates have roughly a-$L_{\text{potential}}^{1.35}$ power law dependence. As a result, the total CPU requirements for the global and local update methods also increase with $L_{\text{potential}}$. Binary tree search and local update of transition probabilities were employed for all cases in Fig. 12. Roughly 100% (10%) of the computational time is spent in updating the interactions with the global (local) update method.

The computational requirements for different algorithms discussed above are consistent with discussion of CPU scaling rules in Sect. 6, and underscore the need for efficient algorithmic implementations to access large length and time scales. Local search and updates have to be implemented in all modules of the code. Otherwise, the efficiency is poor. However, this computational speed-up is often insufficient for accessing realistic lengths and times in physical systems displaying multiscale phenomena. Such challenges, imposed by large separation of length and time scales, were discussed in Sect. 4 and Table 1. In the rest of the paper, spatial and temporal coarse-graining techniques are discussed that can overcome these challenges.

8 Spatial coarse-grained Monte Carlo (CGMC) simulations

Due to it's computational cost, the KMC method has mostly been implemented for small lattices (e.g., up to $\sim 10^3 \times 10^3$ lattice sites in 2D) with periodic boundary conditions. Recent research efforts aim at accelerating the KMC method via multiscale approaches (see Table 1). A review on several multiscale approaches has been discussed in Vlachos [36]. In this section, we discuss a particular multiscale method, called the spatial coarse-grained Monte Carlo (CGMC) method.

8.1 The concept

The CGMC method enables simulation of larger length and time scales at a reasonable computational cost by grouping lattice sites into coarse cells (spatial coarse-graining) [83,84,86]. The idea of coarse cells is reminiscent of the renormalization group (RG) theory [139]. Unlike RG, used for studying equilibrium critical phenomena, the CGMC method is derived via a coarse-graining, non-equilibrium statistical mechanics approach for the evolution of far from equilibrium as well as equilibrium systems. The CGMC method incorporates the correct microscopic physics, and thermal fluctuations [79], which are essential for correctly modeling fluctuation-driven and correlated phenomena, such as nucleation, growth, and phase transitions. Since both KMC on a microscopic lattice and a coarse lattice use the same algorithm, hereafter, the former is referred to as the *microscopic (lattice) KMC method*.

The CGMC method can be implemented using either the null-event or any rejection-free algorithm discussed above. Furthermore, the search and update methods discussed above for microscopic KMC simulation apply directly to the CGMC method as well.

A coarse-grained difference-differential equation is derived to describe the time evolution of coarse-grained variables, such as the coarse-grained system configuration, transition probabilities, and interatomic/intermolecular interactions. A hierarchy of coarse-grained models, spanning from a microscopic lattice (zero coarse graining) to a spatially homogeneous description (infinite coarse-graining) is obtained by simply controlling the degree of coarse-graining [36,140,141]. Increased coarse-graining results in reduced computational cost typically at the expense of a larger computational error.

The coarse-grained transition probabilities are derived from the underlying microscopic lattice via a systematic procedure that ensures correct asymptotic limits in the case of zero (the microscopic model) and infinite (a global mean field model) coarse-grainings. The entire coarse-grained model is expressed in terms of coarse-grained variables, namely the coarse cell size and the populations within a coarse cell.

The main assumption in CGMC is that local equilibrium is achieved within a coarse cell. The local equilibrium assumption requires fast processes within a coarse cell to relax to a quasistationary probability distribution function (pdf), and only slow processes are sampled by projecting their transition probabilities onto the quasistationary pdf of the fast processes. The quasistationary pdf can be computed exactly, using multigrid techniques, or approximately, using lower moments of the pdf (estimated via statistical mechanics-based analytical stochastic closures). The local equilibrium assumption is not as limiting as it may appear at first glance, because it is a general characteristic of dissipative particle dynamics.

The CGMC method is most suitable for problems where there is a need for KMC simulation of really large domains. Unfortunately, most analytical stochastic closures are not exact and they result in loss of information and error in solution. Non-uniform coarse lattice generation (adaptive CGMC method), based on error estimates from information theory, and a hierarchy of stochastic closures (all the way up to using multigrid methods) can be employed to overcome the issue of accuracy (see below). Next we summarize the reasons rationalizing the acceleration of the CGMC method, followed by an elaboration of the aforementioned concepts.

8.2 Speedup resulting from the CGMC method

The CGMC method overcomes several of the aforementioned challenges in KMC simulation (see Sect. 4 and Table 1). Acceleration is achieved for the following reasons:

(1) Computation of the quasi-stationary pdf of the fast processes and projection of the transition probabilities of the slow (rare) events on the pdf bypasses the serious problem of stiffness (co-existence of large and low barriers) in real systems. CPU advantages can be tremendous (proportional to the separation of time scales). This approach is reminiscent of the computational singular perturbation (CSP) and low-dimensional manifold (LDM) methods that have been used very successfully in deterministic models.

(2) Spatial coarse-graining first reduces the total number of processes (because of having fewer coarse cells, m) from the microscopic limit of $N_L{}^*N_p$ to the coarse limit of m^*N_p. This immediately reduces computational cost linearly, when global search and update methods are employed. Second, spatial coarse-graining reduces the range of coarse interaction potentials. For example, when $L_{\text{potential}} = 5$ and five lattice sites are grouped uniformly for a 1D lattice, the number of interacting neighbors per microscopic site and coarse cell are given by 10 and 2, respectively. This aspect reduces the computational time for computing interactions and for search and update steps and enables simulation of large length and time scales.

(3) Spatial coarse-graining results in additional speed-up for surface diffusion. Specifically, when an atom/molecule jumps from one coarse cell to a different coarse cell in a single step, a coarse time increment is taken (proportional to the square of the cell-to-cell centers distance) compared to a microscopic lattice jump.

(4) Finally, the one process-at-a time limitation of the KMC method is overcome by coupling the CGMC method with the τ-leap method (temporal acceleration), as discussed in Sect. 9.1.

8.3 Coarse-grained variables and coarse-grained master equation

This section first introduces the notation and discusses the idea of separation of scales in a somewhat abstract manner. The first step in CGMC is to group q_k lattice sites to form m coarse cells, denoted as $C_k, k = 1, \ldots, m$. Such a coarse lattice is shown in Figs. 13 and 14 for the 2D case. Here, $q_k \geqslant 1$ is an integer that determines the degree of coarse-graining. Site conservation requires that $\sum_{k=1}^{m} q_k = N_L$, and m is also an integer. So far only rectangular/cuboid coarse cell tessellations have been employed because of the relative ease in deriving the corresponding coarse master equation and transition probabilities [83,84,86]. For uniform coarse-graining, site conservation requires that the coarse lattice consists of $m = N_L/q$ uniformly sized coarse cells of size q.

The vector of time-dependent coarse-grained occupancies of all cells, denoted as $\underline{\eta}$, is the main observable in a CGMC simulation. Each element of $\underline{\eta}$ gives the population of a particular chemical species residing on a particular lattice site type (such as top, hollow, and bridge sites) in a particular cell C_k. In the limit of no coarse-graining, $\underline{\eta} = \underline{\sigma}$ and $m = N_L$, whereas in the limit of maximum coarse-graining, $\underline{\eta}$ has a dimensionality $\dim(\underline{\eta}) = \dim(\underline{\sigma})/N_L$ and $m = 1$.

For simplicity in notation, we discuss the CGMC method for a binary system on a single-site lattice. Extension to more complex systems is straightforward [142].

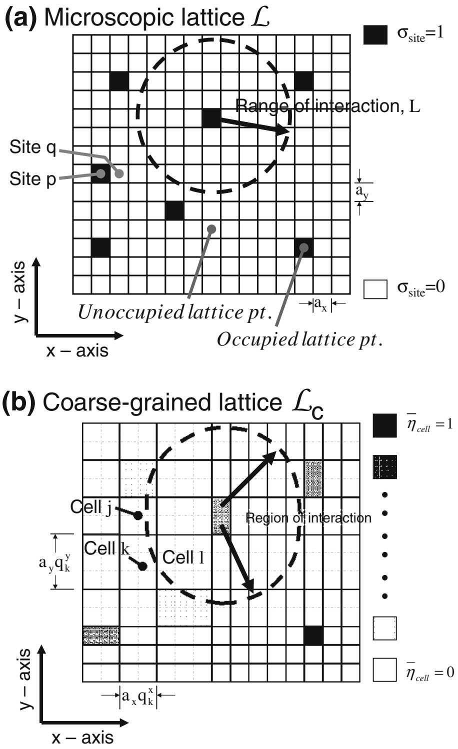


Fig. 13 Schematic of a two-dimensional (**a**) microscopic and (**b**) coarse-grained lattice. In (**b**) the microscopic lattice is shown in *dashed lines*. From Chatterjee et al. [86]

Hereafter, we focus on the submonolayer model (Sect. 2.2.3). Derivations in the CGMC method presented here are valid for any two-body interaction potential $J(r)$. Extensions to multibody interaction potentials, such as the embedded atom [143] and the Stillinger and Weber [144] potentials, are possible.

The coarse-grained occupancy in C_k, $\eta_k = \sum_{v \in C_k} \sigma_{\text{v}}$, gives the number of atoms/molecules in C_k, where $0 \leq \eta_k \leq q_k$. The cell coverage, i.e., the fraction of occupied sites in C_k, is denoted as $\bar{\eta}_k = \eta_k / q_k$ and the spatially averaged coverage over the entire lattice is denoted as $\theta = \sum_{k=1}^{m} \eta_k / N$. As shown in Fig. 13, $\bar{\eta}_k$ has quantized values from 0 to 1 with increments of $1/q_k$.

The coarse-grained difference-differential equation is derived from the microscopic difference-differential equation by exploiting the aforementioned time scale separation between microscopic processes via singular perturbation analysis. The overall approach is reminiscent of adiabatic elimination in stochastic differential equations [145]. In passing, we should note that coarse-graining of the master equation and time scale separation are very important issues and have also been discussed for well-mixed systems, e.g., [77,78,80–82,146].

