Computational Nanoscience

NSE C242 & Phys C203

Spring, 2008

Lecture 10:

Beyond Monte Carlo

February 21, 2008

Elif Ertekin and Jeffrey C. Grossman

Today's lecture (… is schizophrenic)

- … will consist of a smattering of topics as we conclude our discussion of pure Monte Carlo and move on to phase transitions next week.
- We will discuss some of the common pitfalls of Monte Carlo methods
- And then briefly review Grand Canonical Monte Carlo
- And on to **Kinetic Monte Carlo** as a way of bringing time back into our model
- Time-permitting, we will discuss the art of random number generation.

Pitfalls of Monte Carlo

Choosing the initial configuration …

Most of the considerations that are pertinent in determining a good initial configuration in Molecular Dynamics also apply to Monte Carlo.

In MD, one aims at avoiding an initial configuration with particles too close to one another, as this results in large accelerations, and consequently in a longer time required for the system to “settle down”.

The physical analogy is that the system is initially not in thermal equilibrium with the reservoir at the selected temperature, and must be allowed enough time to relax thermally toward the final equilibrium state.

Thus, it is usually necessary to discard from the configurational ensemble the first configurations generated by the Metropolis random walk, until it can be established that the system has reached thermal equilibrium.

Pitfalls of Monte Carlo

Single vs. Global Moves …

Generally speaking, one can expect a higher acceptance if one particle at a time is displaced.

Consider the following simple argument: let the probability of accepting the move of the first particle be η_1, the probability of accepting the move of the second particle be η_2 and so on.

Suppose for simplicity that $\eta_1 = \eta_2 = \ldots \eta_N = 0.5$. Then single-particle moves would be accepted roughly half of the time.

On the other hand, if all particles are moved at once it is easy to see that the probability of accepting the "global move" will be the product $\eta_1 \times \eta_2 \times \ldots \eta_N \approx 2^{-N} << 1$.

Note, though, that at low temperatures, or in the vicinity of a phase transition, single-particle moves become increasingly inefficient and the case for "global" configurational updates becomes stronger.

Pitfalls of Monte Carlo

Critical Slowing Down … more on this later

Roughly speaking, the problem arises as the system tends to behave, in the vicinity of a phase transition, more and more “as a whole”, i.e., its different parts are highly correlated as the system approaches an ordered state (such as in the melting-freezing transition).

In more precise language, the characteristic *correlation length* of the system diverges at a critical point.

As a result, it becomes very easy for the system to be locked into a so-called metastable state, or a state from which the system eventually migrates but in an increasingly larger amount of time, particularly as the system size is increased.

Often, little can be done to overcome such a problem. Sometimes, however, it is possible to overcome, at least partially, this difficulty by attempting to perform global moves, allowing the system to change its state radically (more on this in the context of the Ising model).

Pitfalls of Monte Carlo

Statistically Correlated Configurations

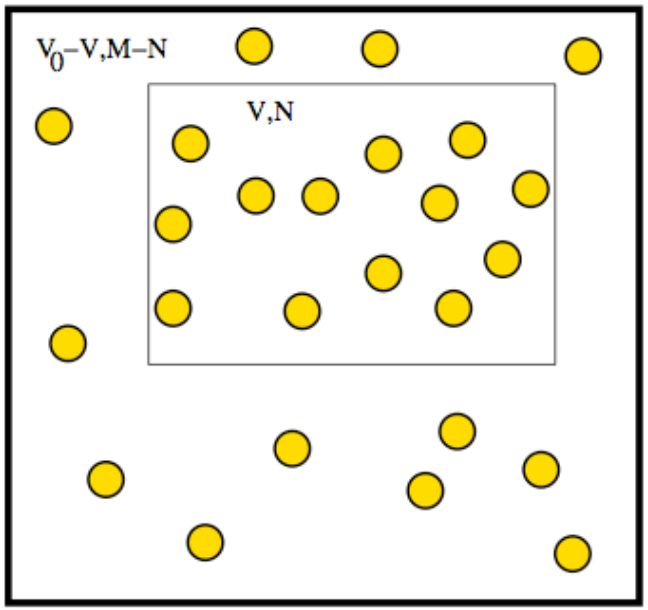
Particularly in Markov simulations, it is quite possible that successive configurations are not necessarily independent of each other (temporal correlations).

A good Monte Carlo simulation must use a large enough number of configurations so that the total number of configurations is larger than a few "characteristic time" scales.

This is more of a problem at low temperatures (since most of the proposed configurations are rejected), where it might take a large number of Monte Carlo steps to get statistically independent configurations.

Beyond Metropolis to Grand Canonical Monte Carlo

- It is possible to conduct Monte Carlo simulations in the Grand Canonical Ensemble, where (μ,P,T) are held fixed.
- Note how the number of particles in the system is now free to vary
- Why would one want to do this?



procedure is the same as before:

(1) Find the partition function for the system.

