Inverted List Kinetic Monte Carlo with Rejection applied to Directed Self-Assembly of Epitaxial Growth

Michael A. Saum[1], Tim P. Schulze[1,*], and Christian Ratsch[2]

[1] *Department of Mathematics, University of Tennessee, Knoxville, Tennessee, 37996, USA.*
[2] *Department of Mathematics and Institute for Pure and Applied Mathematics, University of California in Los Angeles, Los Angeles, California 90095, USA.*

Abstract. We study the growth of epitaxial thin films on pre-patterned substrates that influence the surface diffusion of subsequently deposited material using a kinetic Monte Carlo algorithm that combines the use of inverted lists with rejection. The resulting algorithm is well adapted to systems with spatially heterogeneous hopping rates. To evaluate the algorithm's performance we compare it with an efficient, binary-tree based algorithm. A key finding is that the relative performance of the inverted list algorithm improves with increasing system size.

AMS subject classifications: Primary: 82B80, 82B21; Secondary: 82D25,65C05

PACS: 68.43.Hn,68.55.J-
Key words: Epitaxial Growth, Kinetic Monte Carlo, Binary-Tree Search

1 Introduction

A typical Kinetic Monte Carlo (KMC) [1,3,5,7] model/simulation of epitaxial growth features a small set of distinct rates related to a corresponding small set of different local environments. KMC simulations become more complex, however, when the number of distinct rates becomes large. Such situations arise, for example, when there is strain in the system, or more generally, when the potential energy surface (PES) is varying continuously.

In this paper, we implement an efficient KMC algorithm [13] that effectively handles arbitrary rate distributions through a combination of rejection and inverted list techniques. This generalizes the well-known Bortz-Kalos-Lebowitz (BKL) [3] algorithm. A key feature of the BKL "N-Fold way" is the use of a binning strategy along with what we refer to as "inverted lists". Briefly, if a list $\{e_k\}$ stores the spatial location of an event at (i,j) in memory location k, an inverted list $\{e_{(i,j)}\}$ allows one to quickly perform the

*Corresponding author. *Email addresses:* `msaum@math.utk.edu` (M.A. Saum), `schulze@math.utk.edu` (T.P. Schulze), cratsch@math.ucla.edu (C. Ratsch)

reverse operation of identifying the memory location corresponding to the event at (i,j). This type of data structure is needed to efficiently rearrange the events within the bins after an event is executed. This feature is easily overlooked in the original work, however, and, without the inverted lists, a binary tree algorithm [2] is faster—scaling like $O(\log N)$, where N is the number of independent random events to be sampled from. In an earlier publication, Schulze [12] points out that an inverted-list BKL algorithm has an operation count that is fixed as the system size N increases and compares the practical performance of the two algorithms in the frequently occurring special case where there is a small number, $M \ll N$, of distinct rates.

When combined with rejection, the inverted list technique requires one to partition the rates into M categories, the boundaries of which must be chosen by the user so as to keep the rejection percentage low. There is a tradeoff between the reduced rejection offered by increasing the number of categories and the increased cost of searching through additional categories. This is the first work to fully implement and study the performance of the combined algorithm and we have two aims:

1. give an example of a physical system where such a generalization is needed.
2. demonstrate that the partitions can be chosen so that the inverted list algorithm outperforms the binary search.

As an example of a situation calling for the generalized algorithm, we take up the study of directed self-assembly during epitaxial growth, as recently modeled by Niu et al. [9], who used an island dynamics model and the level-set technique. This continuum approach is based on the seminal work of Burton-Cabrera-Frank (BCF) [4]. The authors show that regular arrays of nano patterns can be obtained if the PES for surface diffusion is varied. Such variations can, for example, result from buried defects [6], and the subsequent long-range strain field in the substrate. We also offer a few comments on the tradeoffs between the continuum and atomistic models. As a rule, atomistic simulations allow for a more faithful description of all relevant microscopic processes, but, as a result, are limited to smaller length and time scales.

In the first section of this paper we briefly review the model and the KMC algorithm. In the second section we compare results obtained from the KMC simulations with the recent results of Niu et al. [9] and illustrate some additional situations that can be addressed using KMC. In the final section we compare the performance of the inverted list with rejection KMC algorithm with a binary-tree based algorithm [2].