Equation 2 is written separately for the fast and slow processes, which gives rise to a low-dimensional stochastic manifold. Summing up contributions for the fast processes within coarse cell C_k, $k = 1, \ldots, m$, results in m difference-differential equations given by

$$\mathrm{d} \sum_{i \in C_k} \sigma_i(\tilde{t}_f) = \mathrm{d}\tilde{t}_f \left[\sum_{j} \sum_{i \in C_k} \Gamma^{+}_{ij,f}(\underline{\sigma}) - \sum_{j} \sum_{i \in C_k} \Gamma^{-}_{ij,f}(\underline{\sigma}) \right], \quad k = 1, \ldots, m. \tag{42}$$

Here the subscript f in $\Gamma^{+}_{ij,f}$ and $\Gamma^{-}_{ij,f}$ denotes fast processes. As an example, two time scales were considered in Chatterjee and Vlachos [79], namely, adsorption and desorption processes were the slow events and surface diffusion was the fast process. Slow processes are not executed over short time scales. When a coarse cell is locally equilibrated in a short relaxation time, determined by $\Gamma^{+}_{ij,f}$ and $\Gamma^{-}_{ij,f}$, the local quasi-stationary pdf of states is obtained. When the system is locally relaxed, one has

$$\sum_{j}\left\langle\sum_{i\in C_k}\Gamma^{+}_{ij,f}(\underline{\sigma})\right\rangle-\sum_{j}\left\langle\sum_{i\in C_k}\Gamma^{-}_{ij,f}(\underline{\sigma})\right\rangle=0,\quad k=1,\ldots,m. \tag{43}$$

The quasi-stationary pdf is independent of the initial conditions and depends only on mesoscopic constraints, such as local mass, energy, and temperature arising from the slow processes in each coarse cell. The configuration space evolves over the slow time scales of $\tilde{t}_s$ interest according to

$$\mathrm{d}\sum_{i\in C_k}\sigma_i(\tilde{t}_s)=\mathrm{d}\tilde{t}_s\left[\sum_{j}\sum_{i\in C_k}\Gamma^{+}_{ij,s}(\underline{\sigma})-\sum_{j}\sum_{i\in C_k}\Gamma^{-}_{ij,s}(\underline{\sigma})\right],\quad k=1,\ldots,m. \tag{44}$$

The CGMC solves Eq. 44. Spatial correlations, activation barriers, etc., required in Eq. 44, are determined only after the coarse cell has locally been equilibrated according to Eq. 43.

Equation 43 makes evident of a theoretical difficulty, namely, how to connect the ensemble averaged coarse-grained transition probability, given by $\bar{\Gamma}_j(k)=\left\langle\sum_{i\in C_k}\Gamma_{ij}\right\rangle$, for correlated microscopic processes in terms of the coarse variable η_k. Note that overbars used in this paper denote the corresponding coarse-grained variables depicted in brackets. Assuming constant frequencies, the coarse-grained adsorption transition probability for C_k is given by

$$\bar{\Gamma}_{\mathrm{a}}(k)=\left\langle\sum_{i\in C_k}\Gamma_{i\mathrm{a}}\right\rangle=\sum_{i\in C_k}\nu_{\mathrm{a}}(1-\langle\sigma_i\rangle)=\nu_{\mathrm{a}}(q_k-\eta_k),\quad k=1,\ldots,m \tag{45}$$

and the coarse-grained desorption transition probability is given by

$$\bar{\Gamma}_{\mathrm{d}}(k)=\left\langle\sum_{i\in C_k}\Gamma_{i\mathrm{d}}\right\rangle=\nu_{\mathrm{d}}\left\langle\sigma_i\mathrm{e}^{-U_{\mathrm{binding}}(i)/kT}\right\rangle,\quad k=1,\ldots,m. \tag{46}$$

Spatial correlations on the microscopic lattice make the connection between $\left\langle\sigma_i\mathrm{e}^{-U_{\mathrm{binding}}(i)/kT}\right\rangle$ and the coarse variable (constraints) η_k difficult.

Likewise, spatial correlations determine the coarse Hamiltonian given by

$$\bar{H}(\underline{\sigma})=\sum_{k=1}^{m}\left(\bar{H}_1(k)+\bar{H}_2(k)+\bar{H}_{\mathrm{ext}}(k)\right), \tag{47}$$

where

$$\bar{H}_1(k)=-\sum_{i\in C_k}\sum_{\substack{i'\in C_k\\ i'\neq i}}\left\langle J(|i-i'|)\sigma_i\sigma_{i'}\right\rangle \tag{48}$$

is the contribution from interactions between C_k and other cells C_j ,

$$\bar{H}_2(k) = -\sum_{i \in C_k} \sum_{\substack{i' \in C_{k'} \\ k' \neq k}} \left\langle J(|i - i'|)\sigma_i \sigma_{i'} \right\rangle \tag{49}$$

is the contribution from interactions within C_k, and

$$\bar{H}_{\text{ext}}(k) = \bar{h}\eta_k \tag{50}$$

gives the contribution of the coarse external field $\bar{h}$.

8.4 Analytical stochastic closures

Analytical expression for the exact quasistationary pdf with nearest-neighbor interactions is known for the 1D lattice for certain ensembles from statistical mechanics [12]. However, exact pdfs for 2D and 3D lattices are not available for the general case. Either approximate analytical stochastic closures, such as the local mean-field (MF), quasi-chemical (QC), and cluster expansions, are employed in place of Eq. 43, or Eq. 43 is solved numerically via multigrid techniques to obtain the quasi-stationary microscopic pdf. These different approaches are discussed next.

8.4.1 Local mean-field (MF) approximation

In the local MF approximation, a uniform distribution of particles is assumed and any local correlations between particles inside a coarse cell are disregarded, i.e., $\langle \xi_1 \xi_2 \rangle \approx \langle \xi_1 \rangle \langle \xi_2 \rangle$. Here ξ_1 and ξ_2 are two independent random variables. The intra- and inter-cell terms of the coarse-grained Hamiltonian derived using the local MF approximation is given in Katsoulakis and Vlachos, Chatterjee and Vlachos, Katsoulakis et al., and Chatterjee et al. [84–87]

$$\bar{H}_1(k) = -\frac{1}{2}\sum_{k=1}^{m} \bar{J}_{kk}\eta_k(\eta_k - 1) \tag{51}$$

and

$$\bar{H}_2(k) = -\frac{1}{2}\sum_{k=1}^{m} \sum_{\substack{k' = 1 \\ k' \neq k}}^{m} \bar{J}_{kk'}\eta_k\eta_{k'}, \tag{52}$$

respectively. Here

$$\bar{J}_{kk} = \frac{\sum_{i \in C_k} \sum_{\substack{i' \in C_k \\ i' \neq i}} J(|i - i'|)}{q_k(q_k - 1)} \tag{53}$$

is the coarse-grained potential within cell C_k and

$$\bar{J}_{kk'} = \frac{\sum_{i \in C_k} \sum_{\substack{i' \in C_{k'} \\ k' \neq k}} J(|i - i'|)}{q_k(q_k - 1)} \tag{54}$$

is the coarse-grained potential between cells C_k and $C_{k'}$ obtained using Haar-wavelets. The coarse-grained potentials are real-valued constants evaluated at the beginning of a CGMC simulation for a given mesh. The term $q_k(q_k - 1)$ results from the lack of self-interaction term due to a hard-potential at zero separation (exclusion principle).

The coarse-grained interaction energy between adsorbed sites, using the local MF approximation, is given by

$$U_{\text{binding}}(k) = \bar{J}_{kk}(\eta_k - 1) + \sum_{\substack{k' \\ k' \neq k}} \bar{J}_{kk'}\eta_{k'}. \tag{55}$$

Neglecting the spatial correlations in Eq. 46 one gets

$$\Gamma_{\text{d}}(k) = \sum_{\text{v}} \nu_{\text{d}} \left\langle \sigma_{\text{v}} \mathrm{e}^{-U_{\text{binding}}(\nu)/kT} \right\rangle \approx \nu_{\text{d}}\eta_k \mathrm{e}^{-\bar{U}_{\text{binding}}/kT}. \tag{56}$$

As before it is assumed that desorption obeys Arrhenius dynamics. Note that by using the MF closure, $\bar{\Gamma}_{\text{d}}$ and $\bar{U}_{\text{binding}}$ can directly be expressed in terms of the coarse variable η_k (compare Eqs. 56 and 51 with Eqs. 46 and 47, respectively).

Numerical examples have shown that the CGMC method with the local MF approximation is reasonably accurate and extremely computationally efficient [84–87]. Scaling laws for CPU were derived in terms of the degree of uniform spatial coarse-graining, q [84,85]. For instance, it was found that the CPU requirements decrease as q^3 for short-ranged potentials and q^4 for long-ranged potentials for diffusion (canonical ensemble) with linear search and global updates. Even with a modest coarse-graining of q = 3 × 3, these scaling laws imply that the CPU requirements of CGMC decrease by a factor of up to 6,500 in comparison to the microscopic KMC method. CPU requirements for diffusion on a 2D lattice using binary search and local updates are discussed for the first time later in this section. Large CPU savings with high accuracy can be expected with non-uniform coarse lattices discussed later.

8.4.2 Local quasi-chemical (QC) approximation

The QC approximation (or the equivalent Bethe–Peierls approximation) is employed within each cell C_k (termed as local QC theory) when nearest-neighbor interactions are present and cell boundaries can be neglected. In the local QC approximation, the partition function for the coarse cell C_k is given by Hill [12]

$$Q(q_k, \eta_k, T) = Q_s{}^{\eta_k} Q_{NN}, \tag{57}$$

where $Q_s = \mathrm{e}^{-h/kT}$ is the partition function of a single adsorbed particle on the lattice,

$$Q_{NN} = \left[\frac{q_k!}{\eta_k!(q_k - \eta_k)!}\right]^{1-Z} \frac{(Zq_k/2)!}{(Z\eta_k/2 - N_{01}^{NN}/2)!(N_{01}^{NN}/2)!^2(Z(q_k - \eta_k)/2 - N_{01}^{NN}/2)!} \tag{58}$$

is the configurational contribution to the partition function Q, N_{01}^{NN} (N_{11}^{NN}) is the number of pairs with one site (both sites) occupied in C_k and Z is the coordination number of a 2D square lattice. Ultimately, the spatial correlations of nearest-neighbors are obtained in terms of N_{01}^{NN} and N_{11}^{NN}.

