LBNL-41775

ERNEST ORLANDO LAWRENCE BERKELEY NATIONAL LABORATORY

The Applicability of Certain Monte Carlo Methods to the Analysis of Interacting Polymers

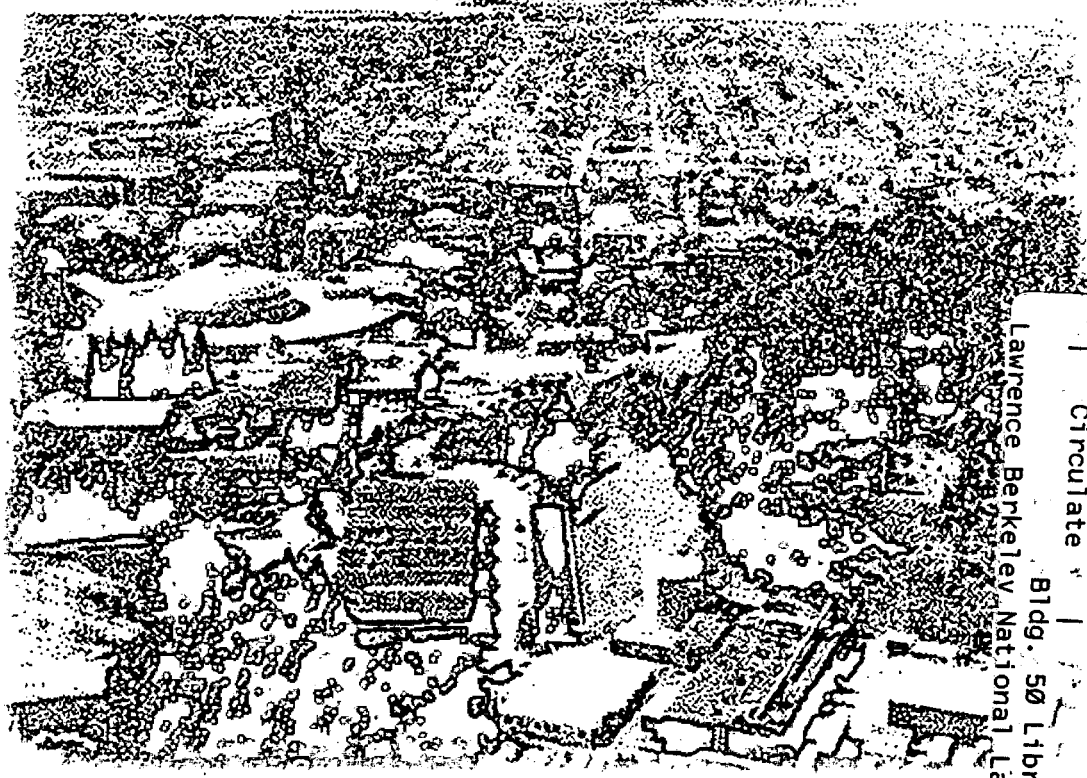
Donald M. Krapp, Jr.

Computing Sciences Directorate

Mathematics Department

May 1998

Ph.D. Thesis



REFERENCE COPY
Does Not
Circulate
Bldg. 50 Library - Ref.
Lawrence Berkeley National Laboratory
LBNL-41775
Copy 1

DISCLAIMER

This document was prepared as an account of work sponsored by the United States Government. While this document is believed to contain correct information, neither the United States Government nor any agency thereof, nor the Regents of the University of California, nor any of their employees, makes any warranty, express or implied, or assumes any legal responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by its trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof, or the Regents of the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof or the Regents of the University of California.

LBNL-41775

THE APPLICABILITY OF CERTAIN MONTE CARLO METHODS TO THE ANALYSIS OF INTERACTING POLYMERS*

Donald Martin Krapp, Jr.
Lawrence Berkeley National Laboratory
University of California
Berkeley, CA 94720, USA

Ph.D. Thesis

May 1998

*This work was supported in part by the Office of Energy Research, Office of Computational and Technology Research, Mathematical, Information, and Computational Sciences Division, Applied Mathematical Sciences Subprogram, of the U.S. Department of Energy, under Contract No. DE-AC03-76SF00098.

The Applicability of Certain Monte Carlo Methods to the Analysis of Interacting Polymers

Copyright © 1998

by

Donald Martin Krapp, Jr.

The U.S. Department of Energy has the right to use this document
for any purpose whatsoever including the right to reproduce
all or any part thereof

Abstract

The Applicability of Certain Monte Carlo Methods to the Analysis of Interacting Polymers

by

Donald Martin Krapp, Jr.

Doctor of Philosophy in Applied Mathematics

University of California at Berkeley

Professor Alexandre Chorin, Chair

We consider polymers, modelled as self-avoiding walks with interactions on a hexagonal lattice, and examine the applicability of certain Monte Carlo methods for estimating their mean properties at equilibrium. Specifically, we use the pivoting algorithm of Madras and Sokal and Metropolis rejection to locate the phase transition, which is known to occur at $\beta_{crit} \approx 0.99$, and to recalculate the known value of the critical exponent $\nu \approx 0.58$ of the system for $\beta = \beta_{crit}$. Although the pivoting-Metropolis algorithm works well for short walks ($N < 300$), for larger N the Metropolis criterion combined with the self-avoidance constraint lead to an unacceptably small acceptance fraction. In addition, the algorithm becomes effectively non-ergodic, getting trapped in valleys whose centers are local energy minima in phase space, leading to

convergence towards different values of ν. We use a variety of tools, e.g. entropy estimation and histograms, to improve the results for large N, but they are only of limited effectiveness. Our estimate of β_{crit} using smaller values of N is 1.01 ± 0.01, and the estimate for ν at this value of β is 0.59 ± 0.005. We conclude that even a seemingly simple system and a Monte Carlo algorithm which satisfies, in principle, ergodicity and detailed balance conditions, can in practice fail to sample phase space accurately and thus not allow accurate estimations of thermal averages. This should serve as a warning to people who use Monte Carlo methods in complicated polymer folding calculations. The structure of the phase space combined with the algorithm itself can lead to surprising behavior, and simply increasing the number of samples in the calculation does not necessarily lead to more accurate results.

Contents

List of Figures v

1 Introduction 1

2 Self-Avoiding Walks and Polymers 6
2.1 Introduction 6
2.2 Self-Avoiding Walks 7
2.3 Polymers 15
2.3.1 Rouse and Zimm Models of a Polymer 27
2.4 Random Walks as Models for Polymers 31

3 Markov Chains, Brownian Motion and Stochastic Differential Equations 34
3.1 Introduction 34
3.2 Probability and Markov Chains 35
3.3 Brownian motion 38
3.4 Stochastic Differential Equations 40

4 Thermodynamics and Phase Transitions 46
4.1 Introduction 46
4.2 Thermodynamics Background 47
4.3 Phase Transitions 51

5 Numerical Methods 66
5.1 Introduction 66
5.2 Generating SAW's 66
5.3 Histograms and Sampling 80
5.4 Entropy Estimation 84

6 Numerical Results **88**
6.1 Introduction 88
6.2 Acceptance fraction and trapping 89
6.3 Weighting the Runs 99
6.4 Estimating β_{crit} and ν 116

7 Conclusions **125**

Bibliography **129**

List of Figures

2.1 Hexagonal lattice with integer coordinates: compared with a lattice formed by regular hexagons (hexagons whose interior angles are all equal), our lattice is contracted by a factor of $\sqrt{3}$ in the y direction. . 7

2.2 Interaction energy E vs. distance from monomer r for the Lennard-Jones potential with $\epsilon = 1.0$ and $\sigma = 0.5$. 22

5.1 N_{NN} and N_{NNN} edges on the square lattice, and (A)-(E) four examples of the edge interaction on the hexagonal lattice: (A)-(D) each have one interaction, (E) has two according to Coniglio, et al. 76

5.2 The original histogram is from a Monte Carlo run of 2.5×10^7 iterations on a two-dimensional hexagonal lattice with $\beta = 0.3$ using the pivoting and Metropolis algorithm described in Section 5.2. The other three curves were generated using the shifted probabilities in Equation 5.15. 84

6.1 Fraction of attempted transformations rejected because of self-intersection vs. length of the walk N for various values of β. For each value of N and β, we first initialized the walks by performing the pivoting algorithm with $T = \infty$ until 10^5 transformations were accepted; we then did then same with $T = \frac{1}{\beta}$. After that, we performed at least 5×10^7 steps of the pivoting algorithm with Metropolis rejection and recorded the number rejections due to self-intersection. We repeated this whole procedure for five separate runs and averaged the results. 90

6.2 Fraction of attempted transformations rejected because of energy vs. length of the walk N for various values of β. The statistics were calculated from the same runs in the same way as in Figure 6.1. Note that the rejection percentage actually decreases with increasing N. See the text for a more detailed discussion of this. 91

6.3 Fraction of attempted transformations accepted vs. length of the walk N for various values of β. Again, the statistics were calculated from the same runs in the same way as in Figure 6.1 and Figure 6.2. 92

6.4 Number of energy rejections/(total iterations-rejections because of self-intersection) vs. length of the walk N for various values of β. Again, the statistics were calculated from the same runs in the same way as in Figure 6.1 and Figure 6.2. 94
6.5 Average end-to-end distance vs. number of iterations for $\beta = 0.9$ and $N = 1500$. In this case, iterations means the number of times we were able to perform a pivoting transformation without self- intersection; the actual number of accepted walks (i.e. after the Metropolis rejection step) was between 1.1×10^6 and 2.5×10^6. See the text for a further discussion of this point. All runs started with the same initial configuration; we first set $\beta = 0.0$ and did 400000 iterations (for $\beta = 0.0$ this means 400000 walks had been accepted), then we set $\beta = 0.9$ and did 400000 more iterations before begining to calculate the end-to-end distance. 95
6.6 Two "trapped" configurations for $N = 16$. 97
6.7 Number of nearest-neighbor interactions for a sequence of walks with $\beta = 0$ and $N = 1000$. In this case, there is no Metropolis rejection and all walks have equal probability. 98
6.8 Transition probability vs. walk number for the same sequence of walks as in Figure 6.7 but with $\beta = 1.0$. Transition probability means in this case the probability that the Metropolis rejection step would allow us to go from walk 0 to walk i through each of the intermediate walks 1, 2, ..., $i - 1$. 99
6.9 Log(transition probability) vs. walk number for the same sequence of walks as in Figure 6.7 but with $\beta = 1.0$. Transition probability means in this case the probability that the Metropolis rejection step would allow us to go from walk 0 to walk i through each of the intermediate walks 1, 2, ..., $i - 1$. 100
6.10 Entropy calculation results for various values of β and m with $N = 1000$; we did 6×10^5 inital iterations, and then calculated the entropy for 2×10^6 iterations of the algorithm. In these cases, the walk segments were 1,...,m and $N - m + 1$,...,N. Note that all of the estimates start out very close to the "equal probability" value $\ln 12 \approx 2.4849$ of the entropy for $m = 3$. We decided to plot the difference of the two estimates instead of the estimates themselves since for some values of β the estimates are so close that the curves for the two segments overlap. 103
6.11 Free energy vs. run for $\beta = 1$, $N = 1500, 2000$. 105
6.12 Entropy vs. run for $\beta = 1$, $N = 1500, 2000$. 106
6.13 End-to-end distance vs. run for $\beta = 1$, $N = 2000$. mean(F) is the mean calculated using Equation 5.23. mean(S) is the mean calculated assuming the average energies E_i of each run are roughly the same. . 108

6.14 Probability of walks vs. energy (as measured by number of nearest neighbor interactions) for $\beta = 1$ and $N = 1000, 1500, 2000$. 109
6.15 Number of walks accepted after self-avoidance check and Metropolis step vs. bound for number of individual walks in Markov chain for $\beta = 1$ and $N = 1000, 2000$. 111
6.16 Number of walks accepted after self-avoidance check and Metropolis step vs. entropy for the runs in Figure 6.11 and Figure 6.12. 113
6.17 Critical exponent ν vs. β for various values of N. Each pair (β, N) had between 5 and 20 runs; for each run i, we did between 400000 and 800000 inital iterations and then computed the average end-to-end distance $R_{ee}(i, N, \beta)$ for between 3×10^7 and 6×10^7 iterations. In this figure, we combined the $R_{ee}(i, N, \beta)$ without any weighting to get $R_{ee}(N, \beta)$. At every value of β, we then fit a line to the plot of $\log R_{ee}(N, \beta)$ vs. $\log N$ using the set of values of N listed in the graph. 114
6.18 Same as Figure 6.17 with different values of N. 115
6.19 Same as Figure 6.17, except that when we combined the $R_{ee}(i, N, \beta)$ to get $R_{ee}(N, \beta)$, we first found the maximum ra_{max} of all of the ra_i for each pair (N, β) and eliminated the runs whose ra_i was less than $0.8ra_{max}$; we then used the remaining runs to calculate the average. The reasoning behind this is that if we look back at Figure 6.16, we see that since r_a^i reflects the entropy S_i, then those runs with small r_a^i and correspondingly small S_i will make a negligable contribution to the overall average. What about weighting the remaining runs after eliminating those with small r_a^i? Well, in this Figure we give them all equal weight, but in Figure 6.21 we weight them using the r_a^i (recall that the entropy method in effect just selects out one of the runs and gives it a weight disproportionately large compared to those of the other runs. 117
6.20 Same as Figure 6.19 with different values of N. 118
6.21 Same as Figure 6.19, except that when we combined the $R_{ee}(i, N, \beta)$ to get $R_{ee}(N, \beta)$, we weighted each according to the number of walks r_a^i accepted after the Metropolis step. This corresponds to Equation 6.1. 119
6.22 Same as Figure 6.21, except that we used Equations 6.2 when doing the weighting. 120
6.23 Same as Figure 6.19, except that we used the histogram method to fill in the curves near $\beta = 1$. 121
6.24 The exponent ν from Equation 6.3 as a function of $\frac{1}{N}$. We set $M = 10$ and tried this for a number of β in the vicinity of our estimate of β_{crit}. 123
6.25 The exponent ν from Equation 6.3 as a function of $\frac{1}{N}$ using the radius of gyration R_g in place of R_{ee}. Again, $M = 10$. 124

Acknowledgements

I would like to thank my family, my advisor Alexandre Chorin, the National Science Foundation, the Mathematics Department at the University of California at Berkeley, the members of the Mathematics Department at Lawrence Berkeley National Labs, and the Department of Energy. Without the help and support of these people and organizations, I would not have had the privilege of studying at Berkeley, nor would I have been able to write this thesis. Thank you.

This work was supported in part by the Office of Energy Research, Office of Computational and Technology Research, Mathematical, Information, and Computational Sciences Division, Applied Mathematical Sciences Subprogram, of the U.S. Department of Energy, under Contract No. DE-AC03-76SF00098, and by a pre-doctoral Applied Mathematics Fellowship from the National Science Foundation.

Chapter 1

Introduction

We begin with a brief introduction to this thesis and a broad outline of the main topics which will be discussed in more detail in the following chapters.

The main objective of this thesis is to examine the applicability of Monte Carlo methods and standard numerical techniques to an apparently simple problem which contains some of the difficulties associated with other, more complicated problems – such as polymer folding – that are currently of interest in physics, biology and chemistry. The example we consider is the following: self-avoiding walks (SAW's) on a hexagonal lattice in two dimensions with a nearest-neighbor interaction. We define all of these concepts in Section 2.2, where we also define universality classes and critical exponents.

Our example is important because it is a prototype of problems like polymer folding; in fact, there is an intimate connection between polymers and self-avoiding

walks. In Section 2.3 we describe some of the characteristics of polymers and introduce several models of polymers such as the Rouse and Zimm models (see Section 2.3.1). We then illustrate in Section 2.4 how certain self-avoiding walks can model certain kinds of polymers.

In Section 3.2 we briefly summarize the probability theory that we need in our computations; we also give definitions of a Markov process and a Markov chain. Since the Rouse and Zimm models mentioned above are formulated as stochastic differential equations, we give an introduction to this theory and a rigorous definition of a stochastic differential equation in Section 3.3 and Section 3.4.

For an arbitrary but fixed positive integer N, the interacting SAW's of length N on a hexagonal lattice form our system. Throughout much of the thesis we will be attempting to compute various "thermal averages" for this system, such as the average end-to-end distance. In Section 4.2 we define exactly what this means; we also define and discuss thermodynamics concepts such as the partition function, the free energy, and the entropy.

The way we test the applicability of Monte Carlo methods to our system is by trying to locate the known phase transition which takes place for these interacting walks at a certain critical inverse temperature $\beta = \beta_{crit} \approx 0.99$. What is a phase transition, and what does "critical inverse temperature" mean? These concepts are defined in Section 4.3, where we also give an extended example of such a transition and discuss some of the difficulties encountered when one tries to model a phase

transition on a computer.

How do we actually try to detect such a transition? The algorithm we use consists of Madras-Sokal pivoting combined with Metropolis rejection (see Section 5.2); this algorithm allows us to generate self-avoiding walks efficiently and to use these to estimate thermal averages. In addition to pivoting and Metropolis rejection, we also use various numerical techniques such as histograms (see Section 5.3) and entropy estimation (see Section 5.4) to help reduce the number of calculations and compute averages more accurately.

Even for this seemingly simple system, we encounter a number of difficulties which are described in detail in Section 6.2. Two of these are unacceptably small acceptance percentages and trapping in valleys centered around local energy minima in phase space.

Both of these problems occur for medium-length SAW's (SAW's with length $N \approx 300$) and for inverse temperatures β near $\beta_{crit} \approx 0.99$. The histogram method is useful here since it allows us to use the data from one run at a given β_0 to estimate the thermal averages for other values of β near β_0; this considerably reduces the number of calculations necessary near β_{crit}. However, we cannot use this method to go too far away from the β_0 at which the original run is done (see Section 5.4).

To increase the number of acceptances, we cannot simply do more iterations of the pivoting-Metorpolis algorithm. Because the algorithm becomes trapped in valleys in phase space, for different runs there is a convergence towards different thermal

averages (see Section 6.2). One way out of this difficulty is to do several runs; we can hope that in this way we might be able to explore a larger portion of phase space. The problem then arises as to how to put the results from the various runs together.

The entropy method is useful in helping to weight the thermal averages from various runs. However, it too has its limitations: the entropy estimations gives us an indication of which runs might be more important, but for long enough SAW's (length $N > 800$), we still cannot get accurate values of the critical exponent (see Section 6.3).

Nevertheless, the entropy method does prove useful. We discover a direct relationship between our entropy estimate for a given run and the number of walks r_a accepted after the Metropolis rejection step. The best results are obtained for relatively short SAW's by doing several runs and using r_a to weight the averages from the runs. Using these data, our estimates for the critical temperature and critical exponent are $\beta_{crit} = 1.01 \pm 0.01$ and $\nu = 0.59 \pm 0.005$. These are in good agreement with the previous values $\beta_{crit} \approx 0.99$ (based on numerical work by Coniglio, et al. [14]) and $\nu \approx 0.58$ (from theoretical calculations, numerical work, and polymer experiments described in Section 6.4). However, the reason we are able to decide which size SAW's to use in these estimates of ν and β_{crit} is that we already knew from previous work what the values of ν and β_{crit} should be. Thus, our algorithm is only partially successful, and if we did not know the values of ν and β_{crit} it would be extremely difficult to decide for which values of N the Monte Carlo method yields

accurate thermal averages.

What do we learn from this? Even with supposedly simple interactions and a powerful algorithm such as pivoting with Metropolis rejection, it is not necessarily true that one can compute accurate thermal averages by Monte Carlo sampling. This should serve as a warning to people who use Monte Carlo in complicated polymer folding calculations. The structure of the phase space combined with the algorithm itself can lead to surprising and complicated behavior, and simply increasing the number of samples in a Monte Carlo calculation does not necessarily lead to more accurate estimates of thermal averages.

Chapter 2

Self-Avoiding Walks and Polymers

2.1 Introduction

In this chapter we will give a short introduction to self-avoiding walks (SAW's), critical exponents and universality classes, and discuss various types of non-interacting and interacting walks. Then we will move on to a discussion of polymers, including the excluded volume and hydrodynamic effects and the effect of the quality of the solvent on the polymer, as well as the excluded volume and the Θ point; we will also talk about several well-know models of polymers (for example, the Rouse and Zimm models), the assumptions which these models make, and their advantages and disadvantages. Finally, we will briefly discuss the connection between self-avoiding walks and polymers.

2.2 Self-Avoiding Walks

Most of the following discussion is based on Lawler [41] and Grimmett [27].

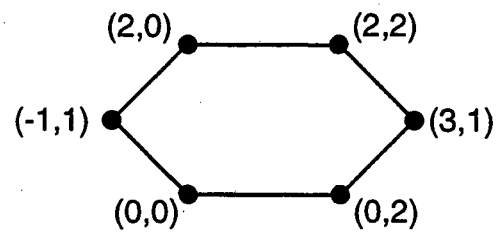


Figure 2.1: Hexagonal lattice with integer coordinates: compared with a lattice formed by regular hexagons (hexagons whose interior angles are all equal), our lattice is contracted by a factor of $\sqrt{3}$ in the y direction.

We first begin with a few definitions. For two points $x = (x_1, \ldots, x_d)$ and $y = (y_1, \ldots, y_d)$ in $\Re^d$, define $d_i(x,y) = |x_i - y_i|$, and let $d_d(x,y) = \sum_{i=1}^{d} d_i(x,y)$. Suppose $a_1, a_2, a_3 \in \mathcal{Z}$, $p_1 = (0,2)$, $p_2 = (3,1)$, $p_3 = (-3,1)$, and $h \in \{(0,0),(2,0),(-1,1)\}$. The hexagonal lattice is the graph consisting of the set $\mathcal{H}$ of all points (called vertices) in $\Re^2$ of the form $a_1p_1 + a_2p_2 + a_3p_3 + h$ together with edges connecting all points $x, y \in \mathcal{H}$ such that either i) $d_1(x,y) = 2$ and $d_2(x,y) = 0$, or ii) $d_1(x,y) = 1$ and $d_2(x,y) = 1$. This is illustrated in Figure 2.1. The square lattice is $\mathcal{Z}^2$ with edges between two points x, y if $d_m(x,y) = 1$; similarly, the cubic lattice is $\mathcal{Z}^3$, again with edges between all x, y such that $d_m(x,y) = 1$. Two points x, y connected by an edge are called adjacent, and the edge between two such points is denoted by $< x, y >$.

The coordination number l of a lattice is the number of edges connected to each point on the lattice: for a square lattice, $l = 4$, for a cubic, $l = 6$, and for a hexagonal, $l = 3$. A path is an alternating sequence of distinct vertices and edges x_0, e_0, x_1, e_1,

$\ldots$, e_{n-1}, x_n with $e_i =< x_i, x_{i+1} >$; the length of such a path is n, and the path is said to connect x_0 to x_n. A circuit is an alternating sequence of vertices and edges x_0, e_0, x_1, e_1, $\ldots$, e_{n-1}, x_n, e_n, x_0 such that x_0,e_0,$\ldots$,x_{n-1} is a path, and e_n is the edge between x_0 and x_n. We can view a two dimensional lattice as a subset of $\Re^2$: a plaquette is the subset of $\Re^2$ enclosed by a circuit of minimal length. For example, on the hexagonal lattice such circuits are of length 6 and plaquettes are hexagons; on the square lattice, these circuits have length 4, and plaquettes are squares.

A self-avoiding walk of length N is a sequence of points ω_0, ω_1, ..., $\omega_N \in \mathcal{Z}^d$ such that $\omega_i \neq \omega_j$ for $0 \leq i < j \leq N$. $|\omega_i - \omega_j| = 1$ if we are on a square or a cubic lattice; on a hexagonal lattice in two dimensions this distance can be either 2 or $\sqrt{2}$ (see Figure 2.1). Now, let $A(l, N)$ be the number of SAW's for a given l and N: what can we say about A? If we allow self-intersection, the number of such walks is l^N, so $A(l, N) < l^N$; since the SAW cannot go back to the point it most recently visited, $A(l, N) \leq l(l-1)^{N-1}$. Also, if we only take non-negative steps in each direction, then this will generate an SAW, and there are $(\lceil \frac{l}{2} \rceil)^N$ of these (where $\lceil y \rceil$ is the smallest integer greater than y). Thus, $A(l, N) \geq (\lceil \frac{l}{2} \rceil)^N$.

Suppose $B(N)$ is some function of N. We define $B(N) \asymp C^N$ to mean

$$\lim_{N \to \infty} \frac{\ln B(N)}{N \ln(C)} = 1 \tag{2.1}$$

We can then say the following about the behavior of $A(l, N)$ for large N:

Theorem 1 $\exists \tau \in [\lceil \frac{l}{2} \rceil, l-1]$ *such that* $A(l, N) \asymp \tau^N$ *where* τ, *which is called the connectivity constant, depends on the lattice and the dimension.*

See Lawler [41] for the proof.

So the number of SAW's grows exponentially with the length N. Well, not quite. The above theorem does not tell us the whole story: it only says that $A(I,N) = \tau^N r(N)$ where $\lim_{N\to\infty}[r(N)]^{\frac{1}{N}} = 1$. The conjectured behavior of this $r(N)$ (called the enhancement factor) is as follows:

$$r(N) = \begin{cases} N^{\gamma-1} & d < 4 \\ (\ln N)^{-a} & d = 4 \\ C & d > 4 \end{cases} \tag{2.2}$$

where C is a constant independent of N, $a = \frac{1}{4}$, and γ is the susceptibility exponent which depends only on the dimension of the lattice d (for $d = 1$, $\gamma = 1$; $d = 2$, $\gamma = \frac{43}{32}$; $d = 3$, $\gamma = 1.16$) and not on the lattice's detailed structure.

What happens if we require the SAW to return to the origin (i.e. $\omega_0 = \omega_N$)? Let the number of such SAP's (self-avoiding polygons) be $B(l,n)$. Hammersley [28] proved that

$$B(l,N) \asymp \tau^N \tag{2.3}$$

It is also conjectured that

$$\frac{B(l,N)}{A(l,N)} \asymp N^{-\delta} \tag{2.4}$$

although this has not yet been proven. The exponent δ is one example of what are called critical exponents (γ in Equation 2.2 is another example), each of which describes the behavior of some function of our SAW's as the length N approaches ∞; see Section 4.3 for a more in-depth discussion of these exponents. Although the

connectivity constant τ is the same for both SAW's and SAP's, for SAP's there is no enhancement factor $r(N)$; in fact, $r(N)$ in Equation 2.2 comes from the fact that the ends of an SAW are less "constrained" by self-avoidance than the more central parts of the walk (see de Gennes [15] for more details).

In the next few paragraphs, we will be talking about other critical exponents of SAW's defined in terms of averages of some quantities (such as end-to-end distance). Before doing this, we should specify what probability measure we will be using to compute these averages. There are two cases to distinguish: interacting and non-interacting SAW's. We can define non-interacting walks as those for which the measure is the uniform measure: in other words, each SAW has equal probability. For interacting SAW's, we need to define what exactly an interaction is. One way this can be done is as follows: if the points in an SAW of length N are labelled $x_0,\ldots,x_N$, we count the number N_{NN} of pairs (x_i, x_j) (where $i+2<j$) that share a common edge on the lattice; we then give this SAW a probability $\exp(\beta N_{NN})$ where β is a numerical parameter (see Section 4.2). There are other definitions of the interaction which involve, for example, counting the number of pairs (x_i, x_j) which can be connected by a path with two edges on the lattice. We refer the reader to Section 5.2 for more detailed examples of such interactions.

In addition to the δ defined above, we can also examine other critical exponents which describe both interacting and non-interacting SAW's. For our discussion, the most important of these concerns the average of the squared end-to-end distance of

the SAW $< |\omega_N - \omega_0|^2 >$. The mean square displacement exponent ν is defined by

$$< |\omega_N - \omega_0|^2 > \asymp N^{2\nu} \tag{2.5}$$

where the measure used in the average $<>$ depends on whether the SAW's are interacting or not.

If we allow self-intersections, then $\nu = \frac{1}{2}$. For interacting SAW's, the beginning of Section 4.3 contains a description of one case; for non-interacting SAW's, the following is the dependence of ν on the dimension d of the lattice as argued by Flory [25]

$$\nu = \begin{cases} \frac{3}{(d+2)} & d \leq 4 \\ \frac{1}{2} & d > 4 \end{cases} \tag{2.6}$$

Although Flory's argument does not constitute a proof, the formula is correct for $d = 1$ and believed to be exact for $d = 2$. Numerical simulations suggest that for $d = 3$, $\nu = .59...$ (see Coniglio, et al. [14] for details); for $d = 4$, it is expected that

$$< |\omega_N - \omega_0|^2 > \asymp N(\ln N)^{\frac{1}{4}} \tag{2.7}$$

Before continuing, we should point out that none of the above arguments actually proves the existence of the critical exponents ν and γ. In fact, surprisingly little is known theoretically about these exponents; for example, up to now the existence of ν has only been proven for dimensions $d = 1$ and $d > 4$ (see Slade [63] and Slade [64]). Thus, when we say "believed to be exact", we are referring to computational results; the reader should keep this in mind during the following discussion.

Consider a set of points S in $\Re^2$. One way to define the dimension of S is the following: cover $\Re^2$ with non-overlapping squares of unit area, and count how many of these squares intersect S; call this number n_1. Then cover $\Re^2$ with squares of area $\frac{1}{2}$, and let n_1 be the number of such squares which intersect S. Continue this procedure with squares of area $\frac{1}{3}$, $\frac{1}{4}$, ... The dimension d_S of S is defined to be

$$d_S = \lim_{m \to \infty} \frac{\ln n_m}{\ln m} \tag{2.8}$$

This is called the boxcounting dimension. For other definitions of dimension and ways to actually compute it see Peitgen, Jurgens and Saupe [57], Peitgen and Saupe [58] and Parker and Chua [56].

For $d \geq 5$, $\nu = \frac{1}{2}$, so SAW's and plain random walks have the same dimension: $d_{crit} = 4$ is called the critical dimension for the SAW. The existence of such a d_{crit} in Equation 2.6 and Equation 2.2 tells us that the geometric constraint of self-avoidance doesn't play a very big role in the dynamics if we are in a high enough dimension.

In addition to the SAW described above, there are other walks on lattices which allow varying degrees of self-intersection. For example, consider the set Λ_N of all random walks of length N, let $J(\omega)$ be the number of self-intersections of the walk ω, and give each walk ω a probability

$$U_\beta(\omega) = \frac{\exp(-\beta J(\omega))}{< \exp(-\beta J) >} \tag{2.9}$$

where the average $<>$ is taken over all $\omega \in \Lambda_N$. When $\beta = 0$, we have a simple random walk; when $\beta \to \infty$ we get the SAW. This probability measure is called the

weakly self-avoiding walk, and it is conjectured (but not yet proven) that the critical exponents for the weakly self-avoiding walk are the same as for the SAW. I.e. if we define $\nu(\beta)$ by

$$< |\omega_N - \omega_0|^2 >_{U_\beta} \asymp N^{2\nu(\beta)} \tag{2.10}$$

then $\nu(\beta) = \nu$ if $\beta > 0$.

There is also another set of walks, called k-tolerant, which belongs to the same universality class as the weakly self-avoiding walks (i.e. it is conjectured that both types of walks have the same critical exponents). A walk is k-tolerant if it returns to the same point up to but not more than k times; for example, the standard SAW is a 1-tolerant walk. Instead of having a probability $U_\beta(\omega)$ which decreases exponentially with the number of self-intersections, we instead weight equally all walks in Λ_N which are k-tolerant, and those walks which do not are given a probability of 0.

