Journal of Computational Physics 230 (2011) 1359–1369

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

Billion-atom synchronous parallel kinetic Monte Carlo simulations of critical 3D Ising systems

E. Martínez [a], P.R. Monasterio [b], J. Marian [c,*]

[a] IMDEA-Materiales, Madrid 28040, Spain
[b] Massachusetts Institute of Technology, Cambridge, MA 02139, USA
[c] Lawrence Livermore National Laboratory, Livermore, CA 94551, USA

A R T I C L E   I N F O

Article history:
Received 25 May 2010
Received in revised form 2 November 2010
Accepted 4 November 2010
Available online 11 November 2010

Keywords:
Kinetic Monte Carlo
Parallel computing
Discrete lattice
Ising system

A B S T R A C T

An extension of the synchronous parallel kinetic Monte Carlo (spkMC) algorithm developed by Martinez et al. [J. Comp. Phys. 227 (2008) 3804] to discrete lattices is presented. The method solves the master equation synchronously by recourse to null events that keep all processors' time clocks current in a global sense. Boundary conflicts are resolved by adopting a chessboard decomposition into non-interacting sublattices. We find that the bias introduced by the spatial correlations attendant to the sublattice decomposition is within the standard deviation of serial calculations, which confirms the statistical validity of our algorithm. We have analyzed the parallel efficiency of spkMC and find that it scales consistently with problem size and sublattice partition. We apply the method to the calculation of scale-dependent critical exponents in billion-atom 3D Ising systems, with very good agreement with state-of-the-art multispin simulations.

Published by Elsevier Inc.

1. Introduction

Kinetic Monte Carlo (kMC) [1] has proven an efficient and powerful tool to study non-equilibrium processes, and is used in fields as different as population dynamics, irradiation damage, or crystal growth [2,3]. The most widely used variants of the method are the rejection-free Monte Carlo *time residence* algorithm [4] and the so-called *n*-fold method, or BKL in reference to its authors [5]. Although kMC is generally capable of advancing the time scale significantly faster than direct, time-driven methods, it suffers from numerical limitations such as *stiffness* [6], and time asynchronicity. This has spurred the development of more powerful variants such as coarse-grained kMC [7], first-passage kMC [8], and other accelerated methods [9]. Additionally, a number of parallelization schemes for kMC have been proposed, including rigorous and semi-rigorous algorithms based on asynchronous kinetics [10–12]. These methods rely on cumbersome roll-back procedures to avoid causality errors, *i.e.* event time incompatibilities associated with processor communications. For this reason, most applications of interest are studied using approximate schemes (non-rigorous) for computational convenience. In spite of this, calculations using asynchronous parallel kMC have provided numerous insights in several studies, most notably crystal growth [13].

Recently, we have developed and alternative algorithm based on a synchronous time decomposition of the master equation [14]. Our parallel kinetic Monte Carlo method, eliminates time conflicts by recourse to *null* events that advance the internal clock of each processor in a synchronized fashion without altering the stochastic trajectory of the system. The method has been demonstrated for continuum diffusion/reaction systems, which represents a worst-case application scenario for

* Corresponding author.
E-mail address: marian1@llnl.gov (J. Marian).

0021-9991/$ - see front matter Published by Elsevier Inc.
doi:10.1016/j.jcp.2010.11.006

two reasons. First, the maximum time step gain is limited by the intrinsic length scale of the problem at hand, which in concentrated systems may not be large; second, spatial boundary errors are difficult to eliminate due to the unbounded nature of diffusion in a continuum setting. This latter feature also limits the parallel efficiency of the algorithm, as global communications are needed during every Monte Carlo step. In any case, in our synchronous parallel kMC method (spkMC), the parallel error can always be computed intrinsically and reduced arbitrarily (at the expense of efficiency). In this paper, we extend spkMC to lattice systems, where diffusion lengths are quantized and boundary errors can be eliminated altogether. First, we adapt the algorithm proposed in Ref. [14] to discrete systems. Due to its widespread use and well-known properties, we have chosen the three-dimensional (3D) Ising system as our underlying lattice model. Second, we analyze the performance of the method in terms of stochastic bias (error) and parallel efficiency. We then apply spkMC to large systems near the critical point, which provides a demanding testbed for the method, as this is where fluctuations are exacerbated and convergence is most difficult.

2. The 3D ising model

The Ising model is one of the most extensively studied lattice systems in statistical mechanics. It consists of a lattice of N sites occupied by particles whose state is described by a variable σ_i that represents the spin of each particle and can take only the value ± 1. For pair-interactions, the Hamiltonian that gives the energy of configurational state $\sigma = \{\sigma_i\}$ takes the form:

$$\mathcal{H}(\sigma) = -J\sum_{\langle i,j\rangle}\sigma_i\sigma_j - H\sum_i \sigma_i, \tag{1}$$

where J is the coupling constant between neighboring pairs $\langle i,j\rangle$ and H is an external (magnetic) field. The time evolution of the system is assumed to be described by a master equation with Glauber dynamics [15], which states that the probability $p(\sigma,t)$ of finding the system in state σ at time t obeys the equation:

$$\frac{\partial p(\sigma,t)}{\partial t} = \sum_i [\mathcal{W}_i(\sigma)p(\sigma,t) - \mathcal{W}_i(\sigma')p(\sigma',t)], \tag{2}$$

where σ' denotes the configuration obtained from σ by flipping the ith spin with transition rate $\mathcal{W}_i(\sigma)$:

$$\mathcal{W}_i(\sigma) = \frac{\lambda}{2}[1 - \sigma_i \tanh(2\beta\Delta E_i)]. \tag{3}$$

Here λ is a positive constant that represents the natural frequency of the system, β is the reciprocal temperature, and $\Delta E_i = -J\sum_{\langle i,j\rangle}\sigma_j - H\sigma_i$ is the energy associated with spin i, which follows directly from Eq. (1). In what follows we consider only internally-driven systems ($H = 0$).