- the "system" is now a coupled system

(2) Determine transition rules are consistent with it - detailed balance.

after M. Dijkstra, http://www.phys.uu.nl/~mdijkstr:

$$\sum_{N=0}^{M} \frac{V^N}{N!\Lambda^{3N}} \int d\mathbf{s}^N \exp[-\beta U(\mathbf{s}^N; V)] \times \frac{(V_0 - V)^{M-N}}{(M-N)!\Lambda^{3(M-N)}}$$

Beyond Metropolis to Grand Canonical Monte Carlo

after M. Dijkstra, http://www.phys.uu.nl/~mdijkstr:

The probability of acceptance of a trial move in which a particle is transferred from the ideal reservoir to our system

$$\frac{\mathrm{acc}(N \to N+1)}{\mathrm{acc}(N+1 \to N)} = \frac{V^{N+1}(V_0 - V)^{M-N-1} \exp[-\beta U(\mathbf{s}^{N+1})]}{(N+1)!(M-N-1)!} \times \frac{(N)!(M-N)!}{V^N (V_0 - V)^{M-N} \exp[-\beta U(\mathbf{s}^N)]}$$

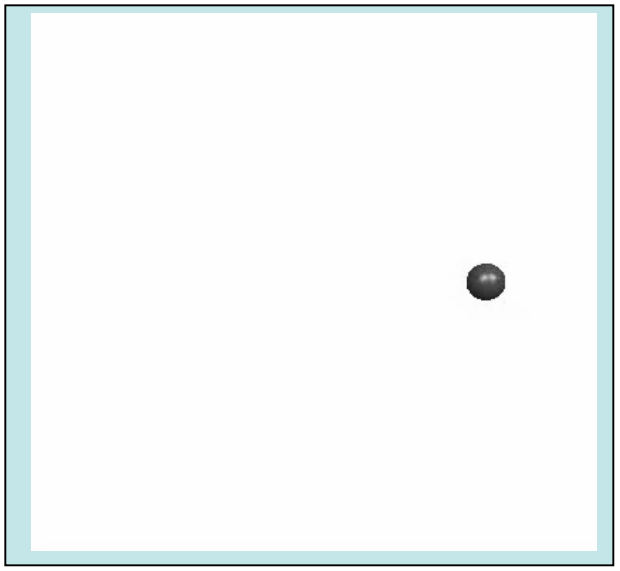
$$= \frac{V(M-N)}{(V_0 - V)(N+1)} \exp[-\beta(U(\mathbf{s}^{N+1}) - U(\mathbf{s}^N))]$$

$$= \frac{V}{\Lambda^3 (N+1)} \exp[\beta(\mu - U(\mathbf{s}^{N+1}) + U(\mathbf{s}^N))] \qquad (2)$$

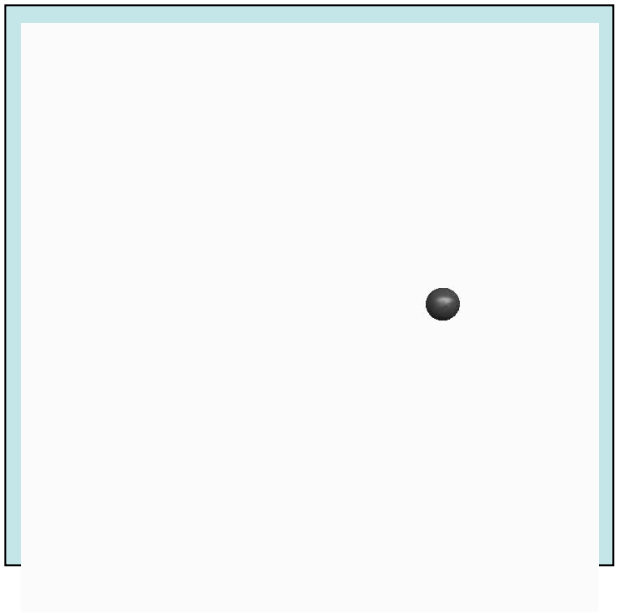
where we considered the limit that the reservoir tends to infinity, i.e. $V_0 \to \infty, M \to \infty$, while $(M-N)/(V_0 - V) \to M/V_0 = \rho = \exp[\beta\mu]/\Lambda^3$.