Why would all of these different kinds of walks be in the same class? After all, if we consider, say, a 6×10^{200}-tolerant walk, how could this possibly look like an SAW (i.e. a 1-tolerant walk)? Firstly, we should be clear about the phrase "looks like". Fix k and suppose we model our k-tolerant walk on the computer; if the length of our walk $N \approx k$, then the behavior of our k-tolerant walk will be significantly different from that of an SAW. In fact, if k is really big (e.g. 6×10^{200}), then for any walk size which we can model on our computer, the k-tolerant walk will look like a random walk. But remember the definition of the symbol $\asymp$: we want to look at the walk as $N \to \infty$. So, what we really should be doing is fixing k and letting N get much

bigger than k: the conjecture then is that all of the k-tolerant walks (and all of the weakly self-avoiding walks) are asymptotically the same – in other words, they are in the same universality class with the same critical exponents. So the answer to the second question posed at the beginning of the paragraph is that for N large enough, a 6×10^{200}-tolerant walk and an SAW do have the same critical exponents, but from a computational point of view, if we wanted to show this we would need to use walks of length $N \gg 6 \times 10^{200}$, which is not possible on the computers of today. If we were to use an N much smaller than 6×10^{200}, the calculated critical exponents would not be the same as those for an SAW.

What about the first question (i.e. the justification for assuming this asymptotic equivalence)? Again, we do not have any proofs, but numerical evidence and heuristic arguments (see Oono [51], Shapir and Oono [62], Malakis [44], and Oono and Freed [52]) lend at least some credence to the claims above.

At this point, the reader may be wondering whether or not we can find any kind of quasi/weakly/tolerant self-avoiding walk which has different critical exponents from the walks already discussed. The answer is that we can. Suppose, instead of weighting each walk in Λ_N, we decide to start with a given walk ω of length N and extend it, choosing the next site x among the nearest neighbors of the last point in the walk ω_N with a probability based on the number of times we have already visited it $V(x)$. Let this probability be

$$\frac{\exp(-\beta V(x))}{\sum_y \exp(-\beta V(y))} \tag{2.11}$$

where the sum is over the nearest neighbors of ω_N. This is called the myopic self-avoiding walk (myopic because it only looks at nearest neighbors before deciding where to go next), and, although it has not yet been proven, numerical work gives the following result for the critical exponent ν:

$$\nu = \begin{cases} \frac{2}{3} & d = 1 \\ \frac{1}{2} & d > 1 \end{cases} \tag{2.12}$$

with a possible log correction for $d = 2$ (see Equation 2.7). Thus, $d_{crit} = 2$, and so both the exponent ν and d_{crit} are different from the previously described SAW.

In fact, we might expect a priori that the myopic walk belongs to a different universality class, since: 1) we are using only local information at ω_N to decide our next move; 2) unlike the weakly self-avoiding walks, the probability of a particular walk varies not only according to the total number of self-intersections but also according to their distribution. For example, a myopic walk with r intersections at one point has in general a different weight than a myopic walk with one intersection at r different points; however, if we use the weak SAW measure, both walks have the same weight.

2.3 Polymers

Now that we have examined SAW's, we take a closer look at polymers. Most of the discussion below can be found (with considerably more details) in Doi and Edwards [17] and de Gennes [15]. In what follows, everything will be in three dimensions unless otherwise indicated.

What exactly is a polymer? We can think of it as a long chain of molecules (called monomers) linked together by some sort of chemical bond. This is the definition of a polymer, but what about the physical mechanism behind the formation of polymers? Polymers form when the condition required to add one molecule to a chemical system is almost independent of the systems's size. For example, going from one molecule $(A)_1$ to $(A)_1-(A)_2$ may be different from going from $(A)_1-(A)_2$ to $(A)_1-(A)_2-(A)_3$; however, going from $(A)_1-...-(A)_n$ $(n \gg 1)$ to $(A)_1-...-(A)_{n+1}$ in many cases is almost identical to going from $(A)_1-...-(A)_{n+1}$ to $(A)_1-...-(A)_{n+2}$; this process is called a polymerization reaction and can lead to extremely long chains.

The bonds between the monomers determine the general shape of the polymer, and we can classify polymers as either rigid or flexible based on these bonds. What exactly does flexibility mean? Suppose we choose a given monomer m and fix the positions of its neighbors $m-1$ and $m+1$. Next, we find the points in space where the energy of m has a local minimum. If the barriers between these energy minima are small compared to a given temperature T, then we say the polymer is dynamically flexible at this temperature; if the difference in energy between each of these minima is small (again compared to T), then the polymer is called statically flexible. Of course, this definition is not exact, and there are polymers which are flexible in only one sense, rigid in both senses, etc. In the following sections we will be concerned chiefly with polymers that are both dynamically and statically flexible.

What about the individual monomers? Although their molecular weight deter-

mines that of the polymer, and although the bonds between successive monomers are what keeps the polymer together, in some cases the exact shape of the monomers does not play a large role in determining the macroscopic properties of the polymer. For example, take two polymers of the same length N and conformation, but whose monomers are in one case circles and in the other case triangles: the difference in viscosity between these two scales as $N^{-\frac{1}{2}}$ (see Doi and Edwards [17] for the definition of viscosity and the proof of this scaling law), so for a long enough polymer, from the point of view of viscosity it does not matter if the monomers are circles or squares. Warning: we do *not* mean to say that the presence of monomers does not affect the dynamics of the polymer: the excluded volume effect, which is caused by the short-range repulsion between the monomers, does have a significant influence on the dynamics (see the discussion below).

Now suppose that we take a bunch of identical polymers and put them in some kind of fluid (this fluid is called the solvent). What will happen is determined by a number of things. The interaction between the polymers and the solvent is one factor, and based upon this interaction the solvent is called either good or poor. Good means the solvent molecules are attracted to the polymer over a wide range of temperatures, and in a good solvent each segment of the polymer tends to contact solvent molecules rather than other segments; poor means the polymers do not interact strongly with the solvent and tend to precipitate out of the solvent when the temperature is changed or the polymer concentration is increased.

We can get a more quantitative description of the quality of a solvent (see de Gennes [15]) as follows. Consider a "polymer-solvent" solution on a cubic lattice in three dimensions; let the fraction of sites occupied by monomers be Φ, and let the volume of a lattice cube be a^3. The amount of volume around each monomer that cannot be occupied by other monomers will be denoted by v (this is called the excluded volume). Also, recall that the free energy F is defined to be $E - TS$ where E is the energy, T is the temperature, and S is the entropy. For E, the relevant parameters are χ_{MS}, χ_{MM}, and χ_{SS}, which denote the energy of the monomer-solvent, monomer-monomer, and solvent-solvent interactions respectively.

For low concentrations Φ, at each lattice site the free energy of mixing, which is defined to be the free energy of the solution minus the weighted average of the free energies of the pure polymer and pure solvent, can be written as

$$\frac{F_{mixing}}{T} = \frac{\Phi}{N}\ln N + \frac{1}{2}\Phi^2(1-2\chi) + \frac{1}{6}\Phi^3 + ... \tag{2.13}$$

where the Flory interaction parameter χ is defined as

$$\chi = \chi_{MS} - \frac{1}{2}(\chi_{MM} + \chi_{SS}) \tag{2.14}$$

It should be noted that χ is usually an increasing function of the temperature, although the exact dependence is very complicated (see de Gennes [15] for the derivation of Equation 2.13).

The coefficient of the Φ^2 term in Equation 2.13 can be interpreted as the strength of the interaction between monomers; this gives us an expression for the excluded

volume v, namely $a^3(1-2\chi)$. When χ is near zero, the solvent is similar to the monomer and $v \approx a^3$; this is what is meant by a good solvent. Poor solvents, on the other hand, have a relatively large χ, and so $v < 0$; the borderline case is $\chi = \frac{1}{2}$ and $v = 0$.

In addition to the quality of the solvent, another factor determining the behavior of our polymer-solvent soup is the concentration of the polymers: depending upon whether this is very high, medium, or very small, the solution is called respectively concentrated, semidilute, or dilute. A more quantitative definition goes as follows: for a polymer of length N, let $\mathbf{R}_m$ be the location of the m-th monomer. We define the center of mass $\mathbf{R}_G$ by

$$\mathbf{R}_G = \frac{1}{N+1} \sum_{m=0}^{N} \mathbf{R}_m \tag{2.15}$$

and the radius of gyration R_g by

$$R_g^2 = \frac{1}{N+1} \sum_{m=0}^{N} |\mathbf{R}_m - \mathbf{R}_G|^2 \quad . \tag{2.16}$$

Let C be the average distance between the center of masses of the polymers. We then say that the solution is dilute if $C \gg R_g$; if $C \approx R_g$, the solution is semidilute; and if $C \ll R_g$ then the solution is concentrated. In our work, we will be considering dilute solutions: in a good solvent, this means the effect of the interaction between different polymers is negligible compared with the solvent-polymer interaction.

We are now going to take a closer look at one of the polymers swimming around in the solvent. A number of models have been proposed to describe the dynamics of this polymer: two of the simplest are the freely-jointed chain and the Gaussian

chain. The freely-jointed chain model considers the polymer as a chain of N links, each of fixed length a and able to point in any direction independently of each other. For the Gaussian chain model, the lengths of the bonds have independent Gaussian distributions, all of which have the same mean and variance. For both models, the end-to-end vector of the polymer is a sum of the form $\sum_{i=1}^{N} \mathbf{r}_i$ where the $\{\mathbf{r}_i\}$ are randomly oriented vectors either of fixed length a (for the freely-jointed chain) or whose lengths have the appropriate Gaussian distribution (for the Gaussian chain).

If we define the average squared end-to-end distance r_N by

$$r_N =< |\mathbf{R}_N - \mathbf{R}_0|^2 >^{\frac{1}{2}} \tag{2.17}$$

then it turns out that for both models

$$r_N^2 = bN \tag{2.18}$$

where b is a (model-dependent) constant called the effective bond length; in fact, in the calculation of r_N, the local structure of the chain affects only this constant b — it does not affect the power of N on the right hand side of Equation 2.18.

The following is an example of this calculation for the freely-jointed chain taken from de Gennes [15].

$$\begin{aligned} < (\mathbf{R}_N - \mathbf{R}_0)(\mathbf{R}_N - \mathbf{R}_0) > &= \sum_{i=1}^{N}\sum_{j=1}^{N} < \mathbf{r}_i \dot{\mathbf{r}}_j > \\ &= \sum_{i=1}^{N} < \mathbf{r}_i \dot{\mathbf{r}}_i > \\ &= Na^2 = bN \end{aligned} \tag{2.19}$$

where $b = a^2$. Since the orientations are independent, we have

$$< \mathbf{r}_i \dot{\mathbf{r}}_j >=< \mathbf{r}_i >< \mathbf{r}_j >= 0 \tag{2.20}$$

unless $i = j$. In fact, even if there is an interaction between the segments i,$i\pm1$,...,$i\pm p$ (where p is fixed and finite), we still have $r_N^2 \approx bN$ as long as N is large enough (see de Gennes [15] for details).

However, both the freely-jointed chain and the Gaussian chain have neglected a fact which significantly changes the dynamics of the polymer: since each segment of the polymer has some finite volume, a given segment cannot enter the region already occupied by other segments. This makes the monomer-monomer interaction considerably more complicated; additionally, since our polymers are immersed in a solvent, when considering the force between monomers the solvent-monomer interaction must also be taken into account. The effective inter-monomer force then consists of a short-range repulsive interaction along with an attractive tail (see Oono [51] for more details). One frequently used model of this is the Lennard-Jones potential, given by

$$E = 4\epsilon((\frac{\sigma}{r})^1 2 - (\frac{\sigma}{r})^6) \tag{2.21}$$

where ϵ is the minimum value of the energy and σ determines the length of the short-range repulsive interaction. An illustration of this is given in Figure 2.2.

One can see in Figure 2.2 that the interaction energy E can be naturally decomposed into two parts: repulsive and attractive. In general, the interaction potential ϕ of the repulsive interaction between monomers m and l is usually assumed to have

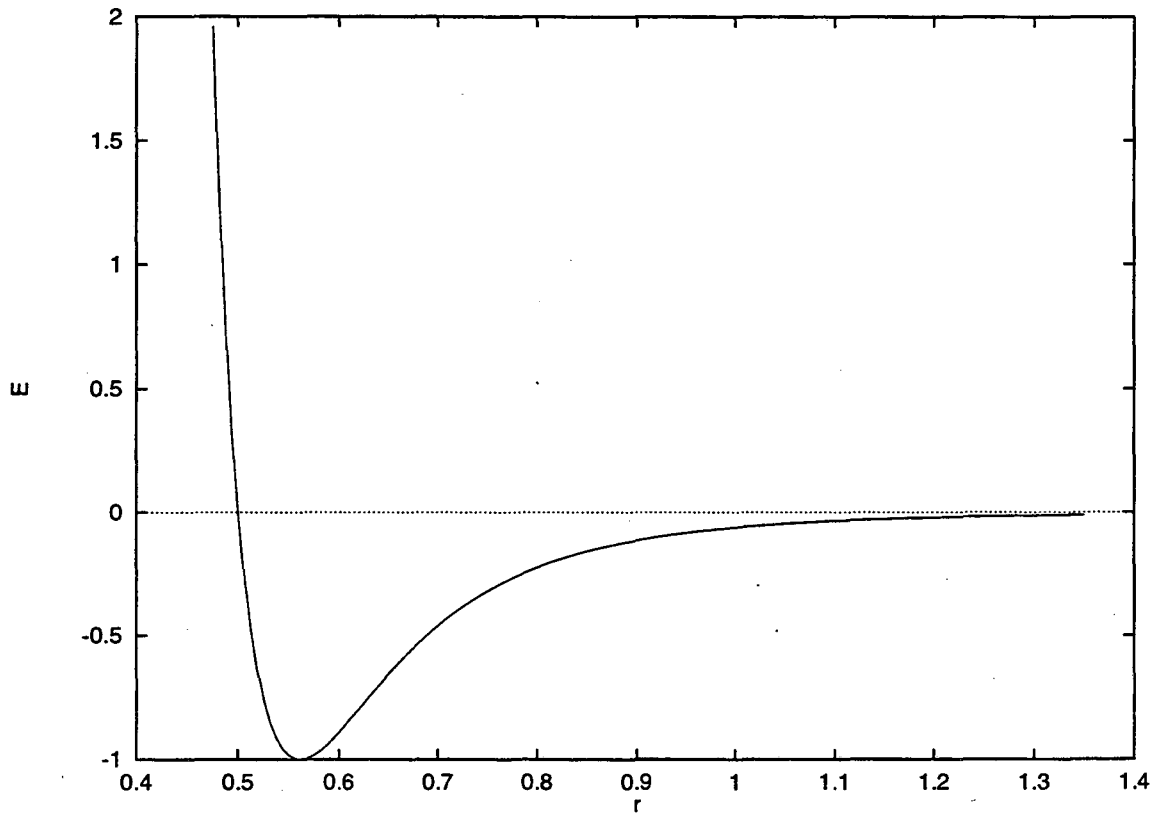


Figure 2.2: Interaction energy E vs. distance from monomer r for the Lennard-Jones potential with $\epsilon = 1.0$ and $\sigma = 0.5$.

the form

$$\phi(\mathbf{R}_m - \mathbf{R}_l) = vk_BT\delta(\mathbf{R}_m - \mathbf{R}_l) \tag{2.22}$$

where v is the excluded volume mentioned above; if the reader does not like the delta function here, we can replace it with some smooth positive function with a peak at the origin, which decays rapidly away from the origin, and whose integral over $\Re^d$ is one.

When looking at the global properties of the polymer ω (such as end-to-end distance), it is not necessary to examine in detail the local monomer structure. In fact, instead of the discrete potential for the monomer-monomer interaction

$$V(\omega) = \frac{1}{2}\sum_{\substack{l,m=0\\ l\neq m}}^{N} vk_BT\delta(\mathbf{R}_m - \mathbf{R}_l) \tag{2.23}$$

(where the $l = m$ monomer self-interaction term has been omitted), the following continuous version is often used

$$V(\omega) = \frac{1}{2}vk_BT\int^{\tau=N}\int_{|\tau-\sigma|>a}^{\sigma=N} d\tau d\sigma\delta(\mathbf{c}(\tau) - \mathbf{c}(\sigma)) \tag{2.24}$$

where $\mathbf{c}(\tau)$ is the polymer curve paramaterized by τ, and where a is the cutoff parameter which corresponds to the condition $l \neq m$ in Equation 2.23. See Oono [51] for a detailed discussion of the continuous version and the parameter a.

For a given polymer and solvent, v is a function of temperature; the temperature $T = T_\Theta$ for which $v = 0$ is called the Θ- or Flory temperature, and at $T = T_\Theta$ the polymer is in an intermediate state between the scrunched-up phase at low temperatures and the stretched-out phase at higher temperatures.

When there is a short-range repulsive effect, the polymer swells and stretches out more than a polymer of the same number of monomers which has no such repulsive interaction: this effect is called the excluded volume effect, and one result is that r_N in Equation 2.17 is larger for a given N.

The excluded volume effect takes into account the interaction between segments which are not necessarily near each other along the polymer chain: this is one example of what is called a 'long'-range or global interaction. We should be careful here: 'long' in this case does not refer to the spatial range of the interaction but to the distance between the interacting monomers along the polymer chain (called contour distance). How can we define long and short (or global and local)? The answer is that there is in general no unambiguous definition (see Oono [51]). However, for our purposes we do not need to draw a very fine distinction between the two, and we can avoid any difficulties by defining a long-range interaction as one which involves segments of the polymer whose contour distance is of the order of the length of the polymer (the excluded volume effect fits this criterion); short-range will be everything else.

There is another phenomenon we would like to discuss before moving on to more detailed models of polymers: it is called the hydrodynamic interaction. To understand this, let us look at the motion of the monomers a little more closely. We denote the linear velocities of the monomers $0, 1, ..., N$ by $\mathbf{V}_0, \mathbf{V}_1, ..., \mathbf{V}_N$, their angular velocities by $\mathbf{W}_0, \mathbf{W}_1, ..., \mathbf{W}_N$, and the forces and torques acting on them by $\mathbf{F}_0, \mathbf{F}_1, ..., \mathbf{F}_N$ and

$\mathbf{T}_0, \mathbf{T}_1, ..., \mathbf{T}_N$. The equation of motion of the monomers then has the general form

$$\begin{pmatrix} \mathbf{V}_m \\ \mathbf{W}_m \end{pmatrix} = \sum_i \mathbf{H}_{mi} \cdot \begin{pmatrix} \mathbf{F}_m \\ \mathbf{T}_m \end{pmatrix} \tag{2.25}$$

which defines the mobility or interaction matrix $\mathbf{H}_{mi}$.

Now, suppose the monomers are small spheres; the reason we do this is so that rotational motion does not enter into the problem below. If there are no torques acting on the monomers, then we write

$$\mathbf{V}_m = \sum_i \mathbf{H}_{mi} \cdot \mathbf{F}_i \tag{2.26}$$

If the velocity of each monomer were determined only by the forces acting on it, then $\mathbf{H}_{mi}$ would be diagonal with the form

$$\mathbf{H}_{mi} = \frac{\mathbf{I}\delta_{mi}}{\zeta} \tag{2.27}$$

where ζ is the friction constant of the monomers: the exact value of ζ depends on the viscosity η_s of the solvent and on the exact shape of the monomers (for example, for a sphere of radius a, $\zeta = 6\pi\eta_s a$), and the linear relationship between $\mathbf{V}_m$ and $\mathbf{F}_m$ holds for our spheres only as long as $\mathbf{F}_m$ is small.

The problem here is that the velocity of a given monomer m is not only affected by $\mathbf{F}_m$, but also by the forces acting on the other monomers $\{\mathbf{F}_i | i = 1, ..., m-1, m+1, ..., N\}$. The reason is that the force $\mathbf{F}_j$ moves monomer j; then, because of the monomer-solvent interaction, monomer j moves some of the molecules of the solvent near it; this motion propagates to the solvent molecules near monomer m, which

in turn interact with monomer m. This is the hydrodynamic effect, and if this is taken into account, the mobility matrix $\mathbf{H}_{mi}$ becomes non-diagonal. Going back to the discussion above about long-range forces, if $\mathbf{H}_{mi}$ is a full matrix, then we have a long-range interaction, whereas if $\mathbf{H}_{mi}$ is diagonal (or tri-diagonal), then we have a short-range one.

More can be said quantitatively about this non-diagonality if we make a few assumptions about the fluid in which the monomers are floating around (i.e. the solvent). In particular, we assume the fluid is incompressible, and we also assume that the Reynolds number R is small. This is called the Stokes' approximation (see Chorin and Marsden [13]), and it gives us the following equations:

$$\nabla \cdot \mathbf{u} = 0 \tag{2.28}$$

$$\frac{\partial \mathbf{u}}{\partial t} = -\nabla p + \frac{1}{R}\Delta \mathbf{u} + \mathbf{F}_{ext} \tag{2.29}$$

where $\mathbf{u} = \mathbf{u}(x(t), y(t), z(t), t)$ is the velocity field of the fluid, $\mathbf{F}_{ext}$ are the external forces, and p is the pressure.

What we do now is calculate the flow field induced by the external forces acting on the monomers assuming the monomers are points. The result is (see Doi and Edwards [17] for the calculation)

$$\mathbf{u}(\mathbf{r}) = \sum_{i=0}^{N} \mathbf{H}(\mathbf{r} - \mathbf{R}_i) \cdot \mathbf{F}_i \tag{2.30}$$

where, if we let $\mathbf{r} = r\hat{\mathbf{r}} = r(\hat{r}^1, \hat{r}^2, \hat{r}^3)$ with $|\hat{\mathbf{r}}| = 1$, $\mathbf{H}$ is given by

$$\mathbf{H}(\mathbf{r}) = \frac{1}{8\pi\eta_s r}(\mathbf{I} + \hat{\mathbf{r}}\hat{\mathbf{r}}) \tag{2.31}$$

with $\hat{r}^i\hat{r}^j$ being the ij element of the 3×3 matrix $\hat{\mathbf{r}}\hat{\mathbf{r}}$. Since the monomers move with the fluid, their velocities are given by

$$\mathbf{V}_m = \mathbf{u}(\mathbf{R}_m) = \sum_{i=1}^{N} \mathbf{H}(\mathbf{R}_m - \mathbf{R}_i) \cdot \mathbf{F}_i \tag{2.32}$$

with $\mathbf{H}(0) = \frac{1}{\eta}$ as in the case where we neglected the hydrodynamic effect. We now identify $\mathbf{H}(\mathbf{R}_m - \mathbf{R}_i)$ with $\mathbf{H}_{mi}$.

2.3.1 Rouse and Zimm Models of a Polymer

Now that we know more about the polymer-solvent solution, we can use this to construct models of polymers which are hopefully more realistic that the Gaussian and freely-jointed chains mentioned above.

The first description of a polymer that we consider in this section is the Rouse model. In this case, the polymer is modelled as a bunch of beads connected by springs, and both the hydrodynamic interaction and the excluded volume effect are disregarded. Thus, the mobility matrix $\mathbf{H}_{mi}$ is the same as in Equation 2.27, and the interaction potential is

$$U = \frac{k}{2} \sum_{m=1}^{N} |\mathbf{R}_m - \mathbf{R}_{m-1}|^2 \tag{2.33}$$

where $k = \frac{3k_BT}{b^2}$ is the spring constant, b is the effective bond length, and $\mathbf{R}_m$ is the position of the mth bead.

To get the equation of motion for the mth bead (monomer), we use Newton's law and assume the mass is negligible; thus, the forces add up to zero. These forces are

friction with the solvent, the spring interactions with the monomers $m+1$ and $m-1$, and some "random" force to model the (temperature-dependent) random collisions of our monomer with the surrounding fluid. So, for $m = 1, ..., N-1$, the equation of motion can be written

$$\zeta \frac{d\mathbf{R}_m}{dt} = k(\mathbf{R}_{m+1} - 2\mathbf{R}_m + \mathbf{R}_{m+1}) + \mathbf{f}_m \tag{2.34}$$

and for $m = 0, N$ we have

$$\zeta \frac{d\mathbf{R}_0}{dt} = k(\mathbf{R}_1 - \mathbf{R}_0) + \mathbf{f}_0 \tag{2.35}$$

$$\zeta \frac{d\mathbf{R}_N}{dt} = k(\mathbf{R}_{N-1} - \mathbf{R}_N) + \mathbf{f}_N \tag{2.36}$$

where the $\{\mathbf{f}_m\}$ as written above are "Gaussian white noise", with $\mathbf{f}_m = (f_m^1, f_m^2, f_m^3)$, and with moments given by

$$< \mathbf{f}_m(t) >= 0 \tag{2.37}$$

$$< f_m^i(t) f_l^j(t') >= 2\zeta k_B T \delta_{ml} \delta(t - t') \tag{2.38}$$

Again, we have to interpret these equations in the appropriate sense: for the corresponding Wiener process, we should have a variance of $2\zeta k_B T$ (see Section 3.4 and Section 3.3 for more rigorous definitions of "random" force, "Gaussian white noise", and Wiener processes).

The Rouse model's key assumption is that the dynamics of the polymer is governed by local interactions; however, for a real polymer the hydrodynamic interaction does play a role in the dynamics, and in fact experimental results do not agree with some

of the Rouse model's predictions. For example, consider the center of mass $\mathbf{R}_G(t)$ as a function of time (see Equation 2.15), and define the self-diffusion constant D_G by

$$D_G = \lim_{t\to\infty} \frac{1}{t} < |\mathbf{R}_G(t) - \mathbf{R}_G(0)|^2 > \tag{2.39}$$

which is a measure of how much the polymer as a whole drifts around in the surrounding solvent. The Rouse model predicts that $D_G \propto N^{-1}$, but experiments give $D_G \propto N^{-\frac{1}{2}}$.

The lack of agreement between the Rouse model and experiment leads naturally to the question: what happens if we take into account the hydrodynamic interaction? Going back to Equation 2.31, we see that the interaction matrix $\mathbf{H}_{ml}$ is non-linear in $\mathbf{R}_m - \mathbf{R}_l$. Zimm replaced $\mathbf{H}_{ml}$ by its average with respect to the equilibrium distribution function at the Θ-temperature; in this model, the mobility matrix $\mathbf{H}_{ml}$ is given by $h(m-l)\mathbf{I}$ where

$$h(m-l) = \begin{cases} \frac{1}{\zeta} & m = l \\ \frac{1}{\sqrt{6\pi^3|m-l|}\eta_s b} & m \neq l \end{cases} \tag{2.40}$$

See Doi and Edwards [17] for the detailed calculation.

We should note that this is valid only near $T = T_\Theta$ and only near equilibrium, and we should also note that since the excluded volume v is zero at T_Θ, we are in effect neglecting the excluded volume interaction. Later on we will examine this assumption more closely.

Just as for the Rouse model, we now want to get the equation of motion for the mth bead (monomer), we again use Newton's law: the only difference is that

the spring interactions with the monomers $m+1$ and $m-1$ is smeared out by the hydrodynamic interaction (i.e. the non-diagonal $\mathbf{H}$) so that every monomer interacts with monomer m. We thus arrive at the Zimm model, which is as follows:

$$\frac{d\mathbf{R}_m(t)}{dt} = \sum_{l=0}^{N} h(m-l)(k(\mathbf{R}_{l+1} - 2\mathbf{R}_l + \mathbf{R}_{l-1}) + \mathbf{f}_m) \tag{2.41}$$

where we define $\mathbf{R}_{N+1}$ to be $\mathbf{R}_N$ and $\mathbf{R}_{-1}$ to be $\mathbf{R}_0$. Again, the key difference here is that $\mathbf{H}_{ml}$ is non-diagonal, and so the interaction between the monomers is non-local.

Although the Zimm model's prediction of the dependence of D_G (see Equation 2.39) on N agrees with the experimental results near $T = T_\Theta$, it is not clear that the Zimm model will yield meaningful results if we attempt to apply it to polymers when the temperature is not near T_Θ. What about if we include the excluded volume interaction? Instead of considering only the spring potential in Equation 2.33, we need to add the following (spatially) short-ranged potential:

$$U_1 = \frac{1}{2} v k_B T \sum_{m,l=1}^{N} \delta(\mathbf{R}_m - \mathbf{R}_l) \tag{2.42}$$

where δ is either the Dirac delta function or of the form $c\exp(\frac{-|\mathbf{x}|^2}{\epsilon})$, where c normalizes δ so that its integral over $\Re^d$ is one, and where $0 < \epsilon \ll 1$ describes the narrowness of the peak at 0. If we also wish our model to be applicable away from T_Θ, we must start from the matrix given by Equation 2.31. Note that this $\mathbf{H}(\mathbf{r})$ depends upon the configuration of the polymer; in other words, we can say that $\mathbf{H} = \mathbf{H}(\mathbf{r}(t))$ and so will change from step to step (in a numerical scheme), unlike the constant matrix $\mathbf{H}_{ml}$ in Equation 2.40.

Both the Rouse model and the Zimm model are examples of the more general Langevin equation, which can be written for each monomer as

$$\frac{\partial}{\partial t}\mathbf{R}_n(t) = \sum_m \mathbf{H}_{nm}(-\frac{\partial U}{\partial \mathbf{R}_m} + \mathbf{f}_m(t)) + \frac{1}{2}k_B T \sum_m \frac{\partial}{\partial \mathbf{R}_m}\mathbf{H}_{nm} \tag{2.43}$$

with $\mathbf{f}_m(t)$ the random force term as above, and with U the interaction potential. The first sum is just the force on the m-th monomer $\frac{\partial U}{\partial \mathbf{R}_m} + \mathbf{f}_m(t)$, which gets smeared out by the hydrodynamic interaction given by $\mathbf{H}_{nm}$. The second sum is because of the drift that could happen, for example, when the diffusion $D = \frac{k_B T}{\zeta}$ depends on the position of the monomer.

2.4 Random Walks as Models for Polymers

In the previous two sections, we have discussed in detail two topics which at first glance are seemingly unrelated: self-avoiding walks and polymers. However, there is in fact a deep connection between the two; in the next few paragraphs, we will use a few examples to more clearly illustrate this connection.

The reader will have noticed that the average end-to-end distance of a polymer r_N when the excluded volume effect is disregarded (e.g. for the Gaussian model see Equation 2.17) scales according to the same power of N as the end-to-end distance of a random walk on a lattice; when the excluded volume effect is present, it turns out that for a dilute solution of polymers in a good solvent at temperatures higher than the Θ temperature, r_N scales just like it does for a non-interacting SAW on a lattice.

We can extend the analogy even further: as noted above, χ in Equation 2.14 is a function of temperature, and thus so is the excluded volume v; at $T = T_\Theta$, $v = 0$. If we want to calculate the statistics of a dilute polymer solution in a good solvent at the Θ point in two dimensions, we can use another SAW called the kinetic growth walk (KGW) (see Poole, et al. [60]), which is an SAW on a hexagonal lattice that starts at the origin and takes random steps until it cannot proceed any further without intersecting itself. For the KGW only nearest-neighbor interactions are considered.

At this point it should be noted that for real polymers, even though the two-body interaction may disappear at the Θ point, the three-body (and four-body, etc.) interactions may still be present. If we take into account these three-body interactions, it turns out that the coefficient v of the Φ^2 term in the expansion of the free energy (see Equation 2.13) becomes $\tilde{v} = v + w_0 a^3$ where a^3 is the volume occupied by a monomer and w_0 measures the strength of the three body interaction. In real experiments it is actually $\tilde{v}$ that is measured and not v; the temperature at which this $\tilde{v}$ vanishes is called the $\tilde{\Theta}$ temperature, and may be significantly different from the Θ temperature. On the lattice, we model this by including including (a subset of) the three-body interactions in our energy calculations. Again, there is a corresponding SAW called the SKW (see Section 4.3 for more details) which allows us to calculate the critical exponents of the polymer at this temperature in two dimensions. It should be noted that it is not clear that the Θ and $\tilde{\Theta}$ points should belong to the same universality class (see Section 4.3 for a definition).

In summary, depending upon the temperature and the quality of the solvent, we can sometimes choose an appropriate random walk on a lattice and use this to calculate the critical exponents instead of numerically integrating a system of differential equations in $\Re^d$ (as would be necessary with the Rouse model, for example).

Chapter 3

Markov Chains, Brownian Motion and Stochastic Differential Equations

3.1 Introduction

In this chapter we will first discuss the necessary probability and measure theory background, including the definition of a probability space and Markov chains and processes. We will then summarize some results about Wiener processes, which is used in the Rouse and Zimm models (see Section 2.3.1) to describe the collisions between polymer and solvent. Since some of the models for polymers discussed in section 2.3.1 are formulated in terms of stochastic differential equations, we will also give

a rigorous definition of what stochastic differential equations such as Equation 2.34 and Equation 2.41 actually mean.

