UCRL-JRNL-226852

Synchronous Parallel Kinetic Monte Carlo

E. Martínez, J. Marian, M. H. Kalos

December 16, 2006

Journal of Computational Physics

Synchronous Parallel Kinetic Monte Carlo

E. Martínez, J. Marian and M. H. Kalos

Lawrence Livermore National Laboratory
Livermore, CA 94551

Abstract

A novel parallel kinetic Monte Carlo (kMC) algorithm formulated on the basis of perfect time synchronicity is presented. The algorithm provides an exact generalization of any standard serial kMC model and is trivially implemented in parallel architectures. We demonstrate the mathematical validity and parallel performance of the method by solving several well-understood problems in diffusion.

Key words: kinetic Monte Carlo, parallel computing, diffusion, scalability
PACS: 02.70.-c, 02.70.Uu, 61.72.Ss

1 Introduction

Kinetic Monte Carlo (kMC) is an extremely powerful method to simulate the time evolution of Markovian processes [1,2]. kMC relies on the *a priori* knowledge of a given set of transition rates characterizing the simulated processes, which are assumed to obey Poisson statistics. The scope of applications for kMC is extraordinarily wide, ranging from epidemiology and population kinetics to surface growth or radiation damage. Because of its versatility, ease of implementation, and wide range of applications, kMC is a prime candidate for parallelization with the recent advent of tera- and peta-scale computing capabilities. However, the difficulty of parallel kinetic Monte Carlo (pkMC) simulations lies in the intrinsic asynchronicity underlying discrete events, each one characterized by a different rate. In mathematical terms, a necessary condition that must be satisfied by any parallel kMC algorithm is that it be rigorous, *i.e.* that it solve the same master equation as the sequential (serial) method. This does not necessarily imply that both approaches give the same sequence of events, but that, on average, both result in the same statistical distributions, and give the same kinetic evolution.

Early attempts to use parallel algorithms achieved some speedup but failed to satisfy this provision [3,4]. To date, most approaches that do ensure this compatibility between sequential and parallel processes rely on asynchronous kinetics, with different processors simulating events independently and then accepting or rejecting them on the basis of domain correlation schemes that may severely limit the computational efficiency [5–9].

Most of the recent work in this area has been inspired by Lubachevsky's original paper [5], which provides an exact parallel algorithm that simulates discrete-event kinetics. This class of algorithms attempts to advance a 'virtual time horizon' (VTH) asynchronously by a combination of kMC steps whose progression is controlled by relatively cumbersome acceptance/rejection techniques. The progress rate of the simulation depends on the density of local minima of the instantaneous VTH, which in turn depends on the relative occurrence of event roll-backs across domain boundaries. Depending on the problem at hand, VTHs can display a strongly fluctuating behavior, for which ingenious roughness-suppressing algorithms have been proposed [6,7]. Another interesting alternative for parallel discrete event simulations is Jefferson's *timewarp* algorithm [9]. The timewarp paradigm provides a protocol for minimizing the number of conservative synchronization updates by ignoring causality errors, which are later detected and retraced for their resolution. Nevertheless, owing to their intrinsic implementation complexity, limited use has been made of these classes of methods, and the development and application of rigorous efficient parallel algorithms for kMC simulations remains a significant challenge.

In this article we propose an exact, synchronous, parallel generalization of the rejection-free n-fold kMC method of Bortz, Kalos and Lebowitz [10] (herein referred to as BKL for brevity). Our algorithm ensures a flat VTH construction, thereby rendering all communications between domains essentially trivial and suppressing the need for roll-backs. This results in an ease of implementation not achieved for earlier parallel algorithms. The paper comprises two main sections: first we describe the algorithm in detail and discuss its potential parallel performance; secondly, we demonstrate correctness and scalability by solving several well-understood diffusion problems.

In BKL, a system with N walkers, each with rate r_i ($i = 1 \ldots N$), is evolved in time by randomly selecting an event with probability r_i/R_{tot}, where $R_{tot} = \sum_i^N r_i$ is what we hereafter term the *frequency line*, *i.e.* the aggregate of all the individual r_i. The system is then advanced in time by randomly sampling from exponential distribution $\exp\left(-R_{tot}\delta t_{BKL}\right)$.

2 Parallel kMC algorithm

The algorithm is based on the minimal process method as originally proposed by Hanusse and Blanché [11,12]. Our procedure goes as follows. The computational cell is divided into K arbitrary subdomains Ω_k (where, for consistency with the parallel computing literature, K is the number of *processing units*). A parallel kMC step consists of the following:

(1) A frequency line is constructed for each Ω_k as the aggregate of the individual rates, r_{ik}, of all the walkers located within each subdomain:

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

where n_k and R_k are, respectively, the number of walkers 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 choose the maximum rate, R_{max} as:

$$R_{max} = \max_{k=1,\ldots K} \{R_k\}.$$

(3) We assign a *null* event with rate r_{k0} to each frequency line in each subdomain k such that:

$$r_{k0} = R_{max} - R_k,$$

where, in general, the r_{k0} will all be different. Of course in Ω_α, $\alpha \in \{k\} \mid R_\alpha \equiv R_{max}$, $r_{\alpha 0} = 0$, and there is no possibility of null events.

(4) In each Ω_k an event to be carried out is sampled with probability $p_{ik} = r_{ik}/R_{max}$, including null events chosen with $p_{k0} = r_{k0}/R_{max}$. For this step, we must ensure that independent sequences of random numbers be produced for each K, using an appropriate parallel random number generator.