Neglecting boundary effects ($q_k \to \infty$), the coarse-grained Hamiltonian is given by Chatterjee and Vlachos [79]

$$\bar{H}(\eta) = \sum_{k=1}^{m} (\bar{H}_2(k) + \bar{h}\eta_k) = -\sum_{k=1}^{m} \frac{Zq_k J_{NN}}{2} \left[\bar{\eta}_k - \frac{2\bar{\eta}_k(1-\bar{\eta}_k)}{\zeta_{NN}+1} \right] + \sum_{k=1}^{m} \bar{h}\eta_k. \quad (59)$$

Here J_{NN} is the isotropic NN interaction potential between two neighboring adsorbed particles. Likewise, the coarse-grained transition probability for desorption is given by Chatterjee and Vlachos [79]

$$\bar{\Gamma}_{\mathrm{d}}(k) = \left\langle \sum_{\mathrm{v} \in C_k} \Gamma_{\mathrm{d}}(v) \right\rangle = 0, \quad \text{when } \bar{\eta}_k = 0, \quad (60)$$

$$\bar{\Gamma}_{\mathrm{d}}(k) \approx \nu_{\mathrm{d}} q_k \mathrm{e}^{-J_{NN}Z/kT}, \quad \text{when } \bar{\eta}_k = 1 \quad (61)$$

and

$$\bar{\Gamma}_{\mathrm{d}}(k) \approx \nu_{\mathrm{d}} \eta_k \left[\frac{(\zeta_{NN} - 1 + 2\bar{\eta}_k)(1-\bar{\eta}_k)}{(\zeta_{NN} + 1 - 2\bar{\eta}_k)\bar{\eta}_k} \right]^{Z/2} \mathrm{e}^{-J_{NN}Z/2kT}, \quad \text{when } 0 < \bar{\eta}_k < 1. \quad (62)$$

8.4.3 Advantages and limitations of analytical closures

An advantage of the local MF and QC approximations is that analytical expressions for the coarse energetics and transition probabilities are available in terms of the coarse observable η_k. Extension of this approach to diffusion on adaptive lattices is not as straightforward as suggested by Eqs. 42–44 (for uniform meshes this is not as difficult a task). As evident from macroscopic scaling laws, such as the Stokes–Einstein's law, incorporation of the correct length scales in the transition probability for diffusion between coarse cells is important to obtain the correct time scales. In Katsoulakis and Vlachos, and Chatterjee et al. [84,86], a systematic approach was introduced to derive the diffusion transition probability on arbitrary complex lattices.

The local MF approximation is exact in the limit of infinitely high temperatures, and/or zero interaction strength and/or infinitely long interaction potential. As the range of interactions or lattice dimensionality increase, the accuracy of the local MF closure improves. Thus, fairly accurate simulations should be expected for 3D systems even with short-ranged interactions. Similarly, the local QC approximation is a good approximation (but not exact) when the potential entails nearest-neighbor interactions.

Detailed discussion on information loss associated with the local MF approximation is given in Sect. 8.6. More sophisticated approximations, such as cluster expansions [12,147, M. Katsoulakis et al. submitted], can also be employed to improve the accuracy. However, the accuracy usually improves slowly with several moments of the pdf at the cost of increasing mathematical complexity and computational requirements. In contrast to traditional cluster expansions [12], the expansion in M. Katsoulakis et al. (submitted) is not done around the high or the low-temperature limits but around the CG approximations, which are already very good approximations. Hence just one correction term gives good predictions even for nearest neighbors. Thus, further developments in this type of expansions are most fruitful.

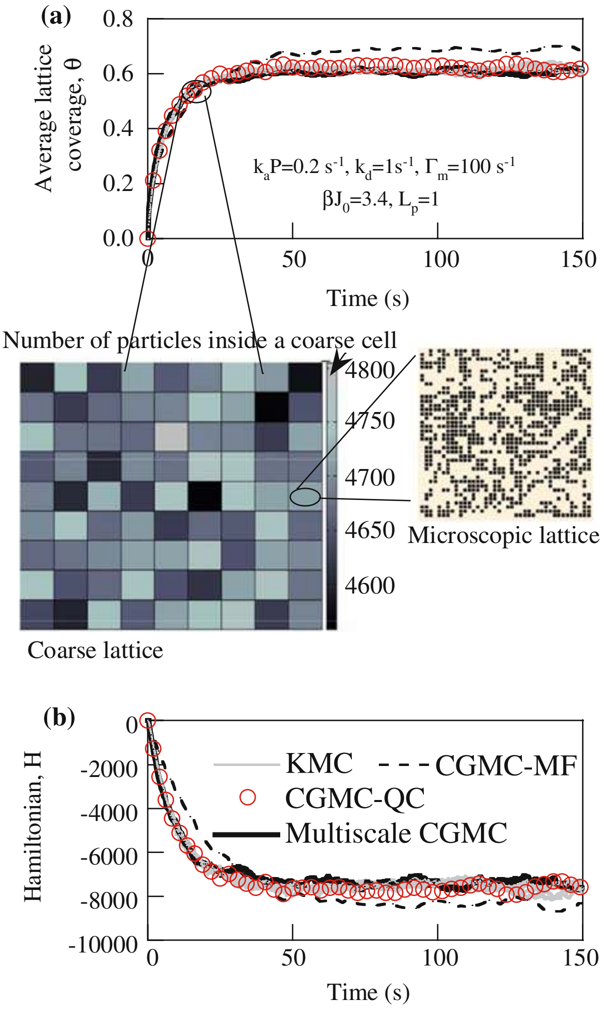


Fig. 14 Comparison of (**a**) time-dependent average lattice coverage and the (**b**) Hamiltonian of various methods for simultaneous adsorption, desorption, and surface diffusion. Initially, the lattice has no adsorbed atoms. Local equilibrium within a coarse cell is attained via fast diffusion processes. A snapshot of the coarse lattice ($m = 10 \times 10$ coarse cells) and the microscopic lattice ($q_m = 40 \times 40$ lattice sites) from the CGMC simulation with a two-level grid at time $t = 10$ s is also shown in the middle. KMC = microscopic simulation, CGMC-MF = spatially coarse-grained MC (*CGMC*) with local mean-field approximation, CGMC-QC = CGMC with local quasi-chemical approximation, multiscale CGMC = CGMC on two grids with stochastic closure to explicitly handle separation of time scales. From Chatterjee and Vlachos [79]

8.5 Multigrid methods

In Chatterjee and Vlachos [79], a multigrid numerical scheme was employed to generate the quasi-stationary pdf. In this method, two grids were used. A coarse lattice is employed to numerically solve Eq. 44 over large length and time scales. A small microscopic lattice of size q_m is also embedded within each coarse cell C_k to solve the master equation for fast processes only over short time scales (see Figs. 13 and 14), i.e., until quasi-steady state (Eq. 43 in this example) is achieved. The microscopic KMC simulation uses the coarse variable η_k as a constraint. In turn, the transition probabilities of slow processes on the coarse grid over long times are projected on the pdf of the fine grid KMC simulation.

Significant acceleration results when $q \gg q_m$ for two reasons. First, one computes the correct pdf of fast modes, via the microscopic KMC method, using only a small fraction q_m/q of the entire domain. Second, additional acceleration results from the time scale separation between simulations on the two grids. Specifically, one evolves the system over coarse (slow events) time scales, using correct transition probabilities obtained from the underlying pdfs determined via brief KMC simulations on short time scales (the duration of KMC simulation is such that quasi-steady state is established; statistical methods to ensure this are given in Chatterjee and Vlachos, and Samant and Vlachos [79,81]). Further acceleration can result by maintaining the same pdf for a period of time and by firing multiple processes at once via the τ-leap CGMC method discussed later [91].

An important advantage of the multigrid method is that it has no error, within numerical accuracy, as long as the local equilibrium assumption is valid. In other words, the method is exact for any range potentials and any temperature (as long as the assumptions of separation of length and time scales are valid). Furthermore, it is relatively easy to implement.

8.5.1 A priori computation of pdfs using look-up tables or neural networks

Instead of employing on-the-fly approaches for evaluating spatial correlations, one can generate a priori a database of pdfs using microscopic lattice KMC simulations for various atomic/molecular configurations. However, this may become tedious for complex systems and may require interpolation and/or tabulation, when a large number of configurations are present. In such cases, more sophisticated fitting methods, such as a neural network [148], can be trained to predict the quasi-stationary pdf with macroscopic thermodynamic constraints, such as mass and temperature, as inputs.

The large acceleration obtained with the neural network is accompanied with difficulties in determining input sets during training, some initial training time, complexity of implementation of the method, and loss of accuracy due to insufficient sampling during training. All of these issues are, though, surmountable given good implementation.

8.6 Adaptive coarse lattice generation via error estimates based on information theory

It has been found through simulation and Large Deviation theory [83–87] that the local MF approximation is accurate in the limit of infinitely long ranged interactions, i.e., $L_{\mathrm{p}} \to \infty$ and/or zero interactions and/or infinite high temperatures, i.e., $|\beta J_0| \ll 1$. In practice, the local MF approximation can be reasonably accurate for L_{p}=3 in 2D and at moderate temperatures, e.g., $|\beta J_0| < 1$ [84]. For strong, short ranged interactions, and/or low temperatures, the local MF assumption results in information loss of the atomic positions (local spatial correlations are absent).

A posteriori error estimates (upper bounds which are based on the solution on the coarse lattice) in a CGMC simulation with the local MF assumption, resulting from information loss due to spatial coarse-graining, were developed using information theory methods [85,87,88,149]. The error bounds were expressed in terms of the level of local coarse-graining, the interaction potential, and the local coverage. The relative entropy difference between the microscopic and coarse-grained Gibbs measures is [88]

$$\int \log\left(\frac{\mathrm{d}\bar{\mu}}{\mathrm{d}\mu}\right)\mathrm{d}\bar{\mu} = O\left(\frac{N_{\mathrm{L}}q}{L_{\mathrm{potential}}}\right). \tag{63}$$

Here, μ is the Gibbs measure for the microscopic lattice. Using Eq. 63 for a non-uniform lattice, an estimate of the error in computing the Hamiltonian (the difference between the exact microscopic and the coarse-grained Hamiltonians) is given by Chatterjee et al. [88]

$$\mathrm{ER} = \sum_{k=1}^{m} \mathrm{ER}_k \tag{64}$$

as a sum of error contributions from all cells, where

$$\begin{aligned}\mathrm{ER}_k = {}& 4\frac{j_{kk}}{q_k(q_k-1)}\left\langle \eta_k(q_k-\eta_k)\left[\eta_k(\eta_k-1)+(q_k-\eta_k)(q_k-\eta_k+1)\right]\right\rangle \\ &+4\sum_{\substack{k' \\ k'\neq k}} \frac{j_{kk'}}{q_k q_{k'}}\left\langle q_k^2\eta_k(q_k-\eta_k) - 2\eta_k\eta_{k'}(q_k-\eta_k)(q_{k'}-\eta_{k'})\right\rangle .\end{aligned} \tag{65}$$

Here $j_{kk'} = \max |J(i-i') - J(i''-i''')|$ is the maximum difference in the interactions between lattice points of C_k and $C_{k'}$, $i, i' \in C_k$ and $i'', i''' \in C_{k'}$. It was shown in Chatterjee et al. [88] via numerical analysis that Eq. 65 accurately captures the functional trends of actual numerical errors, so it could be used as a guide of designing adaptive lattices.