2 Growth Model and KMC Algorithm

Before presenting the results of our simulations, we briefly discuss the KMC algorithm. KMC simulation of surface evolution requires one to keep track of possible events which can move the system between states. For our purposes, at each lattice location on a large two-dimensional rectangular grid with lattice constant chosen to be unity, we consider five possible events which can occur: either a deposition event at location (i,j) or a diffusion (hopping) event from location (i,j) to one of the four nearest neighbor sites $\{(i+1,j),(i,j+1),(i-1,j),(i,j-1)\}$. There are thus a total of $N = N_{\text{Dep}} + N_{\text{Hop}}$ events $\{e_k\}_{k=1}^{N}$ which can occur with rates $\{q_k\}_{k=1}^{N}$. KMC requires us to repeatedly sample and update this discrete distribution. Since deposition will be uniform in our simulations, this can be treated separately and we focus on the algorithm for sampling the hopping rates.

The hopping rate at location (i,j) is given by the harmonic approximation to transition state theory:

$$q_{ijd} = \nu \exp(-\beta(\Delta E)_{ijd}), \tag{2.1}$$

where ν is the hop attempt frequency and $d \in \{0,1,2,3\}$ indicates one of four directions to hop in. KMC simulations often represent the potential energy barrier for surface diffusion of atoms as

$$\Delta E = E_s + nE_n,$$

where E_s is the energy barrier for surface diffusion of an adatom, n is the number of in-plane lateral nearest neighbors, and E_n is the additional energy barrier for higher coordinated atoms due to the nearest neighbor bonds.

In the work of Niu et al. [9] the effect of spatial variations in E_s are studied by considering independent contributions to the transition energy from the adsorption and transition state:

$$E_s = E_{\text{trans}} - E_{\text{ad}},$$

where

$$\begin{aligned} E_{\text{trans}} &= A_0 + A_t \sin(\pi x / L_t + \phi_t) \\ E_{\text{ad}} &= A_a \sin(\pi x / L_a + \phi_a), \end{aligned}$$

and the parameters $\{A_0, A_t, A_a, L_t, L_a, \phi_t, \phi_a\}$ provide flexibility in controlling the shape of the PES.

Away from steps, the role of E_s is essentially the same in both the KMC and the island dynamics models. To make the comparison between the two models, we therefore follow the earlier example and limit the spatial variation in the hopping rates to the E_s contribution, while maintaining a fixed value of E_n. The parameter E_n accounts for detachment of edge atoms and influences the critical size of small islands in the island dynamics model [10]. In the KMC model, it plays a somewhat broader role, influencing edge diffusion and vacancy formation, for example. In making our comparisons, we adjust E_n to match the detachment rate D_{det} used in the island dynamics simulations.

To convert to integer (lattice) coordinates, E_{trans} is evaluated at $(x',y') \in \{(i+1/2,j),(i,j+1/2),(i-1/2,j),(i,j-1/2)\}$ and E_{ad} is evaluated at $(x,y) = (i,j)$. When $A_t = A_a = 0$, we have $E_s = A_0$ and recover the constant E_s situation present in many KMC models. Note also that we require the constraint $E_{\text{ad}} < E_{\text{trans}}$ in order to ensure $E_s > 0$.

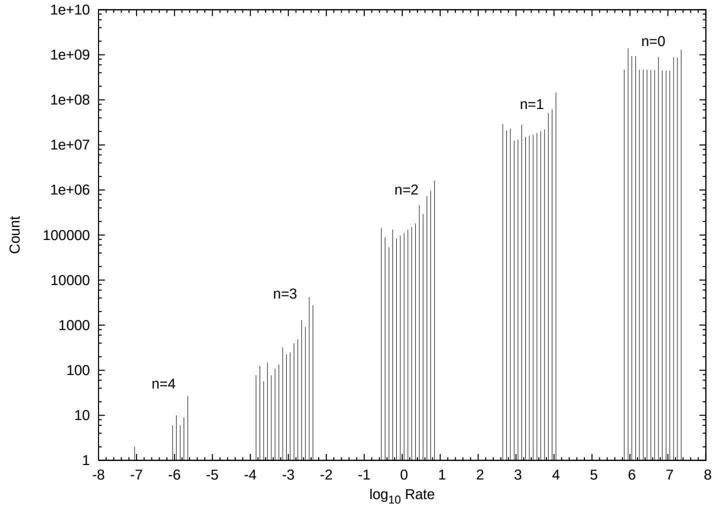

Figure 1: Rate Distribution. This figure illustrates a histogram of the rates of events executed during a simulation undergoing 100 ML of deposition. Note that both axes are $\log_{10}$ scale and that the rates cluster into subsets according to coordination number, with the fastest and most frequently realized events on the right.