Many discrete systems can be mapped exactly or approximately to the Ising system. The grand canonical ensemble formulation of the lattice gas model, for example, can be mapped exactly to the canonical ensemble formulation of the Ising model. Also, binary alloy hamiltonians with nearest-neighbor interactions in rigid lattices can also be expressed as Ising hamiltonians. These mappings allow us to exploit results and behaviors of the Ising model to answer questions about the related models. In addition, the Ising system is particularly useful to study second-order phase transitions. The temperature T_c at which such transitions occur is known as the critical temperature. During the phase transition, thermodynamic quantities diverge according to power laws of T, whose exponents are known as the critical exponents. The nature of the phase transition is determined by whether the order parameter is continuous at T_c [16]. In a ferromagnetic system such as the Ising model, the order parameter is the net magnetization $m(\sigma)$:

$$m(\sigma) = \frac{1}{N}\sum_i \sigma_i. \tag{4}$$

For simple cubic lattices in 3D, $N = L^3$ is the number of sites and L the lattice size. As we approach the critical temperature from $T > T_c$, uncorrelated groups of spins align themselves in the same direction. These clusters grow in size, known as the correlation length ξ, which too diverges at the critical point. At $T = T_c$, one may theoretically encounter arbitrarily large areas with correlated spins pointing in one direction. In finite systems the upper limit of ξ is the system's dimension L. Thus, the challenge associated with simulating Ising systems during the phase transition is then ensuring that the error incurred by simulating a finite-size lattice is sufficiently small for the critical exponents to be calculated with certainty. This has spurred a great many Monte Carlo simulations of very large lattices in the hope of finding converged critical exponents (cf., *e.g.*, Ref. [17]).

When the system is in a ferromagnetic state, m decays from the spontaneous magnetization value m_0 with time as $m \propto t^{-\kappa/\nu z}$, where κ and ν are the critical exponents for m_0 and ξ, respectively. From the known value of the ratio $\kappa/\nu = 0.515$ [18] in 3D, one can obtain z from the slope of the m-t curve, obtained for several L, at the critical point. To study the finite-size dependence of T_c, high-order dimensionless ratios such as the *Binder* cumulant have been proposed:

$$U_4 = \frac{\langle m^4\rangle}{\langle m^2\rangle^2}, \tag{5}$$

which takes a value of $U_4 \approx 3$ when $T > T_c$ (when the magnetization oscillates aggressively around zero), and goes to zero at low temperatures, when $m = m_0$. As mentioned earlier, at the critical point the correlation length diverges, and therefore U_4

does not depend on L. kMC equilibrium calculations of U_4 for several lattice sizes can then be used to calculate the value of the reciprocal critical temperature $\beta_c = J/kT_c$, whose most accurate estimate is presently $\beta_c = 0.2216546$ [19,20].

3. Parallel algorithm for lattice systems

3.1. General algorithm

The basic structure of the algorithm is identical to that described in Ref. [14]. First, the entire configurational space is partitioned into K subdomains Ω_k. Note that, in principle, this decomposition need not be necessarily spatial (although this is the most common one), and partitions based on some other kind of load balancing can be equally adopted. However, without loss of generality, in what follows it is assumed that the system is spatially partitioned:

1. **A frequency line is constructed for each Ω_k as the aggregate of the individual rates**, r_{ik}, **corresponding to all the possible events within each subdomain**:

   $$R_k = \sum_i^{n_k} r_{ik},$$

   where n_k and R_k are, respectively, the number of possible events and the total rate in each subdomain k. Here $R_{tot} = \sum_k^K R_k$ and $N = \sum_k^K n_k$.
2. **We define the maximum rate**, R_{max}, **as**:

   $$R_{max} \geqslant \max_{k=1,\dots K}\{R_k\},$$

   This value is then communicated globally to all processors.
3. **We assign a *null* event with rate r_{0k} to each frequency line in each subdomain k such that**:

   $$r_{0k} = R_{max} - R_k,$$

   where, in general, the r_{0k} will all be different. We showed in Ref. [14] that the condition for maximum efficiency is that step (2) become strictly an equality, such that:

   $$\exists\Omega_\alpha, \quad \alpha \in \{k\}, \quad |R_\alpha \equiv R_{max} \rightarrow r_{\alpha 0} = 0,$$

   i.e. there is no possibility of null events. However, in principle, each subdomain can have any arbitrary r_{0k} as long as all the frequency lines in each Ω_k sum to the same global value. This flexibility is one of the most important features of our algorithm. In fact, as we noted in Ref. [14], in physical problems where the maximum meaningful time step is capped, R_{max} should always be adjusted (increased) accordingly.
4. **In each Ω_k an event is chosen with probability** $p_{ik} = r_{ik}/R_{max}$, **including null events with** $p_{k0} = r_{k0}/R_{max}$. For this step, we must ensure that independent sequences of random numbers be produced for each Ω_k, using appropriate parallel pseudo random number generators.
5. **As in standard BKL, a time increment is sampled from an exponential distribution**:

   $$\delta t_p = -\frac{\ln \xi}{R_{max}},$$

   where $\xi \in (0,1]$ is a suitable random number. Here, by virtue of Poisson statistics, δt_p becomes the global time step for all of the parallel processes.
6. **Communicate boundary events.** A global call will always achieve communication of boundary information. However, depending on the characteristics of the problem at hand, local calls may suffice, typically enhancing performance. This will be discussed in Section 5.

3.2. The sublattice method for solving boundary conflicts

As we have shown, this algorithm solves the master equation exactly for non-interacting particles. When particles are allowed to interact across domain boundaries, suitable corrections must be implemented to avoid boundary conflicts. For lattice-based kinetics with short-ranged interaction distances this is straightforwardly achieved by methods based on the chessboard sublattice technique. This spatial subdivison method has been used in multispin calculations of the kinetic Ising model since the early 1990s [21–23]. In the context of parallel kMC algorithms, Shim and Amar were the first to implement such procedure [24], in which a sublattice decomposition was used to isolate interacting domains in each cycle. The minimum number of sublattices to ensure non-interacting adjacent domains depends on a number of factors, most notably system dimensionality.[1] In 3D, the chessboard method requires a subdivison into a minimum of either two or eight sublattices,

[1] In 2D, four sublattices are sufficient to resolve any arbitrary mapping, as established by the solution to the 'four-color problem' [25].