(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 it becomes a global time step for all of the parallel processes.

The imposition of fixed length for the frequency lines in all Ω_k processes guarantees exact synchronicity. This is a key aspect of our algorithm. Time is advanced exactly the same amount in all processors, which enables direct communication across domain boundaries trivially, without the need for sophisticated rejection minimization schemes across boundaries commonly found in other parallel (asynchronous) methods (cf. Ref. [13]). However, note that, because in principle the spatial decomposition may be arbitrary (*i.e.* it does not affect the global kinetics), optimal efficiency is not guaranteed *per se.*

Evidently, an optimum decomposition will be that which renders $\left\{\sum_k r_{k0}\right\}$ minimum, but the solution is not unique (in fact it need not even be 'spatial') and the catalog of options is quite varied. For example, in his proposed improvement of the original minimal process method, ben-Avraham chose a cell-coarsening scheme that preserves the half-life of the particle concentration [12]. Our method of choice is to perform a domain decomposition using the method of orthogonal recursive bisection (ORB) [14] so as to conserve the total accumulated rate after each recursive partition. In the ideal limit of numbers of walkers that are an exact multiple of K, with equal rates, such decomposition produces perfectly scaling gain by imposing $\{r_{k0} = 0, \forall\ k\}$. In this sense, the *utilization ratio* (UR), defined as the probability that a 'real'– rather than 'null'– event will occur in each processing unit, is $\left(1 - \frac{1}{KR_{max}}\sum_k r_{k0}\right)$.

As interdomain migration occurs, however, the $\{r_{k0}\}$ must be recomputed to continue ensuring synchronicity. In other words, step 6 of our algorithm consists of the following conditional block:

```
(6)
                    if COMM, then go to step 1
                    else, go to step 4
```

where the condition COMM implies communication among processing units.

The intrinsic parallel efficiency of the method is governed by the utilization ratio. A domain decomposition (or any other distributed decomposition) that prescribes $\{r_{k0} = 0, \forall\ k\}$ will yield ideal parallel scaling (UR=100%). Under these conditions, $R_{max} = R_{tot}/K$ and, hence, on average $\delta t_p = K\delta t_{BKL}$. UR=100% is the theoretical speedup limit and acts as an upper bound to the time step gain. Of course, generally, for discrete systems with non-integer rates r_i, UR$\leqslant$ 100%, $R_{max} \geqslant R_{tot}/K$, and $\delta t_p \leqslant K\delta t_{BKL}$.

Steps (1) to (6) above provide a rigorous, synchronous, parallel algorithm in closed form. So far, no numerical arguments have been made as regards the computational efficiency of the method. However, in the event that the time evolution of the density profile results in a spatial redistribution of particles that deviates from the original optimum decomposition, the utilization ratio may drop enough to lead to poor parallel performance. The metrics chosen to establish reasonable tolerance limits on UR are typically problem-dependent (*e.g.* diffusion coefficients, cell sizes, etc). In general, when this occurs, the domain decomposition must be updated, either by performing some type of dynamic load balancing, or by generating a new decomposition (such as a global ORB). Irrespective of the method chosen, this step can be integrated into our kMC algorithm in the form of a closed loop between steps 3 and 4:

```
    if UR < TOL, then
        update domain decomposition (perform ORB)
        go to 3
```

where TOL is a problem-dependent tolerance. It is worth stressing that this is an optional modification that does not detract from the generality of our algorithm, since correct kinetics arise independent of the domain partition scheme chosen. Refreshing the domain decomposition is aimed simply at optimizing the parallel performance.

3 Applications

We now turn to the validation of our algorithm by comparing with some simple cases of known analytical solution. First, we study pure diffusion without volumetric terms, *i.e.* according to the following master equation:

$$\Big[-D\nabla^2+\frac{\partial}{\partial t}\Big]\rho(\mathbf{x};t)=0 \tag{1}$$

where $\rho(\mathbf{x};t)$ is the time and space-dependent particle number density and D is the diffusion coefficient. We study diffusion in an n-dimensional square domain of side a, Ω_a, subject to the following boundary conditions:

(i) Absorbing ('black box') boundary conditions:

$$\rho(\mathbf{x};0)=\rho_0\prod_{\alpha=1}^{n}\cos\left(\frac{\pi x_\alpha}{a}\right),\mathbf{x}\in\Omega_a$$
$$\rho(\mathbf{x};t)=0,\mathbf{x}\in\partial\Omega_a$$

where ρ_0 is a constant.

(ii) Periodic boundary conditions (PBC):

$$\rho(\mathbf{x};0)=\rho_0\prod_{\alpha=1}^{n}\left[\frac{1}{a}-\cos\left(\frac{2\pi x_\alpha}{a}\right)\right],\mathbf{x}\in\Omega_a$$
$$\rho(x_1,x_2,\ldots,x_\alpha,\ldots,x_n;t)$$
$$=\rho(x_1,x_2,\ldots,x_\alpha\pm a,\ldots,x_n;t),\ \forall\ \alpha$$

