universität wien

DIPLOMARBEIT

Titel der Diplomarbeit

Coil-to-crystal transition of a polymer chain with square-well interactions: A transition path sampling simulation study

angestrebter akademischer Grad

Magister der Naturwissenschaften (Mag.rer.nat.)

Verfasser:	Christian Leitold
Matrikel-Nummer:	0604626
Studienrichtung (lt. Studienblatt):	Physik
Betreuer:	Univ.-Prof. Dr. Christoph Dellago

Wien, am 27. Juli 2011

Danksagung

Rückblickend mag es einem als eine Selbstverständlichkeit erscheinen, in aller Ruhe das studieren zu können, was am meisten den persönlichen Interessen entspricht. In Wahrheit muss dem nicht unbedingt so sein, umso mehr ist es ein Privileg, wenn man diese Möglichkeit tatsächlich erhält. Zuallererst möchte ich daher meiner Familie danken, die mir durch ihre fortwährende Unterstützung das Physikstudium überhaupt erst ermöglicht hat.

Meinem Betreuer Christoph Dellago möchte ich für die Möglichkeit danken, in einer exzellenten Gruppe diese Diplomarbeit verfassen zu können. Von ihm konnte ich sehr viel lernen, insbesondere auch, worauf es ankommt, wenn man an vorderster Front Wissenschaft betreibt. Auch allen meinen Kollegen in der Gruppe möchte ich für zahlreiche wertvolle wissenschaftliche und weniger wissenschaftliche Diskussionen und die allgemein sehr gute, lockere Arbeitsatmosphäre danken.

Ich wäre in meinem Studium nie so weit gekommen, hätte ich nicht schon frühzeitig immer wieder auf die Unterstützung zahlreicher anderer Studierender aus meinem Jahrgang zurückgreifen können. Von ihnen habe ich gelernt, dass es in der Physik in allererster Linie auf Teamwork ankommt. Dafür, und auch für die weit über die Universität hinausreichenden Freundschaften, die sich im Laufe der letzten fünf Jahre ergeben haben, bin ich sehr dankbar.

Zuletzt möchte ich meiner Freundin Dagmar Tanda danken, die mich gerade im vergangenen Studienjahr im wahrsten Sinn des Wortes über alle Grenzen und Zeitzonen hinweg kontinuierlich unterstützt und ermutigt hat.

Contents

1. Introduction **1**

2. The square-well potential polymer chain model **3**
2.1. Mean-square radius of gyration 5
2.2. Analytic results for small chains 5
2.3. Phase transitions and phase diagram 7

3. Simulation Methods **13**
3.1. Metropolis Monte Carlo 13
3.1.1. Metropolis algorithm 14
3.2. Transition path sampling 14
3.2.1. Transition path ensemble 15
3.2.2. Generating new trajectories out of existing ones 17
3.2.3. Shooting moves 18
3.2.4. Shifting moves 19
3.2.5. Committor analysis and transition state ensemble 21
3.3. Structural analysis 22
3.3.1. Pair correlation function 22
3.3.2. Steinhardt bond order parameters 23
3.3.3. Connection coefficients 25

4. Implementation **27**
4.1. Metropolis Monte Carlo simulation 27
4.2. Transition path sampling 28

5. Path Sampling Statistics **31**
5.1. The path survival function 31
5.2. Simulating the simulation 32
5.3. Analytic calculation of the path survival function 33
5.4. Mean time between path updates 35
5.5. Tree representation of path acceptance 40
5.6. Calculation for a simple acceptance model and comparison with a simulated TPS run 41

6. Results and discussion **47**
6.1. Metropolis Monte Carlo results 47

6.2. Committor analysis 52
6.2.1. Finding a reaction coordinate 53
6.2.2. Transition state ensemble 56

7. Final Remarks **61**

A. Calculation of the mean-square radius of gyration **63**

B. Calculation of the average time between path updates **65**

1. Introduction

Proteins are one of the most essential parts of any biological organism, taking part in virtually every process within living cells. Among many other functions, proteins make our muscles move, carry the oxygen in our blood and neutralize harmful intruders like bacteria and viruses. The first important information about any protein is the sequence of its building blocks, the amino acids, organic molecules consisting of elements such as carbon, hydrogen, oxygen and nitrogen. Biochemists call the sequence of amino acids the primary structure of a protein. Still, key to the biological function is the way the amino acids are structured within the molecule, the way how they are folded, which determines a protein's secondary and tertiary structure. Everybody knows that an egg undergoes a remarkable transition when cooked, yet this transition solely changes the structure of the proteins in the egg, but not the sequence of amino acids.