depending on whether only first or farther nearest neighbor interactions are considered. This is schematically shown in Fig. 1, where each sublattice is defined by a specific color. The Figure shows the minimum sublattice block (white wireframe) to be assigned to each processor. These blocks are indivisible and each processor can be assigned only integer multiples of them in spkMC. The implementation of the sublattice algorithm is as follows. Because here Eq. (1) only involves first nearest neighbor interactions, the spatial decomposition performed in this work is such that it enables a regular sublattice construction with exactly two colors. In this fashion, each Ω_k becomes a subcell of a given sublattice, which imposes that each processor must have a multiple of two (or eight, for longer range interactions) number of subcells. Using a sublattice size greater than the particle interaction distance guarantees that no two particles in adjacent Ω_k interact. Step (4) above is then substituted by the following procedure:

4a. A given sublattice is chosen for all subdomains. This choice may be performed in several ways such as fully random or using some type of color permutation so that every sublattice is visited in each kMC cycle. Here we have implemented the former, as, for example, in the synchronous sublattice algorithm (SSL) of Shim and Amar [24]. The sublattice selection is performed with uniform probability thanks to the flexibility furnished by the spkMC algorithm, which takes advantage of the null rates to avoid global calls to communicate each sublattice's probability. Restricting each processor's sampling to only one lattice, however, while avoiding boundary conflicts, results in a systematic error associated with spatial correlations. The errors incurred by this procedure will be analyzed in Section 4.

4b. An event is chosen in the selected sublattice with the appropriate probability, including null events. When the rate changes in each Ω_k after a kMC cycle are unpredictable, a global communication of R_{max} in step (2) is unavoidable. When the cost of global communications becomes a considerable bottleneck in terms of parallel efficiency, it is worth considering other alternatives. For the Ising system, we consider the following:

- The simplest way to avoid global communications is to prescribe R_{max} to a very large value so as to ensure that it is never surpassed regardless of the kinetics being simulated. For the Ising model, this amounts to *calculating the maximum theoretical aggregate rate for an ensemble of Ising spins*. For a given subdomain Ω_k, this is:

$$R'_{max} = \lambda n_k \left[\frac{\exp(-\Delta E_{max})}{1 + \exp(-\Delta E_{max})} \right],$$

  where E_{max} is the theoretical maximum energy increment due to a single spin change:

$$E_{max} = -2(n_b|J| - |H|)$$

  and n_b is the lattice coordination number. This procedure is very conservative and may result in a poor parallel performance.
- *Perform a self-learning process to optimize* R'_{max}. This procedure is aimed at refining the upper estimate of R_{max} by recording the history of rate changes over the course of a spkMC simulation. For example, one can start with the maximum theoretical aggregate Ising rate and start decreasing the upper bound to improve the efficiency. For this procedure, a tolerance to ensure that $R'_{max} > R_{max}$ must be prescribed. A sufficiently-long time history of this comparison must be stored to perform regular checks and ensure that the inequality holds.

Therefore, the algorithm based on the modified steps 4a and 4b above, is only semirigorous for interacting systems due to the sampling strategies adopted to solve boundary conflicts, which introduce spatial correlations that result in a stochastic bias. Under certain conditions spkMC does behave rigorously in the sense that this bias is smaller than the intrinsic statistical error, as we shall see in the following section.

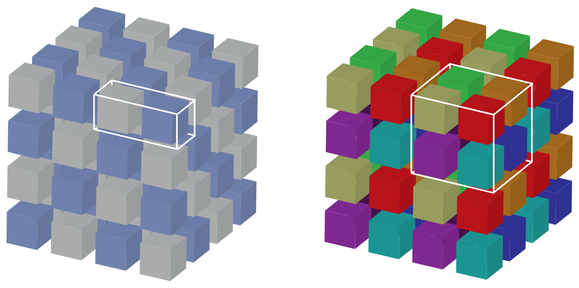

Fig. 1. Sublattice coloring scheme in three dimensions with regular subdivisions. Both two and eight color subdivisions are shown, corresponding to first and farther nearest neighbor sublattice interactions. The white wireframe indicates the indivisible color block assigned to processors.

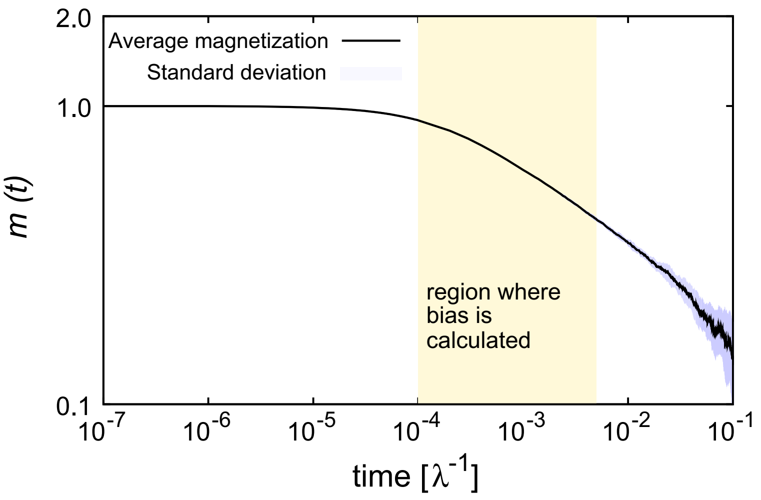

Fig. 2. Time evolution of the magnetization of a 262,144-spin Ising system at the critical temperature. The curve is the result of 20 independent serial kMC runs. The standard deviation, σ_s, is represented by the purple shaded area about the magnetization curve. The golden shaded area marks the region over which the bias is computed.

4. Stochastic bias and analysis of errors

The algorithm introduced above eliminates the occurrence of boundary conflicts at the expense of reducing the sampling configurational space of the system in each kMC step. Because boundary conflicts are inherently a spatial process, this introduces a spatial bias that must be quantified to understand the statistical validity of the spkMC results. Next, we analyze this bias by testing the behavior of the magnetization when the system is close to the critical point (σ_c). All the results shown in this section correspond to the sublattice algorithm using two colors with random selection.

