Computer Physics Communications 185 (2014) 2479–2486

Contents lists available at ScienceDirect

Computer Physics Communications

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

A rigorous sequential update strategy for parallel kinetic Monte Carlo simulation

Jerome P. Nilmeier [*], Jaime Marian

Physical and Life Sciences Directorate, Lawrence Livermore National Laboratory, 7000 East Avenue, Livermore, CA 94550, United States

ARTICLE INFO

Article history:
Received 18 March 2014
Accepted 7 May 2014
Available online 10 June 2014

Keywords:
Kinetic Monte Carlo
Sequential updates
Parallel computing algorithms
Stochastic simulation

ABSTRACT

The kinetic Monte Carlo (kMC) method is used in many scientific fields in applications involving rare-event transitions. Due to its discrete stochastic nature, efforts to parallelize kMC approaches often produce unbalanced time evolutions requiring complex implementations to ensure correct statistics. In the context of parallel kMC, the sequential update technique has shown promise by generating high quality distributions with high relative efficiencies for short-range systems. In this work, we provide an extension of the sequential update method in a parallel context that rigorously obeys detailed balance, which guarantees exact equilibrium statistics for all parallelization settings. Our approach also preserves nonequilibrium dynamics with minimal error for many parallelization settings, and can be used to achieve highly precise sampling.

1. Introduction

Since its development in the 1970s, the kinetic Monte Carlo (kMC) method [1–3] has enjoyed wide popularity, and has been applied to problems far beyond what it was initially designed to model. The kMC approach belongs to a general class of methods known as stochastic *discrete event simulators* [4], which have also attracted much attention and have been used in numerous applications. These simulation techniques are mesoscale by design, as the inputs are often *propensities* – or probabilities per unit time – that are extracted from either simulations, measurements, or both, at microscopic scales. Due to the fact that it is an event driven algorithm, kMC has the potential of vastly extending the accessible timescales of its continuous-time counterparts.

As the demand for simulations of larger system sizes increases, many questions about how best to parallelize these methods remain. The main difficulty arises from the fact that standard discrete time step approaches to the parallelization of deterministic differential equations [5] and molecular dynamics integrators [6] are not directly applicable due to the discrete and stochastic nature of time evolution. Significant progress towards the formulation of a Trotter decomposition has been achieved, however [7], and the subject remains an active area of research.

By far, the dominant paradigm in parallel kMC is the asynchronous approach, working from the idea that parallel processes are run simultaneously, with intermittent bookkeeping to recover either exact or nearly exact statistics. The classic set of rigorous and semi-rigorous approaches proposed by Amar and Shim [8,9], and other derivative algorithms [10], remain as the reference for parallel kMC simulators. These designs can be highly efficient, but the asynchronous strategy gives rise to rough *virtual time horizons*, since each process advances by an independent, stochastically varying time clock. This effect was first noted and addressed by Korniss et al. [11–13]. Martinez et al. [14,15] proposed an elegant solution to the time horizon problem, resulting in a synchronous approach. They build on a null event formulation found in the discrete event community, while also developing a controlled approximation to the master equation for the parallelized process.

In recent years, a number of researchers have developed the notion of *sequential updates* in the context of single process simulations [16–19]. This formulation has been rigorously shown to produce equilibrium distributions by obeying a balance condition, but has not been applied to parallelization contexts. One advantage to the sequential approach in a parallel simulation is that the time steps advance sequentially with each process, and there is no need to worry about time synchronicity across all the processes. A recent sequential approach for parallel simulations has been proposed by Arampatzis, et al. [7]. Since the sequential method can limit parallel efficiency, considerable care is given as to how to treat noninteracting processes simultaneously, making a convincing case that

[*] Corresponding author. Tel.: +1 9252925321.
E-mail addresses: nilmeier1@llnl.gov, jnilmeier@yahoo.com (J.P. Nilmeier), marian1@llnl.gov (J. Marian).

http://dx.doi.org/10.1016/j.cpc.2014.05.024
0010-4655/© 2014 Elsevier B.V. All rights reserved.

this approach can also be efficient. In the SPPARKS simulation suite of Plimpton et al. [20], an efficient implementation that simultaneously updates noninteracting processes is also used.

In this work, we propose to further develop the sequential update paradigm as a parallelization strategy. Here, however, we propose a procedure that obeys detailed balance, and also show that we can recover a very good approximation to the time response, laying the foundation for more detailed treatments in the future. The basis of our approach lies in defining a procedure for generating sequential update schedules in such a way that detailed balance is assured for the endpoints of the schedules. We also discuss on how to use the statistics of the endpoints of each process within the schedule. For the present work, we discuss only the theoretical and algorithmic aspects of the parallel protocol, and will cover implementation and performance related issues in a future work.

This paper is organized as follows. We begin with a theoretical overview of the kMC method and sequential updates. We then formulate the sequential update strategy that preserves detailed balance in the context of parallel simulations. We include a discussion on collecting statistics after each parallel process has run during a schedule sweep. The method is then tested and applied to Ising systems of increasing complexity in Section 3. We conclude with a brief discussion of the results obtained and the conclusions.