Clearly, such complex folding and unfolding transitions are also of interest in physics. However, right now, the full numerical treatment of a real-world protein is rather challenging. After all, a typical protein consists of 100 to 300 amino acids, and each amino acid is built of about 20 atoms, leading to a huge amount of complex interactions. Therefore, to work with proteins in a computer simulation, typically one has to rely on coarse-grained descriptions.

A radically different approach, compared to coarse-grained descriptions, is to try to capture the essential properties of complex molecules like proteins with an even much simpler model. This thesis deals with such a model. Instead of trying to find an accurate description of some specific protein, we just work with a polymer chain of identical, spherical particles with some short-range attractive force. Despite its simple interactions, this model is able to reproduce quite a complex phase behavior. Hence, the polymer chain can serve as a model of short chain-like proteins, as for example Ubiquitin, a regulatory protein that has been found in almost all tissues.

In particular, the chain is able to fold itself into a ground state where the particles are packed very closely together. Contrary to a real protein, this ground state is not unique. Nevertheless, we would like to understand the complex folding transition on the most fundamental level, and computer simulations are able to provide such understanding.

The simulation technique we use to achieve this goal is called transition path sampling. As the name already indicates, it is able to find pathways taking the system from one stable state, say, the unfolded state, to another stable state, which is, in our case, of course the folded state. Afterwards, with the harvested pathways at our disposal, we can follow the transition step by step and search for important intermediate configurations, or transition states. In this thesis, we apply

the transition path sampling technique in order to learn more about the polymer chain. That way, we can look for intermediate states of the transition and try to find a reaction coordinate. A reaction coordinate is a number that conveys the progress of the transition, in other words, how far the reaction or transition has already gone from the initial to the final state. However, for the polymer chain this task remains an open challenge so far.

In addition to the analysis of the polymer chain, we perform a mathematical analysis of the mechanisms that lead to the generation of new transition pathways in the simulation. Then we are able to compare this analysis' predictions with the results from the actual simulation.

2. The square-well potential polymer chain model

The model investigated throughout this work is a single, flexible homopolymer chain. It consists of N identical, spherical monomers with a hard sphere core of diameter σ. The distance L to neighboring monomers is fixed, and in our case, $L = \sigma$ always holds true. Non-neighboring monomers interact via the square-well potential

$$u(r_{ij}) = \begin{cases} \infty & 0 \leq r_{ij} < \sigma, \\ -\varepsilon & \sigma \leq r_{ij} \leq \lambda\sigma, \\ 0 & r_{ij} > \lambda\sigma, \end{cases} \tag{2.1}$$

where $r_{ij} = |\vec{r}_i - \vec{r}_j|$ is the distance between the i-th and the j-th monomer. From this definition it is clear that overlaps of the hard cores are not allowed. Furthermore, the parameter λ defines the range of the attractive interaction. Obviously, $\lambda > 1$, since $\lambda = 1$ is just an ordinary hard sphere potential. Apart from the fixed bond lengths between neighboring monomers and the restrictions arising from the hard core of the square-well potential, the chain is allowed to move freely.

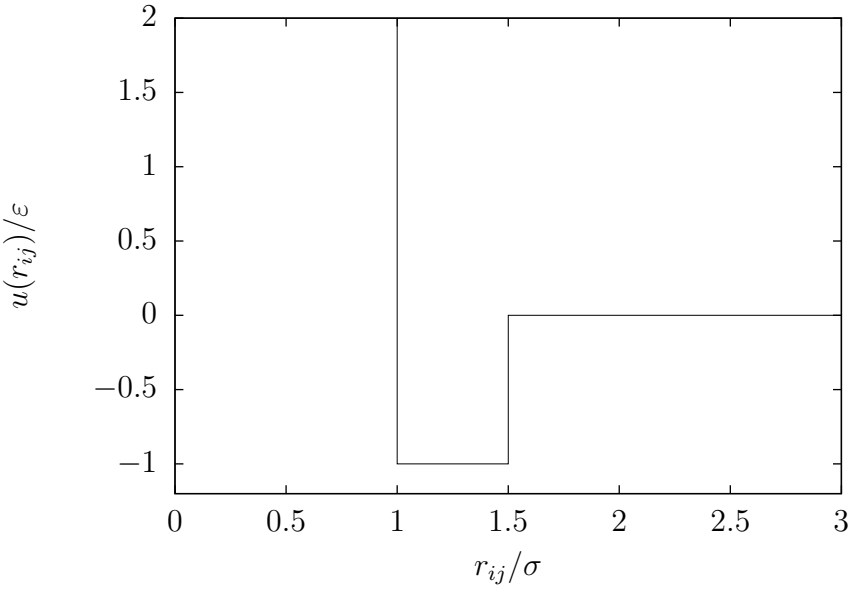

Figure 2.1.: Square-well potential for $\lambda = 1.5$

We can think of the polymer chain as a simple model for small chain-like proteins. With that interpretation, the square-well potential has to be considered as an effective potential, already taking into consideration any solvent effects. However, any temperature dependence of the solvent's properties (and hence of the effective potential) is neglected.