The bias is defined as the difference between a parallel calculation and a reference calculation usually taken as the mean of a sufficient number of serial runs.[2]:

$$\text{bias} = \langle m(\sigma_c)\rangle_p - \langle m(\sigma_c)\rangle_s, \tag{6}$$

where $\langle m(\sigma_c)\rangle_p$ and $\langle m(\sigma_c)\rangle_s$ are the averages of a number of independent runs in parallel and in serial, respectively, for a given total Ising system size. The initial ($m(t = 0) = m_0$) and boundary (periodic) conditions in both cases are identical. In Fig. 2 we show the time evolution of the magnetization of an N=262,144-spin Ising system (2^{18} atoms), averaged over 20 serial runs, used as the reference for the calculation of the bias. The purple shaded region gives the extent of the standard deviation, which is initially very small, when m_0 is very close to one, but grows with time as the system approaches its paramagnetic state and fluctuations are magnified. The shaded region (in gold) between $10^{-4} < t < 5 \times 10^{-3}\lambda^{-1}$ has been chosen for convenience and marks the time interval over which Eq. (6) is solved.[3] The same exercise has been repeated for a 2,097,152-spin (2^{21}) sample with 5 serial runs performed (not shown).

Fig. 3 shows the time evolution of the bias for a number of parallel runs corresponding to the two system sizes studied. The shaded area in the figure corresponds to the interval contained within the standard deviation of the serial case (cf. Fig. 2). Therefore, this analysis yields the maximum number of parallel events that can be considered to obtain a solution statistically equivalent to that given by the serial case.

The figure shows up to what number of parallel processes can the serial and sublattice methods be considered statistically equivalent in the entire range where the bias is calculated. For the 262,144-spin system this is 32, whereas for the 2,097,152 one it is approximately 256. However, runs whose errors are larger than the serial standard deviation at short time scales (*e.g.* $\geqslant$64 and $\geqslant$512 for, respectively, the 262,144 and 2,097,152-spin systems) gradually reduce their bias as time progresses. In fact, at $t \gtrsim 2 \times 10^{-3}\lambda^{-1}$, all parallel runs fall within σ_s. It appears, therefore, that fluctuations play an important role in the parallel runs for low numbers of processes an spkMC cycles. As the accumulated statistics increases (more cycles), this effect gradually disappears. In any event, the bias is never larger than ≈2% for all cases considered here.

Although Fig. 3 provides an informative quantification of the errors introduced by the parallel method, it is also important to separate this systematic bias from the statistical errors associated with each set of independent parallel runs. This is quantified by the standard deviation of the *time-integrated* bias, defined as:

$$\sigma_b = \sqrt{\sigma_p^2 + \sigma_s^2}, \tag{7}$$

[2] If available, an analytical solution may of course be used as a reference as well.

[3] And over where the critical exponent in Section 6 is measured.

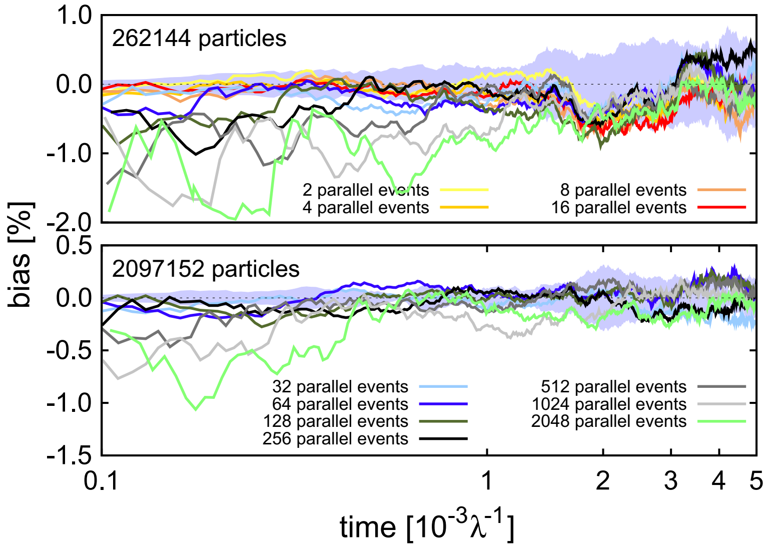

Fig. 3. Time evolution of the parallel bias for two Ising system sizes at the critical point. Results for parallel runs with several numbers of processors are shown. The shaded region corresponds to the interval contained within the standard deviation of the reference (serial) case. Note the logarithmic scale for the abscissa.

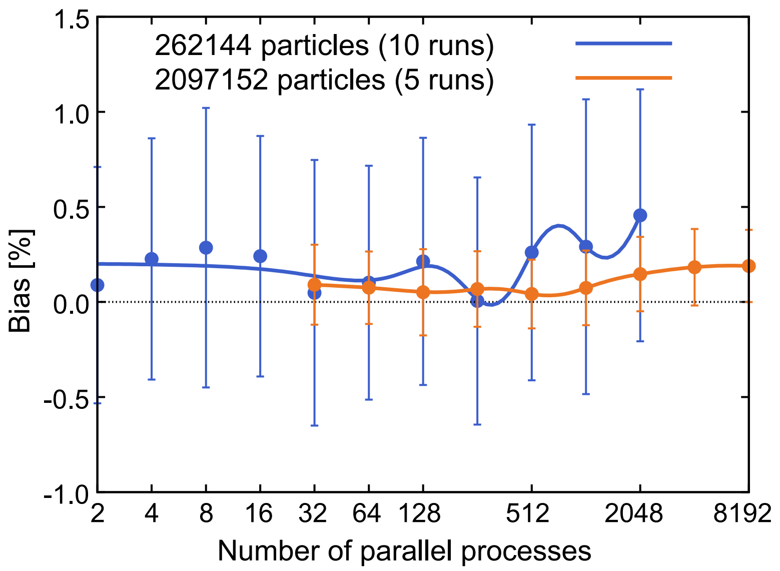

Fig. 4. Systematic parallel bias for 262,144 and 2,097,152-spin 3D Ising systems (obtained, respectively, from 10 and 5 independent runs) as a function of the number of parallel processes. Note that the number of parallel processes is equal to the total number of subcells divided by the number of different sublattices (=2, in our case).

where the terms inside the square root are the parallel and serial variances respectively. We next solve Eqs. (6) and (7) during the time interval prescribed above for the two system sizes considered in Fig. 3. The absolute value of the systematic bias is extracted from a number of independent runs (10 and 5 respectively) and plotted in Fig. 4 as a function of the number of parallel processes. Note that the number of parallel processes is equal to the total number of subcells divided by the number of different sublattices (=2, in our case).