2. Theory: Sequential updates with kinetic Monte Carlo

2.1. The master equation and kinetic Monte Carlo

To begin the discussion, we express our propagation strategy as a master equation. For kinetic Monte Carlo, it is sometimes more convenient to work with the Chapman–Kolmogorov form, as it contains the transition kernel explicitly in the expression. The Chapman–Kolmogorov form [21] of the master equation for a Markov system is

$$T^N p(\sigma; s) = p(\sigma; s+N), \tag{1}$$

where the transition kernel T is expressed in left stochastic form [21], and $p(\sigma; s)$ is the time dependent probability vector for the configuration vector σ at integer time state s. The system is a Markov process, and we advance the system by N steps by applying the T matrix to p an integer number N times. For the present work, the vector of configurations σ is the vector of all 2^{N_S} possible spin states where N_S is the number of spins in a model Ising system, as described in Appendix A. The standard detailed balance for a single step can be expressed as $\pi_i T_{ji} = \pi_j T_{ij}$, where π_i is the equilibrium probability of occupying the ith configuration (spin) state, such that $p(\sigma_i; s=\infty) = \pi_i$. For the cases presented in this work, each time step follows Glauber dynamics [22], as defined in Eq. (A.3).

The time step in Eq. (1) can be advanced by using a Poisson variate for a procedure consisting of N steps, or

$$p_p(\Delta t(N)) = \frac{1}{\tau_S}\frac{\Delta t^{N-1}}{(N-1)!}e^{-\Delta t/\tau_S}. \tag{2}$$

The expectation value of a variate drawn from the distribution in Eq. (2) is $\langle \Delta t(N)\rangle = N\tau_S$, where τ_S is the time scale of the system. For purposes of this work, the time step is advanced only by the average value in order to simplify the analysis. Both approaches give equivalent statistics, however. For n-fold way simulations [1,2], the timescale is computed as the residence time, or the inverse total frequency $\tau_S = 1/R$, where R is the sum of all possible transition rates. For the present work, we regard τ_S as the intrinsic timescale of the simulation. The Poisson variate for an N step process is readily obtained by computing the negative logarithm of N uniform variates and summing them. Using either the Poisson variate or the expectation value, we can advance the time clock as

$$T^N p(\sigma; t) = p(\sigma; t+\Delta t(N)), \tag{3}$$

to generate the time dependent solution to the master equation. For all the cases in this work, the time step is advanced by the expectation value.

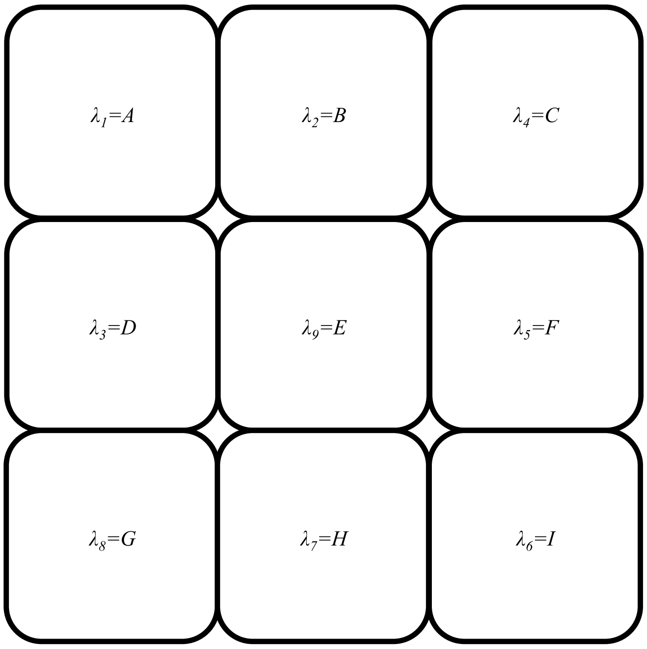

Fig. 1. Example 2D configuration partitioned into $N_D = 9$ domains with a 'checkerboard' pattern and schedule $\Lambda = \{A, B, D, C, F, I, H, G, E\}$. Each domain runs for a fixed number of N_I independent steps with neighboring (or interacting) domain processes held fixed. At the end of each sweep, a new schedule is generated randomly. A vertically striped partitioning would treat the combined domains (A, B, C), (B, E, H), and (C, F, I) each as a single process, resulting in $N_D = 3$.

2.2. Construction of sequential strategy that obeys detailed balance

Here we develop a procedure based on the work of Deem et al. [23], and also Orkoulas et al. [17,19,18], who developed a theory for sequential updates and showed that exact equilibrium distributions can be obtained. The primary motivation for using sequential updates in these works was to accelerate convergence of equilibrium simulations. For the present work, we wish to develop the sequential update procedure as a parallelization strategy, following the ideas introduced by Shim and Amar [8] and Arampatzis et al. [7]. Since we wish to have a parallelization strategy suitable for studying nonequilibrium and dynamical properties, the goal here is to develop a procedure that preserves the dynamic character of native, unparallelized simulations, rather than to have rapid convergence properties. Our procedure can be regarded as an advance in that it introduces a sequential update strategy obeying detailed balance in a parallelization context.

Consider a configuration space that is partitioned into domains, such as that shown in Fig. 1. Each domain is of equal size, and the domain partitioning is held fixed for the duration of the simulation. The general procedure of sequential updating requires the simulation of a single domain process for a number N_I of independent time steps while holding the neighboring domains at a fixed coordinate state. For each process of length N_I, data is first collected from the fixed state of the neighboring processes. For distributed data parallelizations, this data from neighboring