Beyond Metropolis to Grand Canonical Monte Carlo

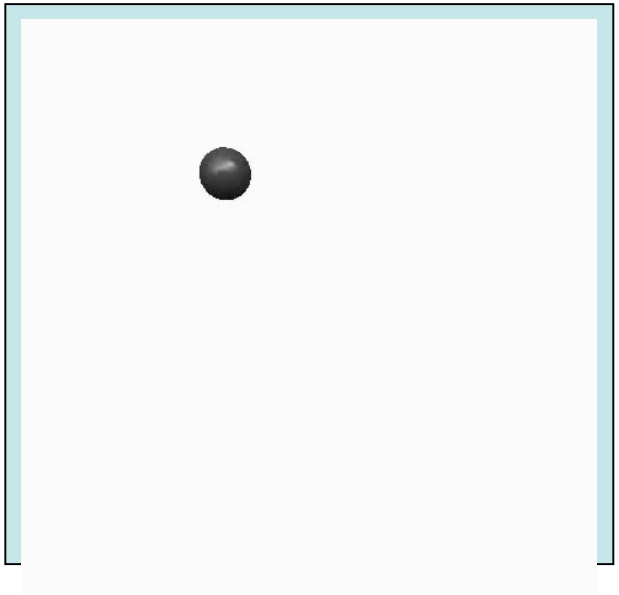
formation of a graphene cap on a spherical catalyst



μ = -5 eV; T = 1000K



μ = -5 eV; T = 2000K



μ = -5 eV; T = 3000K

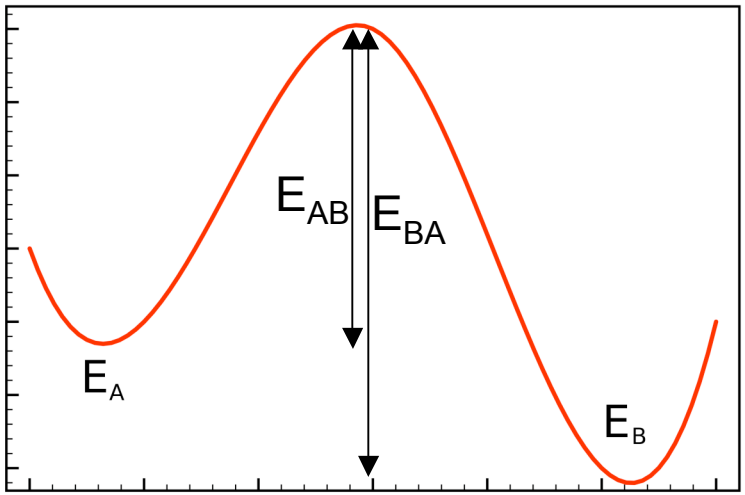
Kinetic Monte Carlo - Motivation

We said earlier that Monte Carlo methods have no intrinsic time scale, but instead represent how a given system samples different configurations at equilibrium.

The Kinetic Monte Carlo method is an extension of basic Monte Carlo that enables one to study the evolution of a system in time.

It enables us to model island growth, thin film growth, self-assembly, physical and chemical vapor deposition processes, etc.

Transition State Theory



The transition rates of the system from state A to state B, and vice-versa, are given by:

$$\Gamma_{AB} = \nu \exp\left[-\beta E_{AB}\right]$$
$$\Gamma_{BA} = \nu \exp\left[-\beta E_{BA}\right]$$

Γ is the transition rate, ν denotes some attempt frequency, and E gives the height of the barrier for the transition.

Notice that pure Monte Carlo simulations take no note of the barriers, but are only concerned with the relative energies of the start and end points.

This is entirely consistent; the relative rates of transition from A to B and from B to A are still correct in Monte Carlo. We need not know the "barrier height" for detailed balance. But, we have no true sense of time, however, and no way to relate transition rates across different types of processes.

Kinetic Monte Carlo Algorithm

BKL Algorithm (Bortz, Kalos, Lebowitz 1975)

1. Identify all the relevant processes for your system.

   - This is not always easy!

   - Many processes are usually present, but you need to select the important ones and leave the others aside.

   - Too many processes in your simulation will mean high computational cost

2. Determine (guess, estimate, calculate …) the activation barrier for each process. Use TST to assign a rate to each process. We'll denote by r_i the rate of the i_{th} process.

3. The total rate at which "anything" happens is then given by

$$R = \sum_i r_i$$

4. Use R to choose from a poisson distribution the time at which the next event happens (first random number).

Kinetic Monte Carlo Algorithm

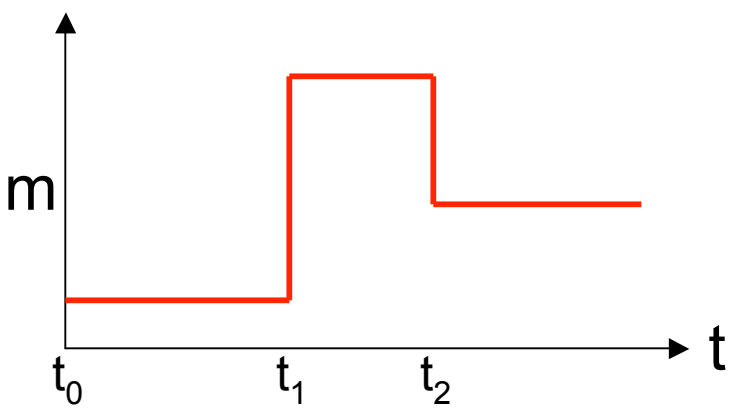
5. Choose which event actually happens (second random number).

$$p_i = r_i / R$$

How does this help, over standard Monte Carlo?

First of all, we now have a clock built in to the simulation.