In the absence of gradients, Eq. 65 is nearly symmetric about the coverage of 0.5 implying that the error for a given cell size is largest around a coverage-value of one-half. As a result, one should refine the mesh at interfaces. Equation 65 guides selection of the cell size around the interface using mesh equidistribution or mesh insertion techniques [87,88]. This approach enables design of optimal adaptive meshes resulting in large accuracy and speed-up. Since $\eta_k \approx O(q_k)$ for large q, it is intuitively expected that $\varepsilon_k \approx O({q_k}^2)$ but this is only an approximate limit. This estimate gives an idea that by doubling the mesh points, one could approximately improve accuracy by a factor of 4 (in reality it is less).

8.7 Numerical examples

Figures 14 and 15 compare CGMC simulations using the local MF, QC and multigrid approach for strong, short ranged interaction ($L_{\mathrm{potential}} = 1$) with simultaneous adsorption, desorption, and diffusion. As expected, the local MF approximation exhibits errors; the local QC approximation is reasonable but not as accurate; finally, the multigrid approach, denoted as multiscale CGMC method in the graph, matches the microscopic KMC solution. Another advantage of the multigrid method is its inherent generality in terms of the number and type of processes, lattices, etc. that can be treated. In fact, as illustrated in Chatterjee and Vlachos [79], one may not even have to derive the coarse-grained transition probabilities in connecting the two (microscopic and coarse) KMC simulators.

As an example of mesh generation guided by Eq. 65, an adaptive coarse lattice was generated for a standing wave connecting the dilute and dense phases at equilibrium in Chatterjee et al. [88] for the case of purely attractive interactions. The isotherm is multivalued when attractive interactions are strong or temperatures are low (specifically, when $J_0/kT > 4$). When the two phases co-exist, a standing wave connects the dense and the dilute phases. A 1D lattice with N_L =16,384 microscopic sites was used for the standing wave calculation with $J_0/kT = 12$ and $L_{\mathrm{potential}} = 128$.

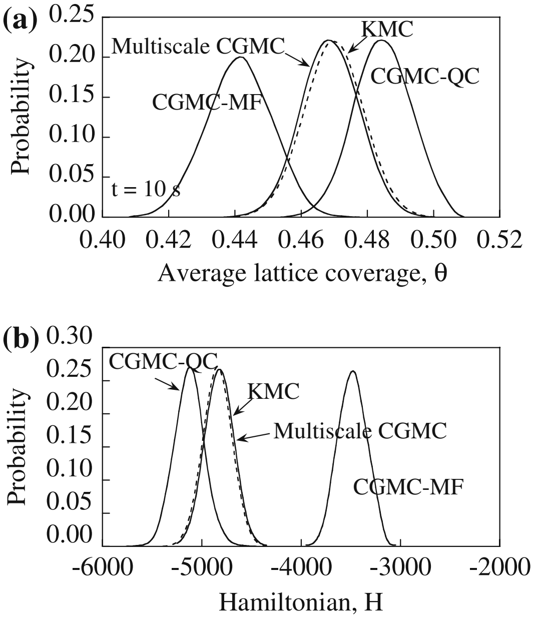


Fig. 15 Probability distribution function (*pdf*) for (**a**) average lattice coverage and (**b**) Hamiltonian at time $t = 10\,\text{s}$ from different methods (acronyms per Fig. 14) for the simulation depicted in Fig. 14. From Chatterjee and Vlachos [79]

As shown in Fig. 16, the standing wave is hardly resolved with a uniform coarse lattice of $q = 64$. The resulting error is large. On other hand, the error can become sufficiently small (lower than a tolerance), by adding more or localizing lattice points around the interface.

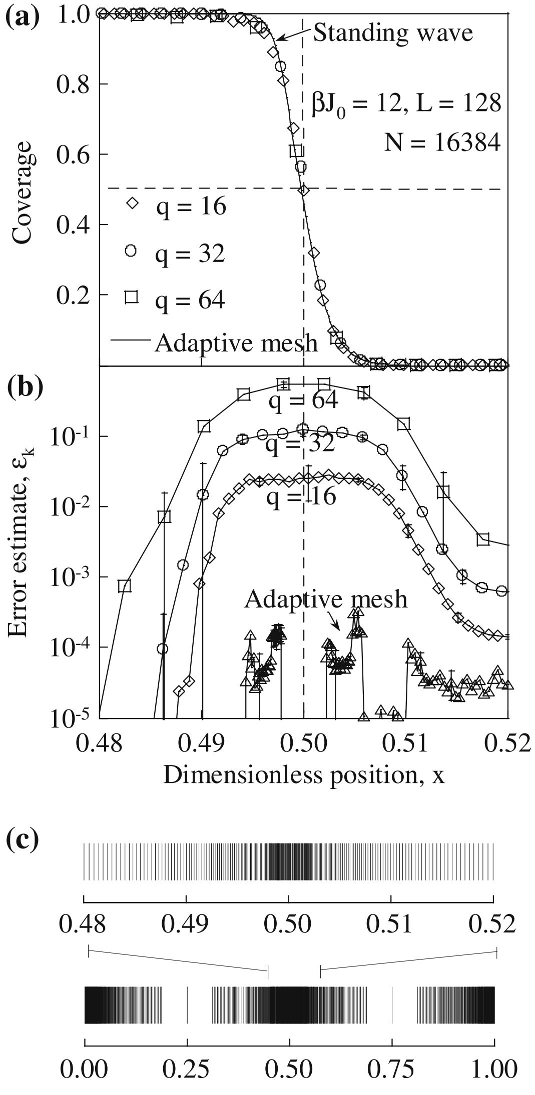
The coarse-grained diffusion transition probability, derived for a 2D (100) lattice [86], is employed to study the diffusion of atoms arranged initially as a stripe (shown at $t_{\text{real}} = 0\,\text{ps}$ in Fig. 17). In this numerical example, microscopic KMC and uniform coarse lattice CGMC simulations were performed using periodic boundary conditions, $J_0/kT = 0.8$ and $L_{\text{potential}} = 3$. Two different uniform coarse-grained lattices, namely, $q = 2 \times 2$ and 4×4 were employed. The microscopic and coarse lattices contain $N_L = 300 \times 300$ microscopic lattice sites.

As time proceeds, atoms diffuse along the x-axis, until the lattice is uniformly covered. As evident from Fig. 17, the coverage profiles obtained by averaging along the y-axis from the CGMC simulations are in excellent agreement with those of the KMC method. The CPU requirements for the microscopic ($q = 1 \times 1$) and coarse ($q = 2 \times 2$ and $q = 4 \times 4$) lattices are given by 36:14:1, respectively. The q^3 CPU scaling law, mentioned above, is not valid here since binary search and local updates were employed for both the microscopic KMC and the CGMC simulations. However, substantial savings are still observed even for moderate levels of coarse-graining.

9 Temporal acceleration (multifiring) methods

The CGMC method still follows the one process-at-a time approach of the microscopic KMC method. While time acceleration is still achieved, as mentioned above, further acceleration can be obtained via temporal coarse graining. This can be achieved by

Fig. 16 (**a**) Standing wave at equilibrium in the presence of strong attractive interactions (or low temperatures) for three uniform meshes and an adaptive mesh. (**b**) Error estimates of CGMC solution for three uniform meshes. Here $N_L = 16,384, J_0/kT = 12$ and $L_{\text{potential}} = 128$. (**c**) Adaptive mesh used. From Chatterjee et al. [88]



executing (firing) multiple processes at once using one of the τ-leap (multifiring) methods.

The τ-leap method is an approximate stochastic simulation method, which was introduced by Gillespie [92] for modeling reaction networks in spatially uniform (well-mixed) systems. There are several variations of the original method, based on different distributions (Poisson versus binomial) and explicit versus implicit schemes [93–97]. Numerical simulations for simple reaction networks and complex biological networks have demonstrated that the method accurately captures the probability density function (pdf) of the time-dependent species populations for small τ. A central assumption in the τ-leap method, termed as the *leap condition* [92], requires τ to be sufficiently small so that the change in the population for all chemical species is small.

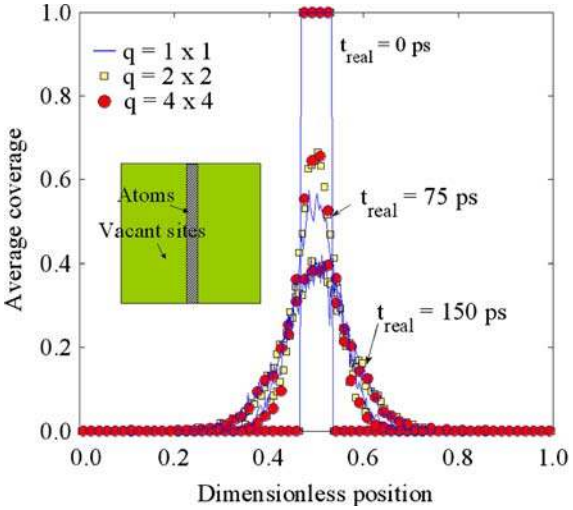


Fig. 17 Comparison of the time-dependent coverage profiles from microscopic (q = 1 × 1) and two CGMC (q = 2 × 2 and 4 × 4) surface diffusion simulations. The inset depicts the initial conditions. Initially atoms are present only in the shaded stripe, and diffuse to vacant sites at $t > 0$. The lattice contains $N_{\mathrm{L}} = 300 \times 300$ microscopic sites

9.1 τ-leaping on coarse-grained lattices

The τ-leap methods cannot directly be implemented on a microscopic lattice, since any surface process would result in a population change that violates the leap condition. For example, desorption of an atom from a site changes the occupation function from 1 to 0. The extension of the τ-leap idea to spatial KMC simulation was introduced in Chatterjee and Vlachos [91]. Specifically, the temporally adaptive coarse-grained Monte Carlo (τ-leap CGMC) method combines coarse-graining in both space and time to study the evolution of the coarse-grained occupation state vector $\underline{\eta}$.