As indicated above, we will compare two efficient KMC algorithms that differ in how they manage the event and rate data structures from which hopping events are determined. The first is a *binary tree algorithm (KMC-BT)*, as described in Blue, Beichl, and Sullivan [2]. We use this as a benchmark to evaluate the performance of an *inverted list algorithm with rejection (KMC-IL)*, as described in Schulze [13].

For the inverted list algorithm, we partition the set of events $\{e_k\}_{k=1}^{N_{Hop}}$ into $M \ll N_{\text{Hop}}$ categories based on rates: $E_m = \{e_k | \hat{r}_{m+1} < q_k \leq \hat{r}_m\}$, where the number and position of the partition boundaries $\{\hat{r}_m\}$ can be chosen freely. As seen in Figure 1, which shows a histogram of the rates of events that where executed in a typical simulation, the distribution of rates can be used as a guide to choosing these rate partition boundaries. In the present case, for example, the rates cluster into subsets based on coordination number. Further, we see that the number of low coordinated events is much larger than the number of high coordinated events and, therefore, low coordinated events are executed much more frequently. Therefore, in the calculations presented below, we will further subpartition the zero- and one-neighbor clusters, with the total number of partions being M. We write the resulting collection of M rate-ordered event lists as $\left\{\{e_{m\ell}\}_{\ell=1}^{c_m}\right\}_{m=1}^{M}$ and identify an individual event as $e_{m\ell}$. We define c_m to be the count of events on event list $m \in \{1,2,\ldots,M\}$. We also maintain an inverted list $\left\{e_{(i,j)}^{-1}\right\}$ which allow us to map $(m,\ell) \leftrightarrow (i,j)$. Note that it is the partitioning of rates into subsets with approximately equal rates that reduces the amount of rejection and it is the inverted list that allows one to update such a structure efficiently after an event has been executed. Without the inverted list, one would have to search the event lists to locate events that need to be moved to another category as a result of a change in their rate.

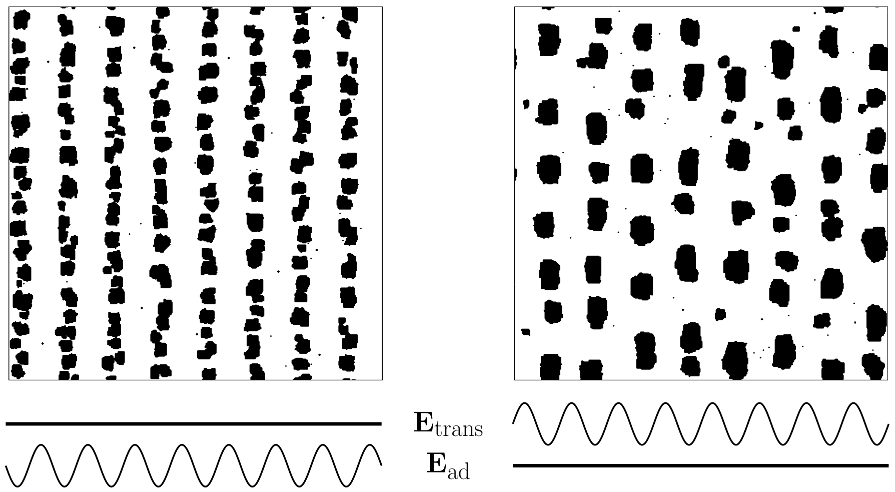

Figure 2: Morphology obtained varying only E_{ad} (left) and varying only E_{trans} (right). The PES envelopes are shown below each picture. These pictures are a snapshot after 0.2 ML of deposition.

In the case where there is no spatial variation in E_s and ΔE is a function only of the coordination number, KMC-IL is rejection free. Otherwise, KMC-IL uses a rejection test using $q_{m\ell}$ and the upper bound $\hat{r}_m$ to determine if the event chosen is to be accepted. Algorithm KMC-IL0 assumes that the hopping rates in each of the four directions are the same and are independent of position (KMC is rejection free in this case); algorithm KMC-IL1 does not make these assumptions.