Here we focus on the two-dimensional (2D) case. In both cases (i) and (ii) the solution of the diffusion equation is given by the time dependent Green's function for an infinite medium with diffusion constant D:

$$G(\mathbf{x},\mathbf{x}_0;t)=\frac{e^{-\frac{(\mathbf{x}-\mathbf{x}_0)^2}{2\sigma^2}}}{\sqrt{4\pi Dt}} \tag{2}$$

where $\mathbf{x}$ and $\mathbf{x}_0$ are the initial and final position of each walker, t is the time, and $\sigma^2 = 2Dt$ is the mean square displacement. For a fixed D, the mean square displacement must be conserved in all Ω_k, from which it follows that, for each walker i, $t_i = \delta t_p R_{max}/r_i$. Both of these cases are eigenvalue problems [15] with known eigenvalues of (i) $\lambda_{abs} = \frac{\pi}{a}\sqrt{2D}$, and (ii) $\lambda_{\text{PBC}} = \frac{\pi}{a}\sqrt{4D}$. Figure 1 shows the comparison between the analytical solution and our parallel algorithm for case (i) with $a = 1$ cm, $D_i = 1$ cm^2·s^{-1}, and $\rho_0 = 131072$ walkers. For these values, $\lambda_{abs} = 4.443$ s$^{-\frac{1}{2}}$, while in a series of runs with $K = 2^n$ ($n = 1, \ldots, 7$) processors we obtained an average value of 4.410 ± 0.042 s$^{-\frac{1}{2}}$. For case (ii) with the same parameters, we show in Fig. 2 the time evolution

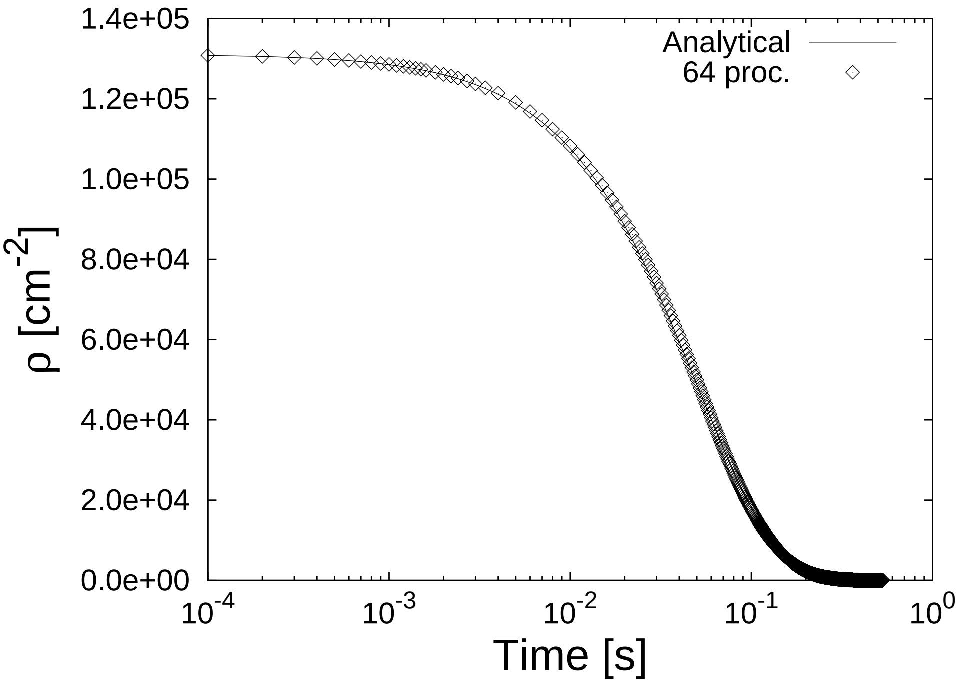

Fig. 1. Comparison between the analytical solution (continuous line) and a parallel run with 64 processors (open diamonds) for the problem of simple diffusion with no particle interactions and absorbing boundaries.

with eigenvalue $\lambda_{\text{PBC}} = 6.28$ s$^{-\frac{1}{2}}$ of the spatial particle distribution (ii). The plot contains the projection of $\rho(\mathbf{x};t)$ on one of the box dimensions at four different times. Note that, for $a = 1$ cm and $D = 1$ cm^2·s^{-1}, the exponent of the time dependent terms is $\sim$40 so that the convergence to ρ_0 is very fast. We obtain $\langle \lambda_{\text{PBC}} \rangle = 6.27$ and an average error of $\pm 4.4\%$ with respect to the analytical value of 6.28 s$^{-\frac{1}{2}}$. For all other parallel runs performed the values were of the same order of magnitude. In this particular case (ii) the initial particle density evolves with time towards a more flattened profile. Thus, the utilization ratio derived from the initial ORB will gradually worsen as walkers diffuse and the mapping between the spatial particle distribution and the initial domain decomposition degrades. For the specific simulation shown in Fig. 2, the UR decreases from its initial value of 97.2% to a steady state value of