The main idea is fairly simple. When the cell sizes $q_k, k = 1, \ldots, m$, of a coarse lattice are sufficiently large, there are enough atoms/molecules within coarse cells that the τ-leap method can be applied without violating the leap condition. Instead of executing one process at a time, a 'bundle' of processes are then executed in all cells during a time interval $[t, t + \tau)$ and the time is advanced by a coarse amount τ.

So far the explicit Poisson and binomial τ-leap methods have been employed in the CGMC method. In the explicit Poisson τ-leap method, the jth process bundle size at cell C_k, γ_{ij}, i.e., the number of times the jth process is executed during τ, is sampled from the Poisson distribution

$$P_{\mathrm{PD}}(\gamma_{ij}; \Gamma_{ij}\tau) = \frac{\mathrm{e}^{-\Gamma_{ij}\tau}}{\gamma_{ij}!}(\Gamma_{ij}\tau)^{\gamma_{ij}}. \tag{66}$$

The average bundle size, based on Eq. 66, is given by $\Gamma_{ij}\tau$. However, since a Poisson distribution random variable is unbounded, negative populations are encountered [95, 150], when γ_{ij} exceeds the available population size within a cell. The explicit binomial τ-leap method overcomes this issue by sampling γ_{ij} from the bounded binomial distribution

$$P_{\mathrm{BD}}(\gamma_{ij}; p_j, \gamma_{\max}^{ij}) = \frac{\gamma_{\max}^{ij}!}{\gamma_{ij}!(\gamma_{\max}^{ij} - \gamma_{ij})!} p_j^{\gamma_{ij}} (1 - p_j)^{\gamma_{\max}^{ij} - \gamma_{ij}}. \tag{67}$$

Here $\gamma_{\max}^{ij}$ is the limiting reactant population size for each reaction, i.e., the number of times this process can be executed in a coarse cell, and $p_j = \min(\Gamma_{ij}\tau/\gamma_{\max}^{ij}, 1)$ is the probability associated with a successful event among $\gamma_{\max}^{ij}$ possible events. The average process bundle size, based on Eq. 67, is given by $\Gamma_{ij}\tau$. Once γ_{ij} for all processes are determined, using Eq. 66 or 67, the populations $\eta_k(t)$ are updated and the time is incremented to $t + \tau$. In order to ensure positive number of particles, the constrained τ-leap method is recommended [95].

Various approaches have been employed to choose the time increment τ [92,95,98]. Our preferred method uses the r-criterion [95] such that

$$\tau = r \min(\eta_k / \sum_{i,j} \Gamma_{ij}). \tag{68}$$

Here $0 < r < 1$ is a user specified temporal coarse-graining factor. Equation 68 is valid when only adsorbed species are consumed, for e.g., when only desorption processes are present. A more general expression for τ in terms of r is given in Chatterjee and Vlachos, and Chatterjee et al. [91,95]. The derivation of Eq. 68 is in essence similar to the stability analysis of the forward Euler method. It was shown in Chatterjee et al. [95] that the value of $r \sim 1$ is closely related to the onset of numerical instabilities in the algorithm. Values of $r \sim 0.1$ give good results for several problems we have studied.

In summary, accurate solutions can be obtained with simultaneous spatial and temporal coarse-graining by employing accurate stochastic closures in CGMC for space and by selecting τ such that the leap criterion, e.g., Eq. 68, is not violated.

9.1.1 CPU comparison for an adsorption-desorption example

CPU scaling laws in terms of the coarse cell size and length of interaction potential can be derived. For example, it was shown that in the case of adsorption and desorption on a uniform coarse lattice, the relative CPUs of CGMC (without temporal acceleration) and τ-leap CGMC are [91]

$$\frac{t_{\text{CPU,CGMC}}}{t_{\text{CPU},\tau-\text{CGMC}}} = r N_{\text{L}} \phi \frac{(7 + 2L_{\text{potential}}/q)\hat{t}_m + 2\hat{t}_{\exp} + 2\hat{t}_{\text{rand}}/m}{(15 + 2L_{\text{potential}}/q)\hat{t}_m + 2\hat{t}_{\exp} + 2\hat{t}_{\text{rand}}/m}, \tag{69}$$

where ϕ is a constant, $\phi = \theta$ when $c_0 < 1/2$, and $\phi = 1 - \theta$ when $c_0 > 1/2$. Since the CPU of the CGMC method is fixed, the CPU of the τ-leap CGMC decreases linearly with increasing the temporal coarse-graining factor r and with the lattice size. Similar relations have been found for other ensembles.

Figure 18 compares the CPU of the CGMC method with and without temporal acceleration. Note that since the generation of Poisson and binomial random numbers is more expensive that that of uniform random deviates, the τ-leap CGMC method becomes more expensive than the CGMC method when the bundle size is small, as shown in Fig. 18. However, for large domains, where substantial spatial coarse-graining is possible, bundle sizes can be large and the CPU savings can be substantial, especially as the time step becomes larger (large r in Fig. 18).

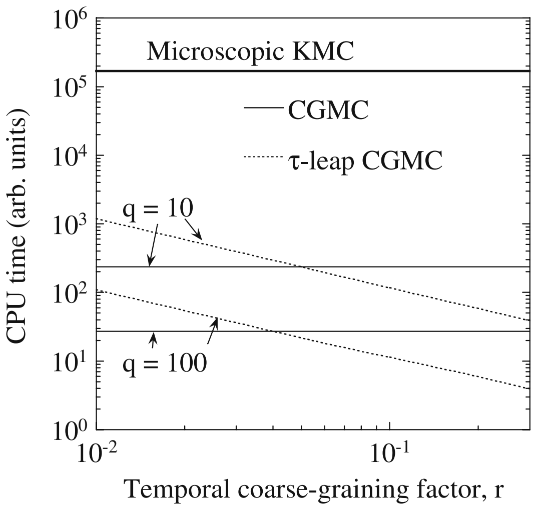


Fig. 18 Computational requirements of the CGMC (without temporal coarse-graining) and the τ-leap CGMC methods for an adsorption–desorption simulation. The parameters are $\nu_a/\nu_d = 0.4$, $J_0/kT = 2$, $L_{\text{potential}} = 30$ and $N_L = 20{,}000$. Uniform lattices of size $q = 10$ and $q = 100$ are employed for the CGMC simulations. The microscopic KMC simulation corresponds to $q = 1$. The r-criterion is used for temporal coarse-graining (see text). From Chatterjee and Vlachos [91]

9.2 τ-leaping on microscopic lattices

The above discussion points to the fact that use of multifiring is straightforward upon spatial coarse-graining but raises the question of whether temporal acceleration is ever possible if coarse-graining in space is not desirable or done. For example, when spatial inhomogeneities span only a few nanometers, a microscopic lattice KMC simulation is the preferred method. The answer to the above question is affirmative. By grouping sites with the same transition probability into the same group or class, i.e., using the n-fold method, a fairly large number of sites in each class are typically present. Each class becomes then a pseudo process (or 'lumped' reaction) and τ-leaping can be applied to the classes without necessarily violating the leap condition. One can then apply the τ-leap method to microscopic spatial KMC simulations (see D.G. Vlachos, submitted, for details).

9.2.1 A crystal growth example

Here we refer again to the earlier example of crystal growth (see Sects. 2.2.1 and 7.1) in order to illustrate the application of the τ-leap method in a microscopic KMC simulation. In the simulations below, the coarse time step has been chosen as that of the microscopic KMC simulation multiplied by an acceleration factor $f(f > 1)$ [95]

$$\tau = f/\Gamma_{\text{tot}}. \tag{70}$$

This is a simpler choice of computing time increments (in comparison to the r-criterion discussed above) and allows a more direct comparison between microscopic and coarse simulations. The Poisson τ-leap method is employed in this example. Results are shown in Fig. 6. Multiple data (seven calculations) for various values of the acceleration factor, f, of the τ-leap method from $f = 15$ to 1,000 were performed at $w/kT = 1$. The proximity of points (triangles), which are hard to discern in the logarithmic scale, indicates a small error (up to ~10%) with respect to the microscopic simulation despite the substantial temporal coarse-graining.

Figure 6d shows the bundles of the τ-leap method versus time for a larger lattice of $N_{\mathrm{L}} = 200 \times 200$. For sufficiently large lattices, the bundles can be large. In this simulation, many adsorption and desorption events occur per time increment, especially for classes with one and two nearest neighbors. In contrast, sites with many nearest neighbors have so low-desorption transition probability that such desorption events rarely occur.

Regarding the performance of the τ-leap method, in all cases it is much faster than the rejection-free method with the n-fold search method and global updating (see Sect. 7.1). The acceleration depends on the size of the time increment τ. As an example, for $w/kT = 1$ the τ-leap method is faster than the rejection-free method by seven times for $f = 15$ and by 140 times for $f = 1,000$. At the same conditions, the CPU of the τ-leap method can be slightly larger and up to a factor of 5 less than the null-event algorithm. Again, τ-leap methods are very efficient when large time steps can be taken. For non-stiff problems, large domains result in large class sizes and better speedup where the benefit of τ-leap methods is substantial.

10 Summary and outlook

In this paper an overview of the traditional and multiscale KMC methods was given. Specifically, the challenges encountered in the KMC simulation were first outlined. A firm foundation of the null-event algorithm was provided and its equivalence to the rejection-free algorithm was established using simple probability theory. Examples from crystal growth and diffusion of defects in 2D served to numerically illustrate the equivalence of the two methods. We further compared these algorithms in terms of implementation ease, memory, and computational cost.

The search and update steps are at the heart of any KMC simulation. Various such methods were discussed, scaling laws were outlined, and numerical examples were provided to compare them.

A number of physical systems exhibit spatial inhomogeneity over length scales much larger than a few nanometers that cannot be captured using small periodic lattices. Simulation of such large domains is currently impossible with microscopic KMC methods. The coarse-grained KMC method is a promising technique for such applications. This method is based on the assumption of local equilibrium in each coarse cell, i.e., on separation of time scales. A number of stochastic closures as well as multigrid methods can be used to resolve some or most of the lost information for the distribution of particles within coarse cells and improve the accuracy of the method. The method can further be improved using spatial adaptivity. The concept of stochastic computational singular perturbation is key to handling separation of time scales. Finally, temporal coarse-graining, using multifiring methods, can overcome the one-at-a time execution of events and can be applied to both spatially coarse and microscopic lattices resulting in further acceleration.