3 Results of the KMC simulations

Without any spatial variations of the PES (i.e., for the case where $A_t = A_a = 0$), we choose $A_0 = E_s = 0.97$ eV and $E_n = 0.28$ eV. For a growth temperature of $T = 700\ K$, and a hop attempt frequency $\nu = 10^{13}\ s^{-1}$, this corresponds to an adatom hopping rate of approximately $10^6\ s^{-1}$ and an edge detachment rate of approximately $10^4\ s^{-1}$. All simulation results presented below, unless otherwise specified, were carried out with a deposition flux $F = 1\ ML/s$, 400×400 surface atom locations, and run on a 2GHz Intel® Core$^{\text{TM}}$2 Duo processor with 4 MB of L2 cache. It should be noted that we have chosen a domain large enough to rule out any finite size effects in the KMC simulations. The interested reader is referred to [11] for a more detailed discussion of these effects. Figure 2 illustrates the morphology obtained for cases where E_{ad} and E_{trans} are varied separately. For the case where only E_{ad} is varied, $L_a = 25$, $A_a = 0.1$ eV,$A_t = 0$, and $\phi_a = \pi$. For the case where only E_{trans} is varied, $L_t = 25$, $A_t = 0.1$ eV,$A_a = 0$, and $\phi_t = 0$. All of these numbers correspond to the variations of the PES in Niu, et al. [9], and the results are similar to those shown in Fig. 1 of Ref. [9].

Figure 3 shows a comparison of spatial island density distributions between KMC and

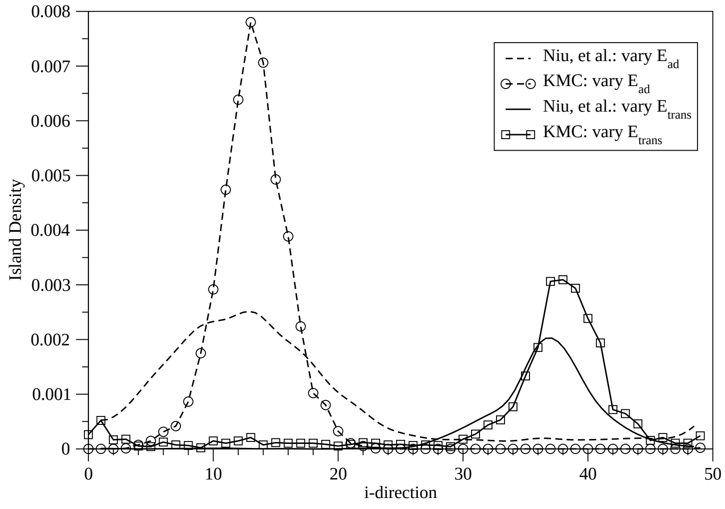

Figure 3: Comparison of spatial island density distributions between KMC and continuum island dynamics calculations. The total densities are different because the meaning of D_{det} is different for KMC and continuum theory.

the results shown in Fig. 2 of Ref. [9]. KMC results are averages of island densities from 30 independent runs which are then averaged across the domain in the *i*-direction due to periodicity of the PES. Continuum results are based on the island dynamics calculations contained in [9]. Note that good agreement is obtained on locations of the distributions. The total densities (i.e., the integrals under the curves) do not match because, while an attempt was made to match D_{det} between KMC and continuum theory, the meaning of D_{det} is different for KMC and continuum theory.

Figure 4 illustrates a case where variations of E_{trans} and E_{ad} vary in-phase ($L_a = L_t = 25$, $\phi_a = \phi_t = \pi$), with the same amplitude ($A_t = A_a = 0.1$ eV), and deposition involves multiple monolayers. This result illustrates that if one could control the PES in such a manner, it would provide a mechanism to grow nanowires. This case took approximately 4 minutes of CPU time for 6.62×10^8 events undergoing 5 ML of deposition.

Figure 5 illustrates using KMC with a varying PES in the Diffusion Limited Aggregation (DLA) regime [14], which is difficult to simulate with a continuum model. Here adatoms are almost surely irreversibly attached once its coordination number becomes one. In an atomistic model, such as KMC, one obtains fractal like geometries, where the arm width of the fractals is determined (and limited) by the discreteness of the size of an atom. In contrast, in continuum models, the arm width is determined by the numerical resolution that has been chosen. Thus, it may take much greater computational resources with continuum models to achieve the same resolution as KMC in the DLA regime. The KMC-DLA case was run with $E_n = 0.5$ eV and $E_s = 0.25$ eV on a 256×256 grid with $T = 250$ K, and it is the combination of lower temperature and decrease in E_s relative to E_n that makes the atoms essentially immobile once they attach to an island.