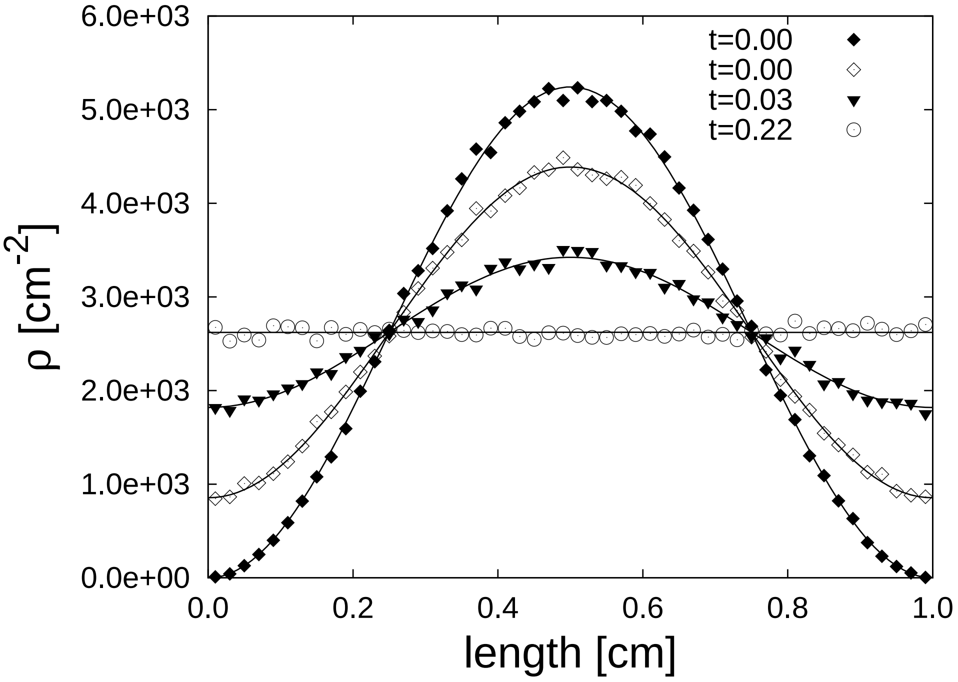

Fig. 2. Comparison between pkMC for 16 processors and the analytical solution of the time evolution of the spatial density profile for the case (ii) of diffusion without particle interactions and periodic boundary conditions.

$\sim$77.5% after homogenization has completed. Figure 3 illustrates the temporal variation of the UR for this case, compared with the case of a flat particle density profile using the same number of processors. While both cases start out at UR$\approx$97%, the domain decomposition that maps the initial sinusoidal particle distribution in (ii) becomes gradually unsatisfactory, resulting in a steady-state UR of $\sim$77.5%, compared to a value of 96.8% for the flat density profile. Although these results, which are perfectly satisfactory, have been obtained for a single ORB, as noted above, nothing precludes carrying out subsequent ORBs to improve the efficiency when the value of UR drops below some problem-specific (arbitrary) tolerance.

Next we turn to the study of cases where particles interact. In particular, we consider the multiparticle reaction $NA \rightarrow 0$ (where N is the number of reacting particles) and the two-species annihilation $A+B \rightarrow 0$. Figure 4 shows the time evolution in 2D of an ensemble of 32768 A-type walkers with $a = 1$ cm, $D = 1$ cm^2·s^{-1}, and r_c, the particle interaction radius, equal to 10^{-5} cm. For an arbitrary value of N the appropriate asymptotic decay is $t^{-\frac{1}{N-1}}$ [16]. Here we have chosen r_c small enough to minimize the number of interactions for which $N > 2$, *i.e.* $\rho(t)$ is expected to scale approximately as $1/t$, which is equivalent to the biparticle ($A + A \rightarrow 0$) annihilation time decay. The figure shows results for 64 processors and a single-CPU BKL run, with excellent agreement between both calculations. Also shown is the $1/t$ asymptotic trend characteristic of the $A+A \rightarrow 0$ reaction. The time evolution of the UR in this

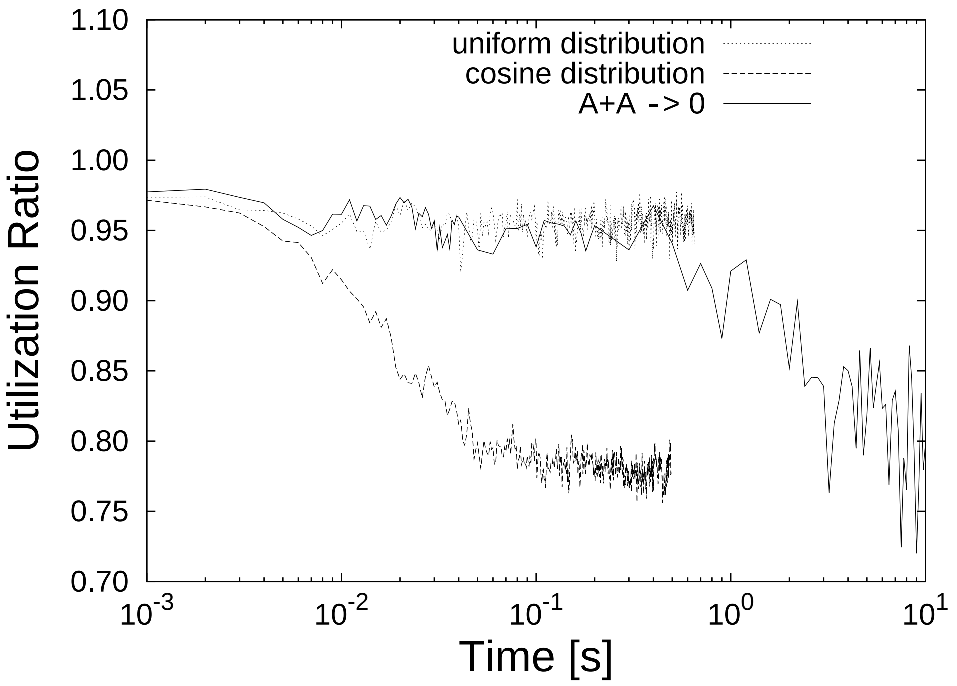

Fig. 3. Time evolution of the utilization ratio (UR) for three different 32-processor cases with periodic boundary conditions: (i) homogeneous particle distribution with no interactions; (ii) cosinusoidal particle distribution with no interactions; (iii) homogeneous distribution with $A + A \to 0$ particle annihilations.

particular case varies with the number of processors K, ranging from 98% to 89% for $K = 4$, and from 97% to about 70% for $K = 32$ (shown in Fig. 3).