3.2 Probability and Markov Chains

The idea behind the discussion below is to get a rigorous definition of Markov processes and transition probabilities; this will be needed when we talk about schemes for computing thermal averages in Section 4.2 and in the chapter on numerical methods (Chapter 5.2). A more detailed discussion can be found in Feller [21] and in the thesis of Chang [9].

We begin with some basic definitions. Suppose Σ is a set, and let $\mathcal{F}$ be a non-empty set of subsets of Σ which is closed under complements and countable unions. $\mathcal{F}$ is called a σ-algebra of subsets of Σ (note that Σ and $\emptyset$ are both in $\mathcal{F}$). Let $A_1,\ldots,A_n,\ldots$ be a countable collection of sets in $\mathcal{F}$ with $A_i \cap A_j = \emptyset$ if $i \neq j$. A probability measure P is a real-valued non-negative function defined on $\mathcal{F}$ with $P(\Sigma) = 1$ and $P(\cup A_i) = \sum P(A_i)$ for any such collection of sets. The triple $(\Sigma, \mathcal{F}, P)$ is called a probability space.

Define $\mathcal{B}^d$ to be the σ-algebra generated by open sets in $\Re^d$; the sets in $\mathcal{B}^d$ are called Borel sets. Let I be some index set, and suppose $\{x_t, t \in I\}$ is a family of $\Re^d$-valued random variables defined on $(\Sigma, \mathcal{F}, P)$: that is, for each t, $x_t : \Sigma \rightarrow \Re^d$ and for each Borel subset B of $\Re^d$, the set $x_t^{-1}(B)$ is in $\mathcal{A}$ (i.e. is $\mathcal{A}$-measurable). The family $\{x_t\}$ is called a stochastic process.

The random variables $\{x_t\}$ are called independent if for every finite subset $x_1,\ldots,x_n$ and for every choice of Borel sets $S_1,\ldots,S_n$, the events $A_i = \{\omega : x_i(\omega) \in S_i\}$ are independent: i.e., $P(\cap A_i) = \prod P(A_i)$. Two collections of measurable sets $\mathcal{G}$ and $\mathcal{H}$ are independent if for any $G \in \mathcal{G}$ and $H \in \mathcal{H}$, $P(G \cap H) = P(G)P(H)$.

The expected value of $E(x_t)$ of $x_t(\sigma)$ is defined to be $\int_\Sigma x_t(\sigma)dP$: if for all t we have $E(x_t) = c$ (where c is independent of t), and if for all t, s $E(x_s x_t) = C(t-s)$, then we say that the process is stationary; $C(t-s)$ is called the covariance of the process.

Suppose now that $x_t \in L^1(P)$ and let $\mathcal{G}$ be a σ-algebra of sets such that $\mathcal{G} \subset \mathcal{F}$. A random variable y_t is called a conditional expectation of x_t relative to $\mathcal{G}$ if y_t is $\mathcal{G}$-measurable and if for all $G \in \mathcal{G}$ we have

$$\int_G y_t dP = \int_G x_t dP \tag{3.1}$$

By the Radon-Nikodym theorem (see Feller [21] for details) such a y_t exists. We then write $y_t = E(x_t|\mathcal{G})$. If $F \in \mathcal{F}$, then the conditional probability $P(F|\mathcal{G})$ is defined by

$$P(F|\mathcal{G}) = E(I_F|\mathcal{G}) \tag{3.2}$$

where the indicator function I_F of the set F is defined by $I_F(f) = 1$ if $f \in F$ and 0 otherwise. We will also use the notation $P(x_t \in F|x_s)$ to mean $P(x_t \in F|\mathcal{X}_s)$ where $\mathcal{X}_s$ is the smallest σ-subalgebra of $\mathcal{F}$ with respect to which x_s is measurable; $\mathcal{X}_{[t_1,t_2]}$ will be the smallest σ-subalgebra of $\mathcal{F}$ with respect to which the $\{x_s|s \in [t_1,t_2]\}$ are

measurable; and $\mathcal{F}_t^+$ will be the smallest σ-subalgebra of $\mathcal{F}$ with respect to which $\{x_s - x_t | t \leq s < \infty\}$ are measurable.

Let $\{x_t, t \in [T_1, T_2]\}$ be a stochastic process, and let $T_1 \leq s \leq t \leq T_2$; it is called a Markov process if it satisfies the following

$$P(x_t \in F | \mathcal{F}_{[T_1,s]}) = P(x_t \in F | x_s) \tag{3.3}$$

with probability one for any $F \in \mathcal{F}$. This means that only the present state of the system (and not any information about the past states) has an effect on the future development: the system has no memory. A Markov chain is a Markov process whose random variables only assume values in a certain finite or denumerably infinite set.

We define the transition probability $P(s, x, t, F)$ to be the distribution such that

$$P(s, x, t, F) = P(x_t \in F | x_s) \tag{3.4}$$

holds with probability one for $s \leq t$. The following equations then hold for $P(s, x, t, F)$:

(a) $P(s, x, t, \cdot)$ is $\mathcal{B}^d$ measurable for fixed s, x_s, t.

(b) $P(s, \cdot, t, F)$ is a probability measure for fixed s, t, F.

(c) the Chapman-Kolmogorov equation is satisfied:

$$P(s, x, t, F) = \int_{\Re^d} P(u, y, t, F) P(s, x, u, dy) \tag{3.5}$$

Alternatively, we also can start with a function satisfying conditions (a),(b) and (c) and construct a Markov process for which Equation 3.4 holds (see Doob [18] for more details).

3.3 Brownian motion

Using the probability background in the previous section, we now will look more closely at how we can model the random collisions of a polymer with the surrounding solvent; the mathematically rigorous way to do this involves the construction of a Wiener process. For a more detailed discussion, see Lamperti [39] Paley and Wiener [54], Karatzas [34] and Knight [36].

Let $(\Sigma, \mathcal{F}, P)$ be a probability space. A Brownian motion process, or a Wiener process, is defined to be a stochastic process $\{x_t, t \geq 0\}$ which satisfies the following conditions:

(i) $x_0(\sigma) = 0$ a.s. (almost surely - with probability one).

(ii) If $0 < t_1 < ... < t_m$, then $x_{t_{i+1}} - x_{t_i}$ are independent for $i = 1, ..., m-1$.

(iii) For $s, t \geq 0$, $x_{t+s} - x_t$ is normally distributed with mean 0 and variance s.

(iv) For almost all $\sigma \in \Sigma$, the function $x_t(\sigma)$ is everywhere continuous in t.

The existence of such a process is not obvious: for a proof, see Lamperti [39]. This process is a model of the physical phenomenon known as Brownian motion: small particles suspended in a fluid move in an irregular manner that results from the numerous collisions between the particles and the molecules in the surrounding fluid.

Note that since the increments of the Wiener process are independent, for any fixed t so are the two σ-subalgebras $\mathcal{X}_t$ and $\mathcal{X}_t^+$ (see Section 3.2 for the definitions).

We will now give a summary of some of the more interesting properties of a Wiener process: Theorems 2, 3 and 4 are all proven in Lamperti [39].

The first property concerns the large scale behavoir of the paths:

Theorem 2 $P(\limsup_{t\to\infty} \frac{x_t}{\sqrt{2t\log\log t}} = 1) = 1$

In fact, since $\{-x_t(\sigma)\}$ and $\{x_t(\sigma)\}$ are both Wiener processes, Theorem 2 holds for both of these, and so $x_t(\sigma)$ has both positive and negative values for arbitrarily large t. Since $x_t(\sigma)$ is a.s. continuous, we conclude that $x_t(\sigma)$ has arbitrarily large zeros.

We know the function $x_t(\sigma)$ is continuous in t: what can be said about its differentiability? Using the following theorem, called the "local law of the iterated logarithm",

Theorem 3 *For each* $t_0 > 0$, $P(\limsup_{h\to 0+} \frac{x_{t_0+h}-x_{t_0}}{\sqrt{2h\log\log h^{-1}}} = 1) = 1$

we can conclude that $P(x_t'\text{exists}) = 0$ for each fixed t. Wiener and Paley gave a more precise characterization:

Theorem 4 *If* $\lambda > \frac{1}{2}$, *the values of* σ *for which there exists a* t *such that*

$\limsup_{h\to 0} |\frac{x_{t_0+h}(\sigma)-x_{t_0}(\sigma)}{h^\lambda}| < \infty$ *form a set of* 0 *measure. If* $\lambda < \frac{1}{2}$, *then except for a set of* σ *of zero measure*

$\limsup_{h\to 0} |\frac{x_{t_0+h}(\sigma)-x_{t_0}(\sigma)}{h^\lambda}| = 0$ *uniformly for all values of* t.

A stronger result is that with probability one, the t set for which x_t exists has Lebesgue measure 0. However, we do not really need the above theorems to prove non-differentiability: we can instead start from the fact that $\frac{x_{t+h}-x_t}{h}$ is normally distributed with mean 0 and variance h (see Lamperti [39]).

3.4 Stochastic Differential Equations

We are now ready to actually say what the equations for the Rouse and Zimm models (see Equation 2.34 and Equation 2.41) actually mean. Suppose Y_t and f are $\Re^d$-valued functions, G is a d by m matrix, and x_t is an m-dimensional Wiener process. Since the derivative of Brownian motion a.s. does not exist (and thus Brownian motion is not of bounded variation), equations like the following

$$dY_t = f(t, Y_t)dt + G(t, Y_t)dx_t, \; Y_0 = c, \; 0 \leq t \leq T < \infty \tag{3.6}$$

need to be looked at more carefully. However, before we begin with our attempt to define this equation more rigorously, let us proceed informally in one-dimensional case. dx_t in the above equation is sometimes written as ξdt where ξ_t is called white noise, which means that ξ_t is a stationary Gaussian stochastic process with constant Fourier transform $f(\lambda)$: if we again denote the covariance of ξ_t by $C(t)$, and if we define $f(\lambda)$ by

$$f(\lambda) = \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{-i\lambda t} C(t) dt \tag{3.7}$$

then $f(\lambda)$ is some constant c independent of λ. The problem here is that this implies that $C(t) = \delta(t)$ where δ is the Dirac delta-function. In particular, we have

$$C(0) = \int_{-\infty}^{\infty} f(\lambda) d\lambda = \infty \tag{3.8}$$

and when $t \neq 0$, $C(t) = 0$, which means that for arbitrarily small s, the random variables ξ_t and ξ_{t+s} are uncorrelated.

As we have presented it, this informal approach contains a number of difficulties: we could continue along this path and make things more rigorous by introducing generalized functions, but we take another route. Instead of trying to define white noise, we are going to define the integral

$$\int_0^t G(s)dx_s = \int_0^t G(s,\sigma)dx_s(\sigma), (0 < t) \tag{3.9}$$

and from this we will be able to make more sense of Equation 3.6. The strategy is to define the integral first for step functions, restrict our G to a certain class of functions that, among other things, can be approximated (in a sense to be defined below) by step functions, and then define the integral of G as the limit (again in a sense to be defined below) of the integrals of these step fuctions. However, we first need some preliminary definitions: for more details and a discussion of white noise, see Arnold [4] or Ito and McKean [33].

A sequence of random variables G_m converges stochastically to a random variable G if for all $\epsilon > 0$ we have

$$\lim_{m\to\infty} P(\sigma : |G_m(\sigma) - G(\sigma)| > \epsilon) = 0 \tag{3.10}$$

This is written

$$st - \lim_{m\to\infty} G_m = G \tag{3.11}$$

Suppose G is a step function on $[0,t]$ with $0 = t_0 < t_1 < ... < t_m = t$ and $G = G(t_{i-1})$ on $[t_{i-1}, t_i)$; we define the integral of G to be

$$\int_0^t G(s,\sigma)dx_s(\sigma) = \sum_{i=1}^{m} G((t_{i-1})(x_{t_i} - x_{t_{i-1}}) \tag{3.12}$$

We now need to examine the more general case: for which G can we define the integral in Equation 3.9? Suppose $\{\mathcal{F}_t | t \geq 0\}$ is a family of σ-subalgebras of $\mathcal{F}$; $\{\mathcal{F}_t\}$ is called non-anticipating (with respect to the Wiener process $\{x_t\}$) if it has the following characteristics:

(a) $\mathcal{F}_s \subset \mathcal{F}_t$ $(0 \leq s \leq t)$

(b) $\mathcal{F}_t \supset \mathcal{X}_t$ $(0 \leq t)$

(c) $\mathcal{F}_t$ is independent of $\mathcal{X}_t$ $(0 \leq t)$

If we have a $d \times m$ matrix function $G(s, \sigma)$ defined on $[0, t] \times \Sigma$, Borel-measurable with respect to s and P-measurable with respect to σ, we correspondingly call G non-anticipating if $G(s, \cdot)$ is $\mathcal{F}_s$-measurable for all $s \in [0, t]$. The set of all such non-anticipating fuctions which are in $L^2(0, t)$ (that is, for which the equation

$$\int_0^t |G(s, \sigma)|^2 ds < \infty \tag{3.13}$$

holds with probability one) is denoted by $M_2[0, t]$.

Let our function G be in $M_2[0, t]$: then we can find a sequence of step functions $\{G^m(s, \sigma)\} \subset M_2[0, t]$ such that

$$st - \lim_{m \to \infty} \int_0^t |G(s) - G^m(s)|^2 ds = 0 \tag{3.14}$$

We then define the integral of G (call this $I(G)$) by

$$\int_0^t G dx = \int_0^t G(s) dx_s = st - \lim_{m \to \infty} \int_0^t G^m(s) ds \tag{3.15}$$

The value $I(G)$ does not depend on the sequence $\{G^m\}$, and with probability one it

is determined uniquely. Again, we refer the reader to Arnold [4] for proofs of these statements and a more detailed discussion.

Let $\{x_t | 0 \leq t \leq T\}$ denote an $\Re^m$-valued Wiener process, and suppose now that we are given some initial condition $c = c(\sigma)$, with c being a random variable independent of $x_t - x_0$ for all $0 \leq t \leq T$. Also, let the $\Re^d$ valued f and the $d \times m$ matrix G be defined on $[0, T] \times \Re^d$. Additionally, if y is a random variable $y(\sigma)$, for fixed (t, y) suppose $f(t, y)$ and $G(t, y)$ are independent of $\sigma \in \Sigma$: in other words, all of the dependence on $\sigma \in \Sigma$ is in the variable y. Define the functions $\overline{f}(t, \sigma) = f(t, Y_t(\sigma))$ and $\overline{G}(t, \sigma) = G(t, Y_t(\sigma))$; we assume that they fulfill the following conditions:

(a) $\overline{f}$ and $\overline{G}$ are non-anticipating with respect to $\mathcal{X}^c_{[0,T]}$, which is defined to be the smallest σ-subalgebra of $\mathcal{F}$ with respect to which c and $\{x_s | 0 \leq s \leq t\}$ are measurable.

(b) with probability one we have:

$$\int_0^t |\overline{f}(s, \sigma)| ds < \infty \tag{3.16}$$

$$\int_0^t |\overline{G}(s, \sigma)|^2 dx < \infty \tag{3.17}$$

(where $|G| = \text{trace}(GG')$)

At this point we are finally ready to give a mathematically more rigorous interpretation of Equation 3.6: namely, we define Y_t to be a solution of Equation 3.6 on the interval $[0, T]$ with initial condition $Y_0(\sigma) = c(\sigma)$ if the following conditions hold:

(a) Y_t is $\mathcal{X}^c_{[0,T]}$-measurable, i.e. non-anticipating for $t \in [0, T]$.

(b) for every $t \in [0,T]$ with probability one we have:

$$Y_t(\sigma) = c(\sigma) + \int_0^t f(t, Y_t(\sigma))dt + \int_0^t G(t, Y_t(\sigma))dx_t \ (0 \leq t \leq T \leq \infty) \tag{3.18}$$

Note that the integrals exist by condition (b) (in particular, since $G \in M_2[0,T]$). See Øksendal [50] for a proof.

What about existence and uniqueness? The reader familiar with the theory of non-stochastic ODE's may suspect that we need some sort of Lipschitz condition and possibly some bound on the growth of f and G: this is in fact the case, as indicated in the following theorem.

Theorem 5 *Suppose f, G, x_t and c are as above, and furthermore suppose that there exists a constant K such that:*

(a) (Lipschitz) $\forall t \in [0,T]$, $x \in \Re^d$, $y \in \Re^d$:

$$|f(t,x) - f(t,y)| + |G(t,x) - G(t,y)| \leq K|x-y| \tag{3.19}$$

(b) (growth bound) $\forall t \in [0,T]$, $x \in \Re^d$:

$$|f(t,x)|^2 + |G(t,x)|^2 \leq K(1+|x|^2) \tag{3.20}$$

Then Equation 3.6 has in $[0,T]$ a unique, with probability one continuous $\Re^d$-valued solution Y_t with the initial condition $Y_0 = c$. In other words, if X_t and Y_t are continuous solutions of Equation 3.6 with the same inital value c, then

$$P(\sup_{0 \leq t \leq T]} |X_t - Y_t| > 0] = 0 \tag{3.21}$$

For the proof, see Arnold [4].

Chapter 4

Thermodynamics and Phase Transitions

4.1 Introduction

In this chapter, we will introduce the thermodynamics concepts relevant to our computations. In particular, we will discuss why we cannot compute thermal averages exactly and how we can get good approximations to them; this will involve the probabiliy theory discussed in Chapter 3. After that, we will give a brief outline of the theory of phase transitions: we will define what a phase transition is, give a detailed example of a phase change, and describe the limitations encountered when using a computer to simulate such a transition.

4.2 Thermodynamics Background

For a system in a heat bath at a constant temperature T, the probability (called the Gibbs probability) of the system being in a state α with energy E_α is given by

$$p_\alpha = \frac{e^{-\beta E_\alpha}}{Z}. \tag{4.1}$$

where $\beta = \frac{1}{k_B T}$ (k_B is Bolzmann's constant) and where the "partition function" Z is defined by

$$Z = \sum_\alpha e^{-\beta E_\alpha}. \tag{4.2}$$

The summation is over all possible states α.

The goal of much of our computations is to calculate the thermal average of some function $X(\alpha)$:

$$< X >= \frac{1}{Z} \sum_\alpha X(\alpha) e^{-\beta E_\alpha}. \tag{4.3}$$

X could be, for example, the end-to-end distance of an SAW.

Suppose that the size N of our system is fixed, and assume that the system has a finite number of states for any given N. For example, if we consider SAW's on a lattice, we can group the SAW's into equivalence classes, each class consisting of those SAW's which are horizontal and/or vertical translates of each other. Alternatively, we could require the SAW's to start at the origin.

For many systems, even for moderately sized N the sum in Equation 4.3 contains an intractable number of terms; thus, we cannot evaluate the thermal averages directly. Instead, we have to somehow sample some of these configurations in such a

way that our estimated average is not too far from the exact value. Since the Gibbs probabilities $e^{-\beta E_\alpha}$ vary over many orders of magnitude, a straightforward random sampling of the states is not in general an efficient method: this would include many terms with negligible contributions to the above sum, and it might leave out terms with large Gibbs probabilities. Instead, what we want to do is choose those configurations α with the largest probabilities $e^{-\beta E_\alpha}$ and use the values of $X(\alpha)$ to estimate $< X >$ (this is called importance sampling). How do we select these α?

What we plan to do is to generate a Markov chain of states: we start with some arbitrary initial state α_i and then specify how to move from one state to the next; if this transformation is chosen correctly, the frequency of a given state in the Markov chain will be close to its Gibbs probability. Recall the assumption above that the number of states M of our system is finite. Suppose that the probability of going from a configuration α_i to α_j depends only on α_i, and not on any previous state of the system (this will give us a Markov process, and the sequence of states that we get from the algorithm below will be a Markov chain): define P_{ij} to be this probability. So, we must have

$$\forall i : \sum_{j=1}^{M} P_{ij} = 1 \tag{4.4}$$

p_i will be the Gibbs probability of the state α_i. We form an $M \times M$ transition matrix P whose elements are $\{P_{ij}\}$, and we define the n-step transition probabilities P_{ij}^n to be:

$$P_{ij}^n = \sum_{k=1}^{M} P_{ik}^{n-1} P_{kj} \tag{4.5}$$

with $P_{ij}^0 = P_{ij}$. Suppose also that we impose the following two requirements on the P_{ij}^n:

1) accessiblilty, or ergodicity: you can reach any configuration from any other one in a finite number of steps.

$$\forall i, j \exists n \geq 0 \text{ such that } P_{ij}^n \neq 0. \tag{4.6}$$

2) we want the $\{p_i\}$ to be a stationary distribution for the Markov chain (see Section 3.2 for the definition).

$$\forall j \sum_{i=1}^{M} p_i P_{ij} = p_j \tag{4.7}$$

Instead of 2), detailed balance, or micro-reversibility is often assumed: this means the following equation is satisfied for all states α_i,α_j:

$$P_{ij} p_i = P_{ji} p_j \tag{4.8}$$

If you sum this equation over i and use Equation 4.4, you get Equation 4.7, so detailed balance is actually a stronger requirement.

If 1) and 2) are satisfied, what can be said about the distribution of states in our newly created Markov chain? Well, what we want is to create a Markov chain in which, for each state, as we move along the chain, the difference between the probability of occurence of the given state in the chain and the Gibbs probability decreases. In other words, if A_0, A_1,...,A_m is the sequence of states we get from our Markov process (where the $\{A_i\}$ are not necessarily distinct), we want

$$\lim_{m \to \infty} \frac{1}{m} \sum_{i=0}^{m} X(A_i) =< X > . \tag{4.9}$$

For our Markov chain, the following argument, taken from Binney, et al. [8], shows that this is the case (see their book for a further discussion): first, although ergodicity (see Equation 4.6) is in general difficult to prove for Markov chains produced by a given algorithm, without it we could not be sure that we are somehow not missing some (or even many) high probability states when generating the chain, so we do need this assumption even if we often cannot verify it in practice.

Now for the detailed balance condition. Suppose we let W_i^n be the probability of occurrence of state i at the nth step of our Markov chain, and define a difference between the actually probabilities p_i and W_i^n

$$D^n = \sum_{i=0}^{m} |W_i^n - p_i| \tag{4.10}$$

We can use detailed balance to show that D^n is a non-increasing function of n as follows:

$$\begin{aligned}
D^{n+1} &= \sum_{i=0}^{m} |W_i^{n+1} - p_i| \\
&= \sum_{i=0}^{m} |\sum_{j=0}^{m} W_j^n P_{ji} - p_i| \\
&= \sum_{i=0}^{m} |\sum_{j=0}^{m} (W_j^n P_{ji} - p_i P_{ij})| \\
&= \sum_{i=0}^{m} |\sum_{j=0}^{m} (W_j^n P_{ji} - p_j P_{ji})| \\
&= |\sum_{j=0}^{m} (W_j^n - p_j)| \\
&\leq \sum_{j=0}^{m} |W_j^n - p_j| = D^n
\end{aligned} \tag{4.11}$$

where we use Equation 4.5 to go from line two to line three, Equation 4.8 to go from three to four, and Equation 4.5 once more to go from four to five.

To be sure, the above argument only says we will eventually get good results: it does not tell use how big m needs to be. Thus, for a given m, we have no guarantee that the averages we compute will be a good approximation to the true value $< X >$: for example, we might have problems near a phase transition, where the variance of the value's $X(\alpha_i)$ will be large, and thus we might need an impractically large number of steps in our Markov chain (see Section 4.3 for more details about this).

4.3 Phase Transitions

To begin our discussion of phase transitions, we will give an extended description of the transition which occurs for interacting self-avoiding walks on a lattice in two dimensions. After the example, we will present a more general definition of a phase transition.

Consider an SAW on a hexagonal (two-dimensional) lattice whose edges interact with each other on a short spatial range, with the number of such interactions being given by one plus the number of hexagons which contain two steps of the walk not connected on that hexagon (Figure 5.1 (A)-(D)) plus twice the number of hexagons that contain three steps of the walk not connected on that hexagon (Figure 5.1 (E)); let N_{NN} be the number of such nearest neighbor interactions, and give the SAW a probability proportional to $e^{\beta N_{NN}}$. It turns out that the average shape of the SAW

depends greatly on β: if β is greater than a certain β_{crit}, the SAW shrinks into a compact blob; if β is below this β_{crit}, the SAW is stretched out as if there were no interaction; at $\beta = \beta_{crit}$, there is an intermediate state (we assume that there are no other possible states). This is one example of a phase transition.

If we invert Equation 2.5, we get the following:

$$N \asymp < |\omega_N - \omega_0|^2 >^{\frac{1}{2\nu}} \tag{4.12}$$

In the following we assume that such an exponent exists; please see Section 2.2 for a more detailed discussion of this point.

We call $\frac{1}{\nu}$ the dimension (see Section 2.2 for a definition of dimension) of the SAW, and this will change as β passes through β_{crit}. In fact, the values of ν in the compact, the stretched out, and the intermediate states are known from heuristic arguments and numerical simulations. When $\beta > \beta_{crit}$, then $\nu = \frac{1}{2}$ (this is as compact as you can get in two dimensions); if $\beta < \beta_{crit}$, $\nu = \frac{3}{4}$ (this corresponds to the Flory exponent given in Equation 2.6 above). What is β_{crit}, and what about the value of ν at $\beta = \beta_{crit}$? The following argument, due to Coniglio, et al. [14], will answer this question.

We consider a new walk on the lattice called the smart kinetic walk (SKW): the idea behind this is to avoid getting trapped in a loop where you cannot get out without self-intersection. The walker starts at the origin and moves one step, labelling the hexagon on the left with a $+$ and the one on the right with a $-$. For the next step, the same procedure is followed, and this continues. When the walker reaches a point

where a random choice could lead to being trapped in a loop, the hexagons on both sides will have already previously been labelled, so the walker must simply choose the path with + on the left and − on the right. The only way for the SKW to terminate is for the walker to return to the origin.

What is the probability of an N-step SKW? For a non-interacting SAW model, the weight will be $\frac{1}{3}(\frac{1}{2})^{N-1}$, since the first step can go in any of the three directions, and each subsequent step can go in any two directions. For the SKW, this factor will be multiplied by $2^{N_{NN}}$ where N_{NN} is the number of times we have been forced to step in a particular direction: i.e. the number of edges with a weight of one. In fact, we can be more precise about N_{NN}: it is one plus the number of hexagons which contain two steps of the walk not connected on that hexagon plus twice the number of hexagons that contain three steps of the walk not connected on that hexagon (see Figure 5.1 and recall the definition given above of the nearest-neighbor interaction for our interacting SAW on the hexagonal lattice). Each time one of the configurations in this figure appears in the walk (i.e., each time we have a nearest-neighbor interaction of the sort (A)-(F)), it means we have been forced to make a step either to avoid self-intersection or to avoid self-trapping, and, on the other hand, each time we have been forced to make such a step, one of these configurations will appear in the walk.

However, $2^{N_{NN}}$ can be written more suggestively as $\exp(\ln 2 N_{NN})$, and this is precisely the weight that our interacting SAW's defined above have at $\beta = \ln 2$. In other words, the statistics of the SKW correspond to those of the interacting SAW's

at the temperature $T = T_{\Theta'} = \frac{1}{\ln 2}$. What can we say about ν at this value of β?

Our walker actually traces out the outer layer of what is called a percolation cluster. Percolation theory can be defined as the study of collective properties of random objects, and percolation clusters are sets of objects (such as points or edges of a graph) which are somehow connected to each other.

The following example will hopefully make this rather abstract definition more concrete: construct some sort of lattice in the plane (see Section 2.2 for the definitions of lattice, edge, plaquette, etc.). Two points will be called nearest neighbors if they have an edge in common; bonds (i.e. edges) are nearest neighbors if they have a point in common; plaquettes are nearest neighbors if they have a common bond on their boundary. Then, choose a probability p and, for each point (point percolation) or each bond (bond percolation) or each plaquette in the newly formed lattice, pick a random number r; if $r < p$, color the selection black, otherwise color it white. Now define a cluster to be a collection of black points (bonds, plaquettes) that can be connected to each other through black points (bonds, plaquettes) that are nearest neighbors. It turns out that for many lattices, the existence of a cluster with an infinite number of points (bonds, plaquettes) depends on p as follows: for $p < p_{crit}$ there is no infinite cluster, but for $p > p_{crit}$ there is. p_{crit} depends on the lattice and on whether points or bonds or plaquettes are being considered. There are many variations on this theme: for example, considering a square lattice and bond percolation, we can have different probabilities p_{hor} and p_{ver} for the horizontal and vertical directions. See Stauffer [65]

for more details.

We now consider plaquettes on a hexagonal lattice. Choose a probability p and randomly label the hexagons on the lattice with $+$ or $-$ according to this p; the resulting lattice will have clusters of $+$'s and of $-$'s, with the borders being bonds with a $+$ hexagon on one side and a $-$ hexagon on the other. If we set $p = \frac{1}{2}$, we get precisely the same cluster which was traced out by our SKW during its walk, and for a percolation cluster, the dimension of the cluster is known to be $\frac{7}{4}$, thus $\nu = \frac{4}{7}$ and so we have found the intermediate state of the interacting SAW at $\beta_{crit} = \ln 2$.

Although we have looked carefully at one example, we have not yet given a general definition of what exactly a phase transition is. One way to look at it is the following: a mechanically isolated system at constant temperature $T \geq 0$ (for example, a self-avoiding walk at fixed β with some interaction between the nodes of the walk) wants to minimize the free energy $F = E - TS$ where E is the energy and S the entropy. At any given T there is a sort of competition between E and S. When T is very small, E plays the dominant rôle, and so those configurations which minimize E are chosen; for our SAW's, these configurations are the scrunched up ones with a lot of nearest-neighbor interactions. On the other hand, when T is large, it is S which plays the bigger rôle, and so those configurations which minimize S are chosen; for SAW's, these are the stretched out ones. At some intermediate or "critical" temperature T_{crit}, E and S balance each other out. So, as we decrease T from $+\infty$, there is a sort of "transition" between the scrunched up and the stretched out "phases". As we

increase the size of our system (i.e. the length of the SAW) this transition becomes more and more abrupt: as N increases, E and S become larger, and so even small changes in T can produce very large changes in F. For $N = \infty$, we would expect the transition at T_{crit} to be very, very abrupt; in fact, if we are lucky we might be able to find some quantity $\Phi(T)$ which has some kind of discontinuity (or non-analytic behavoir) at T_{crit}.

To be more quantitative, one generally accepted definition of a phase transition begins with a variable Φ called an order parameter: the exact defintion of Φ depends upon the system you are considering, but roughly speaking points where Φ has some type of "singular" behavior indicate phase transitions: "singular" could mean $< \Phi >$ is 0 on one side of the transition and non-zero on the other side. For example, suppose we have a lattice of magnetic spins $\{\mathbf{s_i}\}$; we can define $\Phi(\mathbf{x})$ to be the mean magnetization in a small volume δV around $\mathbf{x}$:

$$\Phi(\mathbf{x}) = \frac{1}{\delta V} \sum_{i \in \delta V} \mathbf{s_i} \tag{4.13}$$

Having chosen an appropriate order parameter, we want to use this to better characterize the behavior of the system near a critical point, which can be defined as the point(s) at which our function Φ is non-analytic. To examine this behavior more quantitatively, we define the connected two-point correlation function $G_c^{(2)}$ by

$$G_c^{(2)}(r) \equiv < \Phi(0) \cdot \Phi(\mathbf{r}) > - | < \Phi > |^2 \tag{4.14}$$

This is a measure of the long-range order in the system.

Let us define the symbol $A(r) \sim B(r)$ to mean $\lim_{r\to\infty} \frac{\log A(r)}{\log B(r)} = 1$; similarly, in the following equations $A(T) \sim B(T)$ means $\lim_{T\to T_{crit}} \frac{\log A(T)}{\log B(T)} = 1$.

For r large and $0 < |T - T_{crit}| \ll T_{crit}$,

$$G_c^{(2)}(r) \sim \frac{e^{-r/\zeta}}{r^{d-2+\eta}} \tag{4.15}$$

where η is a critical exponent defined by the above equation, and where ζ is the correlation length: i.e. the order parameter fluctuates on length scales ζ.