The figure shows that the absolute value of the bias is always smaller than the statistical error (*i.e.* the error bars always encompass zero bias). This implies that, in the range explored, a given problem may be solved in parallel and the result obtained can be considered statistically equivalent to a serial run. The bias is roughly constant and always below 0.5% in the entire range explored for both cases. However, it is consistently lower for the larger system size, as are the error bars. This is simply related to the moderation of fluctuations with system size.

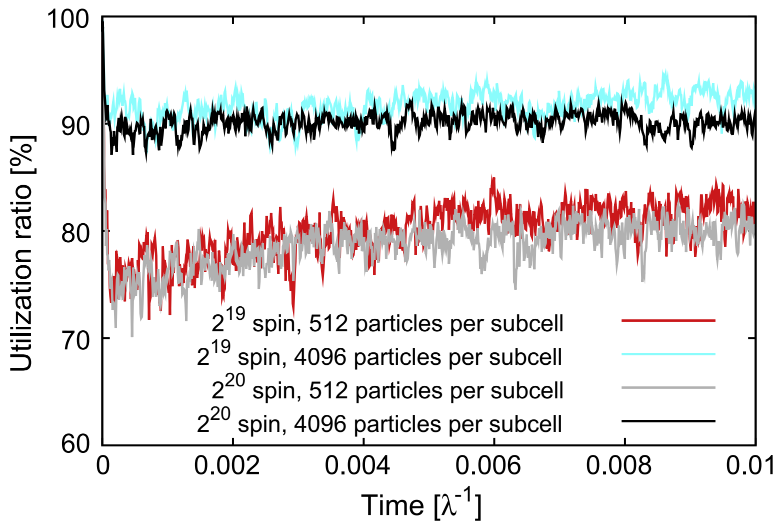

Fig. 5. Evolution of the utilization ratio (UR) with simulated time for 524,288 (2^{19}) and 1,048,576 (2^{20}) particle Ising system. Calculations have been done varying the number of sublattices per processor, which is shown to have a significant impact on the UR.

An analysis such as that shown in Fig. 4 allows the user to control the parallel error by choosing the problem size and the desired number of particles per subcell. Consequently, our method continues to be a controlled approximation in the sense that the error can be intrinsically computed and arbitrarily reduced.

5. Algorithm performance

The algorithm's performance can be assessed via its two fundamental contributions, namely, one that is directly related to the implementation of the minimal process method (MPM) through the null events [26], and the parallel performance *per se*. The effect of the null events is quantified by the utilization ratio (UR):

$$\mathrm{UR} = 1 - \frac{\sum_k r_{0k}}{KR_{max}}, \tag{8}$$

which gives the relative weight of null events on the overall frequency line. The UR determines the true time step gain associated with the implementation of the MPM as [14]:

$$\delta t^* = K \cdot \mathrm{UR} \cdot \delta t_s,$$

where δt^* and δt_s are, respectively, the MPM and standard time steps. This procedure is intrinsically serial, and will result in superlinear scalar behavior if not taken into account for parallel performance purposes. Next, we show in Fig. 5 the evolution of the UR for 524,288 (2^{19}) and 1,048,576 (2^{20}) spin systems. We have done calculations for several numbers of processors and number of particles per subcell. We find that the determining parameter is the latter, *i.e.* for a fixed system size and number of processors used, the UR displays a strong dependence with the number of particles per subcell. The figure shows results for 512 and 4096 particles per subcell, which in the 524,288(1,048,576)-spin system amounts to, respectively, 1024(2048) and 128(256) subcells per processor. In the latter case, the UR eventually oscillates around $\sim$82%, whereas in the former it is approximately 90% (*i.e.* on average, ~18% and 10% of events, respectively, are *null* events).

For its part, the parallel efficiency is defined as the wall clock time employed in a serial calculation relative to the wall clock time of a parallel calculation with K processors involving a K-fold increase in the problem size:

$$\eta = \frac{t_s(1)}{t_p(K)}. \tag{9}$$

The inverse of the efficiency gives the weak-scaling behavior of the algorithm. Due to the absence of fluctuations that exist in other parallel algorithms based on intrinsically asynchronous kinetics (cf., *e.g.*, Ref. [24]), the ideal parallel efficiency of spkMC is always 100%.

Let us now consider the efficiency for the following weak-scaling problem for the case where R_{max} is communicated globally. Assuming that frequency line searches scale linearly with the number of walkers in our system, the serial time expended in simulating a system of N spins to a total time T is:

$$t_s(1) = n_s(t_{exe} + \mathcal{O}(N)),$$

where n_s is the number of cycles required to reach T, and t_{exe} is the *computation* time during each kMC cycle. For its part, the total parallel time for the K-fold system is:

$$t_p(K) = n_p(t_{exe} + \mathcal{O}(N) + t_c),$$

where n_p is the counterpart of n_s and $t_c = t_g + t_l$ is the communications overhead due to global and local calls. In the most general case, the efficiency is then:

$$\eta = \frac{n_s(t_{exe} + \mathcal{O}(N))}{n_p(t_{exe} + \mathcal{O}(N) + t_c)}. \tag{10}$$

As mentioned in the paragraph above, when it is ensured that the serial algorithm also take advantage of the time step gain furnished by the minimal process method, the number of cycles to reach T is the same in both cases, $n_s \equiv n_p$. The parallel efficiency then becomes:

$$\eta = \frac{t_{exe} + \mathcal{O}(N)}{t_{exe} + \mathcal{O}(N) + t_g + t_l}. \tag{11}$$

Next, by virtue of the log P model [27], we assume that the cost of global communications is $\mathcal{O}(\log K)$. The local communication time, t_l, is independent of the number of processors and is proportional to the contact surface between processor subdomains, which scales as $\sim L^{(d-1)}$, where L is the system size and d is the problem dimensionality [28]. For simple cubic domains ($d = 3$), $L \approx N^{\frac{1}{3}}$ and, thus, $t_l \sim \mathcal{O}(N^{\frac{2}{3}})$.