Secondly, we are actually doing something every step (compare to Monte Carlo, where some portion of the steps are rejected, especially at low temperature). Of course, this is somewhat offset by the fact that at the next step, the system might flip back to where it started.



Now, when computing averages, the configurations must be weighted by the time intervals.

$$\langle m \rangle = \frac{1}{T} \int_0^T m(t)\,dt$$

Poisson Distribution for Time Intervals

The probability distribution governing when the next event will happen is a poisson distribution, which we will describe here.

Imagine that time passes in discrete intervals Δ. The probability that nothing has happened after n such intervals is given by

$$Q_n = (1 - R\Delta)^n = \left(1 - \frac{Rt}{n}\right)^n$$

In the limit of continuous time (Δ approaches 0 or n becomes infinite):

$$Q(t) = \exp(-Rt)$$

Then the probability that something has happened between time 0 and t is:

$$P(t) = 1 - \exp(-Rt) = \int_0^t p(t)dt$$

Which means that the relevant probability distribution for events occuring is given by

$$p(t) = R\ \exp(-Rt)$$

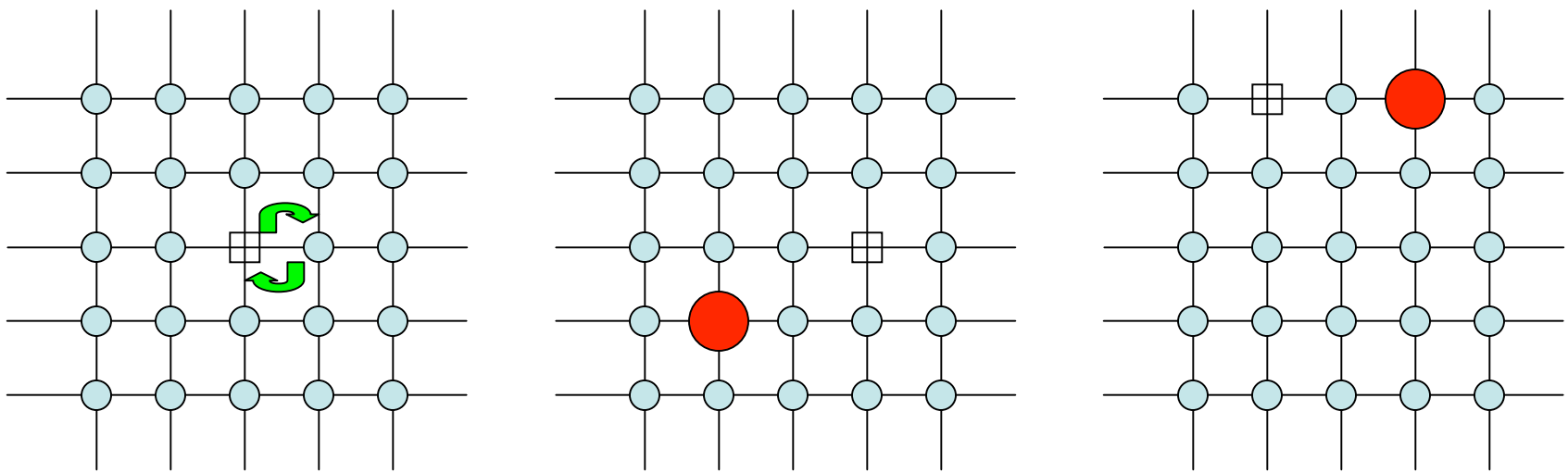
This is called a poisson distribution. It is memoryless, and is the same distribution that governs things such as when your lightbulb will burn out.

Example of KMC - Vacancy Mediated Diffusion

Diffusion in solids is a complex, thermally activated process which can occur through a variety of mechanisms.

We will use KMC to consider vacancy mediated diffusion, in which a vacancy undergoes a “random walk” through a discrete atomic lattice.

The vacancy moves by swapping locations with neighboring atoms.



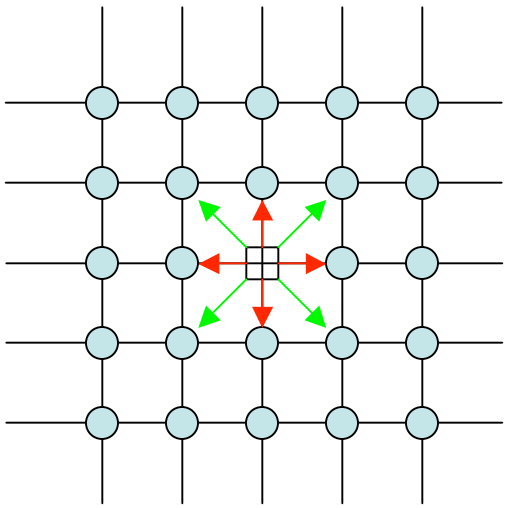
If we choose an atom at random to “trace”, and keep track of its position over the course of the KMC simulation, we can estimate things like diffusion coefficients.