A direct application of these multiscale KMC methods capitalizes on the concept of hierarchical multiscale method (HiMM) introduced in Vlachos [36]. HiMM is a systems analysis tool in which continuum mesoscopic equations (derived in the limit of infinite coarse-graining), coarse-grained KMC methods, and microscopic KMC methods are employed hierarchically to study the stochastic behavior of complex systems. This approach, used primarily for design and control of multiscale systems encompassing multiple system variables, such as temperature and concentrations, starts with a low-level tool (e.g., large coarse-graining, low accuracy, low-CPU requirements), and subsequently refines the information using higher-level tools (small coarse-graining, high accuracy, large CPU requirements). Even though a similar idea is often employed in the multiscale modeling community via different model representations of different scales (hybrid multiscale modeling), the HiMM employs *variable* coarse-graining for the same underlying microscopic model and very importantly preserves the noise between scales. The numerical advantages of this approach are outlined in Vlachos [36]. In essence, the HiMM can be viewed as a *multiresolution* system analysis technique for identifying interesting dynamics and/or equilibrium phase behavior. Recently, HiMM was employed for self-assembled nanopattern (namely, nanoparticles, nanowires, and 'inverted nanodots') formation in heteroepitaxy, resulting from an interplay between competing attractive, and repulsive interactions [140]. Deterministic continuum mesoscopic equations are amenable to non-linear (bifurcation) analysis techniques and enable the relatively easy generation of a dynamic phase diagram of pattern shapes and scaling laws of feature size and shape, in terms of the microscopic model parameters (e.g., interaction potential parameters, substrate temperature, film thickness, and material properties). Next the CGMC method is employed for investigating the role of thermal fluctuations in nucleation of patterns, shape and defects, and for refining the phase diagram.

While all these methods are promising, application to real examples may need some generalization of theory. Ultimately, the success of these methods will be dictated from their ease of implementation and mainly from the resulting speedup in real world problems.

Acknowledgments DGV's multiscale simulation work has been supported over the years by NSF grants (Career Award no. CTS-9702615, CTS-9811088, ITR-0219211, and CTS-0343757) and by the US Department of Energy (DE-FG02-05ER25702 and DE-FG02-04ER25626). AC acknowledges financial support from the University of Delaware, Competitive Graduate School Fellowship. Useful discussions and collaborations with Prof. Markos A. Katsoulakis are also acknowledged.

References

1. Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., Teller, E.: Equation of state calculations by fast computing machines. J. Chem. Phys. **21**, 1087–1092 (1953)
2. Allen, M.P., Tildesley, D.J.: Computer Simulation of Liquids. Oxford Science Publications, Oxford (1989)
3. Frenkel, D., Smit, B.: Understanding Molecular Simulation: From Algorithms to Applications. Academic Press, New York (1996)
4. Auerbach, S.M.: Theory and simulation of jump dynamics, diffusion and phase equilibrium in nanopores. Int. Rev. Phys. Chem. **19**, 155–198 (2000)
5. Binder, K.: Monte Carlo Methods in Statistical Physics, vol. 7. Springer, Berlin Heidelberg New York (1986)
6. Binder, K.: Atomistic modeling of materials properties by Monte-Carlo simulation. Adv. Mater. **4**, 540–547 (1992)
7. Landau, D.P., Binder, K.: A Guide to Monte Carlo Simulations in Statistical Physics. Cambridge University Press, Cambridge (2000)
8. Ciccotti, G., Frenkel, D., McDonald, I.R.: Simulation of Liquids and Solids. Molecular Dynamics and Monte Carlo Methods in Statistical Mechanics. North-Holland, Amsterdam (1987)
9. Dooling, D.J., Broadbelt, L.J.: Generic Monte Carlo tool for kinetic modeling. Ind. Eng. Chem. Res. **40**, 522–529 (2001)
10. Gilmer, G.H., Huang, H.C., de la Rubia, T.D., Dalla Torre, J., Baumann, F. Lattice Monte Carlo models of thin film deposition. Thin Solid Films **365**, 189–200 (2000)

11. Nieminen, R., Jansen, A.: Monte Carlo simulations of surface reactions. Appl. Catal. A: Gen. **160**, 99–123 (1997)
12. Hill, T.L.: An Introduction to Statistical Thermodynamics. Dover, New York (1986)
13. Chakraborty, A.K.: Molecular Modeling and Theory in Chemical Engineering, vol. 28. Academic Press, New York (2001)
14. Broadbelt, L., Snurr, R.: Applications of molecular modeling in heterogeneous catalysis research. Appl. Catal. A: Gen. **200**, 23–46 (2000)
15. Sholl, D.S., Tully, J.C.: A generalized surface hopping method. J. Chem. Phys. **109**, 7702–7710 (1998)
16. Catlow, C.R.A., Bell, R.G., Gale, J.D.: Computer modeling as a technique in materials chemistry. J. Mat. Chem. **4**, 781–792 (1994)
17. Evans, J.W., Miesch, M.S.: Catalytic reaction kinetics near a first-order poisoning transition. Surf. Sci. **245**, 401–410 (1991)
18. Hansen, E.W., Neurock, M.: First-principles-based Monte Carlo simulation of ethylene hydrogenation kinetics on Pd. J. Catal. **196**, 241–252 (2000)
19. Huang, H.C., Gilmer, G.H.: Multi-lattice Monte Carlo model of thin films. J. Comput. Aided Mater. Des. **6**, 117–127 (1999)
20. Jansen, A.P.J.: Monte Carlo simulations of chemical reactions on a surface with time-dependent reaction-rate constants. Comput. Phys. Commun. **86**, 1–12 (1995)
21. Kang, H.C., Weinberg, W.H.: Dynamic Monte Carlo with a proper energy barrier: Surface diffusion and two-dimensional domain ordering. J. Chem. Phys. **90**, 2824–2830 (1988)
22. Kew, J., Wilby, M.R., Vvedensky, D.D.: Continuous-space Monte Carlo simulations of epitaxial-growth, Journal of Crystal Growth. J. Crystal Growth **127**, 508–512 (1993)
23. Khor, K.E., Das Sarma, S.: Quantum dot self-assembly in growth of strained-layer thin films: A kinetic Monte Carlo study. Phys. Rev. B **62**, 16657–16664 (2002)
24. Macedonia, M.D., Maginn, E.J.: Impact of confinement on zeolite cracking selectivity via Monte Carlo integration. AIChE J. **46**, 2504–2517 (2000)
25. Nikolakis, V., Vlachos, D.G., Tsapatsis, M.: Modeling of zeolite L crystallization using continuum time Monte Carlo simulations. J. Chem. Phys. **111**, 2143–2150 (1999)
26. Novere, N.L., Shimizu, T.S.: STOCHSIM: modelling of stochastic biomolecular processes. Bioinformatics **17**, 575–576 (2001)
27. Schulze, T.P.: A hybrid scheme for simulating epitaxial growth. J. Crystal Growth **263**, 605–615 (2004)
28. Zhdanov, V.P., Kasemo, B.: Kinetics of rapid reactions on nanometer catalyst particles. Phys. Rev. B, **55**, 4105–4108 (1997)
29. Gilmer, G.: Computer models of crystal growth. Science **208**, 355–363 (1980)
30. Muller-Krumbhaar, H.: Kinetics of crystal growth. In: Kaldis, E. (ed.) Current Topics in Materials Science, pp. 1–46. North-Holland, Amsterdam (1978)
31. Drews, T.O., Ganley, J.C., Alkire, R.C.: Evolution of surface roughness during copper electrodeposition in the presence of additives - Comparison of experiments and Monte Carlo simulations. J. Electrochem. Soc. **150**, C325–C334 (2003)
32. Lou, Y., Christofides, P.D.: Feedback control of surface roughness of GaAs (001) thin films using kinetic Monte Carlo models. Comput. Chem. Eng. **29**, 225–241 (2004)
33. Gallivan, M.A., Murray, R.M.: Reduction and identification methods for Markovian control systems, with application to thin film deposition. Int. J. Robust Nonlinear Control **14**, 113–132 (2004)
34. Wicke, E., Kunmann, P., Keil, W., Schiefler, J.: Unstable and oscillatory behavior in heterogeneous catalysis. Berichte der Bunsen-Gesellschaft-Phys. Chem. Chem. Phys. **84**, 315–323 (1980)
35. Ziff, R.M., Gulari, E., Barshad, Y.: Kinetic phase transitions in an irreversible surface-reaction model. Phys. Rev. Lett. **56**, 2553–2556 (1986)
36. Vlachos, D.G.: A review of multiscale analysis: Examples from systems biology, materials engineering, and other fluid-surface interacting systems. Adv. Chem. Eng. **30**, 1–61 (2005)
37. Cuitino, A.M., Stainier, L., Wang, G.F., Strachan, A., Cagin, T., Goddard, W.A., Ortiz, M.: A multiscale approach for modeling crystalline solids. J. Comput. Aided Mater. Des. **8**, 127–149 (2002)
38. Miller, R.E., Tadmor, E.B.: The quasicontinuum method: Overview, applications and current directions. J. Comput. Aided Mater. Des. **9**, 203–239 (2002)
39. Maroudas, D.: Multiscale modeling. In: Challenges for the Chemical Sciences in the 21st Century: Information and Communications Report, pp. 133–136. National Academies, Washington, DC (2003)