The two-species reaction $A+B \to 0$ is important in many physical and chemical processes, and has been studied in detail in the literature (*e.g.* Refs. [16,17]). In principle, the kinetics of a random homogeneous bimolecular system with cross-annihilations and equal initial populations $\rho_A(0) = \rho_B(0)$ is governed by two parameters, namely, the capture radius r_c, and the typical diffusion length, $\ell = \sqrt{4D\delta t}$. The relative values of ℓ and r_c give rise to two well-differentiated regimes. In the so-called reaction-limited regime (RLR), $\ell \gg r_c$, and the system obeys an asymptotic decay law of the type $1/kt$, where k is a rate constant. However, in the diffusion-limited regime (DLR), $\ell \lesssim r_c$, spatial fluctuations asymptotically result in the separation of A and B particles into A and B-rich domains. In this case, the kinetics is considerably decelerated and the system evolves as $t^{-\frac{1}{2}}$. Figure 5 shows pkMC calculations for both the reaction- and the diffusion-limited regimes and their corresponding asymptotic decay laws. For the DLR case we have used $\ell = 10^{-3}$ and $r_c = 10^{-2}$ cm, whereas, for RLR, we used values for ℓ and r_c of 10^{-2} and 10^{-5} cm respectively. It is quite clear from the figure that the parallel kMC calculations capture the correct asymptotic kinetics in each case. To further analyze the separation kinetics (or lack thereof) in the RLR and DLR, we show in Figure 6 the $A-B$ pair

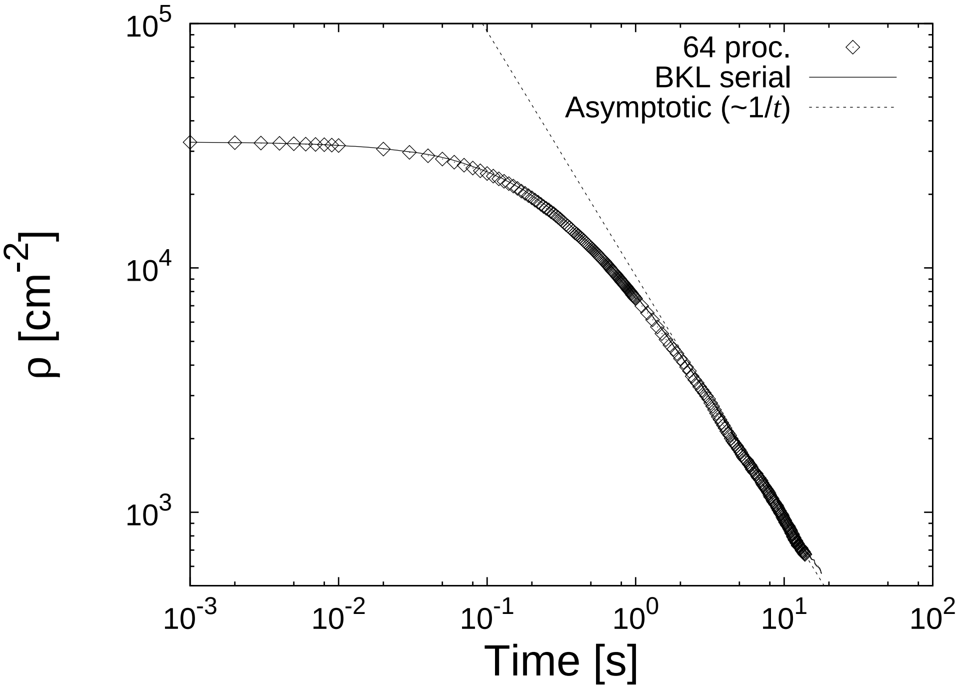

Fig. 4. Comparison between a serial BKL run (continuous line) and a parallel run with 64 processors (open diamonds) for the problem of multiparticle $NA \rightarrow 0$ annihilation with periodic boundary conditions. The asymptotic behavior $\sim 1/t$ expected for this reaction is also shown.

correlation function, $g_{AB}(r)$, for both cases[1]. $g_{AB}(r)$ measures the probability of finding a B-type particle from an A particle, averaged over the entire simulation area. These probabilities are given relative to the overall background particle density in each case, *i.e.* a probability higher than unity at a distance r simply means that, at that distance, the pair density of particles is higher than $\langle \rho_B \rangle$. In the DLR, where particles separate into A and B-rich domains, $g_{AB}(r)$ is initially very low, corresponding to a B-depleted, A-type domain. As the distance is increased, the pair correlation function gradually reaches its background value of 1.0. On the contrary, in the RLR, where homogenization is expected, $g_{AB}(r)$ resembles the pair distribution for an ideal gas. Different amounts of roughness can be appreciated in both curves, presumably indicating short and medium range order. In summary, our parallel calculations satisfactorily capture the time and spatial correlations of a particle population subject to the $A + B \rightarrow 0$ kinetics.