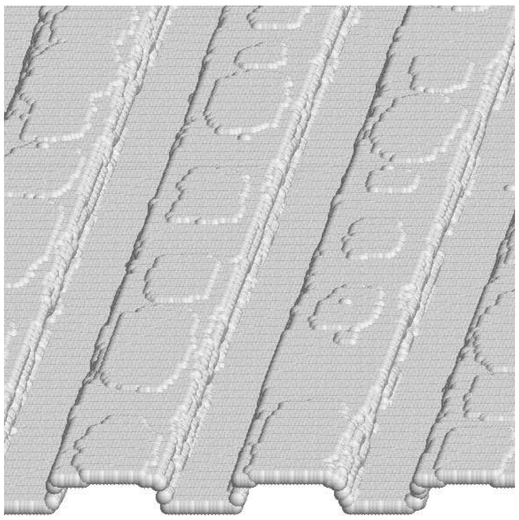

Figure 4: 3-D surface morphology obtained where in-phase variations of E_{trans} and E_{ad} occur. This picture is a snapshot taken after 5.0 ML of deposition, 400×400 grid, $T = 700\ K$, and $F = 1\ ML/s$. This job took approximately 4 minutes of CPU time for 6.62×10^8 events.

This case took approximately 2 seconds of CPU time for 2.16×10^6 events undergoing 0.3 ML of deposition.

A similar simulation using the level set (LS) formalism takes significantly longer, about 10 seconds (depending on the numerical resolution and accuracy chosen). However, the increase in computational time for increasing detachment rate is slower with the level set simulation. This can also be seen in Table 1, where we show a comparison of the computational times for comparable LS and KMC simulations. The key result is that the KMC simulation is significantly faster, but that the slowdown with increasing D_{det} is less pronounced in the LS method. We also note that the number of numerical timesteps needed is orders of magnitude lower with the LS method, which might be advantageous when other external fields such as strain are coupled to the simulation.

4 KMC Algorithm Performance

In Figure 6 we explore the influence of the number of rate categories M on the CPU time per event (left axis) and rejection rate (right axis) when using KMC-IL1. The size of the surface grid for these calculations was 1024×1024, with $A_a = 0.1$ eV, $A_t = 0$ and $F = 0.25$ ML/s. The timing data is the average of ten separate runs undergoing 100 ML of deposition, utilizing the same random seed, and error bars—which reflect variation in system performance only—are one standard deviation above and below the mean. For example, choosing to partition the rate space into 12 bins ($M = 12$) results in a time of 3.48×10^{-7} sec/event at a global rejection percentage of 25.8%. This data was obtained on a 2.6 GHz AMD Opteron$^{\text{TM}}$processor with 1 MB of L2 cache. As anticipated, as the

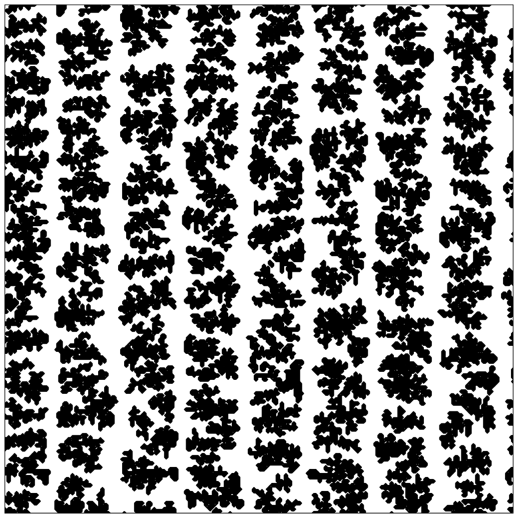

Figure 5: Morphology obtained by varying the PES periodically for DLA case, with a total deposition of 0.3 ML. This case used a 256×256 grid, with $T = 250$ K. This case took approximately 2 seconds of CPU time for 2.16×10^6 events.

Table 1: Comparison of KMC and continuum timing data. These results were obtained on a 200×200 surface undergoing five ML of deposition with no spatial variation of the PES. The continuum results were obtained utilizing a coarse resolution. We believe that the increase in computation time for $D_{\text{det}} = 1000$ with the LS method is an artifact of the timestep selection in this method, and that it could be reduced to a factor close to 1.1.