Empirically, it is also found that for $|T - T_{crit}| \ll T_{crit}$,

$$\zeta \sim |T - T_{crit}|^{-\lambda} \tag{4.16}$$

where λ is another critical exponent.

Why are we worried about these critical exponents? Experimentally it has been observed that many systems which are seemingly quite different have the same critical exponents. This has led to the concept of universality classes, which are collections of systems; all of the systems in a given collection have the same values for their critical exponents. The values depend on, for example, the spatial dimension and any symmetries in the Hamiltonian, but not on the details of the microscopic interactions. In our discussion above of the phase transition of interacting SAW's, we used a hexagonal lattice when we looked at the structure of the intermediate state: universality would lead us to believe that these conclusions are valid on other (e.g. square or triangular) lattices too. Universality would also suggest that, although we chose our definitions of the nearest neighbor interactions on the square and cubic lattices partially because

of the ease and speed with which they can be computed, the resultant statistics of the interacting SAW's may be the same as those which would be observed for other localized (and from a computational point of view possibly less efficient) interactions, although this is not necessarily the case.

We should be clear here about the meaning of critical exponents: in Section 2.2 we defined these for functions of SAW's as the length of the SAW's approaches ∞. Here however, a critical exponent describes the behavior of some function the system in the neighborhood of a critical point T_{crit} (i.e. when $\frac{|T-T_{crit}|}{T_{crit}} \ll 1$). In the literature the term "critical exponent" is used for both cases: for the sake of clarity, we will call the first case "SAW" and the second "thermodynamic" in the following discussion.

Consider Equation 2.5: the "SAW" exponent ν is defined for both interacting and non-interacting SAW's, and for the non-interacting case, we can think of no sensible "thermodynamic" interpretation. However, if we give the interacting SAW's a probability which depends not only on the number of interactions but also on some variable T, then we can examine the end-to-end distance of the SAW's as T is changed and, if we observe non-analytic behavior near some $T = T_{crit}$, we can also consider ν as a "thermodynamic" critical exponent in the neighborhood of this T_{crit}. Such an interpretation may also be possible for other exponents, but we will chiefly be concerned with ν. In what follows, the reader should interpret critical exponent in the thermodynamic sense, unless otherwise indicated.

Phase transitions were originally classified by Ehrenfest according to the smooth-

ness of the chemical potential μ, which is defined by

$$\mu = \frac{\partial E}{\partial N} \tag{4.17}$$

where E is the energy of the system and N is the number of particles. A transition was called n_{th} order if the lowest order derivative of μ to be discontinuous across the transition was the n_{th} derivative. However, the usual classification now uses the latent heat

$$l = \int_{T_c^-}^{T_c^+} C_h(T)dT \tag{4.18}$$

where $C_h = (\frac{\partial <E>}{\partial T})_h$ is the specific heat and h is a thermodynamic variable, e.g. volume, which is held constant. If the latent heat is non-zero, the transition is called *first-order*, otherwise it is called *continuous*.

One of the problems with trying to simulate a phase transition of SAW's (or of any system) on a computer is that a "true" phase transition can only take place for an SAW of infinite length. This is because all the thermodynamics of a system can be derived from the partition function Z; in other words, if we know Z we can find the other interesing thermodynamic variables. For example, the average energy $< E >$ is given by

$$< E >= \frac{1}{Z}\sum_{\alpha} E_\alpha e^{-\beta E_\alpha} = -(\frac{\partial \log Z}{\partial \beta})_V \tag{4.19}$$

where V is the volume; the heat capacity $C_V = (\frac{\partial <E>}{\partial T})_V$ at constant volume can be found from Z using

$$C_V = k_B\beta^2(\frac{\partial 2}{\partial \log Z}\beta)_V \tag{4.20}$$

We refer the reader to Binney, et al. [8] for other calculations of thermodynamic variables using Z.

If the system size N is finite, Z, being the finite sum of analytic functions (exponentials), will itself be analytic; thus, we cannot expect to be able to define some order parameter whose "singular" behavior will indicate a critical point. On the other hand, an infinite sum of analytic functions can in some cases exhibit singularities; in some physical systems, this singular behavior can be associated with a phase transition. Therefore, although we cannot hope to see these singularities in the order parameter for our finite system, if we choose this parameter appropriately, the phase transition may be reflected in the behavior of the parameter. If we study these quasi-transitions as a function of system size, this may tell us something about the phase transition in the infinite system.

Another way to think of the finite-size effect is to go back to the correlation length (see Equation 4.15 and Equation 4.16): as soon as $\zeta \approx N$ where N is some measure of the system size (e.g. the length of the SAW), we will not be able to detect any increase in ζ. For the system of size N, we will thus arrive at our "phase transition" at some β different from β_{crit}.

So, what can be done to take into account the finite size of the system? What we want to do is find the dependence between β_{crit} (or T_{crit}) and N. What exactly is β_{crit} for a finite system? We can define it to be the β at which $\zeta \approx N$.

There is also another criterion that is sometimes useful in detecting the transition

for a finite system. If, for an infinite system, near T_{crit} C_h has the following behavior:

$$C_h \sim t^{-\alpha}, T > T_{crit} \sim (-t)^{-\alpha'}, T < T_{crit} \tag{4.21}$$

where α and α' are critical exponents and $t = \frac{T-T_{crit}}{T_{crit}}$, then one way to locate T_{crit} for finite N is to find the temperature at which the specific heat

$$C_h = (\frac{\partial E}{\partial T})_h = \frac{1}{k_B T^2}_{-1}(< H^2 > - < H >^2) \tag{4.22}$$

has a maximum. If we use this technique, then according to Ferdinand and Fisher [22], we can expect the shift in critical temperature to scale as

$$\delta T_{crit} = T(\infty) - T(N) \sim N^{1/\lambda} \tag{4.23}$$

where λ is the critical exponent defined in Equation 4.16 above.

Similarly, if we look at the free energy F defined by

$$F = -\frac{1}{\beta} \ln Z \tag{4.24}$$

then, for a finite system, $F(N)$ will have the following behavior:

$$F(N) = N^{-(2-\alpha)/\lambda)} \bar{F}(tN^{1/\lambda}) \tag{4.25}$$

where t and α are defined as in Equation 4.21 and $\bar{F}$ is a scaling function. The exact form of $\bar{F}$ and the proportionality constant in Equation 4.23 depend on the details of the system.

It should also be noted that we can only observe a given system for a finite time τ_{obs}: for SAWs, this means that we can only apply our Markov transition matrix

a finite number of times, and so we can only examine a finite sequence of states $\alpha_1, ..., \alpha_M$; for other systems such as the Rouse and Zimm models, this means we can only take a finite number of time steps.

In Equation 4.6 we define ergodicity as follows: for any two states, there is a non-zero probability that we can reach one from the other in some finite number of steps of our Morkov process. This definition does not say anything about how small this probability can be or how many steps we may need. Since we can only observe a system during the time τ_{obs}, the concept of ergodicity on a time scale τ is much more useful from a computational point of view. What exactly does this mean? Remember that we want to explore as many of the (high probability) states of the system as possible, and we want the resulting approximations of thermal averages, which are calculated using the states of the system we have explored, to be close to the exact values. Thus, if we say a system is ergodic on a time scale τ, then if $\tau_{obs} \approx \tau$, our calculations should give thermal averages close to the exact values. The value of τ depends on the particular thermal averages we are calculating as well as the accuracy with which we want to compute these averages.

The ergodic time scale τ is determined by the size of the energy barriers between various minima of the free energy F: the larger the barriers, the longer the time needed to explore different states. If τ is relatively large and we only look at the system for a short period of time, the system may only remain in a small portion of phase space, and so our calculations may result in very inaccurate approximations to

the thermal averages. However, τ is also determined by the particular algorithm we are using to examine the state space. If we have a particularly efficient algorithm, we may be able to move around the energy barriers and explore phase space more quickly than if we were using a less efficient one.

We can also give a more quantitative definition of τ. Define the autocorrelation function $C(k)$ for the variable $X = X(\alpha)$ by

$$C(k) = \frac{< X(\alpha_j)X(\alpha_{j+k}) > - < X(\alpha_j) >^2}{< X(\alpha_j)^2 > - < X(\alpha_j) >^2} \tag{4.26}$$

where α_j is the jth state in our sequence, and the averages are taken over our particular sequence of states $\alpha_1, ..., \alpha_M$. Note that $C(0) = 1$. τ can then be defined as $\tau = k$, where k is such that $C(k) \leq 0.03$ for example.

When β_{crit} is approached, τ becomes large, and so sampling the phase space with our Monte Carlo method becomes more difficult. Why does this happen? To a certain extent, the reason is related to the algorithm that we use. As we approach a phase transition, the correlation length ζ grows as in Equation 4.16: this reflects the increasing long-range order of the system. Thus, we can expect that if we use something like the Madras-Sokal method for SAW's (see Section 5.2) to go from state α_j to state α_{j+k}, we will require more and more transformations in order to undo this long-range order and get to a new, more statistically uncorrelated state (a state α_{j+k} such that in Equation 4.26 $C(k) \leq 0.03$): i.e. we will need more time to move around through the various free energy barriers. The blowing up of τ near β_{crit} is

called *critical slowing down.* The relationship between τ and ζ is often written as

$$\tau = \zeta^z \tag{4.27}$$

where z is the dynamical critical exponent and varies according to the system and the algorithm.

For a finite system, τ must remain finite even at our quasi-transition (i.e. at the point where small changes in the temperature cause large changes in the order parameter Φ or in one of its derivatives), since the free energy barriers between the various states and the correlation length ζ must also remain finite. However, τ does increase, and this requires more computing time for Monte Carlo simulations near the quasi-phase transition. For magnetic spins on a lattice, various algorithms have been developed which help to reduce the value of z, such as the Swendsen-Wang and the Wolff algorithms (see Binney, et al. [8]). For SAWs, however, Madras-Sokal (see Section 5.2) only requires a relatively few number of steps to generate "effectively independent" configurations, so z is less than for many algorithms which only modify one or two steps of the SAW at a time (see Section 5.2).

One question that has not yet been addressed is why study phase transitions at all. The statistics of real polymers in dilute solutions of good solvents are known to correspond to those of a non-interacting SAW. When the attraction between the monomers cannot be neglected, a phase transition similar to the one observed in the interacting SAW's takes place (i.e. a small change in β results in a huge change in, for example, the volume occupied by the polymer), and many biological systems operate

near β_{crit} in order to take advantage of this transition (e.g. it is possible that the collapse transition in DNA is actually such a phase transition). For more examples, see Binney, et al. [8].

Chapter 5

Numerical Methods

5.1 Introduction

In this chapter we will describe the numerical methods that we used in the Monte Carlo calculations, beginning with the (efficient) generation of self-avoiding walks using the pivoting algorithm and Metropolis rejection. Next we will discuss the histogram method for sampling phase space and calculating thermal averages for various temperatures. Then we will outline a method for estimating the entropy of a sequence of walks using only a small portion of each walk.

5.2 Generating SAW's

Self-avoiding walks are used in a number of areas in statistical physics and polymer theory (see Madras and Sokal [43] for references), and until recently an efficient

method for generating SAW's of more than a few hundred steps was not available.

Our goal is to examine the applicability of Monte Carlo methods by looking at an example for which the results are known: the phase transition which occurs for interacting SAW's on a two-dimensional hexagonal lattice. In order to do this, we need to generate a series of SAW configurations for certain values of β and N (the length of the SAW), compute the average end-to-end distance (r_N), and then fit a line to the plot of lnr_N vs lnN for each value of β. Using the graph of the slopes of these lines plotted as a function of β, we want to locate the known critical temperature on the two-dimensional hexagonal lattice and compute the exponent ν for the three states of the SAW (see Section 4.3 for details).

Generating equal probability SAW's is easy in principle; however, doing this efficiently is not at all a simple task: if you procede, for example, by starting at a point and just randomly picking the direction to move in, going to the next point, and continuing this, the chances of generating an SAW of any appreciable length is extremely small.

To be more exact, if we are on a lattice with coordination number l, then the number of random walks Z_N^{RW} of length N is l^N. If we use a (slightly) smarter algorithm that does not allow immediate reversals, then the number of non-reversal random walks Z_N^{NRRW} (the notation, as well as most of this discussion, is taken from Binder and Heermann [7]) still grows exponentially with N as

$$Z_N^{NRRW} = (l-1)^N \tag{5.1}$$

What about the number of SAW's Z_N^{SAW}? In this case, the dependence upon N becomes more complicated, and only for large N (i.e. when $N \to \infty$) can we find a simple form for this dependence, which is given in Equation 2.2 and the discussion preceding it. Using this equation and Equation 5.1 we can find the ratio of Z_N^{SAW} to Z_N^{NRRW}, and thus calculate the efficiency of the simple algorithm for dimensions $d < 4$:

$$\frac{Z_N^{SAW}}{Z_N^{NRRW}} \asymp (\frac{\tau}{l})^N N^{\gamma-1} = \exp(-N \ln \frac{l-1}{\tau} + (\gamma - 1) \ln N) \tag{5.2}$$

So for large N, the probability of producing an SAW decreases exponentially with N; this is called the attrition problem. Thus, the simple algorithm is not extremely useful when trying to generate long walks. [We should note here that the meaning of long depends upon the lattice and the dimension: on the square lattice, for example, long would be a few hundred or less, as our calculation above shows.] Simple sampling can however be used to (efficiently) generate shorter walks, as we will see below in the discussion of Madras-Sokal initialization.

More efficient algorithms to generate equal probability SAW's have been developed, such as the slithering snake algorithm, which consists of removing an edge at one end of the walk and adding one to the other end; an ergodic variant of this also exists for generating SAW's of varying length. See Redner and Reynolds [61], Aragão de Carvalho, et al. [3], Wall and Mandel [68] and Beretti and Sokal [5] and the references cited in these articles. Among such algorithms, one of the most efficient – and the one we have used in our calculations – is due to Madras and Sokal [43] and goes

as follows:

Start with an initial configuration A_0 (i.e. an SAW of length N); the question of how to actually find such an A_0 will be addressed below after we have described the transformation and acceptance parts of the algorithm; suppose for now we have somehow found this A_0.

The next step is to select a random point along the SAW A_0 and apply a lattice transformation to the shorter part of it. A lattice transformation is an element of the symmetry group of the lattice, which consists of rotations, reflections, or combinations of the two. For example, on an n-dimensional square lattice, there are $2^n n!$ such transformations, since every transformation consists of a permutation of an nXn matrix with each diagonal element being ± 1 and every non-diagonal element equal to 0. On a two-dimensional hexagonal lattice, there are six such transformations: rotations of 0, 120 and 240 degrees about the chosen point, and flips about each of the three lines meeting at the chosen point.

To be sure, the chances of generating a new SAW using the Madras-Sokal algorithm also goes to zero as N approaches infinity, but rate at which this decreases is much smaller than for the simple alternative mentioned above: for two-dimensional square lattices, Madras and Sokal [43] estimate that this rate is $\approx N^{-0.19}$. Another advantage of this algorithm is that each transformation changes the SAW fairly radically: we can expect that after a few successful transformations, we will have generated a new, effectively independent configuration.

After applying the lattice transformation, we have to check to see if the new configuration is self-avoiding or not. We could simply check each point against, say, all of the points to the left (or right) of it, but for an SAW of length N this would involve $O(N^2)$ operations. A quicker way would be to create a bit-field with $(2N+1)^d$ elements (where d is the dimension of the lattice) and map each point of the walk into this bit-field: if two points are mapped to the same bit, then we know there is a collision, so the new configuration has to be rejected. The amount of work for this strategy is $O(N)$, but it requires an inordinate amount of storage space for the bit-field: an $N = 500$ walk with $d = 3$ would need more than 125 MB just for the bit-field.

So we want to find a strategy in between the two extremes above: try to reduce the size of the data structure needed for checking for self-intersection, but at the same time stay as close to $O(N)$ amount of work as possible. Instead of a one-to-one mapping of the $(2N+1)^d$ possible coordinates of our walk into the bit-field, we choose a function h (called a hash function) that maps these $(2N+1)^d$ points $\{x_i\}$ into some finite interval of $\mathcal{Z}$.

We proceed in the following manner: We start with the first point in the walk x_0 and store a 0 in the element $h(x_0)$ of an array (this array is called the hash table). Then we go to x_1 and store 1 in the $h(x_1)$ element of the hash table; then we go to x_2, ... Since the size of the hash table will be smaller than $(2N+1)^d$, h will not be one-to-one. What should we do when there is a collision between two points x_i

and x_j, $i < j$ (i.e. when $h(x_i) = h(x_j)$)? If $x_i = x_j$, then we should reject the new configuration since it is not self-avoiding. If $x_i \neq x_j$, we use an auxiliary array and store a j in the i element of this array. Then we continue with the next point x_{j+1}. What happens if $x_i \neq x_j$, but the i element of the auxiliary array is already has a k in it? Then we compare x_j and x_k; if they are equal, stop since the configuration is not self-avoiding; otherwise, store a j in the k element of the auxiliary array, and continue with x_{j+1}. If the k element is occupied by l, compare x_j and x_l as above and repeat this procedure as many times as necessary until finding an empty element in the auxiliary array. This auxiliary array will contain multiple "linked" lists of indices, each list being (part of) an equivalence class of $\mathcal{Z}^d$ under the mapping $h : \mathcal{Z}^d \to \mathcal{Z}$. One of these lists would look like

$$\begin{array}{lcccccccccc} a: & \dots & i & \dots & j & \dots & k & \dots & l & \dots \\ & \dots & \downarrow & \nearrow & \downarrow & \nearrow & \downarrow & \nearrow & \downarrow & \dots \\ b: & \dots & j & \dots & k & \dots & l & \dots & & \dots \end{array} \tag{5.3}$$

with a being the index of the array element, b the value of ath array element, and $h(x_p) = h(x_q)$ for $p, q \in \{i, j, k, l\}$. If there is an x_m in the SAW such that $h(x_m) = h(x_l)$ then an m will be put in the lth element.

The first question is how to choose h. We know that our SAW is connected, so since the points $\{x_i\}$ that we will be checking will be fairly close together, the hash function should separate these points: that is, if $l \ll N$, then $|h(x_i) - h(x_{i+l}| \approx O(N)$. Otherwise, we would have too many collisions and thus spend too much time checking

and inserting indices into the auxiliary list: this would ruin the efficiency of the algorithm. Let $x_i = (x_i^1, ..., x_i^d)$. The function used by Madras and Sokal (and by us) was:

$$h(x_i) = a_1 x_i^1 + a_2 x_i^2 + ... + a_d x_i^d \bmod M. \tag{5.4}$$

where $a_1, ..., a_d, M$ are relatively prime, and

$$a_i \approx M^{\frac{i}{d+1}} \tag{5.5}$$

If the function h and the prime M are chosen correctly, what can be said about the efficiency of the algorithm? In the worst case, we would have to do N work for an insertion, since we could have collisions with every other element. However, if N is not too close to M, we can avoid having a lot of collisions, and the work involved for each point is $O(1)$ (see Knuth [37]), so the total work is $O(N)$.

There are a few modifications to the above algorithm which decrease the constant in front of the N in $O(N)$: if i is the location of the randomly selected pivot point for the transformation, apply the transformation to $x_0, ..., x_{i-1}$ if $i < \frac{n}{2}$; otherwise, apply the transformation to $x_{i+1}, ..., x_N$. Also, when checking for collisions, proceed in the order $x_{i\pm1}, x_{i\pm2}, ...$ since it is more likely for the points near x_i to collide.

The question now is how to choose M, since this will be the size of our hash table (the auxiliary array is the same size as the walk itself). Madras and Sokal found that M can be as low as $2N$ without compromising the efficiency of the hash table algorithm; we chose M to be around $15N$ for our calculations; the reason is that we use the hash table in the energy calculation below, and so we have to take into

account not only the points in the walks but also all of the nearest neighbors of these points.

We now return to the questions posed above about the initialization process: namely, how can we find an initial configuration A_0? Why not simply use a straight line on the square or cubic lattices, or something similar on the hexagonal lattice? The reason why we cannot do this is given by Madras and Sokal [43]: "When analyzing the data produced by a dynamic Monte Carlo method, one assumes that the observations come from an (approximately) stationary stochastic process whose single-time probability distribution is the desired equilibrium distribution." In other words, if we simply set the initial state to be a straight line, even if we know that our algorithm will eventually produce a Markov chain for which the frequency of occurence of a given SAW is close to this SAW's Gibbs probability, averages computed using a finite portion $A_0, \ldots, A_M$ of this Markov chain may be extremely inaccurate, since the beginning part of the finite chain may contain a large number of walks whose Gibbs probabilities are very low. For example, at low temperatures configurations with a small number of nearest-neighbor interactions have relatively low probabilities: however, if we start with a straight line, there will be a number of SAW's with relatively few nearest-neighbor interactions at the beginning of $A_0, \ldots, A_M$.

There are in fact a number of methods for generating an appropriate initial configuration: the problem is that all of these methods are rather time-consuming. The three considered by Madras and Sokal are simple sampling, dimerization and ther-

malization: simple sampling and its drawbacks have already been discussed above; we will give a brief summary of the other two below, see Madras and Sokal [43] for more details.

Dimerization is a recursive algorithm: if we want an N-step SAW, we generate two $N/2$-step ones, piece them together, and check to see if the resulting walk is self-avoiding. If so, we are done; if not, generate two more $N/2$-step SAW's, put them together, and check again for self-intersections. This continues until we have sucessfully pieced two walks together. How do we generate an $N/2$-step SAW? By piecing together two $N/4$-step SAW's. We continue by considering smaller and smaller SAW's until we have reached a level m such that we can use simple sampling to (efficiently) generate SAW's of length $N/2^m$. The time required for dimerization is $O(N^{c_1 \log_2 N + c_2})$, and it does in fact generate SAW's from the equilibrium distribution.

The third method, thermalization, consists of of starting at some arbitrary initial state (for example, a straight rod on a square or cubic lattice), applying the transformation, and discarding the first T observations so that the distribution of A_T is close to the equilibrium distribution. Unfortunately, the time required for this is at least $O(N^2)$, compared with $O(N)$ for the transformation and self-intersection checking.

At first glance it might seem that Madras-Sokal thus does not provide any significant computational advantages over certain other schemes because of the time required for initialization. However, it turns out that the constants in the $O(N^2)$ and $O(N^{c_1 \log_2 N + c_2})$ are small enough that initialization only becomes a problem for fairly

large N: for example, on a square two-dimensional lattice, thermalization is effective up to $N \approx 10000$, and dimerization works up to around $n \approx 3000$.

This is the Madras-Sokal algorithm for generating equal probability SAW's. Assuming that the algorithm satisfies detailed balance and ergodicity (see below for a proof), the resulting Markov chain of SAW's $A_0, A_1, A_2, \ldots$ will then be a stationary stochastic process, and we can use this chain to compute, say, the average end-to-end distance of non-interacting SAW's of a given length N.

However, the interacting SAW's we are studing do not all have equal probability; instead, the probability of a SAW depends on its energy, which in turn depends on the temperature and the number of interactions between the points in the given SAW. We want to compute the energy of an interacting SAW and determine whether the transition should be accepted or not based upon the energy change between the old configuration and the new one. What modifications must be made to the Madras-Sokal algorithm in order to accomplish this?

For a given lattice, the energy of an SAW can depend on (some subset of the) number of nearest neighbor edge interactions N_{NN}, the number of next-nearest neighbor edge interactions N_{NNN}, and so on. For hexagonal lattices, one possibility is according to the definition given by, among others, Coniglio, et al. ([14]), which gives the statistics of a walk at the temperature $T = T_{\Theta'}$: an interaction consists of the SAW traversing one or more sides of a hexagon, leaving, and then returning. It could also happen that the SAW then leaves and returns yet again, in which case there are

two interactions. Examples of such N_{NN} and N_{NNN} edge interactions are given in Figure 5.1.

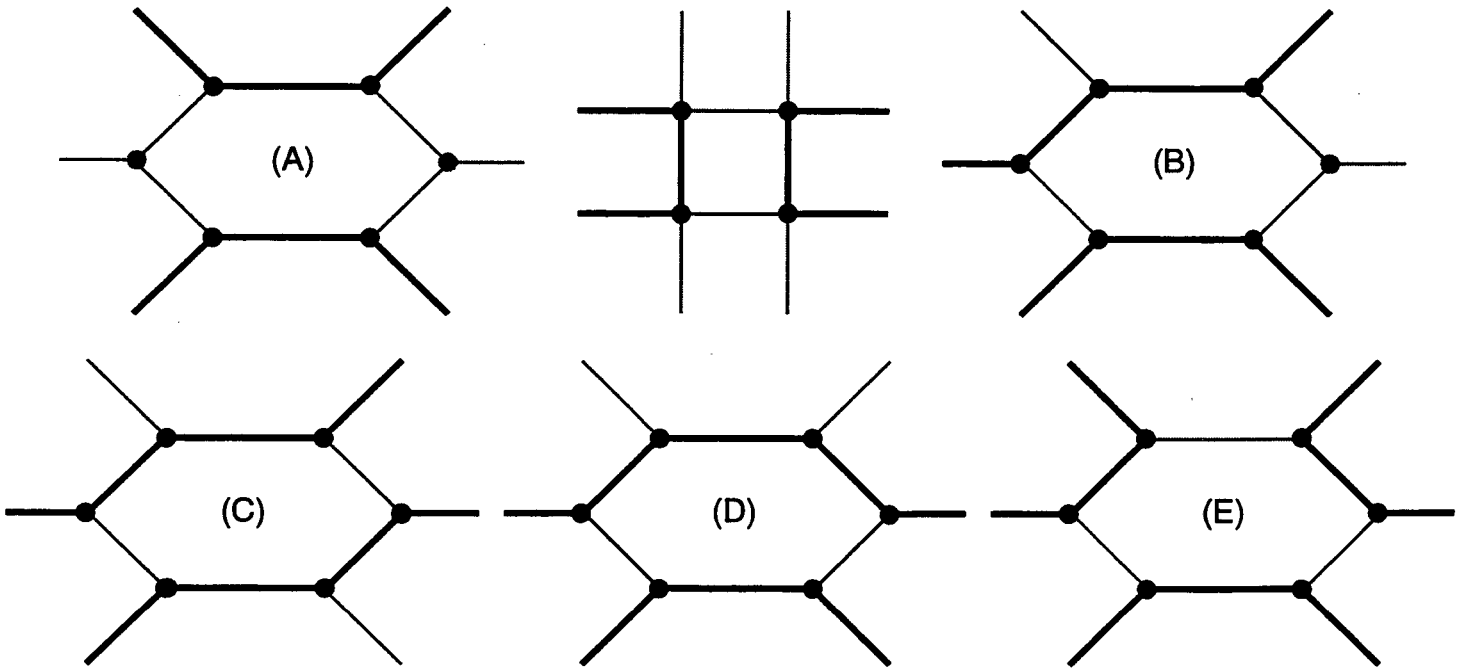


Figure 5.1: N_{NN} and N_{NNN} edges on the square lattice, and (A)-(E) four examples of the edge interaction on the hexagonal lattice: (A)-(D) each have one interaction, (E) has two according to Coniglio, et al.

For the hexagonal lattice, if we follow the definition of edge interactions given by Coniglio, et al. [14], we would also have to check to see if the two edges are in the same hexagon. However, the algorithm would be more complicated: we would have to be more careful about counting the interactions twice - specifically, we would have to worry about the interaction illustrated in part (E) of Figure 5.1 which involves three separate parts of the walk. If the three interacting edges there are labelled i, j, k and if we have already have counted the $\{i, k\}$ and the $\{j, k\}$ interactions, we do not count the $\{i, j\}$ interaction, since the total weight of the interactions in this hexagon

should only be 2, not 3. The reader may say at this point: why not define a simpler N_{NN} and N_{NNN} edge interaction and argue that universality implies we will get the same statistics and behavior as for the more complicated one? This question is all the more valid because the energy calculation takes up most (more than 80 percent on average) of the computing time for each iteration.

Since the code for the interaction defined by Coniglio, et al. is somewhat more complex than it might be for some simpler definitions, and since the actual time penalty is not negligible (compared with, say, a definition which does not need to worry about N_{NNN} interactions), we decided to use simply the N_{NN} interactions when calculating energies. However, there is a caveat: our interaction gives the statistics of a walk at the temperature $T = T_{\Theta}$, and it is not certain that the Θ and Θ' points belong to the same universality class (see Poole, et al. [60] for a more thorough discussion of this point).

The choice of using only N_{NN} interactions also has an effect on the initialization part of the algorithm. Although we did not mention it above, another possibility for initialization near $T = T_{\Theta'}$ could be the SKW algorithm: this generates walks from the ensemble at $\beta = \ln 2$, but this is useful only if we include the appropriate subset of N_{NNN} interactions in our subsequent calculations using the Madras-Sokal algorithm. Since we consider only the Θ point, the corresponding algorithm would be the KGW (see Section 2.4). However, the SKW algorithm is much more efficient than KGW at producing SAW's of any significant length. In fact, since KGW suffers from the

attrition problem illustrated by Equation 5.2, it is impractical except for producing extremely short walks. So, we are left with the three possibilities mentioned above; taking into account the length of our walks and the arguments for and against the three methods, for our calculations we have chosen to use thermalization.

In order to compute the number of nearest-neighbor interactions, you could go through the whole walk and check each point against all of the others to find the nearest neighbor interactions, but this would again involve $O(N^2)$ work. A more efficient way is to use the hash function and the hash table which we have already used above to check for self-avoidance. If we have reached this point, we know the walk is self-avoiding, and the SAW has already been mapped into the hash table. We now find the N_{NN}-neighbors of each point x_i (call these $\{x_{i_k}\}$), and start computing their hash codes $\{h(x_{i_k})\}$, starting with x_{0_k}. On the square lattice, the nearest neighbors of the point (x_i^1, x_i^2) are the points $(x_i^1 \pm 1, x_i^2)$ and $(x_i^1, x_i^2 \pm 1)$; on the cubic, nearest for (x_i^1, x_i^2, x_i^3) are $(x_i^1 \pm 1, x_i^2, x_i^3)$, $(x_i^1, x_i^2 \pm 1, x_i^3)$ and $(x_i^1, x_i^2, x_i^3 \pm 1)$; on the hexagonal, nearest means the other three points which are one step away from x_i, namely $(x_i^1 + 1, x_i^2 \pm 1)$ and $(x_i^1 - 2, x_i^2)$, or $(x_i^1 - 1, x_i^2 \pm 1)$ and $(x_i^1 + 2, x_i^2)$.

If, when computing the hash codes of the neighboring points, we have a collision between x_{i_k} and x_j, and if $i < j$, we check to see if $x_{i_k} = x_j$; if so, we increment the interaction counter N_{NN} (if $j < i$ we do not count the interaction so as to avoid counting each interaction twice). The work here on any of the three lattices is also $O(N)$, although the constant depends on the coordination number of the lattice and

the exact definition of the interaction (whether it includes just nearest neighbors, or also next nearest neighbors, or some other combination).

Now that we have computed the energy of the new SAW, we need to find a criterion for accepting or rejecting it. Remember that we are sampling the phase space in order to compute a thermal average, and so, given enough iterations, we want to eventually reach states α with a large Gibbs probability $\exp(-\beta E(\alpha))$. We used the Metropolis algorithm (see Metropolis, et al. [48]) to decide whether to accept or reject the new SAW: if ΔE is the energy change going from the old walk to the new, then the probability of accepting the new walk is computed as follows:

a) if $\Delta E < 0$, accept the transition

b) if not, pick a random number r between 0 and 1. If $e^{-\beta\Delta E} > r$, accept the new SAW; otherwise, reject it.

The Madras-Sokal algorithm is ergodic (for a proof see Madras and Sokal [43]), and the Metropolis algorithm satisfies the detailed balance condition, since

$$\frac{P(\alpha \to \alpha')}{P(\alpha' \to \alpha)} = \frac{min(1, e^{-\beta(E_{\alpha'}-E_\alpha)})}{min(1, e^{-\beta(E_\alpha-E_{\alpha'})})} = e^{-\beta(E_{\alpha'}-E_\alpha)} = \frac{P(\alpha')}{P(\alpha)}. \tag{5.6}$$

Thus, both of the criteria mentioned in Section 4.2 are fulfilled, and thus we have reason to believe that our thermal averages computed with the Monte Carlo data will approximate well the actual averages (with the caveats discussed in Section 4.3).