From some remarkably general considerations (that I will not describe here), we can relate mean-field quantities such as diffusion coefficients to discrete systems

$$J = -D \bullet \nabla C\ ,\ \frac{\partial C}{\partial t} = \mathrm{D}\nabla^2 \mathrm{C},\ \text{probability distributions governing random walks} \quad \Rightarrow \quad 2dDt = \left\langle R^2 \right\rangle$$

Example of KMC - Vacancy Mediated Diffusion

1. Identify all the relevant processes for your system.



8 “obvious” processes:

- p_1, p_2, p_3, p_4 have barrier E_N
- p_1, p_2, p_3, p_4 have barrier E_D

What are other possible processes?

- two vacancies come into contact with some binding energy

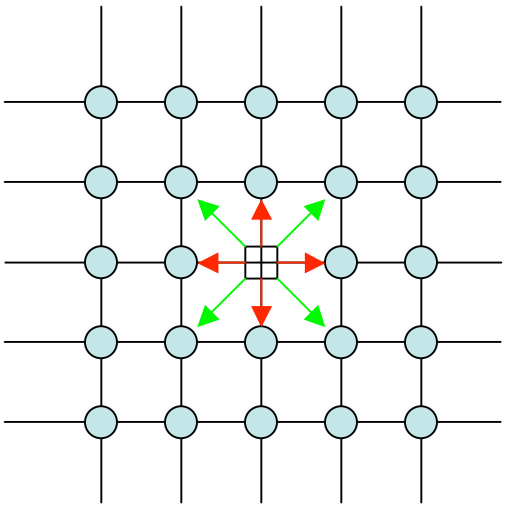
- atoms swap locations with each other

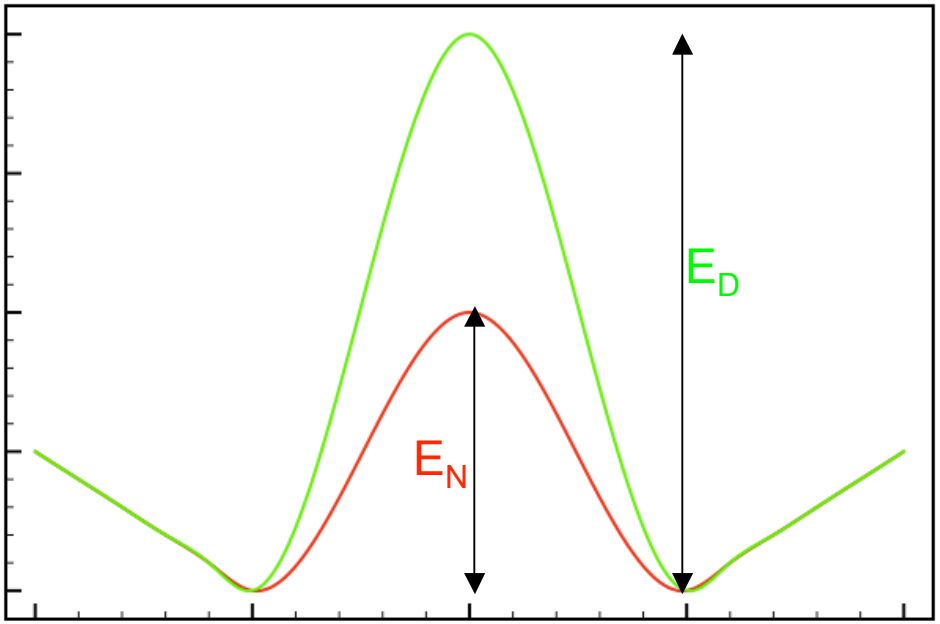
- atom adsorption from gas, desorption to a gas (vacancy destruction or creation)

- etc, etc but we will ignore these

Example of KMC - Vacancy Mediated Diffusion

2. Determine (guess, estimate, calculate …) the activation barrier for each process. Use TST to assign a rate to each process. We'll denote by r_i the rate of the i_{th} process.





What are some ways to determine the transition barriers?

Example of KMC - Vacancy Mediated Diffusion

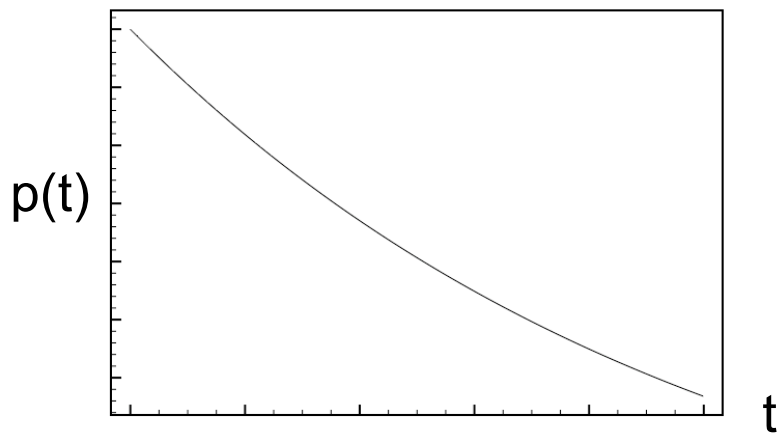
3. The total rate at which "anything" happens is then given by

$$R = \sum_i r_i = 4\nu\left(\exp\left[-\beta E_N\right] + \exp\left[-\beta E_D\right]\right)$$