[1] Here, r is a generic *radial* distance, not to be confused with the rates of the diffusing species r_i

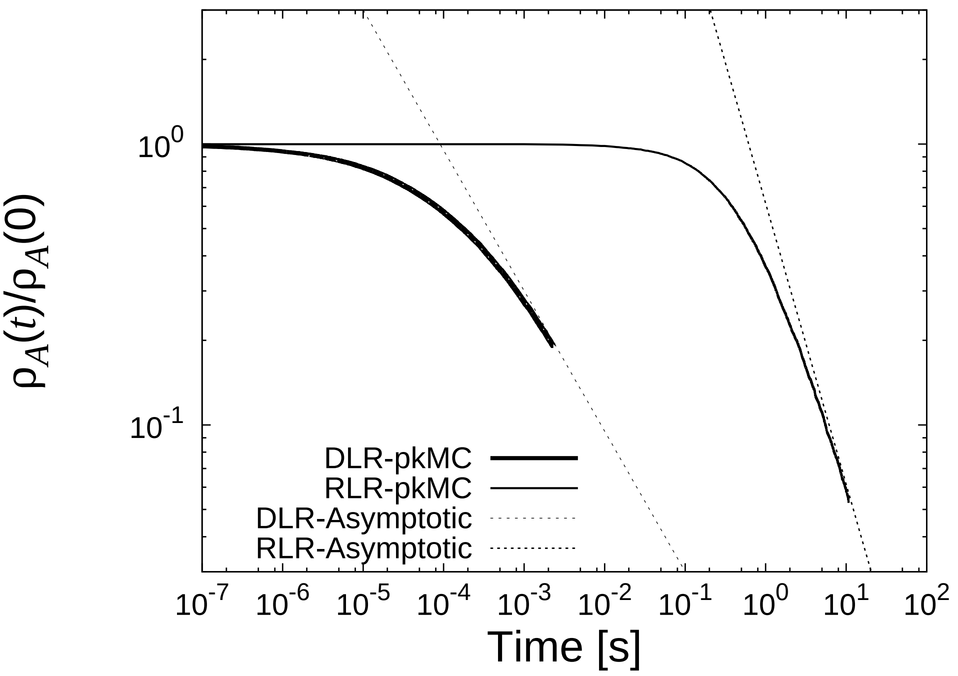

Fig. 5. Two-species annihilation kinetics (only the A-type normalized density is shown) for the reaction ($\ell \gg r_c$) and diffusion ($\ell \lesssim r_c$) limited regimes as obtained with our parallel kMC algorithm. The expected asymptotic decay law in each case is also shown for reference.

4 Performance Analysis

Next we turn to the study of the parallel efficiency of our algorithm as implemented on a distributed-memory Linux cluster with 1.4-GHz Itanium2 processors. We define two metrics for our scalability analysis, namely 'weak' and 'strong' scalability. Weak scalability measures the performance of a parallel algorithm using K processing units when the problem size is increased K-fold. For simplicity, we study this metric on a PBC system with a uniform particle distribution with no interactions. From step (6) of our algorithm, it is clear that, although not strictly necessary for this computation, our program incurs a communications overhead when particles that move across domain boundaries are reassigned to the corresponding processing units.

Figure 7 shows a family of curves for three different numbers of walkers per processor (see legend). In the absence of particle interactions, this metric estimates the cost of parallel communications when all other factors are kept invariant. For the present MPI implementation adopted here, the computational cost correlates directly with the perimeter-to-surface area (*i.e.*, the communication-to-computation) ratio in our system [18]. It has been shown that the communication-to-computation ratio in 2D improves as the aspect ratio of the subdomains tends toward perfect quadrature [19]. Of course, a per-

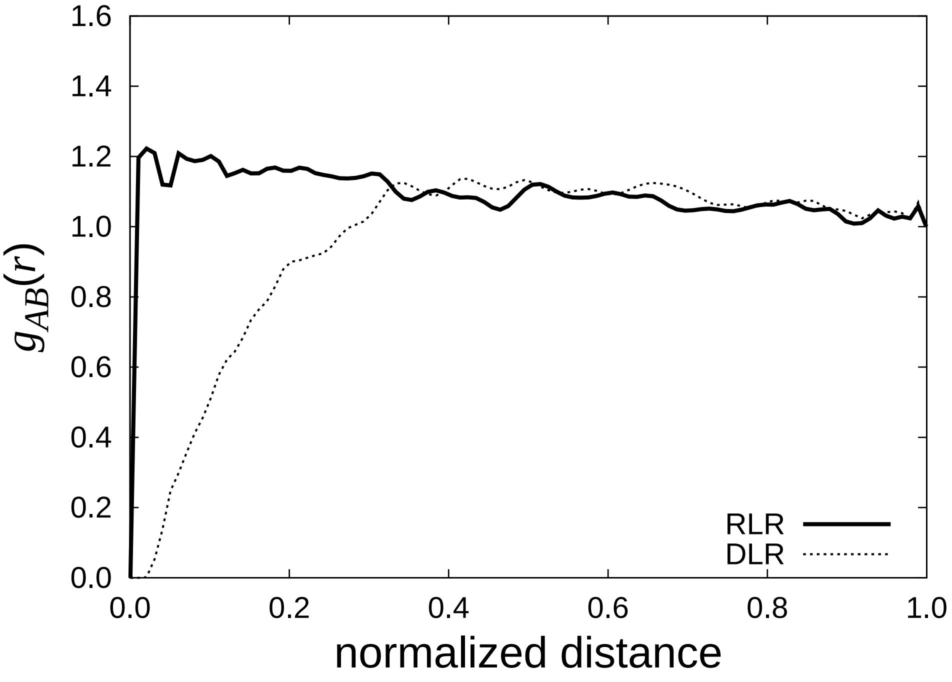

Fig. 6. $A-B$ pair correlation function for the diffusion (DLR) and reaction (RLR) limited regimes. The function measures the probability to find a B particle from an A-type particle, averaged over the entire simulation cell. The particle distance is normalized to half of the box length.

fect quadratic decomposition can only be achieved when K is an even power of 2, *e.g.* 4, 16, 64, etc., which is why the curves in Fig. 7 display abrupt steps at, for example, K=2 and K=8.