5.3 Histograms and Sampling

As stated at the beginning of the previous section, our computations involve generating a series of SAW configurations for various β and N and using this data to compute r_N. These calculations can be extremely expensive, especially for large N; one way to reduce this cost is to reuse the data from a given β_0 to estimate r_N for other nearby values of β. We will outline this procedure (called the histogram method) below; for a more thorough discussion, see Akao [1], Kuchta and Etters [38] and Ferrenberg and Swendsen [23].

The first step in the histogram method is to generate a series of SAW's for the given β_0 and N and store a histogram of the energy and end-to-end distance R_{ee} in two vectors n and r. During this computation, for each SAW b in our Markov chain we compute the number of nearest-neighbor interactions $N_{NN}(b)$ (which determines the energy of this walk) as well as $R_{ee}(b)$. Then we increment $n_{N_{NN}(b)}$ by one and add $R_{ee}(b)$ to $r_{N_{NN}(b)}$. At the end of this computation, we can calculate r_N as

$$r_N(\beta_0) = \frac{1}{m}\sum_{k=1}^{l} r_k \tag{5.7}$$

where l is the number of elements in the vectors n and r and $m = \sum_{k=1}^{l} n_k$ is the total number of walks in our Markov chain.

Now let us look at the actual phase space. Let L be the maximum number of nearest-neighbor interactions that our N-step SAW's can have, and let N_k be the number of walks with k nearest-neighbor interactions. The partition function $Z(\beta_0)$

can be written as

$$Z(\beta_0) = \sum_{k=1}^{L} \exp(-k\beta_0) N_k \tag{5.8}$$

We can also define a probability measure $P_k(\beta_0)$ which tells us the relative weight of the walks with k nearest-neighbor interactions

$$P_k(\beta_0) = \frac{1}{Z(\beta_0)} \exp(-k\beta_0) N_k \tag{5.9}$$

In order to be able to use our Markov chain to calculate thermal averages, we want

$$P_k(\beta_0) \approx \frac{n_k}{m} \tag{5.10}$$

We will assume this is true for the rest of this calculation. If the Monte Carlo data for the original temperature β_0 are not good (i.e. the original histogram is not smooth), then we cannot expect to be able to use these data for calculating averages at another temperature β.

Using the approximation in Equation 5.10 for $P_k(\beta_0)$, we can rewrite Equation 5.7 as

$$r_N(\beta_0) = \sum_{k=1}^{l} \frac{n_k}{m} \frac{r_k}{n_k} \approx \sum_{k=1}^{l} P_k(\beta_0) \frac{r_k}{n_k} \tag{5.11}$$

So far, storing the histogram data has not saved any cpu time. However, if we want to calculate $r_N(\beta)$ for some other β, instead of calculating an entirely new Markov chain, we can just "shift" the data we have for β by $\exp(\beta - \beta_0)$ and use this to estimate $r_N(\beta)$. In other words, we approximate $r_N(\beta)$ by replacing $P_k(\beta_0)$ in Equation 5.11

with

$$P_k(\beta) = \frac{1}{Z(\beta)} \exp(-k\beta) N_k \tag{5.12}$$

From Equation 5.9, we know that

$$N_k = P_k(\beta_0) Z(\beta_0) \exp(k\beta_0) \tag{5.13}$$

We can also use Equation 5.13 to rewrite $Z(\beta)$ in terms of $Z(\beta_0)$ as follows

$$Z(\beta) = \sum_{k=1}^{L} \exp(-k(\beta - \beta_0)) P_k(\beta_0) Z(\beta_0) \tag{5.14}$$

If we substitute Equation 5.13 and Equation 5.14 back in to Equation 5.12, $Z(\beta_0)$ cancels out, and we get

$$P_k(\beta) = \frac{\exp(-k(\beta - \beta_0)) P_k(\beta_0)}{\sum_{j=1}^{L} \exp(-j(\beta - \beta_0)) P_j(\beta_0)} \tag{5.15}$$

The average $r_N(\beta)$ can then be calculated using

$$r_N(\beta) \approx \sum_{k=1}^{l} P_k(\beta) \frac{r_k}{n_k} \tag{5.16}$$

The key point here is that although we have to recalculate the probabilities $P_k(\beta)$, we can reuse the arrays r and n from the calculation for β_0. This can result in substantial savings in compute time.

There are problems with this approach. For example, the Markov chain that we generated for β_0 presumably will allow us to calculate averages over the phase space for this particular value β_0, but how do we konw this chain will give a good sampling of the phase space when $\beta \neq \beta_0$? Although we cannot offer any rigorous proofs,

what we can hope is that if β is close to β_0, then the averages calculated using $P_k(\beta)$ will approximate closely the true averages. How can we judge whether β and β_0 are close enough? One way is to look at the corresponding histogram plots: if the shifted histogram for beta is very unsmooth or if the peak of the shifted histogram lies very near the tail end of the original one, then we might suspect that the estimated averages do not approximate the true averages very well for β.

Figure 5.2 shows an example of the original histogram as well as several ones which have been shifted according the procedure described above. It is clear that the last few histograms are not at all acceptable, whereas the first shifted one is fairly smooth and its peak lies near the peak of the original histogram. What happens with the last few histograms is the phenomena described by Poole, et al. [60]. As the temperature decreases (i.e. as β increases), the walks with more nearest-neighbor contacts become more probable, and so make a larger contribution to the statistics. At $\beta = 0.3$, those walks with a large number of nearest-neighbor contacts are at the far right of the histogram; they do not make a large contribution to the statistics, and they are not sampled effectively by our Monte Carlo algorithm, but this does not greatly affect the statistics *at* $\beta = 0.3$. However, as β increases, in the histogram method these walks become increasingly important for calculating thermal averages; a smaller part of our original sample data becomes more important in the statistics, and so what was originally a small statistical error at $\beta = 0.3$ becomes disastrous as β grows. Poole, et al. call this an effective reduction in the sample size; this reduction limits the amount

that we can shift a given histogram and still get reasonably accurate statistics.

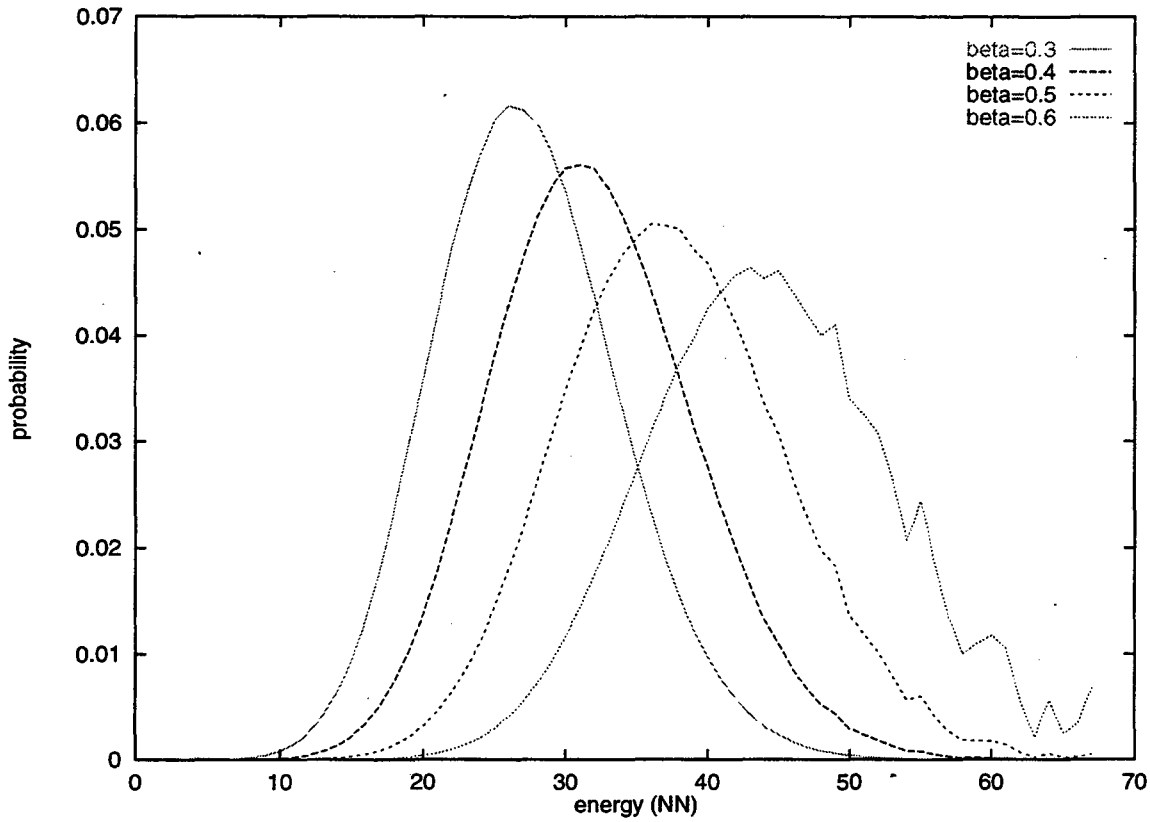


Figure 5.2: The original histogram is from a Monte Carlo run of 2.5×10^7 iterations on a two-dimensional hexagonal lattice with $\beta = 0.3$ using the pivoting and Metropolis algorithm described in Section 5.2. The other three curves were generated using the shifted probabilities in Equation 5.15.

5.4 Entropy Estimation

The motivation behind the histogram method is to use the data from one run at a particular β_0 to help us avoid doing other runs for nearby values of β. The reason we introduce entropy estimation is, in a certain sense, the opposite: we want know how we should combine the data from a series of runs to calculate an average of

some quantity over all these runs. Why we actually need to do this is explained in Section 6.2; here we just describe the entropy estimation itself.

Suppose we have a Markov chain of walks $x_0, x_1, \ldots, x_n$ with each walk consisting of N steps, and suppose we want to calculate the entropy of this chain, where the entropy S is defined as

$$S = -\sum_{i=0}^{n} P(x_i) \log P(x_i) \tag{5.17}$$

with $P(x_i)$ the probability of walk x_i. Except for very small N, we do not know the partition function and so we cannot calculate the probabilities exactly. Instead, what we do is consider a small portion of each walk, say steps 1 through m where $m \ll N$. We then find all of the self-avoiding walks $y_0, y_1, \ldots, y_l$ of length m, and estimate the probability of each walk by the frequency with which it occurs in our Markov chain; call this $\tilde{P}$. This will give us an estimate of the entropy of this portion of the walks; we then scale the estimate to approximate S for the chain of N-step walks. This gives

$$S = -\frac{N}{m} \sum_{i=0}^{l} \tilde{P}(y_i) \log \tilde{P}(y_i) \tag{5.18}$$

We can see why this works if we look at the definition of the entropy in Equation 5.4; suppose we break up a walk of length N into 2 segments of length $m = \frac{N}{2}$, and suppose each of these segments has entropy

$$S_0 = -\sum_{i=0}^{l} \tilde{P}(y_i) \log \tilde{P}(y_i) \tag{5.19}$$

For the walks of length N, we will have l^2 configurations, each of which will have

a probability equal to the product of the probabilites of each segment of length m. Remembering that $\sum_{i=0}^{l} \tilde{P}(y_i) = 1$, we can then calculate the entropy of the N-step walks:

$$\begin{aligned} S &= -\sum_{i=0}^{l}\sum_{j=0}^{l} \tilde{P}(y_i)\tilde{P}(y_j) \log \tilde{P}(y_i)\tilde{P}(y_j) \\ &= -\sum_{i=0}^{l} \tilde{P}(y_i) \sum_{j=0}^{l} \tilde{P}(y_j) \log \tilde{P}(y_j) - \sum_{j=0}^{l} \tilde{P}(y_j) \sum_{i=0}^{l} \tilde{P}(y_i) \log \tilde{P}(y_i) \\ &= -\sum_{j=0}^{l} \tilde{P}(y_j) \log \tilde{P}(y_j) - \sum_{i=0}^{l} \tilde{P}(y_i) \log \tilde{P}(y_i) \\ &= -2S_0 \qquad (5.20) \end{aligned}$$

$$(5.21)$$

A similar calculation works for the general case.

Now, suppose we have a series of M runs, and denote by r_i the average for the ith run of some quantity r. We can use the entropy estimates to put these averages together as follows. Using the free energy $F = E - TS$, where E is the energy, T the temperature, and S the entropy, we can calculate the exact average of r as

$$< r >= \frac{\sum_\alpha r(\alpha) \exp(-\frac{F}{T})}{\sum_\alpha \exp(-\frac{F}{T})} \qquad (5.22)$$

where the sum is over all configurations α in the appropriate phase space. If we know the average energy E_i and average entropy S_i for each run i, we can estimate $< r >$ as

$$< r >\approx \frac{\sum_{i=1}^{M} r_i \exp(-\frac{F_i}{T})}{\sum_{i=1}^{M} \exp(-\frac{F_i}{T})} \qquad (5.23)$$

where $F_i = E_i - TS_i$.

In addition to self-avoiding walks on the hexagonal lattice, this method has also been applied in Monte Carlo calculations of hard sphere lattice gases (see Meirovitch [47]) and vortex filaments (see Chorin [10]), where it has given good results. We have not yet said anything about how good our estimate for $< r >$ will be, and Equation 5.23 is not the only possibility for combining the data from the various runs. For example, we could use the histogram data described in Section 5.3: we simply put together all of the histograms from the separate runs and then use Equation 5.7, with the vectors r and n containing the end-to-end distance and number of walks for all of the runs. Chapter 6 contains a more thorough discussion of this point and some of numerical results for various weighting methods.

Chapter 6

Numerical Results

6.1 Introduction

In this chapter we will discuss in detail some of the difficulties we had with our implementation of the numerical methods in Chapter 5 and how we attempted to resolve these problems. After that we will present and analyze the results of our calculations and attempt to draw some useful conclusions about the simulation of phase transitions using our Monte Carlo algorithm.

We first attempted to do what we thought would be a straightforward Monte Carlo calculation of the critical exponent ν in Equation 2.5 as a fuction of the inverse temperature β on a two-dimensional hexagonal lattice using the pivoting algorithm with Metropolis rejection described in Section 5.2. The goal was to locate the phase transition which takes place at the Θ point $\beta \approx 0.99$ (see Coniglio, et al.

[14] for more details). For $\beta = 0$ all walks have the same probability and there is no nearest-neighbor interaction; the pivoting algorithm at this value of β is very efficient compared with other methods, as is shown in Madras and Sokal [43]. However, we encountered some problems when attempting to extend the algorithm to $\beta \neq 0$ which we will attempt to explain in the following sections.

6.2 Acceptance fraction and trapping

For a given iteration of our algorithm, there are three possible outcomes:

- the new walk is rejected because it is not self-avoiding;
- the new walk is rejected because its energy is significantly larger than the energy of the current walk;
- the new walk is accepted.

For a given run, let M be the total number of iterations of our algorithm, and let r_i, r_e, and r_a be the number of walks rejected because of self-interection, the number rejected because of energy considerations, and the number accepted, respectively. We then have $M = r_a + r_e + r_i$.

In Figure 6.1, Figure 6.2 and Figure 6.3 we show as a function of N and β the fraction of transformations rejected because of self-intersections, the fraction rejected because of energy considerations, and the fraction accepted, respectively. As N becomes larger, it can be seen that the number of accepted walks becomes smaller: this

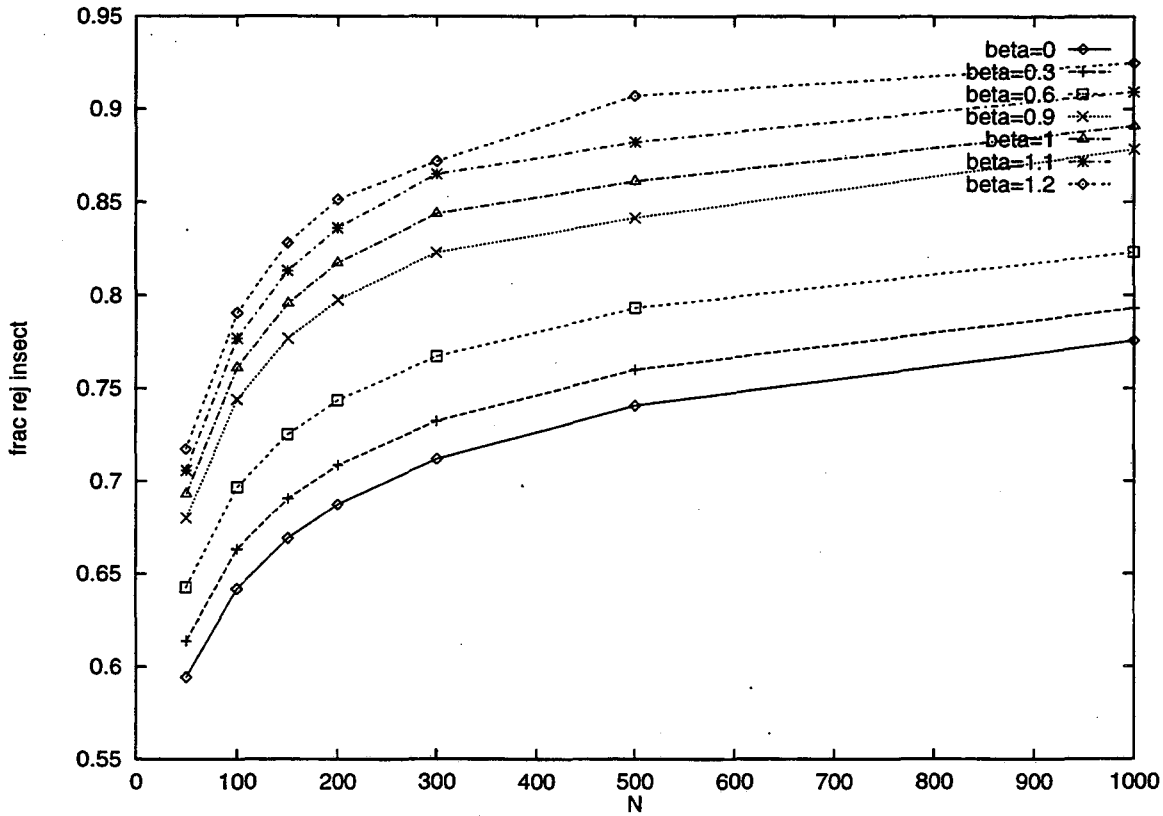


Figure 6.1: Fraction of attempted transformations rejected because of self-intersection vs. length of the walk N for various values of β. For each value of N and β, we first initialized the walks by performing the pivoting algorithm with $T = \infty$ until 10^5 transformations were accepted; we then did then same with $T = \frac{1}{\beta}$. After that, we performed at least 5×10^7 steps of the pivoting algorithm with Metropolis rejection and recorded the number rejections due to self-intersection. We repeated this whole procedure for five separate runs and averaged the results.

is to be expected whether β is zero or not. However, the number accepted becomes extremely small as β approaches the critical value $\beta_{crit} \approx 0.99$. At first glance one might think this would have something to do with an increasing number of Metropolis rejections (i.e rejections because of unfavorable changes in the energy). In Figure 6.2, one can see that the fraction of transformations rejected because of energy considerations is indeed an increasing function of β, although it remains relatively small even near β_{crit}.

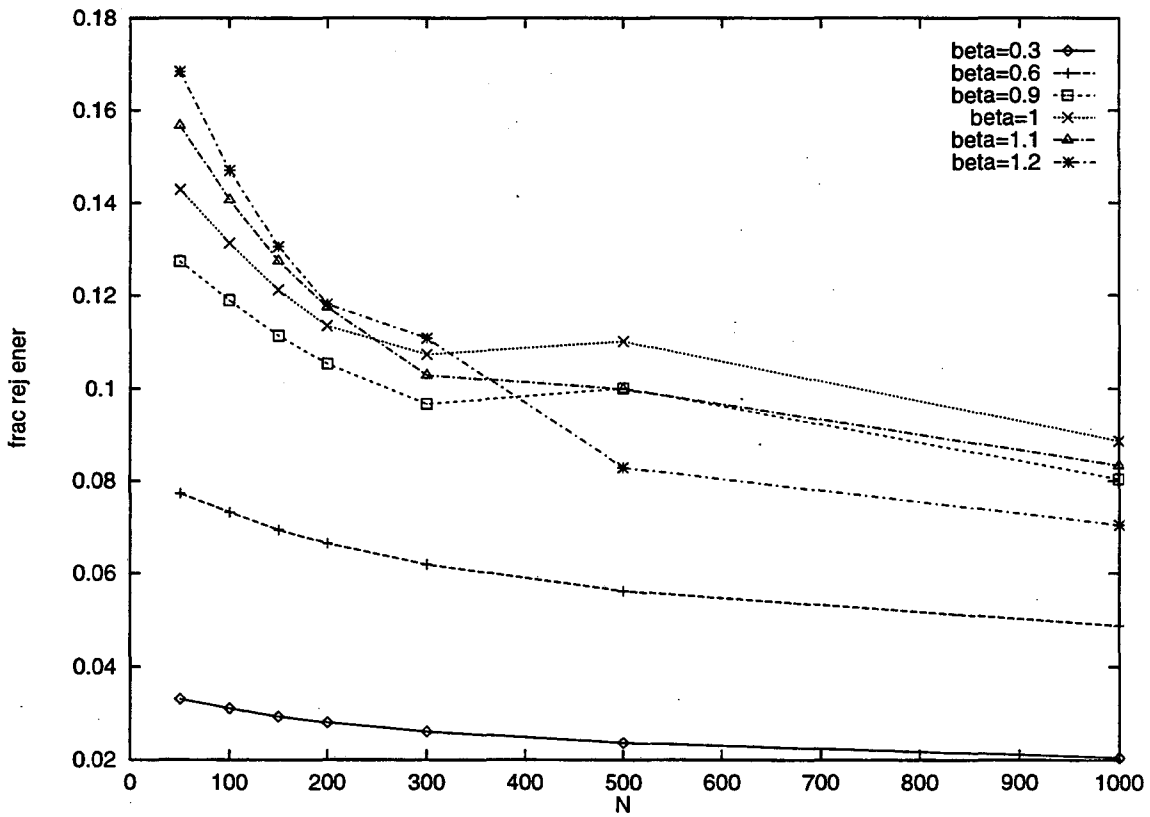


Figure 6.2: Fraction of attempted transformations rejected because of energy vs. length of the walk N for various values of β. The statistics were calculated from the same runs in the same way as in Figure 6.1. Note that the rejection percentage actually decreases with increasing N. See the text for a more detailed discussion of this.

Note that, as illustrated in Figure 6.2, the fraction of Metropolis rejections is actually a decreasing function of N, the number of steps of the walk. The reason for this is that as N increases an increasing percentage of transformations are rejected because of self-intersection, so the algorithm in most cases does not even get a chance to decide if the energy change is favorable or not: self-intersection of the proposed walk causes it to be rejected before the Metropolis energy calculation.

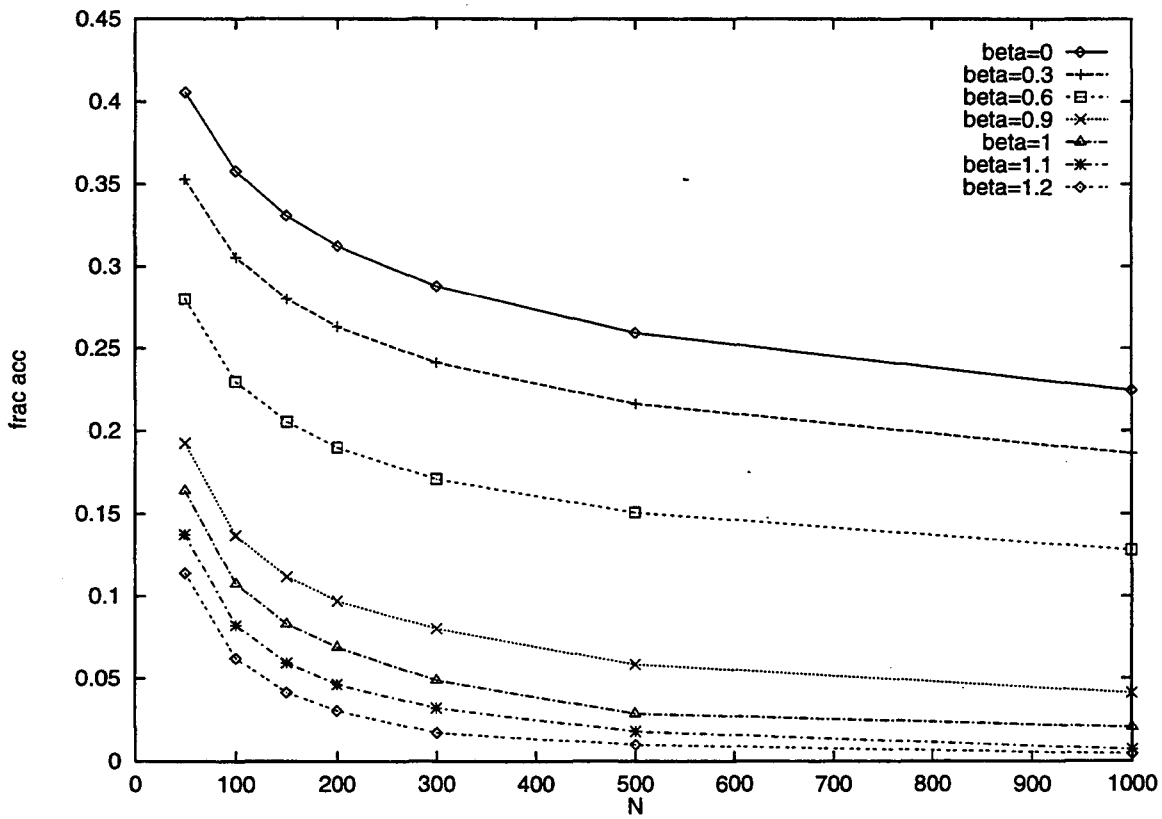


Figure 6.3: Fraction of attempted transformations accepted vs. length of the walk N for various values of β. Again, the statistics were calculated from the same runs in the same way as in Figure 6.1 and Figure 6.2.

In fact, this suggests that what we should be looking at is not the number of energy rejections r_e divided by the total number of iterations M, but instead r_e divided by

the number of iterations of the algorithm which generated self-avoiding walks: i.e. $\frac{r_e}{M-r_i} = \frac{r_e}{r_e+r_a}$. This ratio is shown in Figure 6.4; here the effect of increasing β can clearly been seen in the increasing percentage of rejections due to the Metropolis step. If we go back to Figure 6.1, we can see that there is also a somewhat less dramatic increase in the percentage of rejections due to self-intersection; this can also be attributed indirectly to the Metropolis step, since as β increases, the walks become more contracted, so the probability that a randomly chosen transformation at a randomly chosen step of the walk produces a self-intersecting walk becomes correspondingly greater.

If the rejection percentage increases with N and β, why not just do more iterations of the algorithm? Here we ran into another problem in our calculations which can be termed (computational) non-ergodicity. The non-ergodic behavior can best be illustrated by an example from our data, which is shown in Figure 6.5. In this case, for $\beta = 0.9$ and $N = 1500$ (a medium-sized walk not too far away from $\beta_{crit} \approx 0.99$ we used the same initial configuration and computed the average end-to-end distance R_{ee} as a function of the length of our Markov chain for ten different runs. For each individual run, we had initially attempted to do a running estimate of the variance of R_{ee} and use this as a stopping criterion: however, we found that in some cases the individual runs were converging to (significantly) different values of R_{ee} and thus gave different values for the critical exponent ν. This behavior is clearly illustrated in Figure 6.5.

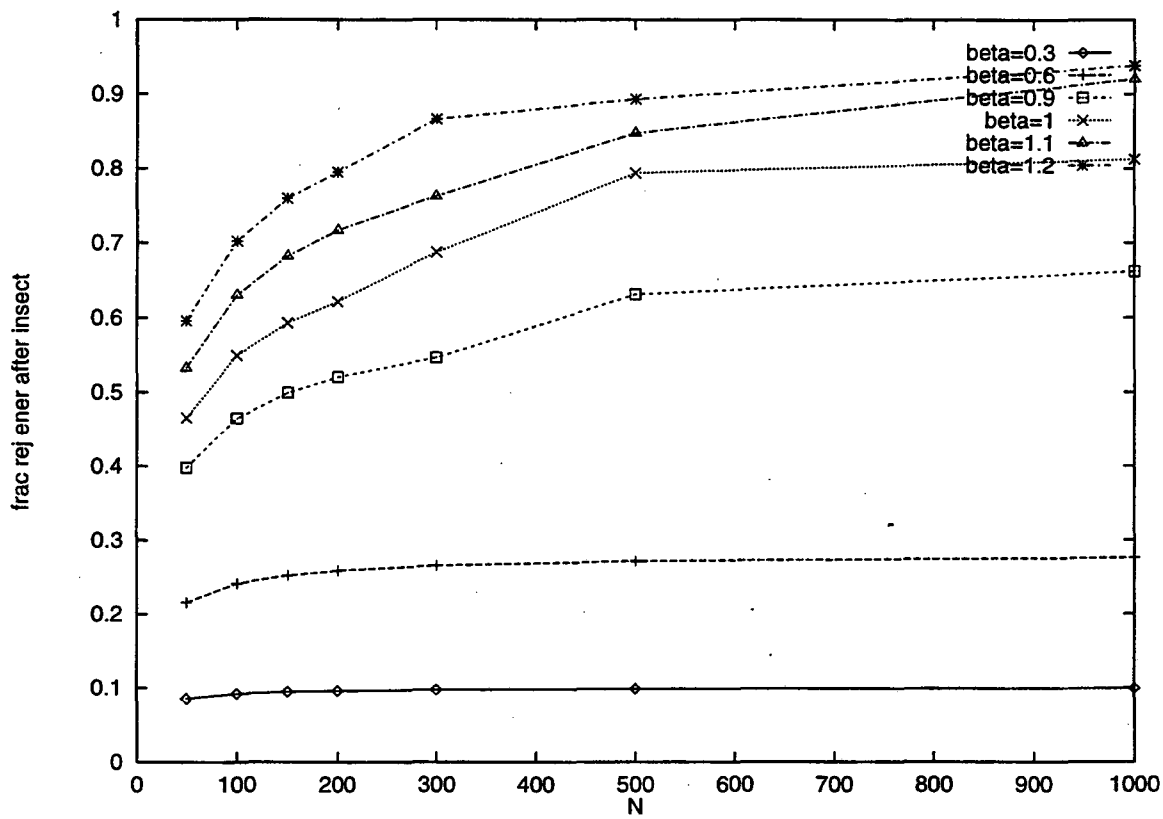


Figure 6.4: Number of energy rejections/(total iterations-rejections because of self-intersection) vs. length of the walk N for various values of β. Again, the statistics were calculated from the same runs in the same way as in Figure 6.1 and Figure 6.2.