If we consider the execution time t_{exe} negligible compared to the communication time, we have:

$$\eta = \frac{c_0 N}{c_0 N + c_1 \log K + c_2 N^{\frac{2}{3}}} = \frac{1}{1 + (c_1/c_0 N)\log K + \left(c_2/c_0 N^{\frac{1}{3}}\right)}, \tag{12}$$

where c_0, c_1 and c_2 are architecture-dependent constants The final expression then reduces to:

$$\eta = \frac{1}{1 + a\log K + b}, \tag{13}$$

where a and b are architecture and problem dependent constants that represent the relative cost of global an local communications, respectively. Thus, the denominator in Eq. (13) contains the three main contributions to the computational overhead of each Monte Carlo cycle, namely: the cost of linear searches (computation cost) and the cost associated with global and local calls (communication cost). This simple model is not necessarily intended as predictive, but as a basis for understanding the algorithm's performance. We do this in the scaling tests performed next.

The tests have been carried out on LLNL's distributed-memory parallel platforms, specifically the "hera" cluster using Intel compilers [29]. The scalability calculations were all performed for 512 particles per subcell, regardless of the number of processors used, for systems with three different numbers of spins per processor, namely 4,194,304 (2^{22}), 2,097,152 (2^{21}), and 1,048,576 (2^{20}). This means that as the number of particles per processor is increased, more subcells are assigned to each processor. Fig. 6 shows the parallel efficiency of the spkMC algorithm as a function of the number of processors used for three

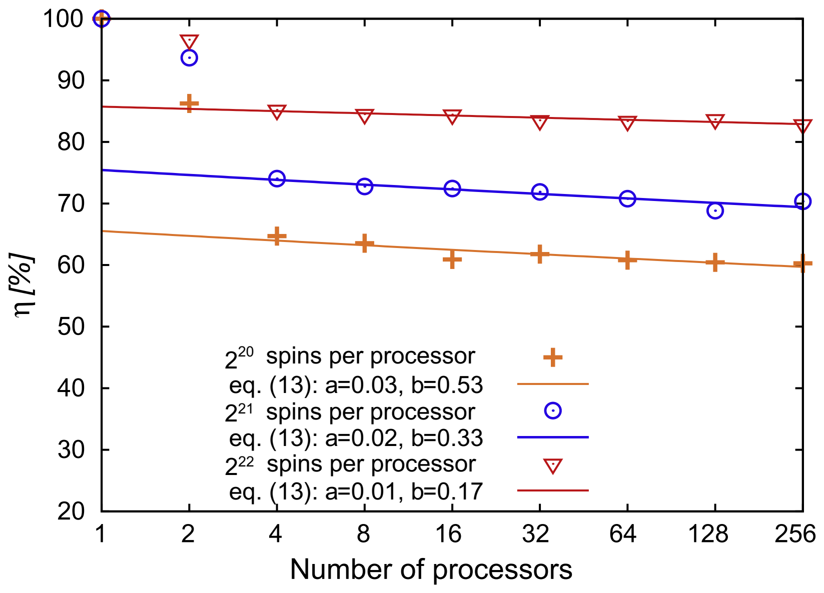

Fig. 6. Parallel efficiency for three different weak-scaling problems, one with 2^{20}, one with 2^{21}, and another with 2^{22} particles per processor of an Ising system at the critical point. The first two points have not been included in the fitting of Eq. (13). The fitting constants a and b are given in each case.

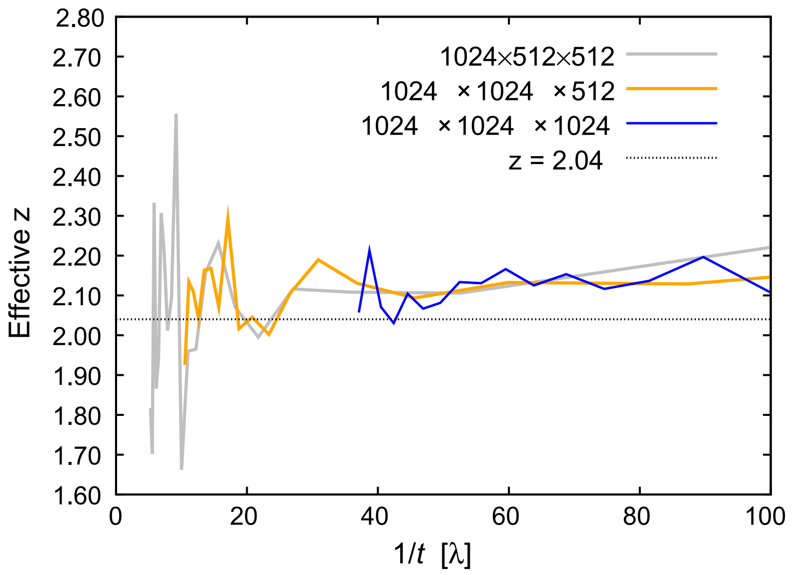

Fig. 7. Effective dynamical critical exponent as a function of inverse time using 1024 processors and 1,048,576 spins per subcell for approximately a quarter (2^{28}), half (2^{29}), and one (2^{30}) billion-spin Ising systems for $t > 0.025\lambda^{-1}$. The horizontal line at $z = 2.04$ marks the consensus value in 3D from the literature.

reference Ising systems at the critical point. The fitting constants a and b are given for each case. As the figure shows, the number of spins per processor has a significant impact on the parallel efficiency, with larger sizes resulting in better performances. The efficiency at $K = 256$ is upwards of 80% for the largest system, and $\approx$ 60% for the smallest system size. The leap in efficiency observed in all cases between 2 and 4 processors is caused by the nodal interconnects (band width) connecting quad cores in the platforms used, and so these two points have not been taken into account for fitting Eq. (13). As expected from Eq. (12), the fitting constant a scales roughly as N^{-1}, while b deviates about 30% from the expected $N^{-\frac{1}{3}}$ scaling. What is clear is that the local communications cost dominates over the $a\log K$ term for any number of processors, an extent that we have confirmed via parallel profiling tests.