In contrast, strong scalability measures the computational speedup when an increasing number of processors is applied to a problem of fixed size. Results for up to K=128 processors for a PBC case with 131072 walkers are presented in Fig. 8. The total scalability, which is seen to be superlinear in the figure, benefits from two distinct contributions, namely (i) the time step gain derived from decreasing the length of R_{tot} (discussed in Section 2), and (ii) a contribution associated with the ORB decomposition implemented here. The binary search method used to select an event out of the frequency line carries an associated computational cost that ideally scales as $\log(N)$ [20]. After performing our ORB, each processor must now perform a search with cost $\log\left(\frac{N}{K}\right)$. In other words, there is a factor of $\log(N)/\log\left(\frac{N}{K}\right)$ speedup related simply to the cost of carrying out smaller binary search tree operations in parallel. Therefore, this speedup is in addition to the time step gain discussed earlier, which can only scale linearly at best (see discussion in Section 2 above). In figure 8 we have separated contributions (i) and (ii) from the total speedup by performing serial calculations in the fashion of Hanusse and Blanché [11], *i.e.* only with time step gain. This has been subtracted out in the curve termed 'parallel',

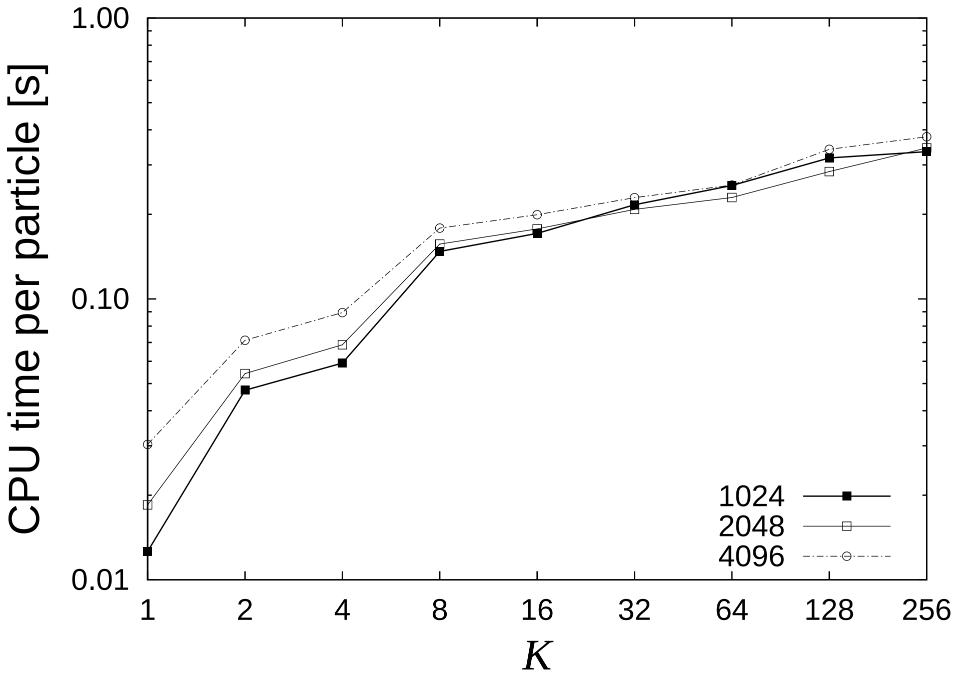

Fig. 7. Weak scalability of our algorithm for a family of curves with different numbers of particles per processor (in legend).

which is therefore the one that better represents parallel performance. As we can see, this true parallel gain scales linearly, with proportionality constant ≈ 0.5, up to about K=32, after which it is seen to gradually saturate. These metrics (Figs. 7 and 8) are simply intended for demonstrating parallel gain as obtained with a first-order implementation of our algorithm. Although the primary focus of this paper is to demonstrate correctness of our parallel method, it is encouraging that the efficiency found in the test problems is very high. Nevertheless, the full assessment of the effectiveness of the algorithm remains for future applications.