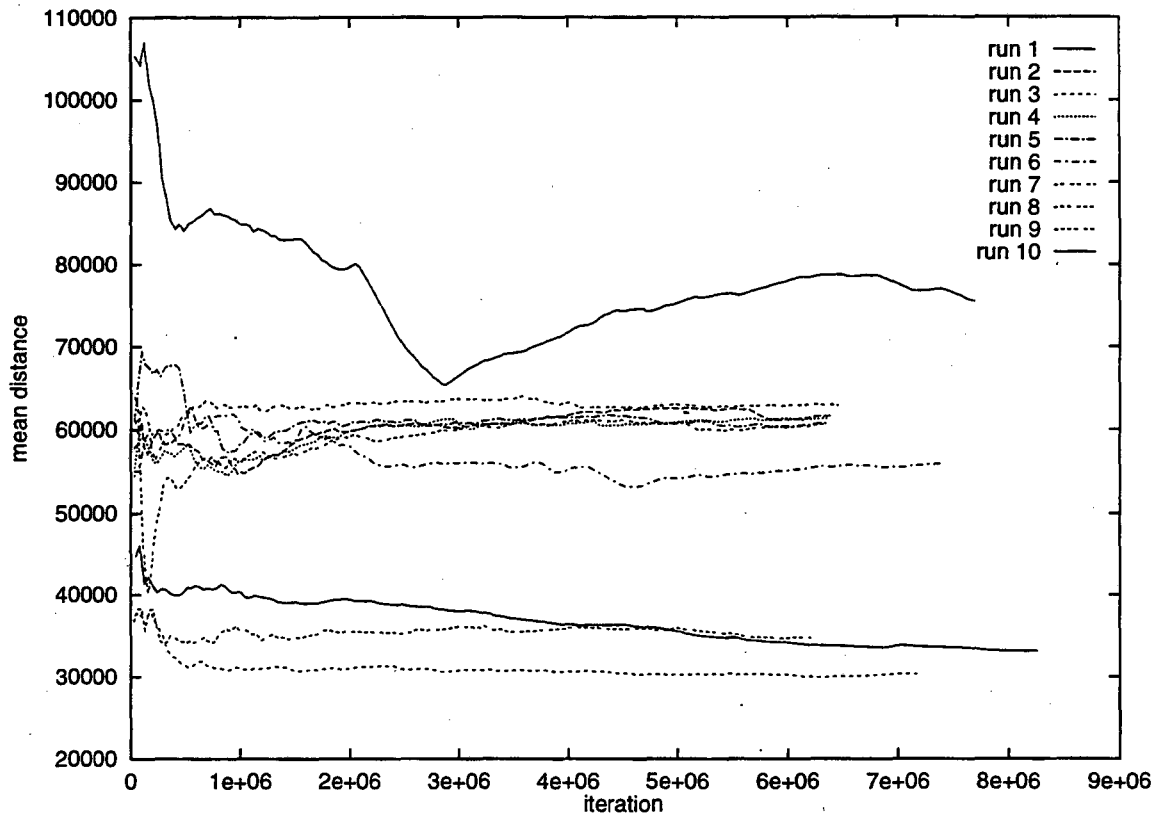


Figure 6.5: Average end-to-end distance vs. number of iterations for $\beta = 0.9$ and $N = 1500$. In this case, iterations means the number of times we were able to perform a pivoting transformation without self- intersection; the actual number of accepted walks (i.e. after the Metropolis rejection step) was between 1.1×10^6 and 2.5×10^6. See the text for a further discussion of this point. All runs started with the same initial configuration; we first set $\beta = 0.0$ and did 400000 iterations (for $\beta = 0.0$ this means 400000 walks had been accepted), then we set $\beta = 0.9$ and did 400000 more iterations before begining to calculate the end-to-end distance.

Although we have no direct proof (one reason is that the size of the phase space for $N = 1500$ is enormous), what we suspect is that the pivoting and Metropolis algorithm becomes computationally non-ergodic for relatively small values of N and values of β near β_{crit}; phase space breaks up into valleys separated by energy peaks which in some cases require an inordinate amount of time to surmount. Although we have a number of walks from each valley, attempting to transform walks from one valley into walks in another in order to illustrate this energy barrier is both difficult and not particularly enlightening. Difficult because we cannot let the algorithm itself do it: since the sample space is so big, the time required to transform one randomly selected walk into another is too long; also, developing an algorithm to directly transform one walk into another while maintaing the constraint of self-avoidance is not trivial. Even if we could do the direct transform, and even if the resulting path from one walk to another had a big energy barrier, it would not be a very convincing proof of the existence of the above-mentioned valleys since we wouldn't know if there were any alternative paths between the two walks with lower barriers.

Extrapolation of results for N small can be of some help, but there are a few caveats. Since we can only do the enumeration up to around $N = 30$, it is not clear that conclusions drawn from these phase spaces for such small N will be applicable at much larger N. At these small N the number of nearest-neighbor interactions is so small that β needs to be extremely large in order to allow the formation of valleys and peaks in phase space. For example, we set $\beta = 20$ and then ran our algorithm for

small values of N (between 10 and 20); as the reader may have suspected, there was almost immediate convergence towards various walks which represented local minima in the phase space. In Figure 6.6 we show two such walks for $N = 16$.

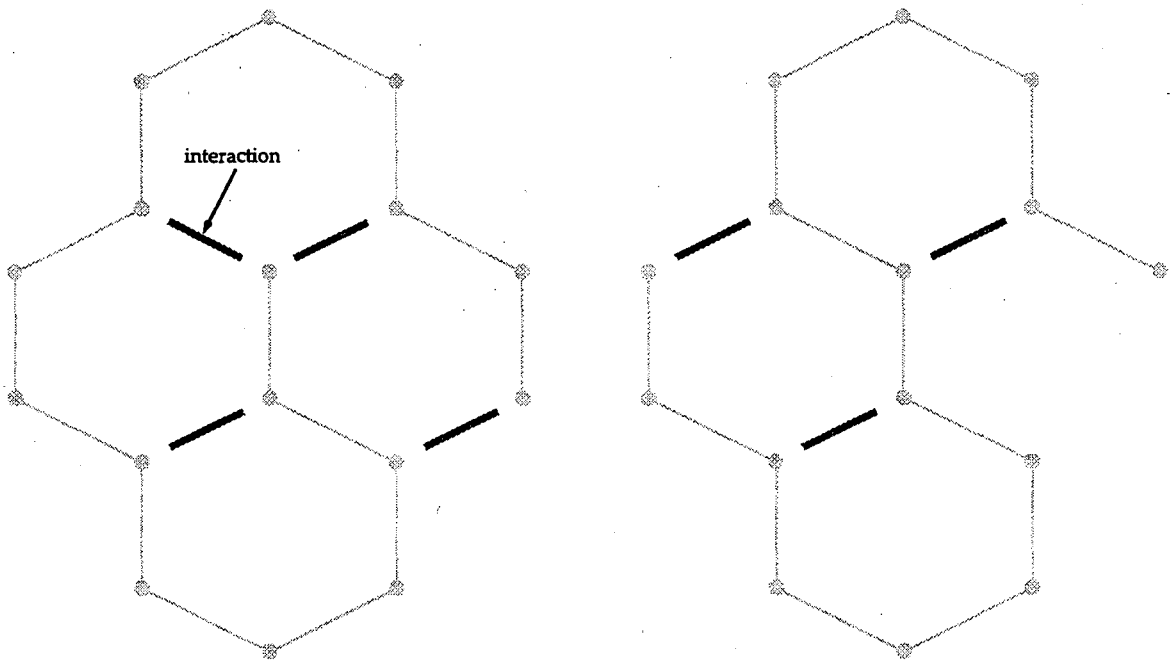


Figure 6.6: Two "trapped" configurations for $N = 16$.

A close examination of the two walks in Figure 6.6 shows that any lattice transformation applied to a part of either of the walks will break some of the bonds (i.e. will decrease the number of nearest-neighbor interactions), and so will be rejected with probability $1 - e^{-\beta}$. If β is large enough, we have from the computational point of view an effectively non-ergodic algorithm; in this case we can define non-ergodic to mean that the number of iterations of the algorithm (and thus the compute time) required to overcome the energy barriers is larger than some given number (say 10^{12}).

We suspect that as N increases, the β required for this to occur gradually approaches β_{crit}.

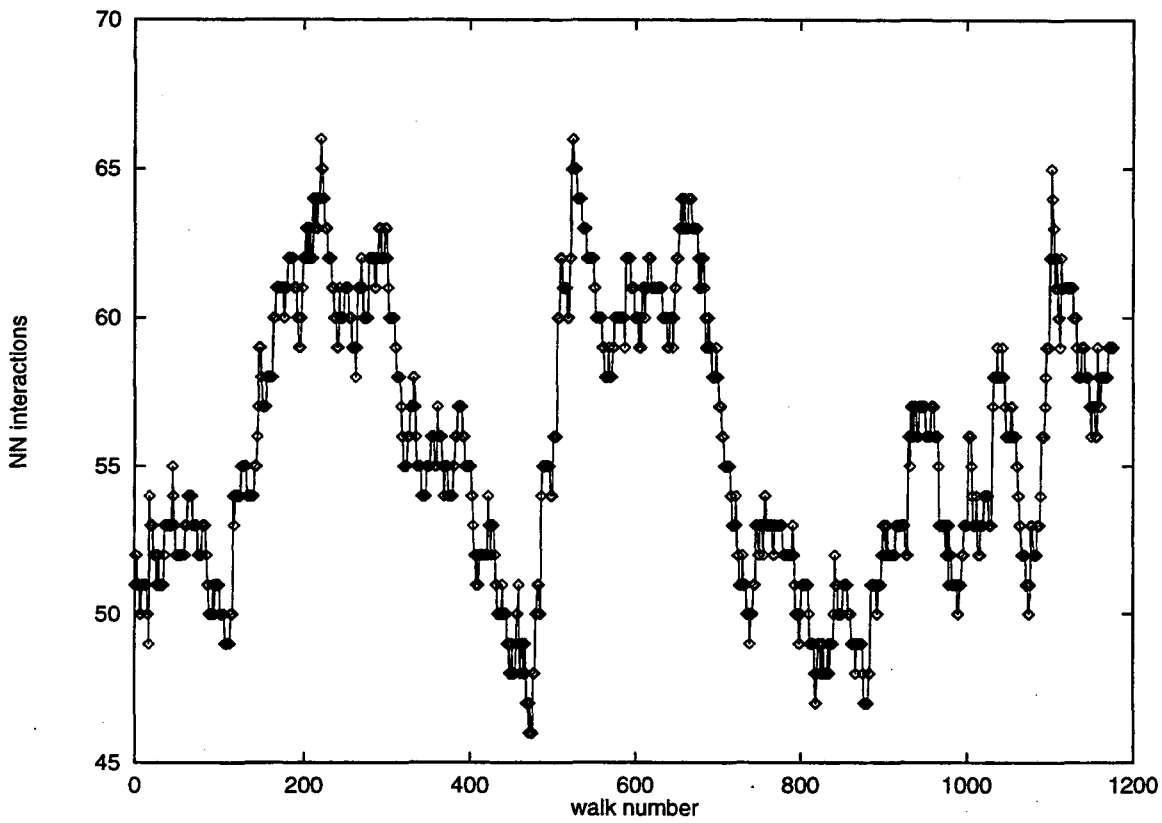


Figure 6.7: Number of nearest-neighbor interactions for a sequence of walks with $\beta = 0$ and $N = 1000$. In this case, there is no Metropolis rejection and all walks have equal probability.

One way of conveying at least heuristically the trapping that we suspect is occuring is to run the algorithm for $\beta = 0$ and produce a series of walks. Since every walk is equally probable, each of the walks in a given series tends to have a different number of nearest-neighbor interactions; this is illustrated in Figure 6.7. We then set $\beta = 1$ and calculate the probability that the algorithm would actually produce this sequence of walks; the result is shown in Figure 6.9. Again, we should stress that this is not

a proof, but it does give us some insight into what effect an increase in β has on our algorithm's sampling of phase space.

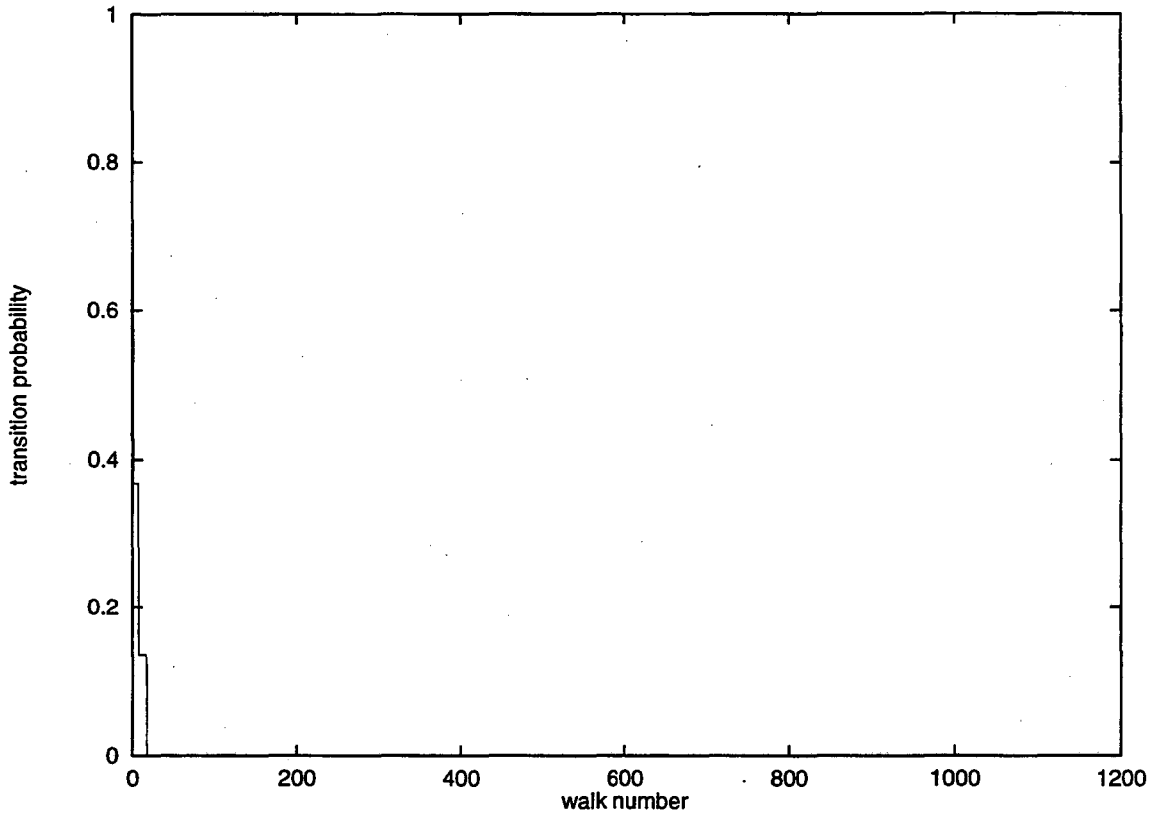


Figure 6.8: Transition probability vs. walk number for the same sequence of walks as in Figure 6.7 but with $\beta = 1.0$. Transition probability means in this case the probability that the Metropolis rejection step would allow us to go from walk 0 to walk i through each of the intermediate walks 1, 2, ..., $i - 1$.

6.3 Weighting the Runs

The behavior illustrated in Figure 6.5 occurred for other values of N and other values of β near β_{crit}; convergence towards significantly different values of the mean end-to-end distance is apparently not atypical, so doing one long run is not an effective

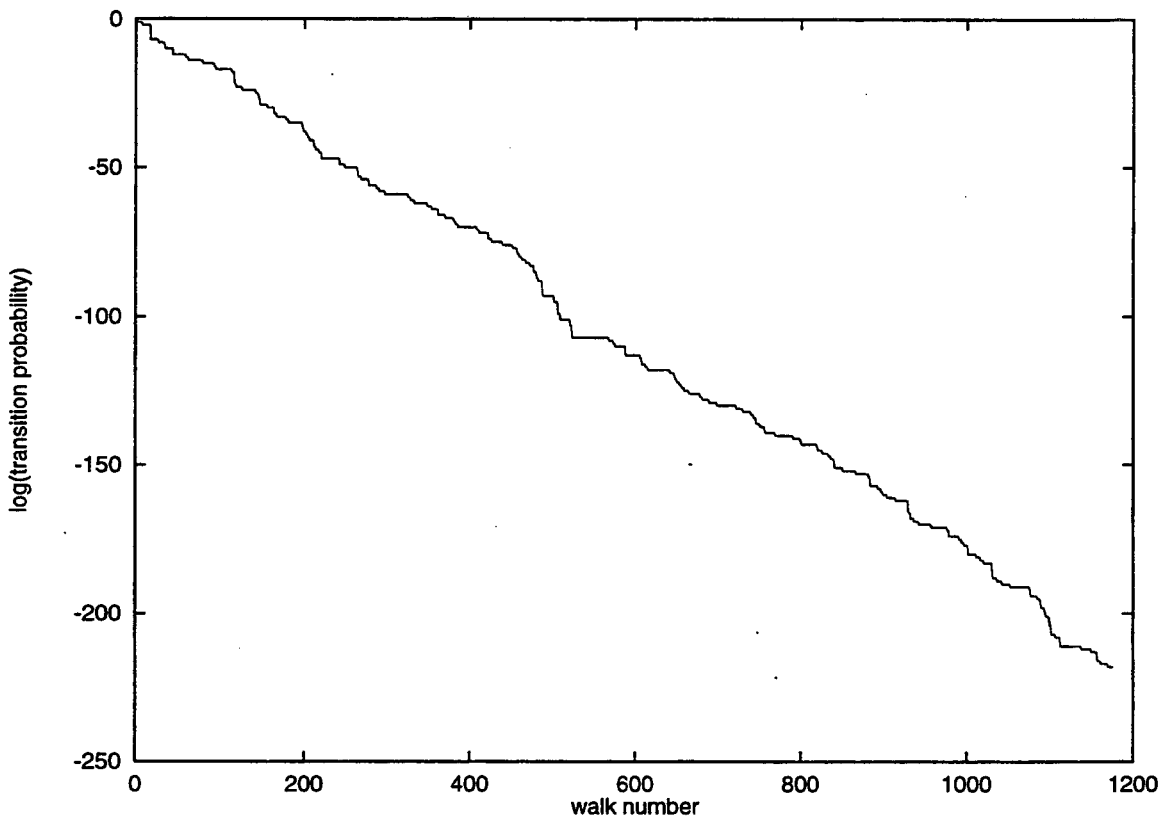


Figure 6.9: Log(transition probability) vs. walk number for the same sequence of walks as in Figure 6.7 but with $\beta = 1.0$. Transition probability means in this case the probability that the Metropolis rejection step would allow us to go from walk 0 to walk i through each of the intermediate walks 1, 2, ..., $i - 1$.

strategy. Given this, how can we compute the mean with any degree of accuracy? If the phase space breaks up into valleys separated by large energy barriers, then one approach might be to explore a number of different valleys and use the resulting thermal averages from each valley to calculate an overall thermal average.

In other words, we want to do a number of different Monte Carlo runs and use the means from all of the runs R_{ee}^i to get a final estimate R_{ee} of the mean. However, we have not said how we should weight each individual mean when calculating R_{ee}. There is no reason to suppose that the valleys contain the same number of walks or the same distribution of end-to-end distances, so it is not clear that equally weighting all of the R_{ee}^i is justified. The problem now is how to decide on a method for weighting the runs.

What is really needed here is a way to estimate how much of phase space is explored by each of the runs; this will allow us to estimate the weights that we should give to the energy valley corresponding to each run. If we go back to Section 5.4, we see that the entropy method allows us to quantify how much of phase space is explored: we can use this algorithm to estimate the entropy of each run; the entropy estimate (along with the average energy) can then be used to get an approximate free energy F_i which we substitute in to Equation 5.23.

We estimate the entropy for a given segment of the walk and for a given Monte Carlo run in the following manner: let m be the number of steps in the segment. For the first step, let b_{m-1} be 0 if the step is horizontal, 1 if it is in the positive y direction,

and 2 otherwise. For the remaining $m-1$ steps, let b_{m-i} be 0 if the step is to the left and 1 if if it is to the right. We thus get a sequence $\{b_i\}$ which uniquely identifies each possible m-step segment. We then form the sum $S_m = \sum_{i=0}^{m-1} 2^i b_i$ which also uniquely identifies each m-step segment; S_m is then stored for each of the walks in our Markov chain. At the end of the run we sort this list, find all of the numbers S_m which appear in the list, and count the number of times that each number appears. This gives us the frequency (i.e. the estimate of the probability) for each configuration, from which we can calculate the entropy estimate using Equation 5.18.

We attempted to do this for a number of values of β and N, but the results, some of which are illustrated in Figure 6.10, were disappointing at best. Initially we tried fairly small portions of the walk ($2 \leq m \leq 5$), but we found that the resulting entropies were extremely close to the "equal probability" values for all of the runs: for example, there are $3 * 2 * 2 = 12$ walks consisting of three steps, and our entropy values for the three-step segments of the walk were extremely close to $\ln 12$, which is the entropy of the ensemble of equal probability three-step walks. So, we next tried to increase m to between 30 and 50 and then used different segments along the walk to see if the entropy estimation would be the same from segment to segment (a consistency check for the entropy calculation).

However, even for $\beta = 0$ we found that the entropy in the center of the walk was significantly smaller than at the ends. The reason for this is related to one of the improvements in efficiency that we used when implementing the pivoting algorithm:

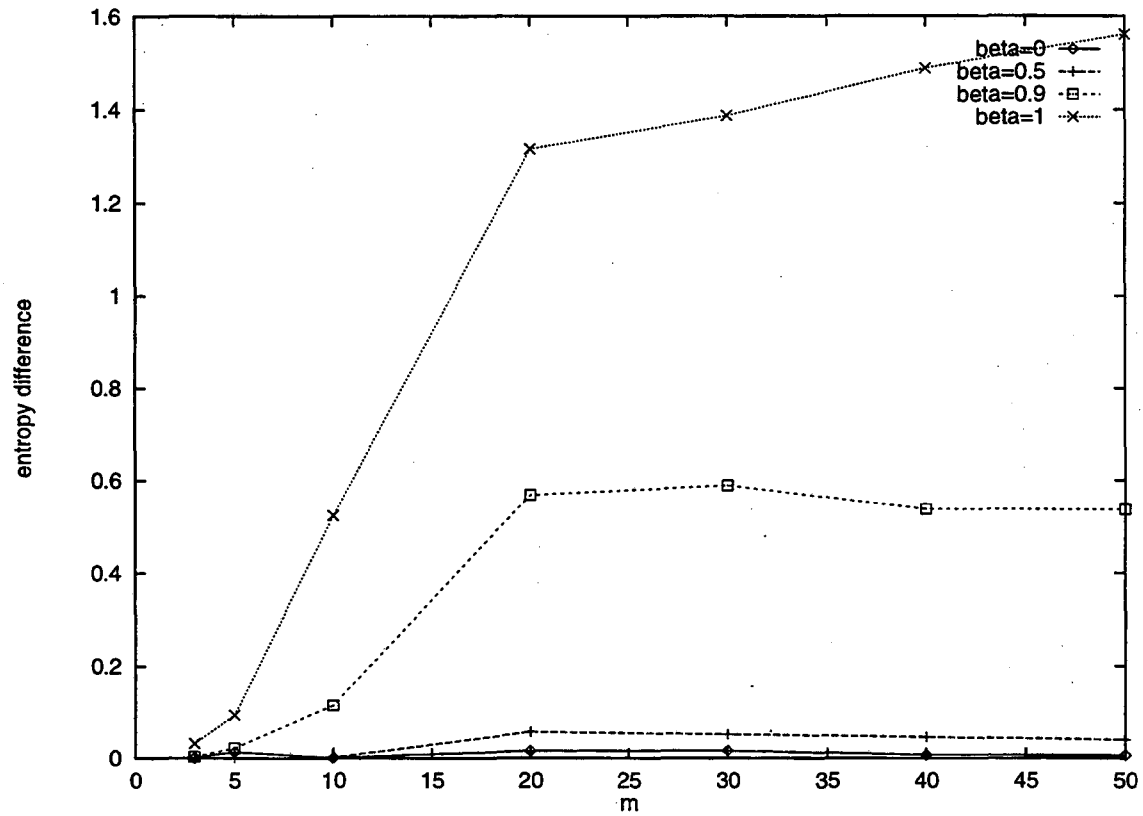


Figure 6.10: Entropy calculation results for various values of β and m with $N = 1000$; we did 6×10^5 inital iterations, and then calculated the entropy for 2×10^6 iterations of the algorithm. In these cases, the walk segments were $1,\ldots,m$ and $N-m+1,\ldots,N$. Note that all of the estimates start out very close to the "equal probability" value $\ln 12 \approx 2.4849$ of the entropy for $m = 3$. We decided to plot the difference of the two estimates instead of the estimates themselves since for some values of β the estimates are so close that the curves for the two segments overlap.

namely, when a given site is selected as the pivot point, we only pivot the smaller part of the walk. Thus, the central part tends to remain in the same configuration for a longer period of time than the ends, so the entropy calculation using the central segment does not accurately reflect how much of phase space is explored by the entire walk and leads to an underestimation of the entropy of the walk.

Because of this, we decided to just use segments of length m at the two ends to estimate the entropy. For values of β near 0, the estimates from the different ends were very close; however, there was no need for entropy estimates for these β values since the there was no significant discrepancy between the averages calculated from various Monte Carlo runs. The values of β for which we needed the entropy estimate were precisely those values for which the entropies calculated at the different ends did not agree (see Figure 6.10).

So, we resorted to the heuristic method of averaging the two entropies to come up with a single estimate; the motivation behind this is that we want the subsection of the walk that we use in estimating the entropy to accurately reflect the entire walk's wandering through phase space. A careful examination of Section 5.4 shows that the arguments there are valid both when we are using consecutive steps of the walk for our subsection and when the "subsection" consists non-adjacent steps of the walk. The only modification necessary is in our function mapping the "subsection" to an integer: if we take non-adjacent steps, in general we would need to do an expansion in base three (instead of base two) since non-adjacent steps could go in any of three

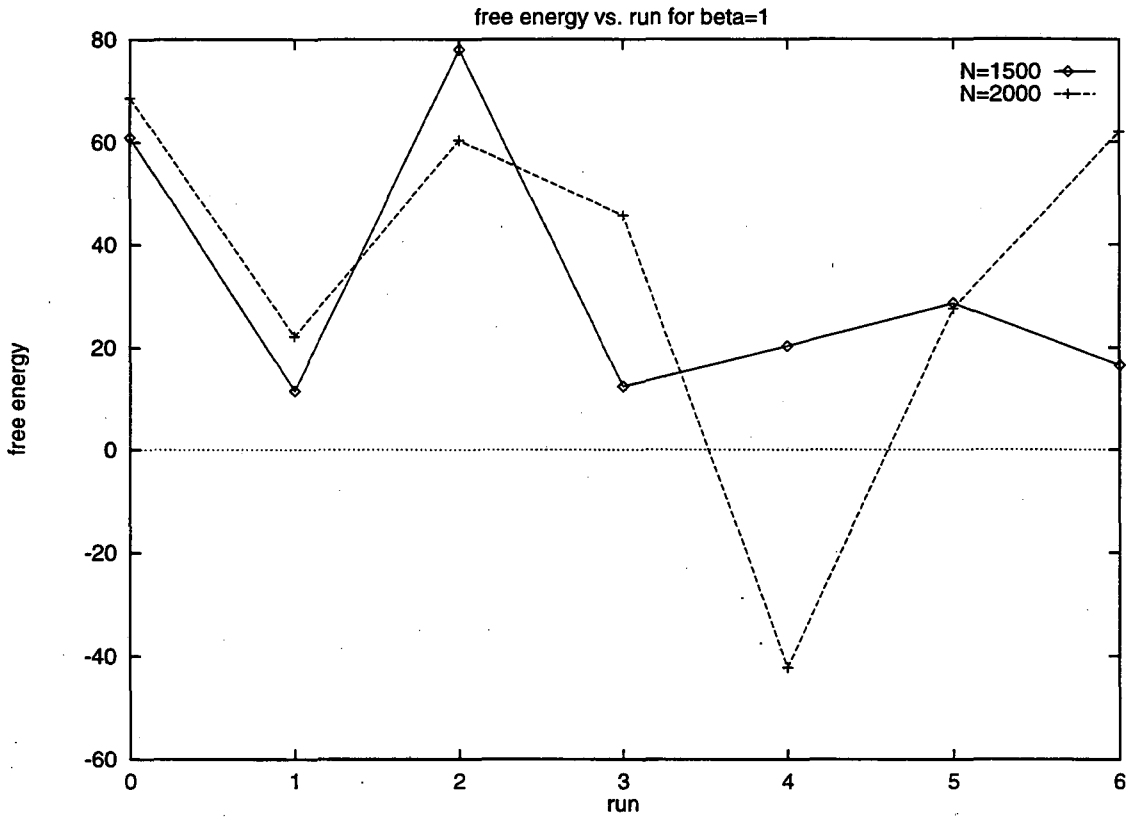


Figure 6.11: Free energy vs. run for $\beta = 1$, $N = 1500, 2000$.

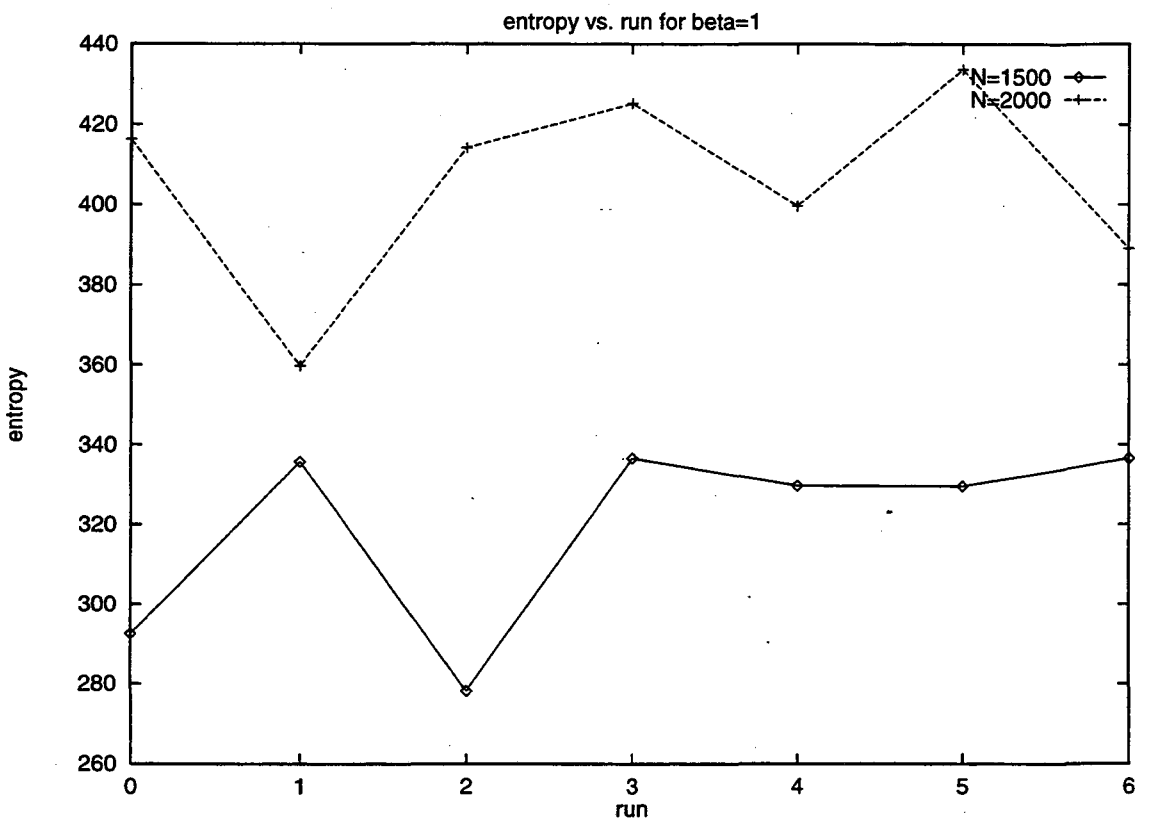


Figure 6.12: Entropy vs. run for $\beta = 1$, $N = 1500, 2000$.

directions, whereas adjacent ones are constrained to two because of self-avoidance. Anyway, the point is that by putting the estimates from the two extremes of the walk together, we might hope to get a more global picture of the walk and so a better estimate of the entropy.