Next we study the case where R_{max} is overdimensioned *a priori* to a prescribed tolerance TOL of the *true* value to avoid global calls. In such case, $R^*_{max} \approx R_N(1 + \mathrm{TOL})$ and $t_g = 0$ so that Eq. (10) becomes:

$$\eta = \frac{t_{exe} + \mathcal{O}(N)}{(1 + \mathrm{TOL})(t_{exe} + \mathcal{O}(N) + t_l)}, \tag{14}$$

stemming from the fact that now the ratio $n_s/n_p = \delta t_p/\delta t_s = R_{max}/R^*_{max}$. Assuming again that t_{exe} is negligible with respect to t_l and the linear term for frequency line searches, the expression for the efficiency takes the form:

$$\eta = \frac{cN}{(1 + \mathrm{TOL})(cN + t_l)} = \frac{1}{(1 + \mathrm{TOL})(1 + b)}, \tag{15}$$

where b is the same as in Eq. (13).

Combining Eqs. (13) and (15), we arrive at the criterion to choose the optimum algorithm:

$$\mathrm{TOL} < \frac{a \log K}{1 + b},$$

i.e. the chosen tolerance must be smaller than the ratio between the global communication cost and all the other overheads. As long as the above inequality is satisfied, avoiding global calls by conservatively setting R_{max} at the beginning of the simulation results in a more efficient use of parallel resources. Again, note that, via the constants a and b, this is problem and machine-dependent, and establishing these with confidence may require considerable testing prior to engaging in production runs.

The idea behind using a tolerance to minimize or contain global calls forms the basis of the so-called *optimistic* algorithms [30,31], where the parameter (s) controlling the parallel evolution of the simulation are set conservatively—either by a self-learning procedure or by accepting some degree of error—and monitored sparingly. For example, for the 2^{20}-spin Ising system, $a = 0.03$, $b = 0.53$, TOL varies between < 0.6 and < 4.7% in the range $2 < K < 256$. These values may not be sufficient to encompass the time fluctuations in R_{max}, but it is expected that for a higher number of processors the efficiency will improve. In any case, as the cost of global calls in our method is small compared to that of local calls, an optimistic implementation of our algorithm may not be warranted. These and more aspects about the parallel efficiency and its behavior will be discussed in Section 7.

6. Application: billion-atom Ising systems at the critical point

We now apply the method to study the time relaxation of large Ising systems near the critical point. As we advanced in Section 2, at the critical point, the relaxation time τ diverges as ξ^z, where $\xi \propto |T - T_c|^{-\nu}$. The scaling at $T = T_c$ is then:

$$m(t) \propto t^{-\kappa/z\nu}. \tag{16}$$

In 3D, we use the known critical temperature $J/kT_c = 0.2216546$ [19,20] to find z. We start with all spins +1 and let $m(t)$ decay from its initial value of unity down to zero. At each time point, we can find the critical exponent z from:

$$z = -\frac{\kappa}{\nu}\left[\frac{d(\log m)}{d(\log t)}\right]^{-1}, \tag{17}$$

where the ratio κ/ν takes the known value of 0.515 [18]. We have carried out simulations with lattices containing $1024 \times 512 \times 512\,(2^{28})$, $1024 \times 1024 \times 512\,(2^{29})$, and $1024 \times 1024 \times 1024\,(2^{30})$ spins. The results are shown in Fig. 7 for critical exponents calculated during $t > 0.025\lambda^{-1}$, from time derivatives averaged over 300 to 500 timesteps.

At long time scales, the critical exponent oscillates around values that range from, roughly, 2.06 to 2.10 depending on system size. This in good agreement with the converged consensus value of ~2.04 published in the literature [32] (shown for reference in Fig. 7). However, as time increases, the oscillations increase their amplitude with inverse system size. Oscillations of this nature also appear in multi-spin calculations, both for smaller [33–35] and larger [36] systems, where the inverse proportionality with system size is also observed. These may be caused by insufficient statistics due to size limitations, as we have shown that, under the conditions chosen for the simulations, our calculations are statistically equivalent to serial ones (cf. Fig. 4). The effect of the system size is also clearly manifested in the relative convergence rate of z. As size increases, convergence to the expected value of 2.04 is achieved on much shorter time scales, *i.e.* fewer kMC cycles.

7. Discussion

We now discuss the main characteristics of our method. We start by considering the three factors that affect the performance of our algorithm:

(i) *Number of particles per subcell*. This is the most important variable affecting the algorithm's performance, as it controls the intrinsic parallel bias and the utilization ratio. Higher numbers of spins per subcell both reduce the bias (cf. Figs. 3 and 4) and increase the UR (Fig. 5), bolstering performance. However, this also results in an increase of the value of R_{max}, which causes a reduction in δt_p. Thus, decreasing the bias and increasing the time step are actions that may work in opposite directions in terms of performance, and a balance between both should be found for each class of problems. For the large Ising systems run here we have used an optimal value of 512 particles per subcell.

(ii) *Number of particles per processor*. This parameter affects the parallel efficiency via the number of spins per processor N_k (for regular space decompositions, $KN_k \approx N$). As N_k increases, a significant improvement is observed. This is directly related to an increase of the computation-to-communications ratio, symbolized by decreases in a and b in Eq. (13).

(iii) *Total system size*. As Figs. 3 and 4 show, for a given sublattice decomposition, a larger system incurs in smaller relative fluctuations in the magnetization, which results in a more contained bias.

Through the constants a and b, the parallel efficiency strongly depends on the latency and bandwidth of the communication network used. An advantageous feature of our method is that, by construction, local calls are the dominant contribution to the asymptotic scaling behavior. This is important because it makes our method ideally scalable at a level only dictated by the system size. Exploring other efficiency-increasing alternatives based on controlling the cost of global calls may result in some efficiency gain, but only marginal and for low values of K. Prescribing a tolerance on the expected fluctuations of R_{max} is one such alternative which is in the spirit of so-called optimistic kMC methods. Ideally its value is set by way of a self-learning procedure that maximizes the efficiency.