4. Use R to choose from a poisson distribution the time at which the next event happens (first random number picked here).

$$p(t) = R\ \exp(-Rt)$$

probability distribution for the next event to occur



How does one choose a number according to a poisson distribution? We want to identify a function f(x) so that computing $f(x_i)=t_i$ generates t_i according to p(t) (where x_i is a random number uniformly distributed between 0 and 1.

$$t_i = -\frac{1}{R}\ln x_i$$

Example of KMC - Vacancy Mediated Diffusion

5. Choose which event actually happens (second random number) from the rate catalog.

$$p_{N_1} = \frac{r_{N_1}}{R} = \frac{\exp[-\beta E_N]}{4(\exp[-\beta E_N] + \exp[-\beta E_D])}$$

$$p_{N_2} = \frac{r_{N_2}}{R} = \frac{\exp[-\beta E_N]}{4(\exp[-\beta E_N] + \exp[-\beta E_D])}$$

$$p_{N_3} = \frac{r_{N_3}}{R} = \frac{\exp[-\beta E_N]}{4(\exp[-\beta E_N] + \exp[-\beta E_D])}$$

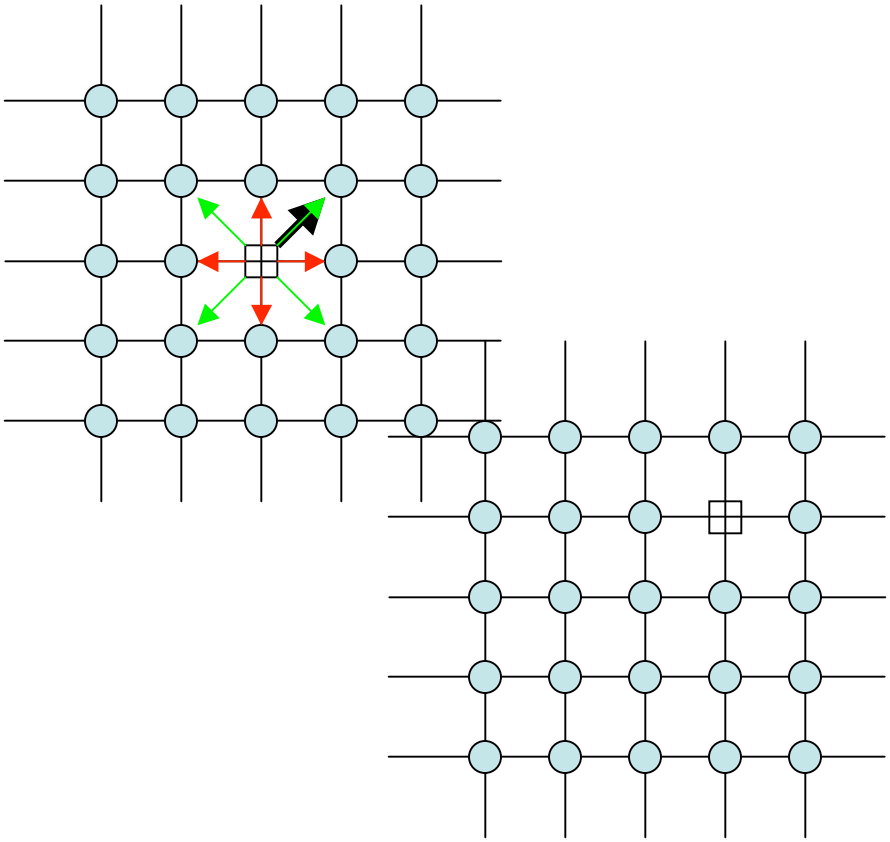
$$p_{N_4} = \frac{r_{N_4}}{R} = \frac{\exp[-\beta E_N]}{4(\exp[-\beta E_N] + \exp[-\beta E_D])}$$

$$p_{D_1} = \frac{r_{D_1}}{R} = \frac{\exp[-\beta E_N]}{4(\exp[-\beta E_N] + \exp[-\beta E_D])}$$

$$p_{D_2} = \frac{r_{D_2}}{R} = \frac{\exp[-\beta E_N]}{4(\exp[-\beta E_N] + \exp[-\beta E_D])}$$

$$p_{D_3} = \frac{r_{D_3}}{R} = \frac{\exp[-\beta E_N]}{4(\exp[-\beta E_N] + \exp[-\beta E_D])}$$

$$p_{D_4} = \frac{r_{D_4}}{R} = \frac{\exp[-\beta E_N]}{4(\exp[-\beta E_N] + \exp[-\beta E_D])}$$



Advance the clock, update the configuration, and record whatever properties you are interested in for the new configuration.