After calculating the average entropy, we then tried to use these values to find an average over all of the separate Monte Carlo runs. Once again, the procedure was not effective, and the reason why can be seen if we look at Figure 6.11 and Figure 6.12 which show, respectively, the free energy and entropy for a series of runs with β near β_{crit}. What we would expect is that the energy and entropy would be approximately equal for $\beta \approx \beta_{crit}$, since this is where the balancing out between the two takes place (see Section 4.3 for a further discussion of this point). From the graphs, it is clear that for some runs this is indeed (at least approximately) the case; for others, this is not true. If we look at Equation 5.23, what we might hope is that the free energy for these "bad" runs will be small enough that their contributions to the average will be negligible. However, even if this is true (and it is for this set of runs), the problem is that the same exponential factor that gets rid of these "bad" runs also gets rid of every other run except the one with the largest free energy. The reason for this is that even though the various good runs have free energies which are within a few percent of each other, the actual differences themselves can be of the order of $5 - 10$; when this difference is exponentiated, runs whose free energies are relatively close may have weights whose ratios can be between e^5 and e^{10}. The same phenomenon occurs if we

use only the entropies (instead of the free energies) to weight the walks, although the "chosen" run may be different for the two weightings. All of this is illustrated in Figure 6.13, where we show the end-to-end distances for the runs as well as the average end-to-end distance computed using the two different weightings.

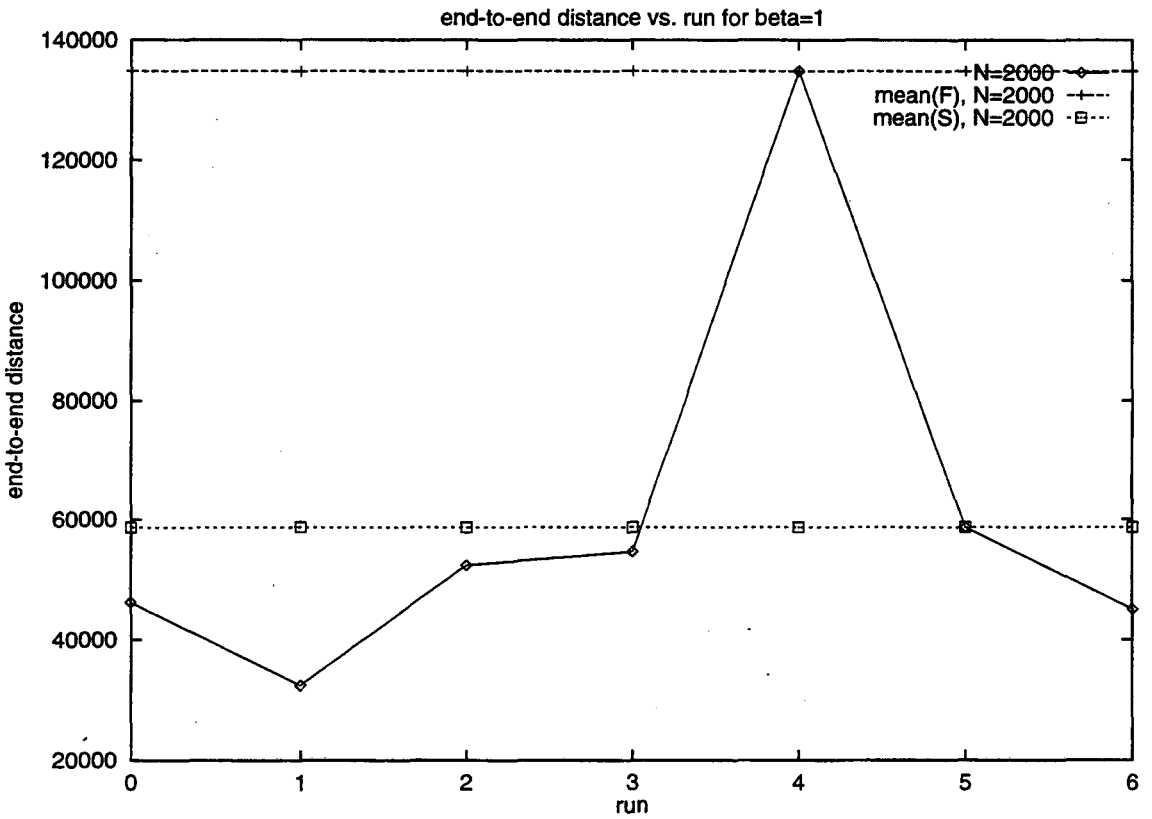


Figure 6.13: End-to-end distance vs. run for $\beta = 1$, $N = 2000$. mean(F) is the mean calculated using Equation 5.23. mean(S) is the mean calculated assuming the average energies E_i of each run are roughly the same.

We decided next to temporarily abandon the entropy method and to instead try to look for other ways to weight the averages from the various runs. The first alternative was simply lumping all of the histograms from the various runs together to try to get a more accurate average histogram. The resulting histograms for β near β_{crit} and for

various N are shown in Figure 6.14; as can be seen from the figure, the quality of the composite histograms is unacceptable for even moderate values of N, although for a given N, each of the component histograms may look quite smooth when viewed by itself. For example, the composite histogram for $N = 2000$ appears to be composed of at least four sub-histograms, each of which is fairly smooth; however, since they are centered around different energies, when put together they do not form a smooth composite histogram.

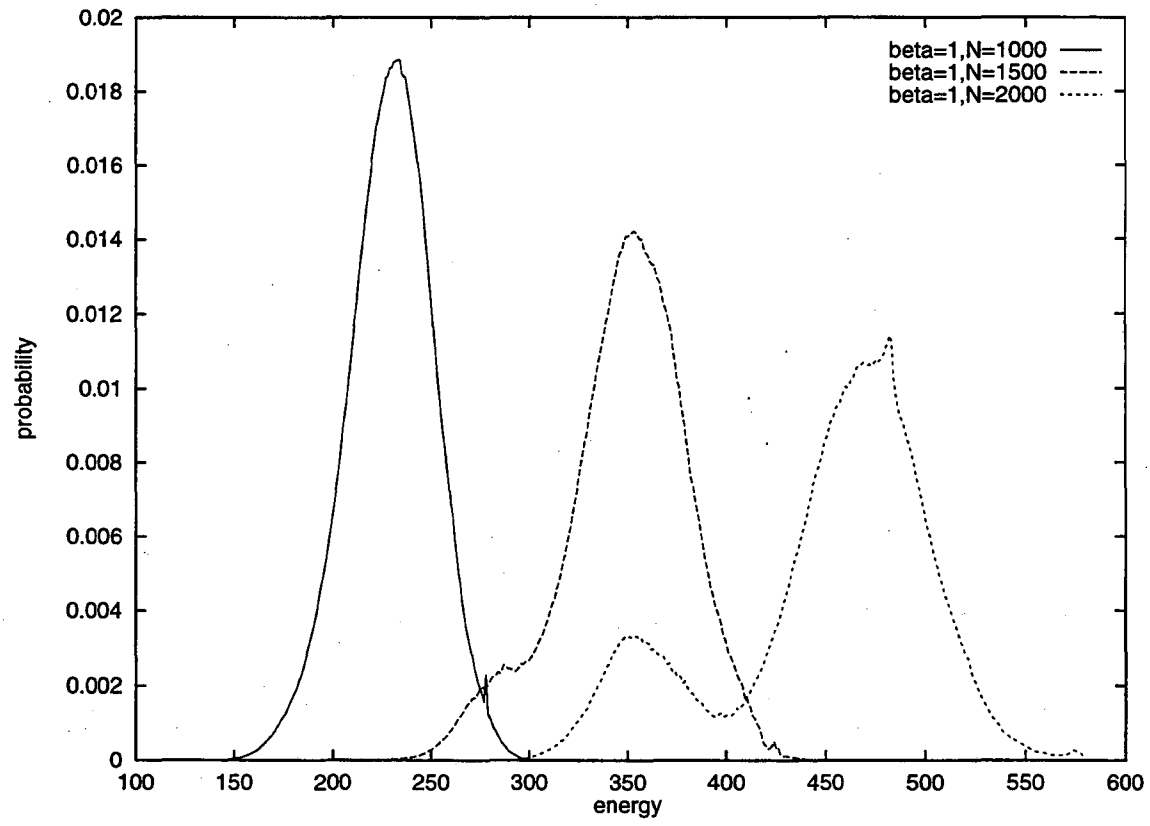


Figure 6.14: Probability of walks vs. energy (as measured by number of nearest neighbor interactions) for $\beta = 1$ and $N = 1000, 1500, 2000$.

When using the composite histogram method to compute thermal averages, what

we are doing is weighting each of the runs according to the walks accepted after the self-avoidance check; what we are not doing is taking into account which runs give us more new walks. In other words, we are only considering the length of each of the separate Markov chains, and not the number of individual walks which each one contains; the reader should recall that a given Markov chain will contain a sequence of identical walks unless and until the Metropolis step accepts a new walk based on energy considerations. The problem here is that the length of a given Markov chain does not tell us how much this chain has explored phase space. So, instead of considering the length, why do not we consider the number of walks r_a accepted after the Metropolis rejection step? The reader may be saying at this point: what happens if we just keep bouncing back between two (or three or a small number) of walks?

For $N > 100$ the phase space itself is huge; one might suspect that the valleys in which each of the runs get trapped are also huge. For any reasonable number of iterations (say $\leq 10^9$), huge means that a walk trapped in a subspace only explores a minuscule portion of the subspace. To examine more closely the "hugeness" of the subspaces, and to see if we might be bouncing back and forth between a few walks, we did some test runs for $N = 1000$ and $N = 2000$ with $\beta = 1$; for each walk generated by the algorithm, we recorded the distance between steps i_k and j_k of the walk for $k = 1 \ldots 9$ and $|i_k - j_k|$ varying as $\frac{kN}{9}$. We then used these pairs to label the walks (we didn't have enough disk space to store all of the walks themselves): by examining this list, we found a lower bound for the number of different walks in our

chain. For all runs, the average of the ratio of this bound to the number of walks ra accepted by the algorithm is 0.95 (see Figure 6.15). In other words, there is very little revisiting of any point in phase space by the algorithm. This does not mean that a given walk does not appear more than one time in our Markov chain (this is not true): what it means is that although we might get stuck at a particular walk for a number of consecutive iterations of the algorithm, once the algorithm accepts another walk, usually we won't go back to the first walk later in the algorithm.

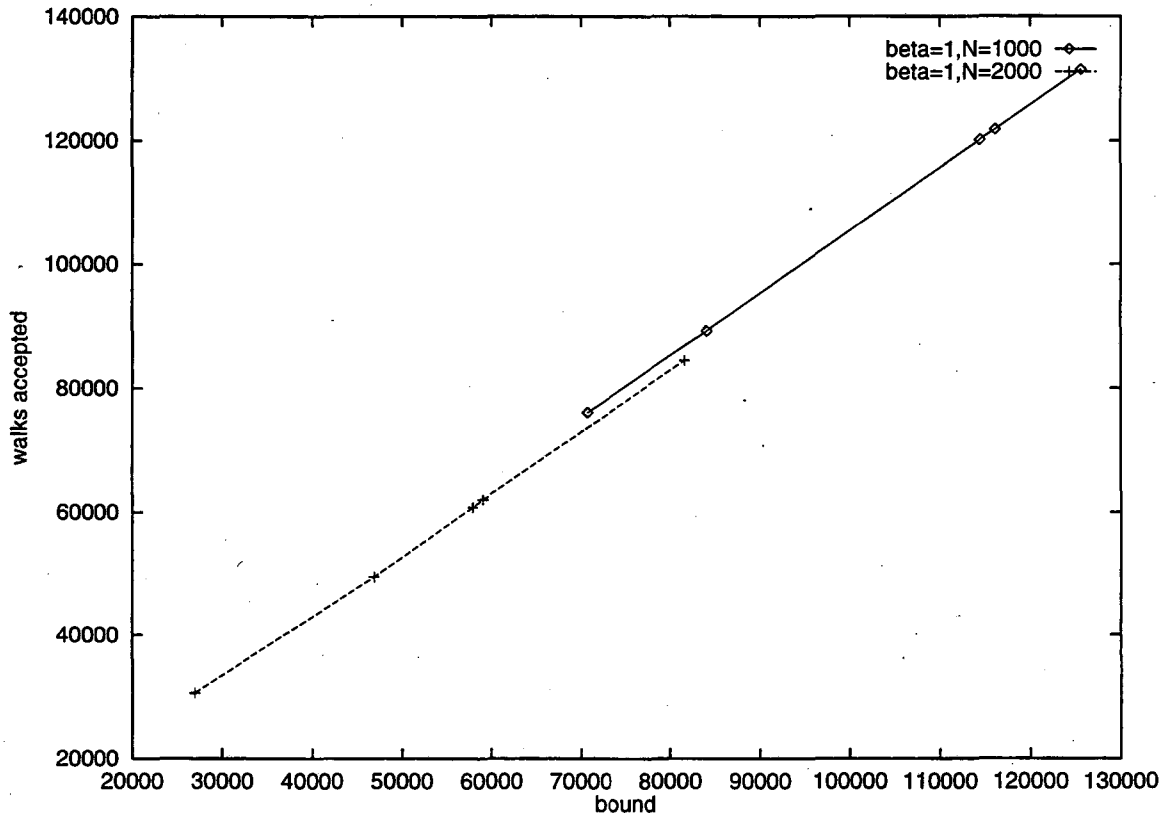


Figure 6.15: Number of walks accepted after self-avoidance check and Metropolis step vs. bound for number of individual walks in Markov chain for $\beta = 1$ and $N = 1000, 2000$.

Thus, for a given run i, r_a^i is very close to the number of distinct walks generated

by our algorithm. One might suspect that the number of distinct walks would also be directly related to the entropy of a particular run. Since the entropy estimates themselves do not provide an accurate method of weighting the runs, we might try to somehow weight run i according to r_a^i. In fact, there is a correlation between r_a and the entropy that we computed above. This is illustrated in Figure 6.16, and this is why we temporarily abandoned the entropy method. Although the fact that we take exponentials of the entropies in order to calculate thermal averages makes these averages inaccurate, the magnitude of the entropies themselves gives us a clue as to which runs we might want to give less weight to and which ones we might want to give larger weights to: since the r_a^i reflect the entropies, instead of using the entropy, why not instead use the r_a^i (which are much easier to compute) to do the weighting?

We illustrate the results of this weighting in the following figures. First, Figure 6.17 and Figure 6.18 show the results of our calculations when we use no weighting at all. These are somewhat disappointing, especially given the large number of iterations that we did for each run. For $N > 300$, the curves give inaccurate values of ν near β_{crit}, which is precisely where we need the most accuracy; however, even for smaller N things do not look good near β_{crit}.

If we examine Figure 6.19 and Figure 6.20, we see that using the r_a^i gives good results up to much larger values of N. To be sure, it is clear from Figure 6.20 that even this weighting gives inaccurate values of ν for $N > 1000$. What about using the histograms themselves instead of weighting the averages directly? Putting them

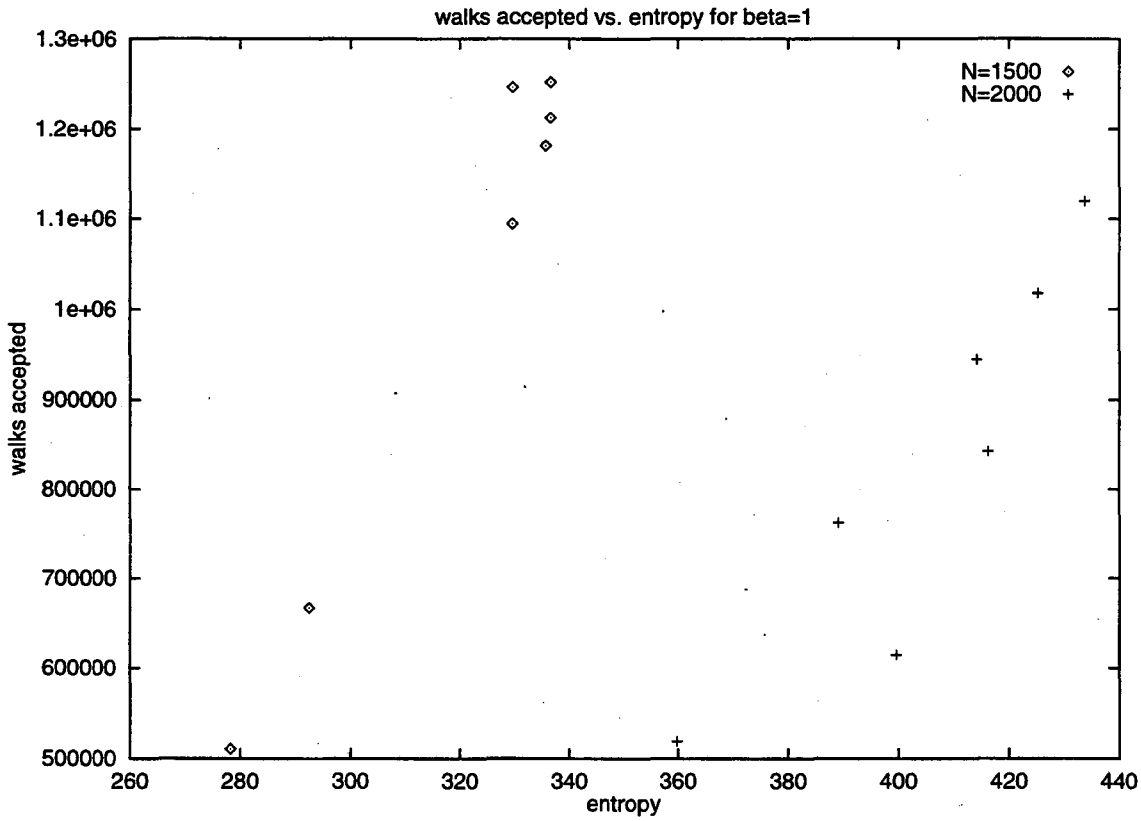


Figure 6.16: Number of walks accepted after self-avoidance check and Metropolis step vs. entropy for the runs in Figure 6.11 and Figure 6.12.

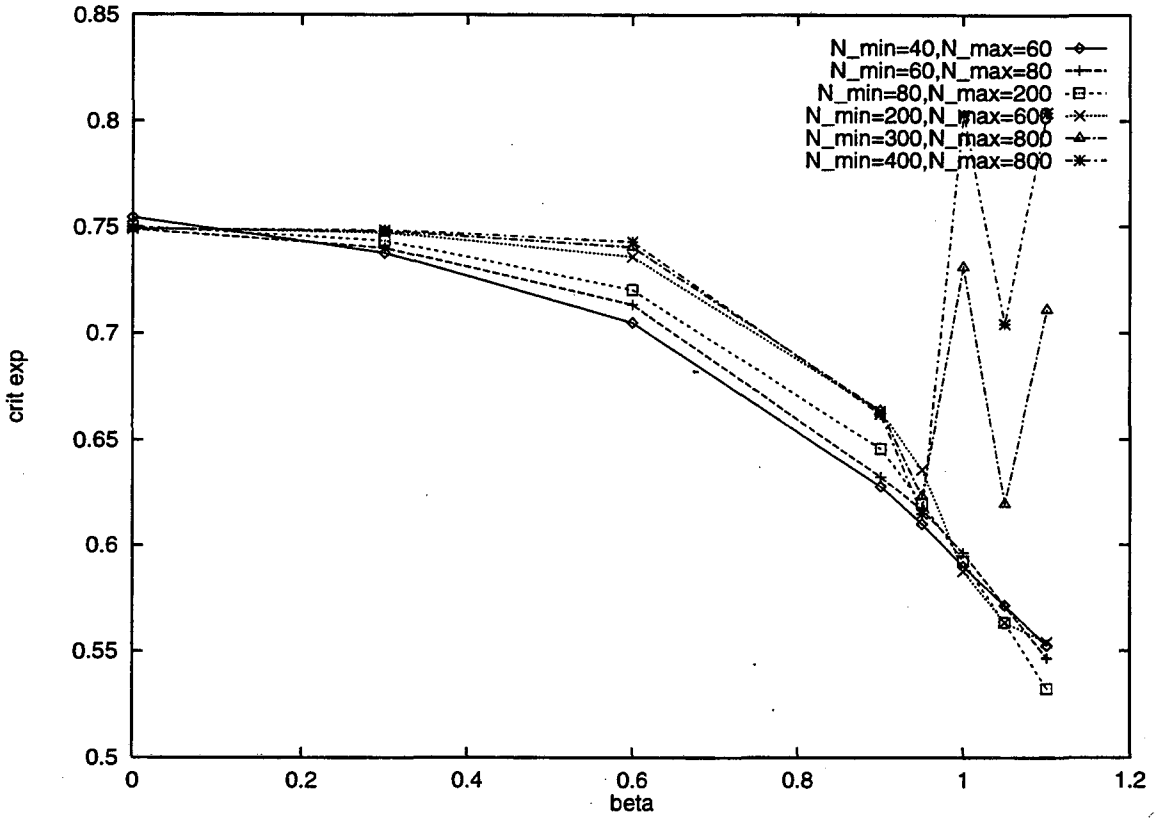


Figure 6.17: Critical exponent ν vs. β for various values of N. Each pair (β, N) had between 5 and 20 runs; for each run i, we did between 400000 and 800000 inital iterations and then computed the average end-to-end distance $R_{ee}(i, N, \beta)$ for between 3×10^7 and 6×10^7 iterations. In this figure, we combined the $R_{ee}(i, N, \beta)$ without any weighting to get $R_{ee}(N, \beta)$. At every value of β, we then fit a line to the plot of $\log R_{ee}(N, \beta)$ vs. $\log N$ using the set of values of N listed in the graph.

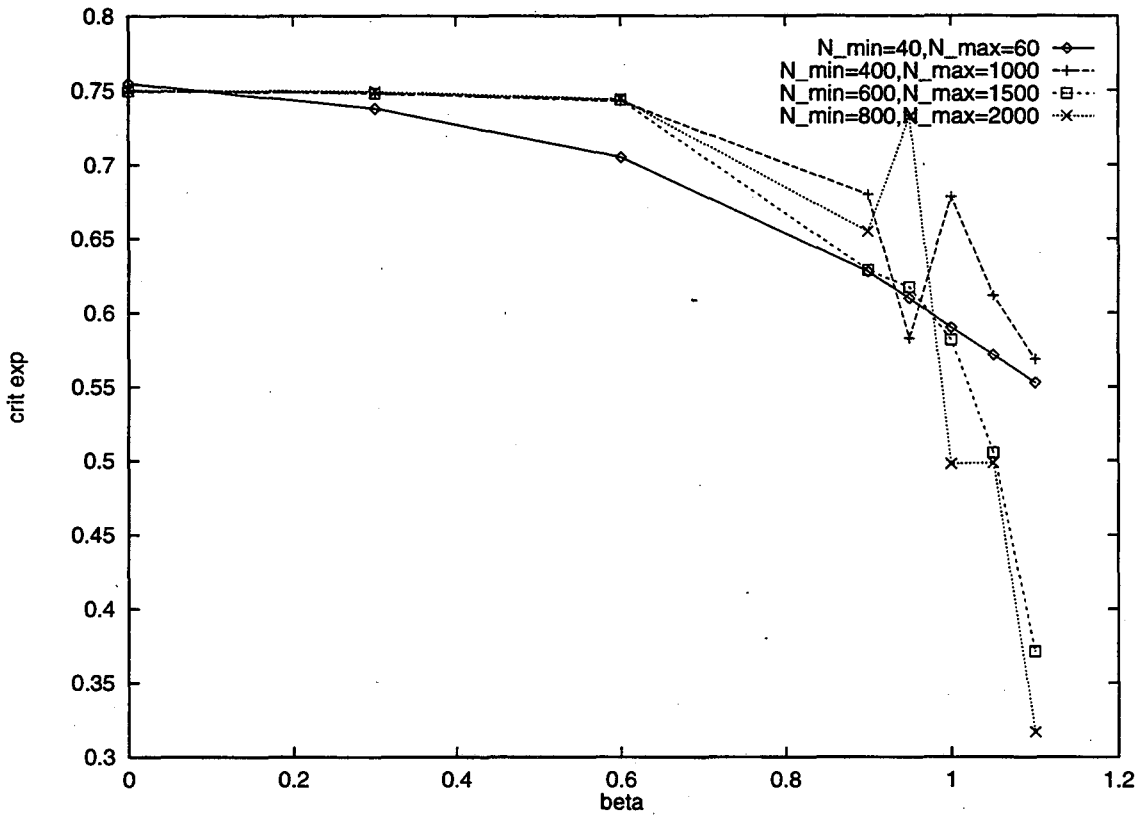


Figure 6.18: Same as Figure 6.17 with different values of N.

together according to the procedure described in Figure 6.19 is identical to putting together the averages; however, if we add the weighting described in Figure 6.21, the results of the two methods are not in general the same. If $r_{ee}(i,j)$ are the end-to-end distances for runs $i = 1, \ldots, M$, with each run consisting of M_i walks, then averaging these averages using weights r_e^i gives

$$< R_{ee} > \approx \frac{\sum_{i=1}^{M} ra_i \frac{\sum_{j=1}^{M_i} r_{ee}(i,j)}{M_i}}{\sum_{i=1}^{M} ra_i} \tag{6.1}$$

However, if we weight the individual histograms and then put them together to form an average, we are actually computing

$$< R_{ee} > \approx \frac{\sum_{i=1}^{M} ra_i \sum_{j=1}^{M_i} r_{ee}(i,j)}{\sum_{i=1}^{M} M_i ra_i} \tag{6.2}$$

The result of this second method is given in Figure 6.22; as can be seen by comparing with Figure 6.21, for our data the two methods give almost identical results.

6.4 Estimating β_{crit} and ν

Even though the curves for large N are not particularly useful, we can still use the curves for small N to try to estimate β_{crit}. To do this, however, we need more sample points near where the curves for different N seem to intersect. We could do many more runs with values of β near the intersection points of the curves in Figure 6.19, but this would be extremely expensive from a computational point of view. Instead, we decided to use the histogram method (see Section 5.3) to fill in the curves near

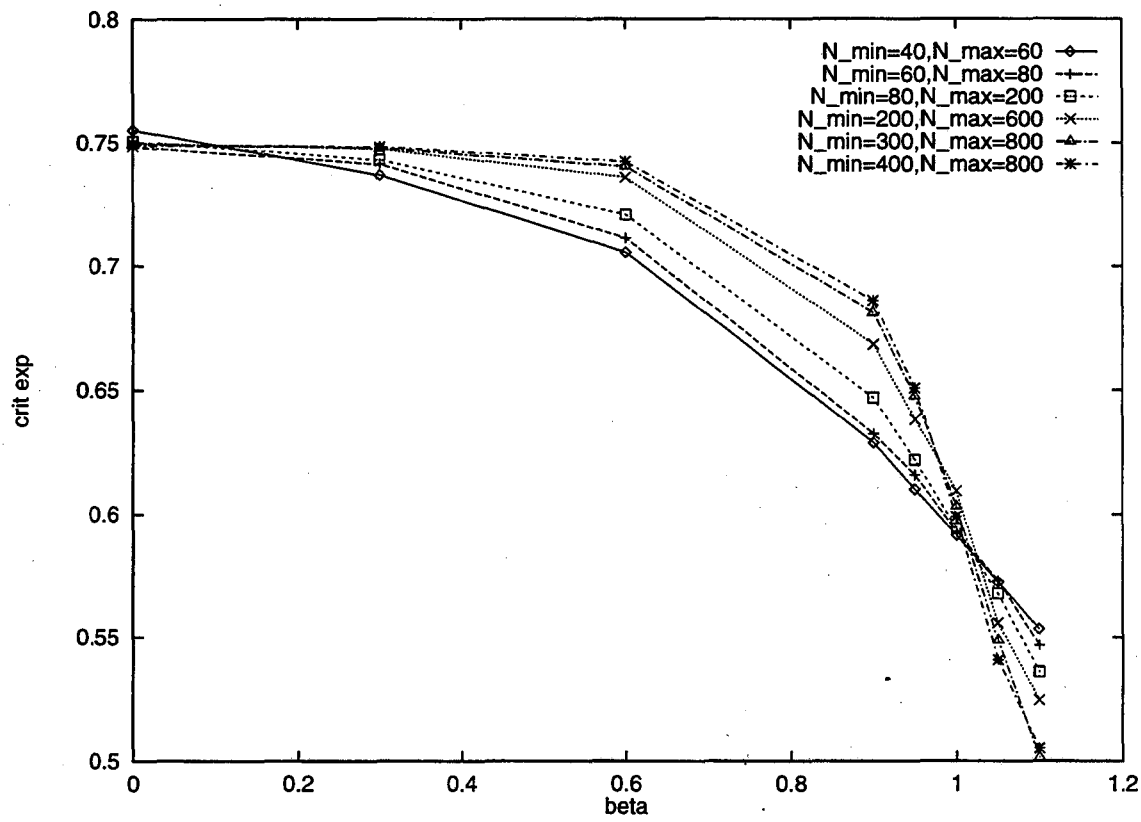


Figure 6.19: Same as Figure 6.17, except that when we combined the $R_{ee}(i, N, \beta)$ to get $R_{ee}(N, \beta)$, we first found the maximum ra_{max} of all of the ra_i for each pair (N, β) and eliminated the runs whose ra_i was less than $0.8ra_{max}$; we then used the remaining runs to calculate the average. The reasoning behind this is that if we look back at Figure 6.16, we see that since r_a^i reflects the entropy S_i, then those runs with small r_a^i and correspondingly small S_i will make a negligable contribution to the overall average. What about weighting the remaining runs after eliminating those with small r_a^i? Well, in this Figure we give them all equal weight, but in Figure 6.21 we weight them using the r_a^i (recall that the entropy method in effect just selects out one of the runs and gives it a weight disproportionately large compared to those of the other runs.

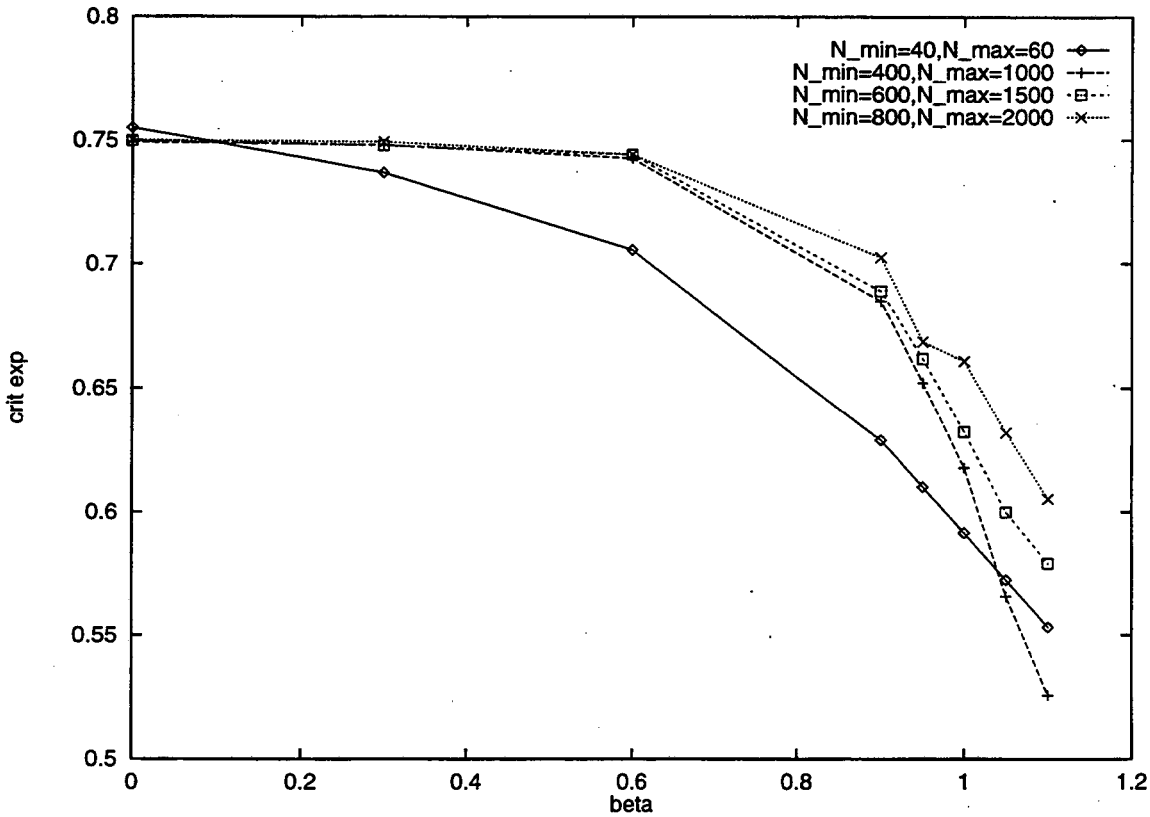


Figure 6.20: Same as Figure 6.19 with different values of N.