40. Grujicic, M., Lai, S.G.: Multi-length scale modeling of chemical vapor deposition of titanium nitride coatings. J. Mater. Sci. **36**, 2937–2953 (2001)
41. Jaraiz, M., Rubio, E., Castrillo, P., Pelaz, L., Bailon, L., Barbolla, J., Gilmer, G.H., Rafferty, C.S.: Kinetic Monte Carlo simulations: an accurate bridge between ab initio calculations and standard process experimental data. Mater. Sci. Semiconductor Process. **3**, 59–63 (2000)
42. Kremer, K., Muller-Plathe, F.: Multiscale simulation in polymer science. Mol. Simul. **28**, 729–750 (2002)
43. Duke, T.A.J., Le Novere, N., Bray, D.: Conformational spread in a ring of proteins: A stochastic approach to allostery. J. Mol. Biol. **308**, 541–553 (2001)
44. McAdams, H.H., Arkin, A.: Stochastic mechanisms in gene expression. Proc. Natl. Acad. Sci. **94**, 814–819 (1997)
45. McAdams, H.H., Arkin, A.: It's a noisy business! Genetic regulation at the nanomolar scale. Trends in Genetics, **15**, 65–69 (1999)
46. Woolf, P.J., Linderman, J.J.: Self organization of membrane proteins via dimerization. Biophys. Chem. **104**, 217–227 (2003)
47. Mayawala, K., Vlachos, D.G., Edwards, J.S.: Spatial modeling of dimerization reaction dynamics in the plasma membrane: Monte Carlo vs. continuum differential equations. Biophys. Chem. **121**, 194–208 (2006)
48. National Research Council (NRC): Beyond the Molecular Frontier: Challenges for Chemistry and Chemical Engineering. National Research Council, The National Academy Press, BCST, www.nap.edu publication (2003)
49. Partnership, C.I.V.T., Chemical Industry Vision2020 Technology Partnership, Chemical Industry R&D Roadmap for Nanomaterials by design. www.ChemicalVision2020.org (2003)
50. Vlachos, D.G.: Molecular modeling for non-equilibrium chemical processes. In: Lee, S. (ed.) Encyclopedia of Chemical Processing, pp. 1717–1726. Taylor and Francis, New York.
51. Voter, A.F.: Introduction to the Kinetic Monte Carlo Method. Radiation Effects in Solids. Springer, NATO Publishing unit, Dordrecht (2006) in press.
52. Gardiner, C.W.: Handbook of Stochastic Methods, 2nd edn. Springer, Berlin Heidelberg New York (1985)
53. Ghez, R.: A Primer of Diffusion Problems. John Wiley & Sons, New York (1988)
54. Vlachos, D.G., Schmidt, L.D., Aris, R.: Kinetics of faceting of crystals in growth, etching, and equilibrium. Phys. Rev. B, **47**, 4896–4909 (1993)
55. Magna, A.L., Coffa, S., Colomo, L.: Role of externded vacancy-vacancy interaction on the ripening of voids in silicon. Phys. Rev. Lett. **82**, 1720–1723 (1999)
56. Domain, C., Becquart, C.S., Malerba, L.: Simulation of radiation damage in Fe alloys: an object kinetic Monte Carlo approach. J. Nucl. Mater. **335**, 121–145 (2004)
57. Sadigh, B., Lenosky, T.J., Theiss, S.K., Caturla, M.J., de la Rubia, T.D., Foad, M.A.: Mechanism of boron diffusion in silicon: An ab initio and kinetic Monte Carlo study. Phys. Rev. Lett. **83**, 4341–4344 (1999)
58. Noda, T.: Modeling of Indium diffusion and end-of-range defects in Silicon using a kinetic Monte Carlo simulation. J. Appl. Phys. **94**, 6396–6400 (2003)
59. Gordon, S.M.J., Kenny, S.D., Smith, R.: Diffusion dynamics of defects in Fe and Fe-P systems. Phys. Rev. B **72**, 214104 (2005)
60. Soneda, N., Rubia, T.D.: Defect production, annealing kinetics and damage evolution in a-Fe: an atomic-scale compuer simulation. Philos. Mag. A **78**, 995–1019 (1998)
61. Dai, J., Kanter, J.M., Kapur, S.S., Seider, W.D., Sinno, T.: On-lattice kinetic Monte Carlo simulations of point defect aggregation in entropically influenced crystalline systems. Phys. Rev. B, **72**, 134102 (2005)
62. Fahey, P.M., Griffin, B.P., Plummer, J.D.: Point defects and dopant diffusion in silicon. Rev. Mod. Phys. **61**, 289 (1989)
63. Flynn, C.P.: Point defects and diffusion. Calderon, Press, Oxford (1972)
64. Vlachos, D.G., Katsoulakis, M.A.: Derivation and validation of mesoscopic theories for diffusion of interacting molecules. Phys. Rev. Lett. **85**, 3898–3901 (2000)
65. Lam, R., Basak, T., Vlachos, D.G., Katsoulakis, M.A.: Validation of mesoscopic theories and their application to computing effective diffusivities. J. Chem. Phys. **115**, 11278–11288 (2001)
66. Gillespie, D.T.: A general method for numerically simulating the stochastic evolution of coupled chemical reactions. J. Comput. Phys. **22**, 403–434 (1976)
67. Gomer, R.: Diffusion of adsorbates on metal surfaces. Rep. Prog. Phys. **53**, 917–1002 (1990)
68. Kapur, S.S., Prasad, M., Crocker, J.C., Sinno, T.: Role of configurational entropy in the thermodynamics of clusters of point defects in crystalline solids. Phys. Rev. B **72**, 014119 (2005)

69. Henkelman, G., Jonsson, H.: Long time scale kinetic Monte Carlo simulations without lattice approximation and predefined event table. J. Chem. Phys. **115**, 9657 (2001)
70. Schulze, T.P.: Kinetic Monte Carlo simulations with minimal searching. Phys. Rev. E **65**, 036704 (2002)
71. Lukkien, J.J., Segers, J.P.L., Hilbers, P.A.J., Gelten, R.J., Jansen, A.P.J.: Efficient Monte Carlo methods for the simulation of catalytic surface reactions. Phys. Rev. E **58**, 2598–2610 (1998)
72. Bortz, A.B., Kalos, M.H., Lebowitz, J.L.: A new algorithm for Monte Carlo simulations of Ising spin systems. J. Comput. Phys. **17**, 10–18 (1975)
73. Snyder, M.A., Chatterjee, A., Vlachos, D.G.: Net-event kinetic Monte Carlo for overcoming stiffness in spatially homogeneous and distributed systems, invited. Comput. Chem. Eng. **29**, 701–712 (2004)
74. Vlachos, D.G.: Stochastic modeling of chemical microreactors with detailed kinetics: induction times and ignitions of H_2 in air. Chem. Eng. Sci. **53**, 157–168 (1998)
75. Resat, H., Wiley, H.S., Dixon, D.A.: Probability-weighted dynamic Monte Carlo method for reaction kinetics simulations. J. Chem. Phys. **105**, 11026–11034 (2001)
76. DeVita, J.P., Sander, L.M., Smereka, P.: Multiscale kinetic Monte Carlo algorithm for simulating epitaxial growth. Phys. Rev. B **72**, 205421 (2005)
77. Haseltine, E.L., Rawlings, J.B.: Approximate simulation of coupled fast and slow reactions for stochastic chemical kinetics. J. Chem. Phys. **117**, 6959–6969 (2002)
78. Cao, Y., Gillespie, D.T., Petzold, L.R.: The slow-scale stochastic simulation algorithm. J. Chem. Phys. **122**, 014116 (2005)
79. Chatterjee, A., Vlachos, D.G.: Multiscale spatial Monte Carlo simulations: multigriding, computational singular perturbation, and hierarchical stochastic closures. J. Chem. Phys. **124**, 064110 (2006)
80. Liu, W.E.D., Eijnden, E.V.: Nested stochastic simulation algorithm for chemical kinetic systems with disparate rates. J. Chem. Phys. **123**, 1941071–19410716 (2005)
81. Samant, A., Vlachos, D.G.: Overcoming stiffness in stochastic simulation stemming from partial equilibrium: a multiscale Monte Carlo algorithm. J. Chem. Phys. **123**, 144114 (2005)
82. Salis, H., Kaznessis, Y.N.: An equation-free probabilistic steady-state approxaimtion: Multigriding, computational singular perturbation, and hierarchical stochastic closures. J. Chem. Phys. **123**, 2141061–21410616 (2005)
83. Katsoulakis, M., Majda, A.J., Vlachos, D.G.: Coarse-grained stochastic processes for microscopic lattice systems. Proc. Natl. Acad. Sci. **100**, 782–787 (2003)
84. Katsoulakis, M.A., Vlachos, D.G.: Coarse-grained stochastic processes and kinetic Monte Carlo simulators for the diffusion of interacting particles. J. Chem. Phys. **119**, 9412–9428 (2003)
85. Katsoulakis, M.A., Majda, A.J., Vlachos, D.G.: Coarse-grained stochastic processes and Monte Carlo simulations in lattice systems. J. Comput. Phys. **186**, 250–278 (2003)
86. Chatterjee, A., Vlachos, D.G., Katsoulakis, M.A.: Spatially adaptive lattice coarse-grained Monte Carlo simulations for diffusion of interacting molecules. J. Chem. Phys. **121**, 11420–11431 (2004)
87. Chatterjee, A., Katsoulakis, M.A., Vlachos, D.G.: Spatially adaptive grand canonical Monte Carlo simulations. Phys. Rev. E **71**, 026702 (2005)
88. Chatterjee, A., Vlachos, D.G., Katsoulakis, M.: Numerical assessment of theoretical error estimates in coarse-grained kinetic Monte Carlo simulations: application to surface diffusion. Int. J. Multiscale Comput. Eng. **3**, 59–70 (2005)
89. Ismail, A.E., Rutledge, G.C., Stephanopoulos, G.: Multiresolution analysis in statistical mechanics. I. Using wavelets to calculate thermodynamic properties. J. Chem. Phys. **118**, 4414–4423 (2003)
90. Ismail, A.E., Stephanopoulos, G., Rutledge, G.C.: Multiresolution analysis in statistical mechanics. II. The wavelet transform as a basis for Monte Carlo simulations on lattices. J. Chem. Phys. **118**, 4424–4431 (2003)
91. Chatterjee, A., Vlachos, D.G.: Temporal acceleration of spatially distributed kinetic Monte Carlo simulations. J. Comput. Phys. **211**, 596–615 (2006)
92. Gillespie, D.T.: Approximate accelerated stochastic simulation of chemically reacting systems. J. Chem. Phys. **115**, 1716–1733 (2001)
93. Rathinam, M., Petzold, L.R., Cao, Y., Gillespie, D.T.: Stiffness in stochastically reacting systems: the implict tau-leaping method. J. Chem. Phys. **119**, 12784–12794 (2003)
94. Tian, T., Burrage, K.: Binomial leap methods for simulating stochastic chemical kinetics. J. Chem. Phys. **121**, 10356–10364 (2004)
95. Chatterjee, A., Vlachos, D.G., Katsoulakis, M.: Binomial distribution based τ-leap accelerated stochastic simulation. J. Chem. Phys. **122**, 024112 (2005)