In any case, the intersection of items (i), (ii) and (iii) above configures the *operational* space that determines the class of problems that our method is best suited for: large (multimillion) systems, with preferably a sublattice division that achieves an optimum compromise between time step gain and lowest possible bias, with the maximum possible number of particles per processor. These are precisely the conditions under which we have simulated critical dynamics of 3D Ising systems, with very good results. We conclude that spkMC is best designed to study this class of dynamic problems where fluctuations are important and there is an unequivocal size scaling. This includes applications on many other areas of physics, such as crystal growth, irradiation damage, plasticity, biological systems, etc., although other difficulties due to the distinctiveness of each problem may arise that may not be directly treatable with the algorithm presented here. We note that, because the parallel bias is seen to saturate for large numbers of parallel processes, and the efficiency is governed mostly by the local communications overhead, which does not depend on K, the only limitation to using spkMC is given by the number of available processors.

8. Summary

We have developed an extension of the synchronous parallel kMC algorithm presented in Ref. [14] to discrete lattices. We use the chess sublattice technique to resolve boundary conflicts, and have quantified the resulting spatial bias. The algorithm displays a robust scaling, governed by the cost of local communications as well as by the spatial decomposition adopted. We have applied the method to multimillion-atom three-dimensional Ising systems close to the critical point, with very good agreement with published state-of-the-art results.

Acknowledgments

We thank M. H. Kalos for his invaluable guidance, suggestions, and inspiration, as this work would not have been possible without him. This work performed under the auspices of the US Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52–07NA27344. E. M. acknowledges support from the Spanish Ministry of Science and Education under the ‘‘Juan de la Cierva’’ programme.

References

[1] M.H. Kalos, P.A. Whitlock, Monte Carlo Methods, John Wiley& Sons, New York, 1986.
[2] K.A. Fichthorn, W.H. Weinberg, Journal of Chemical Physics 95 (1991) 1090.
[3] A.F. Voter, Introduction to the kinetic Monte Carlo method, in: K.E. Sickafus, E.A. Kotomin, B.P. Uberuaga (Eds.), Radiation Effects in Solids, Springer, NATO Publishing Unit, Dordrecht, The Netherlands, 2006, pp. 1–24.
[4] D.R. Cox, H.D. Miller, in: The Theory of Stochastic Processes, Methuen, London, 1965, p. 6.
[5] A.B. Bortz, M.H. Kalos, J.L. Lebowitz, Journal of Computational Physics 17 (1975) 10.
[6] M.A. Snyder, A. Chatterjee, D.G. Vlachos, Computers & Chemical Engineering 29 (2005) 701.
[7] A. Chatterjee, D.G. Vlachos, M.A. Katsoulakis, International Journal for Multiscale Computational Engineering 3 (2005) 135.
[8] T. Oppelstrup, V.V. Bulatov, G.H. Gilmer, M.H. Kalos, B. Sadigh, Physical Review Letters 97 (2006) 230602.
[9] A. Chatterjee, D.G. Vlachos, Journal of Computer-Aided Materials Design 14 (2007) 253.
[10] B.D. Lubachevsky, Journal of Computational Physics 75 (1987) 1099.
[11] S.G. Eick, A.G. Greenberg, B.D. Lubachevsky, A. Weiss, ACM Transactions on Modelling and Computation Simulation 3 (1993) 287.
[12] Y. Shim, J.G. Amar, Journal of Computational Physics 212 (2006) 305.
[13] G. Nandipati, Y. Shim, J.G. Amar, A. Karim, A. Kara, T.S. Rahman, O. Trushin, Journal of Physics: condensed Matter 21 (2009) 084214.
[14] E. Martinez, J. Marian, M.H. Kalos, J.M. Perlado, Journal of Computational Physics 227 (2008) 3804.
[15] J.R. Glauber, Journal of Mathematical Physics 4 (1963) 294.
[16] M.E.J. Newman, G.T. Barkema, Monte Carlo Methods in Statistical Physics, Oxford University Press, 1999.
[17] M. Hasenbusch, International Journal of Modern Physics C 12 (2001) 911.
[18] P.E. Berche, C. Chatelain, B. Berche, W. Janke, European Physical Journal 38 (2004) 463.
[19] C.F. Baillie, R. Gupta, K.A. Hawick, G.S. Pawley, Physical Review B 45 (1992) 10438.
[20] M.A. Novotny, A.K. Kolakowska, G. Korniss, AIP Conference Proceedings 690 (2003) 241.
[21] D.W. Heermann, A.N. Burkitt, Parallel Algorithms in Computational Science, Springer, Berlin, 1991.
[22] P.M.C. Oliveira, T.J.P. Penna, S.M.M. de Oliveira, R.M. Zorzenon, Journal of Physics A: Mathematical and General 24 (1991) 219.
[23] G. Parisi, J.J. Ruiz-Lorenzo, D.A. Stariolo, Journal of Physics A: Mathematical and General 31 (1998) 4657.
[24] Y. Shim, J.G. Amar, Physical Review B 71 (2005) 115436.
[25] K. Appel, W. Haken, J. Koch, Illinois Journal of Mathematics 21 (1977) 439.
[26] P. Hanusse, A. Blanche, Journal of Chemical Physics 74 (1981) 6148.
[27] D.E. Culler et al, Communications of the ACM 39 (1996) 78.
[28] E. Schwabe, V.E. Taylor, M. Hribar, An in-depth analysis of the communication costs of parallel finite element applications, Technical Report CSE-95-005, Northwestern University, EECS Department, 1995 (<http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.31.5316>).
[29] LLNL's ‘‘Hera’’ cluster (<https://computing.llnl.gov/tutorials/linux_clusters/#SystemsOCF>), on which all kMC simulations were performed, uses CHAOS Linux as O/S and runs version 1 of the MPI libraries, with two-sided communications.
[30] M. Merrick, K.A. Fichthorn, Physical Review E 75 (2007) 011606.
[31] A. Kara, O. Trushin, H. Yildirim, T.S. Rahman, Journal of Physical: Condensed Matter 21 (2009) 0842139.
[32] D. Ivaneyko, J. Ilnytskyi, B. Berche, Y. Holovatch, Physica A 370 (2006) 163.
[33] R. Matz, D.L. Hunter, N. Jan, Journal of Statistical Physics 74 (1994) 903.
[34] P. Grassberger, Physica A 214 (1995) 547.
[35] B. Zheng, Physica A 283 (2000) 80.
[36] D. Stauffer, Physica A 244 (1997) 344.