D_{det}	Type	Time (sec/ML)	t/t_{10}
10	KMC	0.91	1
	LS	33.42	1
31	KMC	1.04	1.14
	LS	33.69	1.00
100	KMC	1.32	1.45
	LS	34.65	1.04
316	KMC	1.91	2.10
	LS	37.41	1.12
1000	KMC	2.84	3.12
	LS	67.97	2.03

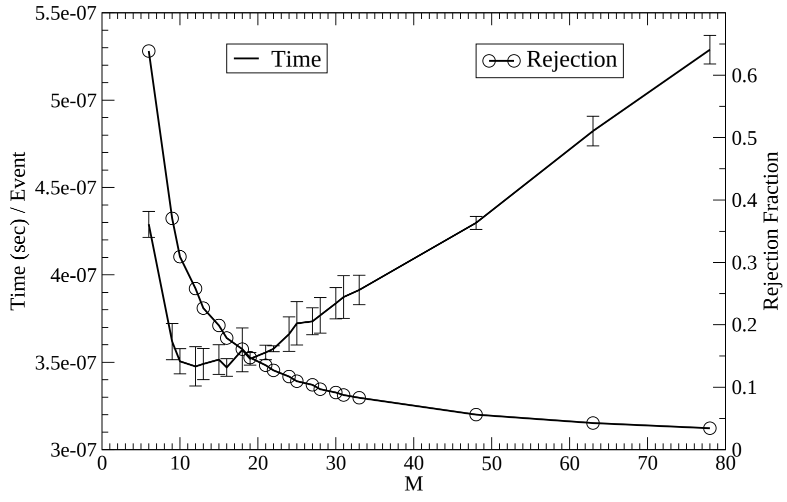

Figure 6: KMC-IL1 Binning Strategy. This figure illustrates various binning strategies based on total number of bins M, as well as global rejection fraction for the particular binning strategy. Note that it is clear that there exists an optimal binning strategy for this particular problem.

number of bins is increased beyond a certain point (roughly 12 bins for these system parameters), the reduced rejection rate is offset by the increased computational cost of selecting a bin and summing the rates.

Figure 7 contains a comparison of performance for the KMC algorithms referenced above for a variety of system sizes. The top panel utilizes the fundamental KMC performance metric, time (sec) / event, and the bottom figure shows the ratios, KMC-BT/KMC-IL{0,1}, for these same metrics. The data is plotted against p, which is representative of the system size $2^p \times 2^p$. The timing data trend is similar to that of previous results published in Schulze [12] for systems with no spatial heterogeneity, where we have extended the results to larger system sizes. Notice that for very large systems, both methods slow down relative to the theoretical $O(\log N)$ (BT) and $O(1)$ (IL) scalings, which would suggest these curves should be linear (BT) and horizontal (IL). We present data below that shows the degredation in performance is due to cache misses, which both alogorithms suffer from as the system becomes very large. This effectively makes CPU operations more expensive as the system becomes large, but as the bottom panel shows, the ratio of the performance of the two alogorithms grows rougly linearly in $p \propto \log N$, as expected, and, for large systems, one obtains excellent speedup of KMC-IL1 over KMC-BT. This data was obtained on a 2.6 GHz AMD Opteron$^{\text{TM}}$processor with 1 MB of L2 cache.

The degradation of KMC-BT performance as p gets large is a combination of two factors, algorithm and memory cache issues. For each event, KMC-BT requires a search