96. Chatterjee, A., Mayawala, K., Edwards, J.S., Vlachos, D.G.: Time accelerated Monte Carlo simulations using the binomial τ-leap method. Bioinformatics **21**, 2136–2137 (2005)
97. Auger, A., Chatelain, P., Koumoutsakos, P.: R-leaping: Accelerating the stochastic simulation algorithm by reaction leaps. J. Chem. Phys. **125**, 084103 (2006)
98. Cao, Y., Petzold, L.R., Rathinam, M., Gillespie, D.T.: The numerical stability of leaping methods for stochastic simulation of chemically reacting systems. J. Chem. Phys. **121**, 12169–12178 (2004)
99. Thostrup, P., Christoffersen, E., Lorensen, H.T., Jacobsen, K.W., Besenbacher, F., Norskov, J.K.: Adsorption-induced step formation. Phys. Rev. Lett. **87**, 126102 (2001)
100. Kratzer, P., Penev, E., Scheffler, M.: Understanding the growth mechanisms of GaAs and InGaAs thin films by employing first-principles calculations. Appl. Surf. Sci. **216**, 436–446 (2003)
101. Fichthorn, K.A., Scheffler, M.: Island nucleation in thin-film epitaxy: a first-principles investigation. Phys. Rev. Lett. **84**, 5371 (2000)
102. Neurock, M., Hansen, E.W.: First-principles-based molecular simulations of heterogeneous catalytic surface chemistry. Comput. Chem. Eng. **22**, S1045–S1060 (1998)
103. Haug, K., Raibeck, G.: Kinetic Monte Carlo study of competing hydrogen pathways into connected (100), (110) and (111) Ni surfaces. J. Phys. Chem. B **107**, 11433–11440 (2003)
104. Truhlar, D.G., Garrett, B.C., Klippenstein, S.J.: Current status of transition-state theory. J. Phys. Chem. **100**, 12771–12800 (1996)
105. Car, R., Parrinello, M.: Unified approach for molecular dynamics and density functional theory. Phys. Rev. Lett. **55**, 2471–2474 (1985)
106. Voter, A.F.: Classically exact overlayer dynamics: diffusion of Rhodium clusters on Rh(100). Phys. Rev. B **34**, 6819–6829 (1986)
107. Vvedensky, D.D.: Multiscale modelling of nanostructures. J. Phys. Cond. Mater. **16**, R1537–R1576 (2004)
108. Maroudas, D.: Multiscale modeling of hard materials: Challenges and opportunities for chemical engineering. AIChE J. **46**, 878–882 (2000)
109. Wadley, H.N.G., Zhou, X., Johnson, R.A., Neurock, M.: Mechanisms, models and methods of vapor deposition. Prog. Mater. Sci. **46**, 329–377 (2001)
110. Raimondeau, S., Vlachos, D.G.: Recent developments on multiscale, hierarchical modeling of chemical reactors. Chem. Eng. J. **90**, 3–23 (2002)
111. Daw, M.S., Foiles, S.M., Baskes, M.I.: The embedded-atom method: a review of theory and applications. Mater. Sci. Rept. **9**, 251–310 (1993)
112. Jacobsen, K.W., Norskov, J.K., Puska, M.J.: Interatomic interactions in the effective-medium theory. Phys. Rev. B **35**, 7423–7442 (1987)
113. Wang, Z., Li, Y., Adams, J.B.: Kinetic lattice Monte Carlo simulation of facet growth rate. Surf. Sci. **450**, 51–63 (2000)
114. Abraham, F.F., Broughton, J.Q., Bernstein, N., Kaxiras, E.: Spanning the continuum to quantum length scales in a dynamic simulation of brittle fracture. Europhys. Lett. **44**, 783–787 (1998)
115. Jónsson, H., Mills, G.: Nudged elastic band methods for finding minimum energy paths of transitions. In: Berne, B., Ciccotti, G., Coker, D.F., (eds.) Classical and Quantum Dynamics in Condensed Phase Simulations, pp. 385–404. World Scientific, Singapore (1998)
116. Wales, D.J.: Energy landscapes: calculating pathways and rates. Int. Rev. Phys. Chem. **25**, 237–282 (2006)
117. Olsen, R.A., Kroes, G.J., Henkelman, G., Arnaldsson, A., Jonsson, H.: Comparison of methods for finding saddle points without knowledge of final states. J. Chem. Phys. **121**, 9776 (2004)
118. Voter, A.F., Montalenti, F., Germann, T.C.: Extending the time scales in atomistic simulation of materials. Annu. Rev. Mater. Res. **32**, 321–346 (2002)
119. Lavrentiev, M., Allan, N., Harding, J., Harris, D., Purton, J.: Atomistic simulations of surface diffusion and segregartion in ceramics. Comput. Mater. Sci. **36**, 54–59 (2006)
120. Trushin, O., Karim, A., Kara, A., Rahman, T.S.: Self-learning kinetic Monte Carlo method: Application to Cu(111). Phys. Rev. B **72**, 1154011–1154019 (2005)
121. Renisch, S., Schuster, R., Wintterlin, J., Ertl, G.: Dynamics of adatom motion under the influence of mutual interactions: O/Ru(0001). Phys. Rev. Lett. **82**, 3839–3842 (1999)
122. Maroudas, D.: Modeling of radical-surface interactions in the plasma-enhanced chemical vapor deposition of silicon thin films. In: Chakraborty, A.K. (ed.) Molecular Modeling and Theory in Chemical Engineering, pp. 252–296. Academic Press, New York (2001)
123. Raimondeau, S., Aghalayam, P., Vlachos, D.G., Katsoulakis, M.: Bridging the gap of multiple scales: From microscopic, to mesoscopic, to macroscopic models. In: Proceedings of the Foundations of Molecular Modeling and Simulation, AIChE Symposium Series No. 325, 97, pp. 155–158. Keystone, Co, USA (2001)

124. Gillespie, D.T.: Exact stochastic simulation of coupled chemical reactions. J. Phys. Chem. **81**, 2340–2361 (1977)
125. Gibson, M.A., Bruck, J.: Efficient exact stochastic simulation of chemical systems with many species and many channels. J. Phys. Chem. A **104**, 1876–1889 (2000)
126. Gilmer, G.H., Bennema, P.: Simulation of crystal growth with surface diffusion. J. Appl. Phys. **43**, 1347–1360 (1972)
127. Reese, J.S., Raimondeau, S., Vlachos, D.G.: Monte Carlo algorithms for complex surface reaction mechanisms: efficiency and accuracy. J. Comput. Phys. **173**, 302–321 (2001)
128. Vlachos, D.G., Schmidt, L.D., Aris, R.: The effects of phase transitions, surface diffusion, and defects on surface catalyzed reactions: Oscillations and fluctuations. J. Chem. Phys. **93**, 8306–8313 (1990)
129. Vlachos, D.G., Schmidt, L.D., Aris, R.: The effect of phase transitions, surface diffusion, and defects on heterogeneous reactions: multiplicities and fluctuations. Surf. Sci. **249**, 248–264 (1991)
130. Fichthorn, F.A., Weinberg, W.H.: Theoretical foundations of dynamical Monte Carlo simulations. J. Chem. Phys. **95**, 1090–1096 (1991)
131. Mayawala, K., Vlachos, D.G., Edwards, J.S.: Computational modeling reveals molecular details of epidermal growth factor binding. BMC Cell Biol. **6**(41), 1–11 (2005)
132. van der Eerden, J.P., Bennema, P., Cherepanova, T.A.: Survey of Monte Carlo simulations of crystal surfaces and crystal growth. Prog. Crystal Growth Characterization **1**, 219–254 (1978)
133. Masel, R.I.: Principles of Adsorption and Reaction on Solid Surfaces. Wiley, NY (1996)
134. Schoeberl, B., Eichler-Jonsson, C., Gilles, E.D., Müller, G.: Computational modeling of the dynamics of the MAP kinase cascade activated by surface and internalized receptors. Nat. Biotechnol. **20**, 370–375 (2002)
135. Dumesic, I.A., Rud, D.F., Aparicio, L.M., Rekoske, J.E., Revino, A.A.: The Microkinetics of Heterogeneous Catalysis. American Chemical Society, Washington, DC (1993)
136. Cormen, T.H., Leiserson, C.E., Rivest, R.L.: Introduction to Algorithms. MIT Press, Cambridge, MA (2001)
137. Cao, Y., Li, H., Petzold, L.R.: Efficient formulation of the stochastic simulation algorithm. J. Chem. Phys. **121**, 4059–4067 (2004)
138. Press, W.H., Flannery, B.P., Teukolsky, S.A., Vetterling, W.T.: Numerical Recipes. Cambridge University Press, Cambridge (1986)
139. Goldenfeld, N.: Lectures on Phase Transitions and the Renormalization Group, chap. 9. Addison-Wesley, New York (1992)
140. Chatterjee, A., Vlachos, D.G.: Systems tasks in nanotechnology via hierarchical multiscale: formation of nanodisks arrays in heteroepitaxy. Chem. Eng. Sci. In press (2007).
141. Chatterjee, A., Snyder, M.A., Vlachos, D.G.: Mesoscopic modeling of chemical reactivity. Chem. Eng. Sci. ISCRE 18, invited, **59**, 5559–5567 (2004)
142. Chatterjee, A., Vlachos, D.G.: Hierarchical coarse-grained models derived from Kinetic Monte Carlo models: Part II: Coarse-grained Monte Carlo method for multiple interacting species, sites and crystallographic surface types. J. Chem. Phys. In preparation (2007)
143. Daw, M.S., Baskes, M.I.: Embedded-atom method: Derivation and application to impurities, surfaces, and other defects in metals. Phys. Rev. B **29**, 6443–6453 (1984)
144. Stillinger, F.H., Weber, T.A.: Computer-simulation of local order in condensed phases of silicon. Phys. Rev. B **31**, 5262–5271 (1985)
145. Haken, H.: Synergetics. Springer, Berlin Heidelberg New York (1977)
146. Rao, C.V., Arkin, A.P.: Stochastic chemical kinetics and the quasi-steady-state assumption: Application to the Gillespie algorithm. J. Chem. Phys. **118**, 4999–5010 (2003)
147. Hill, T.L.: Statistical Mechanics Principles and Selected Applications. Dover, New York (1987)
148. Stinchcombe, K.H., White, H.: Multilayer feedforward networks are universal approximators. Neural Netw. **2**, 359–366 (1989)
149. Katsoulakis, M., Trashorras, J.: Information loss in coarse-graining of stochastic particle dynamics. J. Stat. Phys. **122**, 115–135 (2006)
150. Burrage, K., Tian, T.H., Burrage, P.: A multi-scaled approach for simulating chemical reaction systems. Prog. Biophys. Mol. Biol. **85**, 217–234 (2004)