5 Summary

In summary, we have developed a novel parallel kinetic Monte Carlo algorithm that promises to access time and length scales as-of-yet unexplored in kMC simulations. Our algorithm is based on a perfectly-synchronous parallel decomposition of the master equation, to which it provides an exact solution. The efficiency of the method is contingent on the characteristics of the problem at hand and the optimization facilities of the decomposition chosen. We have demonstrated the rigorousness and performance of our algorithm in a few well understood diffusion problems, with reasonable scaling and excellent agreement between our computational results and analytical and serial cases. Due

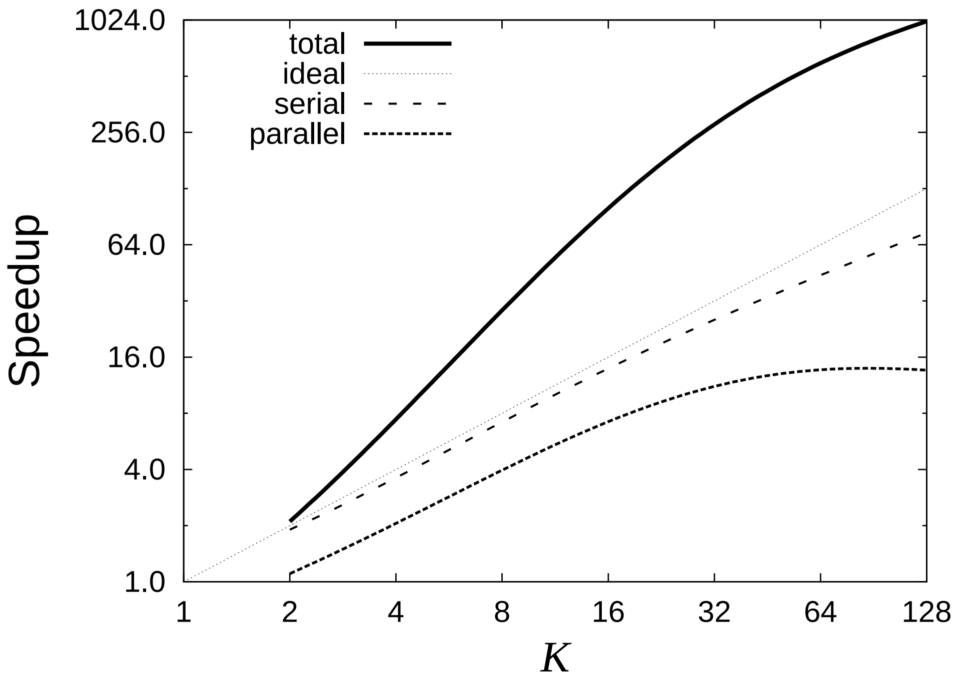

Fig. 8. Strong scalability for our parallel kMC algorithm as implemented in our prototype code. The total speedup can be broken down into a timestep gain which can be calculated using serial calculations (hence 'serial') and a gain related to a reduced binary search cost which can only be obtained in parallel calculations (termed 'parallel').

to its trivial implementation in parallel architectures, our algorithm suggests itself as a more practical alternative to previously published asynchronous methods.

This work has been performed under the auspices of the U.S. Department of Energy by the University of California, Lawrence Livermore National Laboratory under contract No. W-7405-ENG-48 and the Laboratory-directed Research and Development Office under programs 06-ERD-005 and 06-LW-028. The authors gratefully acknowledge J. M. Perlado and J. Knap for support and help with the implementation.

References

[1] M. H. Kalos, P. A. Whitlock, "Monte Carlo Methods" (John Wiley & Sons, New York", 1986)

[2] D. P. Landau, K. Binder, "Monte Carlo Simulations in Statistical Physics" (Cambridge University Press, 2000)

[3] R. Friedberg, J. E. Cameron, J. Chem. Phys. 52 (1970) 6049

[4] R. H. Swendsen, J. S. Wang, Phys. Rev. Lett. 57 (1986) 2607

[5] B. D. Lubachevsky, J. Comp. Phys. 75 (1988) 103

[6] G. Korniss, Z. Toroczkai, M. A. Novotny, P. A. Rikvold, Phys. Rev. Lett. 84 (2000) 1351

[7] G. Korniss, M. A. Novotny, H. Guclu, Z. Toroczkai, P. A. Rikvold, Science 299 (2003) 677

[8] Y. Shim, J. G. Amar, J. Comp. Phys. 212 (2006) 305

[9] D. R. Jefferson, "Virtual Time", ACM Trans. Prog. Lang. & Sys. 7 (1985) 404

[10] A. B. Bortz, M. H. Kalos and J. L. Lebowitz, J. Comp. Phys. 17 (1975) 10

[11] P. Hanusse, A. Blanché, J. Chem. Phys., 74 (1981) 6148

[12] D. ben-Avraham, J. Chem. Phys. 88 (1987) 941

[13] J. G. Amar, Comp. Sci. & Eng. 8 (2006) 9

[14] M. J. Berger, S. H. Bokhari, IEEE Trans. Comp. 36 (1987) 570

[15] *I.e.* the formal solution to eq. 1 may be expressed in terms of the eigenfunctions, ϕ_m, of the operator $-\nabla^2$. Then we may express the Green's functions for the operator $\big[-\nabla^2 + \frac{\partial}{\partial t}\big]$ as $G(\mathbf{x}, \mathbf{x}_0; t) = \sum_m e^{-\lambda_m{}^2 t}\phi_m(\mathbf{x})\phi_m(\mathbf{x}_0)$.

[16] K. Kang, S. Redner, Phys. Rev. A 32 (1985) 435

[17] F. Leyvraz, S. Redner, Phys. Rev. Lett. 66 (1991) 216

[18] D. J. Kerbyson, H. J. Alme, A. Hoisie, F. Petrini, H. J. Wasserman, M. Gittings, Proceedings of the ACM/IEEE SC2001 Conference, 2001

[19] S. Goedecker, A. Hoisie, "Performance Optimization of Numerically Intensive Codes" (SIAM, Philadelphia, PA, 2001)

[20] D. E. Knuth, "The Art of Computer Programming: Sorting and Searching (vol. 3)" (Addison-Wesley Professional, 2nd edition, 1998)