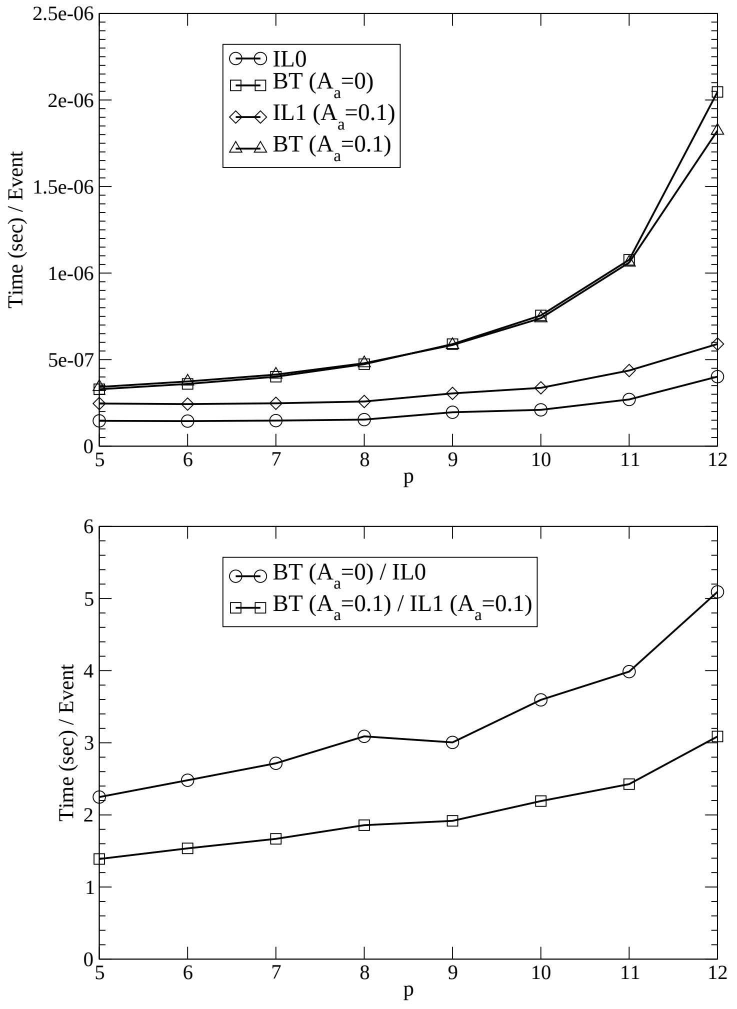

Figure 7: KMC Timing Comparisons: KMC-BT vs. KMC-IL{0,1}. The top figure illustrates the amount of time (sec) per event at different grid sizes, where the grid is $2^p \times 2^p$. The bottom figure illustrates the ratio of the time/event numbers for the binary tree compared to the inverted list algorithm. KMC-IL0 is rejection free and used 5 rate categories ($M = 5$), corresponding to coordination number. For the KMC-IL1 runs with spatially dependent ΔE ($A_a = 0.1$ eV, $A_t = 0$), 12 rate categories ($M = 12$) were used. Note that processor cache effects start to play a large role as p gets large.

of a binary tree of depth $2p+1$, and up to seven traversals of the full depth to update the partial sums in the tree, giving the binary tree algorithm a search and update complexity of $O(\log N)$. Using the inverted list KMC algorithm one obtains an $O(1)$ search and update.

The second factor, related to memory cache utilization, affects both algorithms roughly in proportion to the number of instructions executed. Today's computers utilize a cached memory hierarchy which is based on the locality principle, motivated by the fact that most memory references are made to a small number of locations. Temporal locality is the principle that once a program references a memory location, it is likely that the same memory location will be referenced again soon, which is true in the case of program iteration or recursion. Spatial locality is the principle that a memory location near a recently referenced location is more likely to be referenced than one further away, typically true of data stored consecutively in arrays.

To investigate cache performance issues more thoroughly, we utilized the Cachegrind tool which is a part of the Valgrind [8] distribution. Cachegrind is a flexible and powerful tool that performs simulation of a machine's cache as a program executes. Cachegrind tracks cache statistics (L1 and L2 hits and misses) for every individual line of source code executed by the program. We configured Cachegrind to simulate the AMD Opteron$^{\text{TM}}$ processor with 1 MB of L2 cache. Note that Cachegrind does not model TLB (Translation Lookaside Buffer) misses, which can be up to 100 times more expensive (in CPU cycles) than an L2 cache miss. On the AMD Opteron$^{\text{TM}}$, the L2 TLB can store 512 entries which map 4KB pages between physical and virtual memory. Thus, when a reference is made to a memory location which is not in L2 cache and the page referenced is not in the TLB, then a TLB miss occurs.

Table 2 compares two metrics for the binary tree KMC and the inverted list KMC algorithms for four different system sizes. The first colum gives the number of L2 data cache misses per event and the second the number of CPU instruction per event. The latter column is consistent with the theoretical operation counts given above. In particular, note that the number of instructions per event for the inverted list algoirithm is essentially constant. The first column shows that the slowdown in both algorithms for large values of p is clearly correlated with increased L2 data cache misses.