Example of KMC - Vacancy Mediated Diffusion

Repeat this process as long as reasonable. At the end of the day, you will have something that looks like:

Step	Time	Configuration
1	Δt_1	C_1
2	$\Delta t_1 + \Delta t_2$	C_2
…	…	…
n	$\sum_{i=1}^{n} \Delta t_i$	C_n

From this, we can compute properties of interest such as diffusion coefficients.

Kinetic Monte Carlo vs. MD

MD: choose a potential, choose boundary conditions, and propagate the classical equations of motion forward in time. If potential is accurate, if electron-phonon coupling (non Born-Oppenheimer behavior) is negligible, then the dynamical evolution will be a very accurate representation of the real physical system.

Limitation is the time steps required by accurate integration (10^{-15} s to resolve atomic vibrations), generally limiting the total simulation time to microseconds.

KMC: attempts to overcome this by exploiting that the long-time dynamics of a system typically consist of jumps from one configuration to another.

In KMC, we do not follow trajectories, but treat the state transitions directly. Time scales are seconds or longer (in fact, achievable time varies with simulation temperature by orders of magnitude).

A key feature of KMC is that the configuration “sits” in some local minimum of configuration space for some time. It then transitions out of that state and into others with the transition rates related to the barriers. It does not matter how the system got into the current state in the first place - it is memoryless, and the process is Markovian.

Kinetic Monte Carlo vs. MD

What about the computational time in KMC?

Limited by searching through the rate catalog for the process that has been selected, so for the most elementary searches, KMC computational time will scale linearly with the number of processes. More sophisticated search algorithms can give log(M) scaling.

Why is KMC not exact?

- Inexact barriers. That is, inexactly computed.

- In fact, the TST rate is not exact (harmonic approximation to the minima & saddle point) but are pretty good (within 10-20%).

- Incomplete rate catalog. This is arguably the biggest problem in KMC. Our intuition cannot often capture surprising reaction pathways, and we neglect relevant physics.

Example: Adatom surface diffusion on Al(100)

After Voter, A.F. Radiation Effects in Solids, 2005:

- Until 1990, diffusion of an adatom on an fcc(100) surface was though to occur by a simple hopping from one site to another
- Feibelman (1990) discovered using density functional calculations that the primary diffusion pathway is quite different

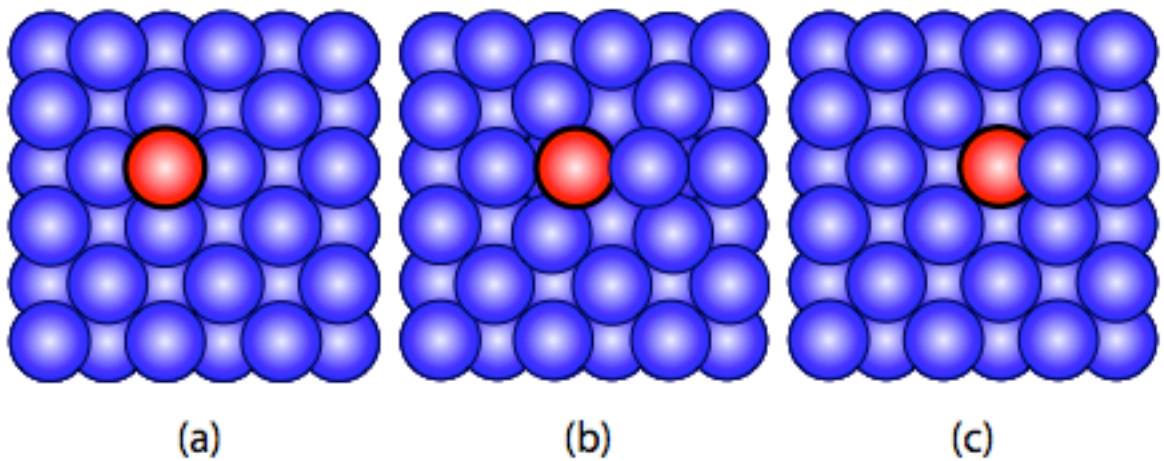


Fig. 6. Exchange mechanism for adatom on fcc(100) surface. (a) initial state; (b) saddle point; (c) final state. This mechanism, unknown until 1990 [52], is the dominant diffusion pathway for some fcc metals, including Al, Pt, and Ir.

- This new mechanism has now been observed for Pt(100) and Ir(100) surfaces via field ion microscopy

Random Number Generation Primer

Generating random numbers effectively and efficiently is a critical part of a good Monte Carlo simulation.

Monte Carlo simulations are subject to systematic errors due to poor random number generation.

Random number generation is still and avid research area - both in developing algorithms for generation and in developing ways to test your sequence for randomness.

For example, a recent concern related to the use of random number generators on massively parallel computers. The sequences on all processors must be distinct and uncorrelated.