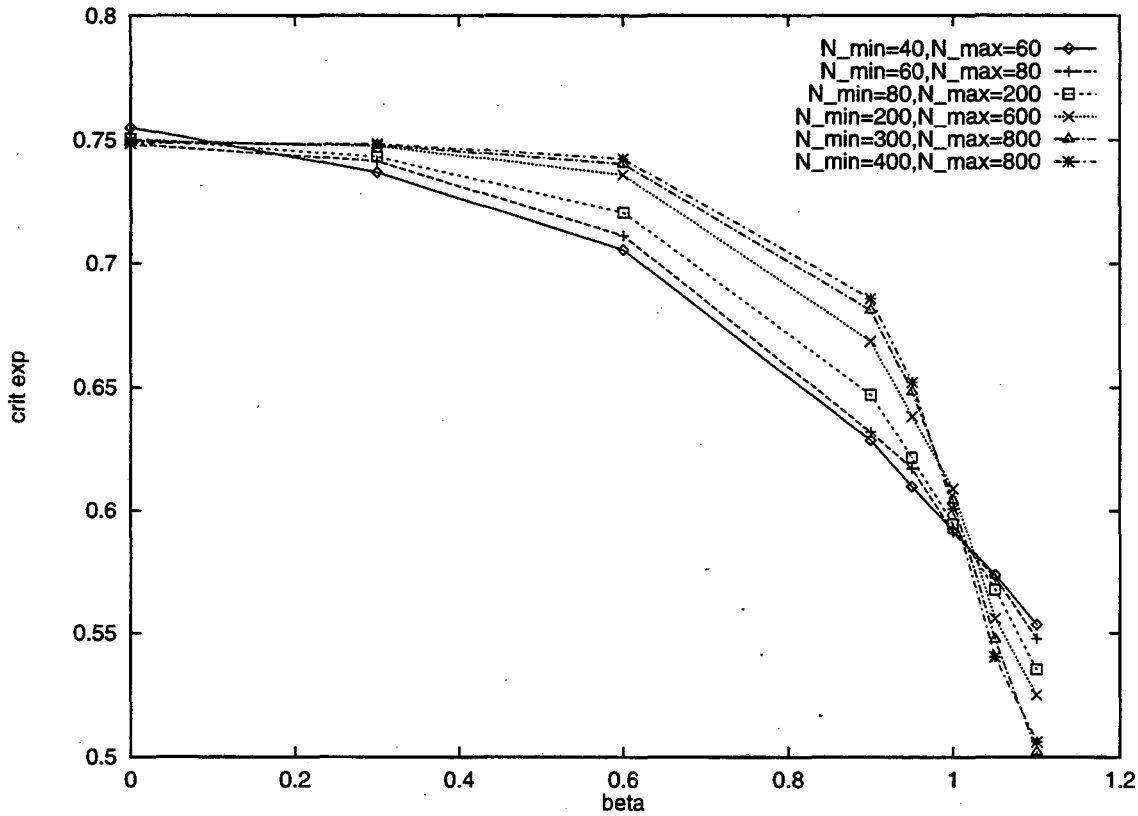


Figure 6.21: Same as Figure 6.19, except that when we combined the $R_{ee}(i, N, \beta)$ to get $R_{ee}(N, \beta)$, we weighted each according to the number of walks r_a^i accepted after the Metropolis step. This corresponds to Equation 6.1.

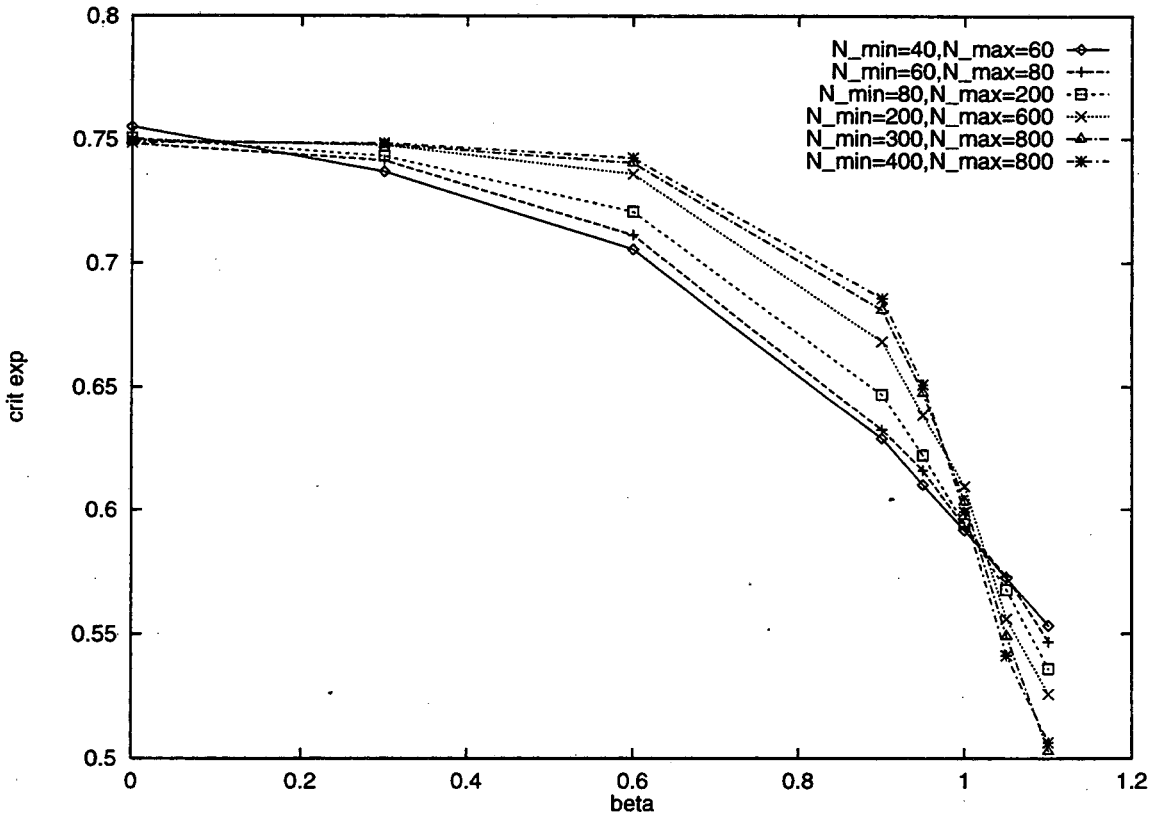


Figure 6.22: Same as Figure 6.21, except that we used Equations 6.2 when doing the weighting.

the suspected value of β_{crit}. The results of this are shown in Figure 6.23. From this Figure, we can estimate β_{crit} to be between 1.01 and 1.02. Unfortunately, the lack of good data for larger N prevents us from accurately determining the dependance of the estimate of β_{crit} on N (see Section 4.3).

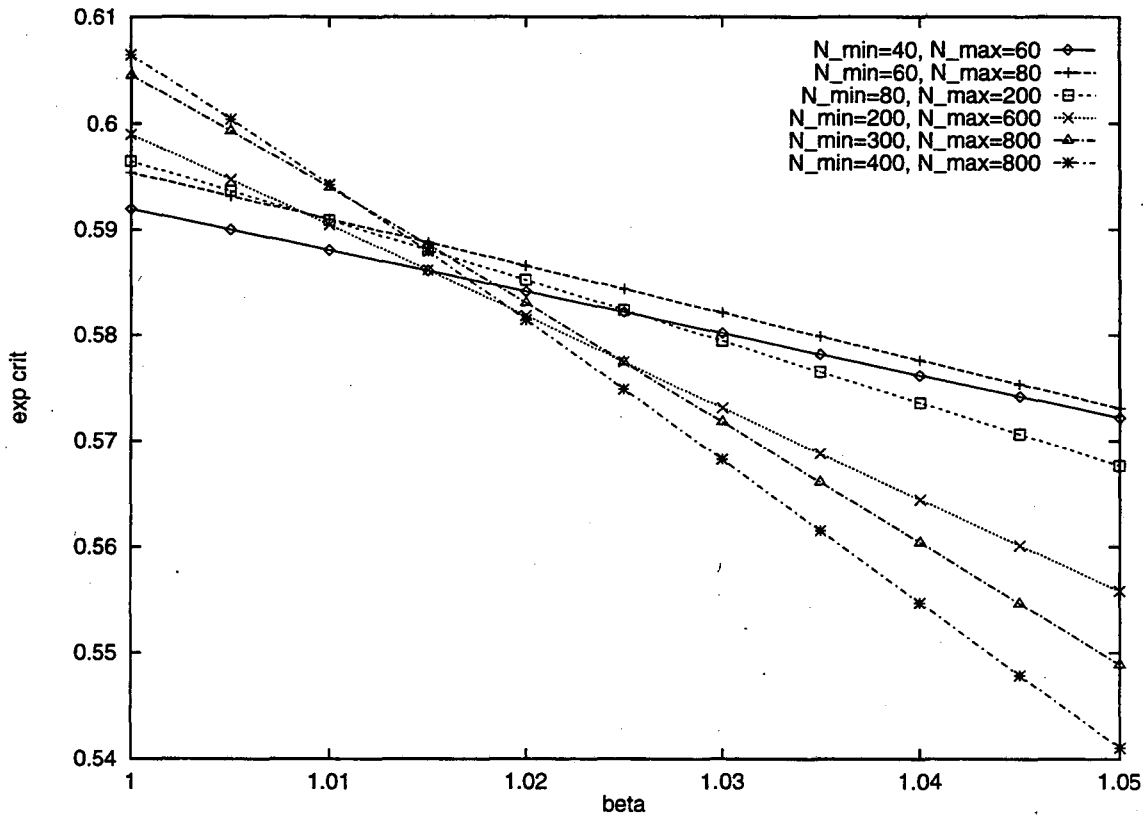


Figure 6.23: Same as Figure 6.19, except that we used the histogram method to fill in the curves near $\beta = 1$.

After having located the approximate value of β_{crit}, we wanted to get a more accurate estimate of the critical exponent ν at this temperature. To do this, we used a procedure from Poole, et al. [60] which goes as follows: if β is fixed, we expect $< R_{ee}^2 >$ to be proportional to $N^{2\nu}$. However, we do not know the constant of

proportionality, so we instead take this equation for two different values of N, say N and $N+M$, divide the two equations, take logs, and solve for ν. This gives us

$$\nu \approx \frac{1}{2}\frac{\log \frac{<R_{ee}(N+M)^2>}{<R_{ee}(N)^2>}}{\log \frac{N+M}{N}} \tag{6.3}$$

We then compute this ratio for various values of N; this gives us several estimates of ν for this particular β. The results of this calculation are shown in Figure 6.24; Figure 6.25 shows the results of the same calculation using the radius of gyration R_g in place of R_{ee}. For M fixed, the accuracy of this procedure decreases as N increases: this is because in Equation 6.3 we are taking the log of a number which is approaching 1, so any errors in the averages become more and more significant with increasing N. In fact, Poole, et al. [60] use $M = 1$, but we were unable to get any useful results for this M, since for some N we actually had $< R_{ee}(N+1) >$ slightly less than $< R_{ee}(N) >$; the corresponding estimate of ν was therefore less than zero.

As can be seen from Figure 6.24 and Figure 6.25, for the estimated value of $\beta_{crit} \approx 1.01$, $\nu \approx 0.59$, which is in good agreement with the previous estimates of $\nu \approx 0.58$ (see Poole, et al. [60] for a list of these and further references).

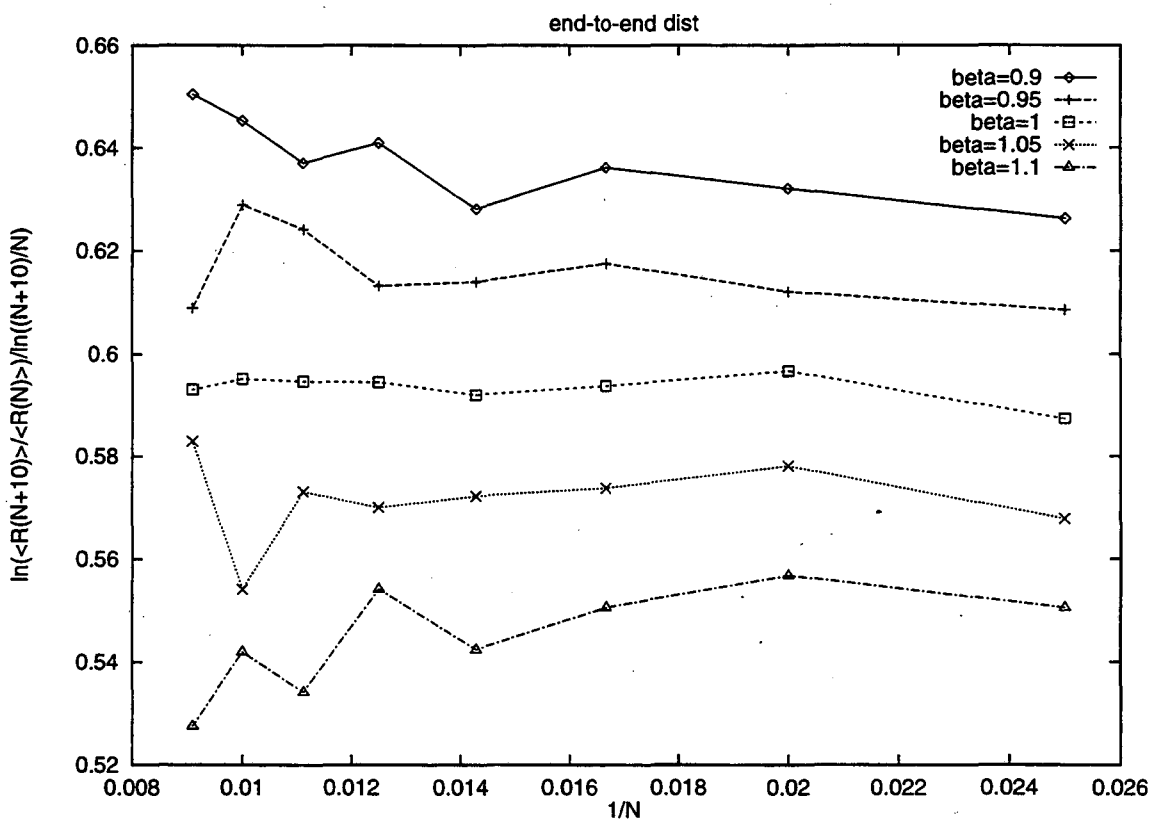


Figure 6.24: The exponent ν from Equation 6.3 as a function of $\frac{1}{N}$. We set $M = 10$ and tried this for a number of β in the vicinity of our estimate of β_{crit}.

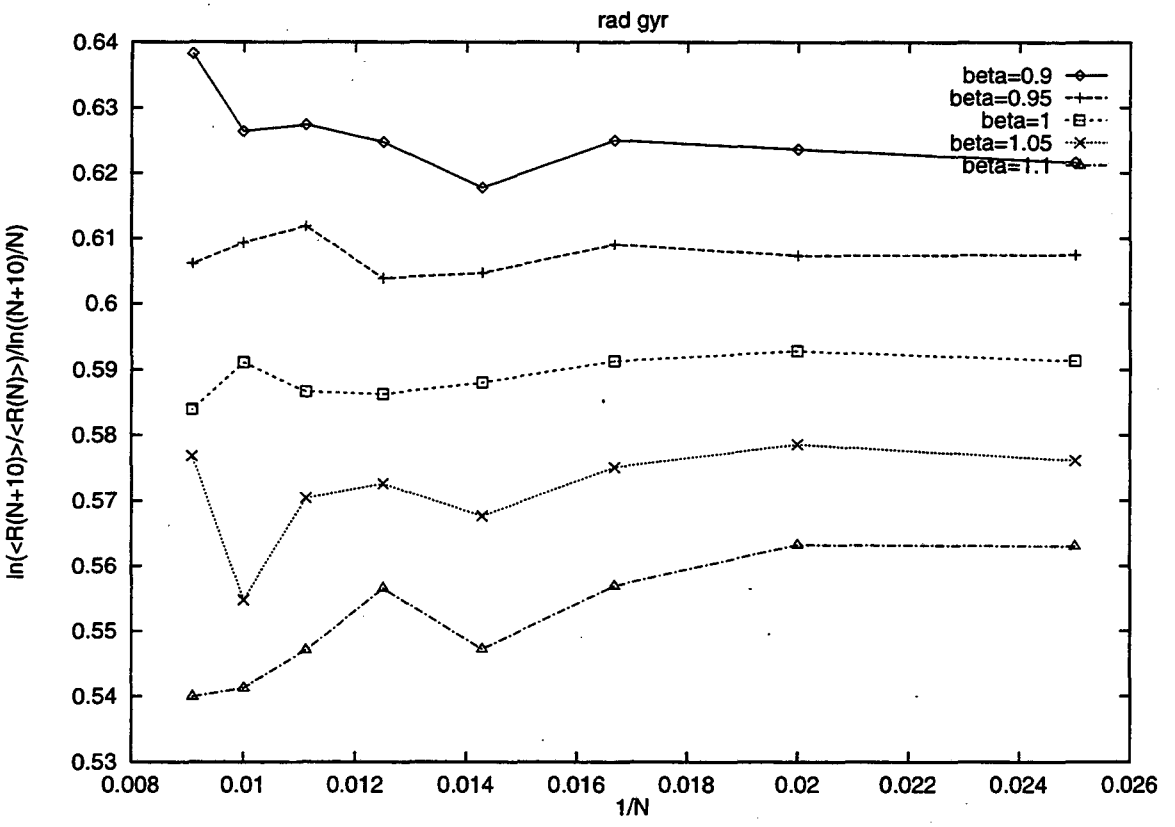


Figure 6.25: The exponent ν from Equation 6.3 as a function of $\frac{1}{N}$ using the radius of gyration R_g in place of R_{ee}. Again, $M = 10$.

Chapter 7

Conclusions

We originally set out in this thesis to put together two well-known Monte Carlo algorithms (pivoting and Metropolis rejection) in order to calculate thermal averages for a particular polymer system. The example we considered was at first glance fairly simple: self-avoiding walks on a hexagonal lattice in two dimensions with a nearest-neighbor interaction. Since the thermal averages had already been calculated for this system by other methods, our calculations served as a test for the applicability of the pivoting and Metropolis rejection algorithms to such a polymer system.

Our example contained some of the characteristics of more complicated problems (e.g. polymer folding) that are currently of interest in physics. However, even for our seemingly simple system, we ran into a number of difficulties such as unacceptably small acceptance percentages and trapping in valleys centered around local energy minima.

Both of these problems occurred for $N > 800$ (where N is the length of the SAW) and β near $\beta_{crit} = 0.99$. In fact, for any N we needed to concentrate our attention on values of β near the suspected value of β_{crit}. The histogram method was useful for this, since it allowed us to use the data from one run at a given β_0 to estimate the thermal averages for other values of β near β_0; this considerably reduced the number of calculations necessary near β_{crit}. However, the histogram method had its limitations: we could not use it to go too far away from the β_0 at which the original run was done.

The low acceptance percentage mentioned above was caused in part by the length of our walks: even when $\beta = 0$ the pivoting algorithm's acceptance fraction decays as $N^{0.19}$. However, the chief factor was the Metropolis rejection step: when β was near β_{crit}, Metropolis rejection tended to accept walks which had a large number of nearest-neighbor interactions. These walks were contracted, and so a large percentage of the subsequent iterations of the pivoting part of the algorithm led to self-intersecting walks and a correspondingly low acceptance percentage.

To increase the number of acceptances, we could not simply do more iterations. Because the algorithm became trapped in valleys in phase space, for different runs there was a convergence towards different thermal averages (which resulted in different values for the critical exponents). One way out of this difficulty was to do several runs, hoping that in this way we would be able to explore a larger portion of phase space. The problem then arose as to how to put the results from the various runs

together.

The entropy method was useful in helping to weight the thermal averages from various runs to produce an overall average. However, it too had its limitations: the entropy estimations gave us an indication of which runs might be more important, but for large enough N, we still could not get accurate values of the exponent ν. This was true even when we tried to put several runs together, each of which had a large number of iterations of the algorithm. In addition, the entropy estimates gave almost all of the weight to the run whose entropy was the highest, and gave very little weight to runs whose entropy was slightly less than this maximum value.

Nevertheless, the entropy method did prove useful. We discoverd a direct relationship between our entropy estimate for a given run and the number of walks r_a accepted after the Metropolis rejection step. The best results were obtained for relatively small N ($N \leq 800$) by doing several runs and using r_a to weight the averages from the runs. Using these data, our estimates for the critical temperature and critical exponent were $\beta_{crit} = 1.01 \pm 0.01$ and $\nu = 0.59 \pm 0.005$, which are in good agreement with the previous values $\beta_{crit} \approx 0.99$ (based on numerical work by Coniglio, et al. [14]) and $\nu \approx 0.58$ (from theoretical calculations, numerical work, and polymer experiments described in Section 6.4). However, the reason we were able to decide which values of N to use in these estimates of ν and $beta_{crit}$ was that we already knew from previous calculations by other researchers what the approximate values of ν and β_{crit} were. Thus, our algorithm was only partially successful, and if we had not known the

approximate values of ν and β_{crit} it would have been extremely difficult to decide for which values of N the Monte Carlo method was yielding accurate thermal averages.

What has been learned? Even with simple interactions and a simple and powerful algorithm such as pivoting with Metropolis rejection, it is not necessarily true that one can compute accurate thermal averages for polymers near T_c using Monte Carlo sampling. Researchers who use Monte Carlo in complicated polymer folding calculations should therefore interpret the results of such calculations with care. Simply increasing the number of samples in a Monte Carlo calculation does not always increase the accuracy of the estimates of thermal averages; phase space's structure together with the algorithm itself can lead to complex and unexpected behavior.

Bibliography

[1] J. H. Akao. *Phase Transitions and Connectivity in Three-Dimensional Vortex Equilibria.* Ph.D. thesis, Univ. of California at Berkeley, Berkeley, 1994.

[2] J. H. Akao. Percolation in a self-avoiding vortex gas model of the lambda transition in three dimensions. *Phys. Rev. E*, 53:6048–6055, 1996.

[3] C. Arag ao de Carvalho, S. Caracciolo, and J. Fröhlich. Polymers and $g|\phi|^4$ theory in four dimensions. *Nucl. Phys. B*, 215[FS7]:209, 1983.

[4] L. Arnold. *Stochastische Differentialgleichungen: Theorie und Anwendungen.* R. Oldenbourg Verlag, München, 1973.

[5] A. Berretti and A. D. Sokal. A new monte carlo method for the self-avoiding walk. *J. Stat. Phys.*, 40:483, 1985.

[6] K. Binder. Critical properites from monte carlo coarse graining and renormalization. *Phys. Rev. Lett.*, 47(9):693–696, 1981.

[7] K. Binder and D. W. Meerman. *Monte Carlo Simulation in Statistical Physics.* Springer-Verlag, New York, 1988.

[8] J. J. Binney, N. J. Dowrick, A. J. Fisher, and M. E. J. Newman. *The Theory of Critical Phenomena.* Oxford University Press, New York, 1992.

[9] C. C. Chang. *Numerical Solutions of Stochastic Differential Equations.* Ph.D. thesis, Univ. of California at Berkeley, Berkeley, 1985.

[10] A. J. Chorin. Equilibrium statistics of a vortex filament with applications. *Commun. Math. Phys.*, 30:619–631, 1991.

[11] A. J. Chorin. *Vorticity and Turbulence.* Springer-Verlag, New York, 1994.

[12] A. J. Chorin and J. H. Akao. Vortex equilibria in turbulence theory and quantum analogues. *Physica D*, 52:403–414, 1991.

[13] A. J. Chorin and J. E. Marsden. *A Mathematical Introduction to Fluid Mechanics.* Springer-Verlag, New York, 1993.

[14] A. Coniglio, N. Jan, I. Majid, and H. E. Stanley. Conformation of a polymer chain at the θ' point: Connection to the external perimeter of a percolation cluster. *Phys. Rev. B*, 35:3617, 1987.

[15] P. G. de Gennes. *Scaling Concepts in Polymer Physics.* Cornell University Press, Ithica, NY, 1979.

[16] P. G. de Gennes. *Introduction to Polymer Dynamics.* Cambridge University Press, New York, 1990.

[17] M. Doi and S. F. Edwards. *The Theory of Polymer Dynamics.* Oxford University Press, New York, 1989.

[18] J. L. Doob. *Stochastic Processes.* Wiley, New York, 1953.

[19] J. Douglas, C. M. Guttman, A. Mah, and T. Ishinabe. A spectrum of self-avoiding walk exponents. *Phys. Rev. E*, to appear.

[20] W. Ebeling. *Lattices and Codes.* Vieweg, Braunschweig/Wiesbaden, 1994.

[21] W. Feller. *An Introduction to Probability Theory and Its Applications*, volume 2. Wiley, New York, 1966.

[22] A. E. Ferdinand and M. E. Fisher. Bounded and inhomogeneous ising models. i. specific-heat anomaly of a finite lattice. *Phys. Rev.*, 185:832, 1969.

[23] A. M. Ferrenberg and R. H. Swendsen. Optimized monte carlo data analysis. *Phys. Rev. Lett.*, 63(12):1195–1198, 1988.

[24] M. E. Fisher and M. N. Barber. Optimized monte carlo data analysis. *Phys. Rev. Lett.*, 28(23):1516–1519, 1972.

[25] P. J. Flory. The configuration of a real polymer chain. *J. Chem. Phys.*, 17:303–310, 1949.

[26] P. J. Flory. *Principles of Polymer Chemistry.* Cornell University Press, Ithaca, NY, 1953.

[27] G. Grimmett. *Percolation.* Springer Verlag, New York, 1989.

[28] J. M. Hammersley. The number of polygons on a lattice. *Proc. Camb. Phil. Soc.*, 57:516–523, 1961.

[29] H. Holden, B. Øksendal, J. Ubøe, and T. Zhang. *Stochastic Partial Differential Equations.* Birkhäuser, Boston, 1996.

[30] N. Ikeda and S. Watanabe. *Stochastic Differential Equations and Diffusion Processes.* North Holland/Kodansha, New York/Tokyo, 2 edition, 1989.

[31] K. Itô. Stochastic integral. *Proc. Imp. Acad. Tokyo*, 20:519–524, 1944.

[32] K. Itô. Multiple wiener integral. *J. Math. Soc. Japan*, 3:157–169, 1951.

[33] K. Ito and H. P. McKean. *Diffusion Processes and Their Sample Paths.* Springer-Verlag, New York, 1965.

[34] I. Karatzas. *Brownian Motion and Stochastic Calculus.* Springer-Verlag, New York, 2 edition, 1991.

[35] P. E. Kloeden and E. Platten. *Numerical Solutions of Stochastic Differential Equations.* Springer Verlag, New York, 1992.

[36] F. B. Knight. *Essentials of Brownian motion and diffusion.* American Mathematical Society, Providence, RI, 1981.

[37] D. E. Knuth. *The Art of Computer Programming*, volume 3. Addison Wesley, Reading, Massachusetts, 1973.

[38] B. Kuchta and R. D. Etters. Features of the histogram monte carlo method: Application to n2 monolayer melting on graphite. *J. Comp. Phys.*, 108:353–356, 1993.

[39] J. Lamperti. *Probability: A Survey of the Mathematical Theory.* W. A. Benjamin, New York, 1966.

[40] L. D. Landau and E. M. Lifshitz. *Statistical Physics.* Pergamon Press, Oxford, 1969.

[41] G. Lawler. *Intersections of Random Walks.* Birkhauser, Boston, 1991.

[42] T. J. Ligocki. *Minimizing Knot Energies Using Simulated Annealing.* Ph.D. thesis, Univ. of California at Berkeley, Berkeley, 1995.

[43] N. Madras and A. Sokal. The pivot algorithm: a highly efficient monte carlo method for the self-avoiding walk. *J. Stat. Phys.*, 50:109–186, 1988.

[44] A. Malakis. The trail problem on the square lattice. *J. Phys. A*, 9:1283, 1976.

[45] H. P. McKean. *Stochastic Integrals.* Academic Press, New York, 1969.

[46] R. Meester and R. Roy. *Continuum Percolation.* Cambridge University Press, New York, 1996.

[47] H. Meirovitch. A monte carlo study of the entropy, the pressure, and the critical behavior of the hard sphere lattice gas. *J. Stat. Phys.*, 30:681–698, 1984.

[48] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equations of state calculations by fast computing machines. *J. Chem. Phys.*, 21:1087–1092, 1953.

[49] O. G. Mouritsen. *Computer Studies of Phase Transitions and Critical Phenomena.* Springer-Verlag, New York, 1984.

[50] B. Øksendal. *Stochastic Differential Equations.* Springer, New York, 4 edition, 1995.

[51] Y. Oono. Statistical physics of polymer solutions: conformation space renormalization group approach. In A. Rice, editor, *Advances in Chemical Physics*, volume 61, pages 301–437. Wiley, New York, 1985.

[52] Y. Oono and K. F. Freed. Conformation space renormalization of polymers. i. single chain equilibrium properties using wilson-type renormalization. *J. Chem. Phys.*, 75:993, 1981.

[53] R. H. J. M. Otten and L. P. P. P. van Ginneken. *The Annealing Algorithm.* Kluwer, Norwell, MA, 1989.

[54] R. E. A. C. Paley and N. Wiener. *Fourier Transforms in the Complex Domain.* American Mathematical Society, New York, 1934.

[55] R. G. Palmer. Broken ergodicity. *Adv. Phys.*, 31:669–735, 1982.

[56] T. S. Parker and L. O. Chua. *Practical Numerical Algorithms for Chaotic Systems.* Springer Verlag, New York, 1989.

[57] H. Peitgen, H. Jurgens, and D. Saupe. *Chaos and Fractals: New Frontiers of Science.* Springer Verlag, New York, 1992.

[58] H. Peitgen and D. Saupe, editors. *Chaos and Fractals: New Frontiers of Science.* Springer Verlag, New York, 1988.

[59] L. Pietronero. Survival probability for kinetic self-avoiding walks. *Phys. Rev. Lett.*, 55(19):2025–2027, 1985.

[60] P. H. Poole, A. Coniglio, N. Jan, and H. E. Stanley. Universality classes of the θ and θ' points. *Phys. Rev. B*, 39:495, 1989.

[61] S. Redner and P. J. Reynolds. Position-space renomalization group for isolated polymer chains. *J. Phys. A*, 14:2679, 1981.

[62] Y. Shapir and Y. Oono. Walks, trails and polymers with loops. *J. Phys. A*, 17:L39, 1984.

[63] G. Slade. The diffusion of self-avoiding random walk in high dimensions. *Comm. Math. Phys.*, 110:661–683, 1987.

[64] G. Slade. The scaling limit of self-avoiding random walk in high dimensions. *Annals of Prob.*, 17:91–107, 1989.

[65] D. Stauffer. *Introduction to Percolation Theory.* Taylor & Francis, Philadelphia, 1985.

[66] R. L. Stratonovich. A new representation for stochastic integrals and equations. *J. Siam Control*, 4:362–371, 1966.

[67] G. E. Uhlenbeck and G. W. Ford. *Lectures in Statistical Mechanics.* American Mathematical Society, Providence, RI, 1963.

[68] F. T. Wall and F. Mandel. Macromolecular dimensions obtained by an efficient monte carlo method without sample attrition. *J. Chem. Phys.*, 63:4592, 1975.

[69] K. G. Wilson and J. Kogut. The renormalization group and the epsilon expansion. *Phys. Rep.*, 12c:75–200, 1974.

Ernest Orlando Lawrence Berkeley National Laboratory
One Cyclotron Road | Berkeley, California 94720

Prepared for the U.S. Department of Energy under Contract No. DE-AC03-76SF00098