While stochastic simulations such as KMC inherently exhibit poor spatial and temporal locality memory access patterns, the performance penalty for KMC-BT relative to KMC-IL is exacerbated due to the increased number of instructions. Traversing and updating the binary tree requires one to access many pages of virtual memory, resulting in many cache misses. Similarly, but to a lesser extent, the inverted list algorithm suffers cache misses when accessing the various lists of events. It may be possible to reduce this effect for the inverted list algorithm through careful programming for the specific application of epitaxial growth, where the dominant calculational cost in the simulation is due to the motion of the adatoms, of which there are relatively few. The inverted list algorithm naturally groups these few, but frequently realized events, together in memory. To exploit this, however, one would need to avoid the cache misses encountered when an

Table 2: Cache Performance Summary. This table summarizes performance characteristics for BT and IL KMC algorithms for various system sizes. Note that L2 cache misses occur more frequently as the system size gets large, especially when comparing BT to IL for the same system size. Note also the linear increase in the number of CPU instructions per event with the BT and the essentially constant CPU instructions per event for IL, as a function of system size p.

Size (p)	Alg.	L2 Miss/Event	CPU Instr/Event
6	BT	0.0042	1184
	IL0	0.0039	453
8	BT	0.122	1462
	IL0	0.064	452
10	BT	0.855	1745
	IL0	0.610	452
12	BT	14.1	2019
	IL0	1.73	452

adatom hops to a site with coordination number four. Since most sites have a coordination number of four, it might be possible to avoid listing these sites entirely and simply use uniform sampling of the grid with rejection to simulate the rare vacancy formation events.

Acknowledgments

MAS was supported by a grant from DOE (DE-FG02-03ER2558). TPS was supported by grants from DOE (DE-FG02-03ER2558) and NSF (NSF-DMS-0707443).

References

[1] F. F. Abraham and G. W. White. Computer simulation of vapor deposition on two-dimensional lattices. *J. Appl. Phys.*, 41:1841, 1970.

[2] James L. Blue, Isabel Beichl, and Francis Sullivan. Faster Monte Carlo simulations. *Phys. Rev. E*, 51:867, 1995.

[3] A. B. Bortz, M. H. Kalos, and J. L. Lebowitz. A new algorithm for Monte Carlo simulation of Ising spin systems. *J. Comp. Phys.*, 17:10, 1975.

[4] W. K. Burton, N. Cabrera, and F. C. Frank. The growth of crystals and the equilibrium structure of their surfaces. *Phil. Trans. Roy. Soc.*, 243(866):299–358, 1951.

[5] G. H. Gilmer and P. Bennema. Simulation of crystal growth with surface diffusion. *J. Appl. Phys.*, 43:1347, 1972.

[6] H.J. Kim, Z.M. Zhao, and Y.H. Xie. Three-stage nucleation and growth of Ge self-assembled quantum dots grown on partially relaxed SiGe buffer layers. *Phys. Rev. B*, 68:205312, 2003.

[7] A. C. Levi and M. Kotrla. Theory and simulation of crystal growth. *J. Phys. Condens. Matter*, 9:299–344, 1997.

[8] Nicholas Nethercote. *Dynamic Binary Analysis and Instrumentation*. PhD thesis, University of Cambridge, 2004.

[9] X. Niu, R. Vardavas, R. E. Caflisch, and C. Ratsch. Level set simulation of directed self-assembly during epitaxial growth. *Phys. Rev. B*, 74:193403, 2006.

[10] C. Ratsch, P. Šmilauer, A. Zangwill, and D. D. Vvedensky. Submonolayer epitaxy without a critical nucleus. *Surf. Sci.*, 329:L599, 1995.

[11] Michael Schroeder and Dietrich E. Wolf. Magic islands and submonolayer scaling in molecular beam epitaxy. *Phys. Rev. Lett.*, 74(11):2062–2065, Mar 1995.

[12] T. P. Schulze. Kinetic Monte Carlo simulations with minimal searching. *Phys. Rev. E*, 65(3):036704, Feb 2002.

[13] Tim P. Schulze. Efficient kinetic monte carlo simulation. *J. Comp. Phys.*, 227:2455–2462, 2008.

[14] T. A. Witten and L. M. Sander. Diffusion-limited aggregation, a kinetic critical phenomenon. *Phys. Rev. Lett.*, 47(19):1400–1403, 1981.