Desired characteristic for random number algorithms: uncorrelated, uniform, of extremely long period.

Random Number Generation Primer

No computer can in fact generate a sequence of true random numbers. The computer uses an algorithm (that is in fact deterministic) to select a number, that's why they are often called "pseudorandom" numbers.

A random number generator is typically a routine that produces a very long sequence of pseudorandom numbers; this sequence is, strictly speaking, *periodic*.

This means that, after drawing a sufficiently large number of pseudorandom numbers, the sequence will start and numbers will repeat themselves.

In principle, the fact that a pseudorandom number generator repeats itself is an obviously undesirable feature.

Random Number Generation Primer

However, typical generators have large periods, such as 2^{32} or so, which normally make them suitable to perform reliably even the most demanding Monte Carlo calculations.

The sequence can be started at any point by the selection of an arbitrary initial integer number called *seed*.

A choice of the seed is as good as any other, it is just a matter of determining where in the “chain” one wants to start, all starting points being obviously equivalent.

A random number generator initialized with the same seed will always produce the same random sequence.

Random Number Generation Primer

commonly used algorithms:

- Linear congruential method - most common

  $x_i = MOD\{(x_{i-1}*a+b),m\}$

  The period of this type of generator is limited by m (often chosen to be the largest prime number that can be represented in a given number of bits).

  Even though it is widely used, the linear congruential generator can give very “non-random” results.

- Shift register algorithms

  $x_n = x_{n-p}$ XOR x_{n-q}

  Start with a table of random numbers and generate a new random number by combining two existing numbers from the table.

  Has been demonstrated that the best choices for (p,q) are Mersine primes satisfying $p^2 + q^2 + 1$ = prime.

- Algorithms based on Fibonacci sequences

Random Number Generation Primer

commonly used approaches to testing your algorithm:

Uniformity Test

- the numbers should be uniformly distributed over the range of selection

Overlapping M-Tuple Test

- check statistical properties for the number of times M-tuples of digits appear in the sequence of random numbers

“Parking Lot” Test

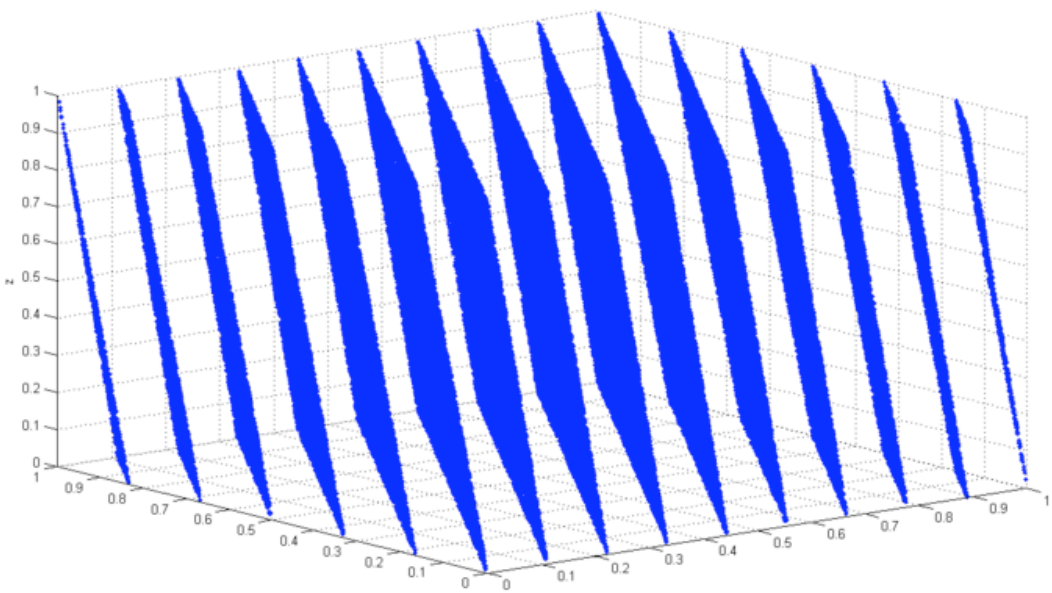
- Plot points in an m-dimensional space where the m-coordinates come from m-successive calls to the generator. Look for patterns.

Random Number Generation Primer

- Example of arguably the lousiest random number generator in history of computers: RANDU
- It is a linear congruential PRNG used since the 1960's into the early 1970's.

$$V_{j+1} = (65539 V_j) \bmod 2^{31}$$

- Subject this kernel to the "parking lot" test:



From the Wikipedia entry on RANDU

Random Number Generation Primer

- "True Random Number Generators" are becoming more in vogue these days
- See, e.g., www.random.org/integers and you can test one out for yourself
- Extracts randomness from physical phenomena (radiation sources, atmospheric noise) and introduces it into a computer
- These are aperiodic and undeterministic -- true random numbers! -- but their efficiency is very poor