Due to the form of the potential, the system has a discrete energy spectrum given as

$$E_n = -n\varepsilon, \tag{2.2}$$

where n is the number of square-well overlaps in the chain. Obviously, the ground state energy depends on the interaction range λ, since this value will affect the number of overlaps in a given chain configuration.

This model, despite its simplicity, has already shown a number of interesting thermodynamic properties. Hence, there have been done a considerable amount of investigations of the system [1–3]. In the remainder of this chapter, we will therefore discuss the results of previous research on the square-well polymer chain.

In the following, reduced units will be used instead of physical units. This means, for example, instead of the temperature T we use the reduced temperature

$$T^* = \frac{k_B T}{\varepsilon}, \tag{2.3}$$

which is a dimensionless number. Here, k_B is the Boltzmann constant. Similarly, the dimensionless length $r_{ij}^* = r_{ij}/\sigma$ will be used. For the sake of simple notation, we will furthermore omit the superscripts for the reduced units.

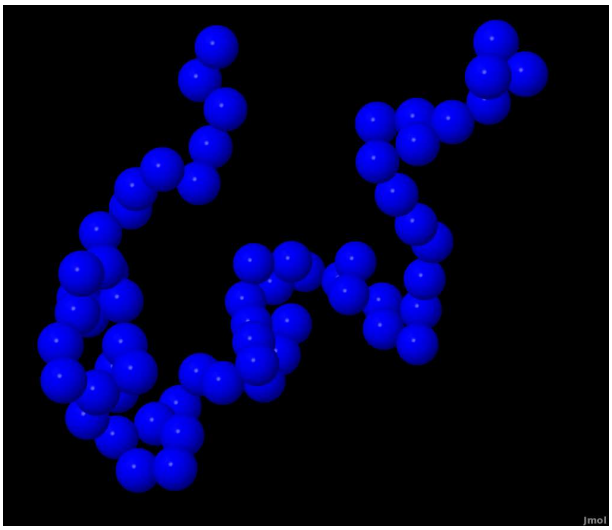

Figure 2.2.: A graphical representation of the $N = 64$ polymer chain. The spheres correspond with the hard cores of the individual monomers.

2.1. Mean-square radius of gyration

One of the most intriguing properties of the polymer chain is its phase behavior under different (reduced) temperatures T and interaction ranges λ. Some early research on this topic was done by Zhou et al. [1]. They have performed Monte Carlo simulations for chains with interactions of Eqn. (2.1) with $\lambda = 1.5$ and lengths up to $N = 64$. One of the quantities they have studied is the mean-square radius of gyration

$$R_g^2 = \left\langle \frac{1}{N} \sum_{i=1}^{N} (\vec{r}_i - \vec{r}_{\text{mean}})^2 \right\rangle, \tag{2.4}$$

which is nothing more than the mean-squared distance of the monomers from the geometrical center of the chain

$$\vec{r}_{\text{mean}} = \frac{1}{N} \sum_{i=1}^{N} \vec{r}_i. \tag{2.5}$$

For equal-mass particles (as it is the case for our chain), the geometrical center coincides with the center of mass. Sometimes, the alternative expression

$$R_g^2 = \left\langle \frac{1}{N^2} \sum_{i<j} r_{ij}^2 \right\rangle \tag{2.6}$$

is found. It is shown in appendix A that this expression is equivalent to Eqn. (2.4).

Zhou et al. have calculated R_g^2 for a variety of chain sizes and temperatures, as can be seen in Fig. 2.3. Later, I will compare my own simulations with these results. The sigmoidal shape of the function already shows that a transition from an expanded high-temperature phase to a compact low-temperature phase takes place.

2.2. Analytic results for small chains

For very small chains, namely for $N = 3$ and $N = 4$, Taylor and Lipson were able to calculate the mean-square radius of gyration, the mean energy of the chain and its heat capacity analytically exact [4]. For these short chains, it is possible to analytically calculate the intramolecular distribution function of the system by integration. Roughly speaking, the intramolecular distribution function $w_{ij}(r)$ tells one the expected distance between the i-th and j-th particle. Having determined that, one can calculate e. g. the mean-square distance between the i-th and j-th particle as [4]

$$\left\langle r_{ij}^2 \right\rangle = 4\pi \int dr r^4 w_{ij}(r). \tag{2.7}$$

The mean-square radius of gyration is then obtained by averaging this quantity over all pairs of particles, in analogy to Eqn. (2.6):

$$R_g^2 = \frac{1}{N^2} \sum_{i<j} \left\langle r_{ij}^2 \right\rangle. \tag{2.8}$$

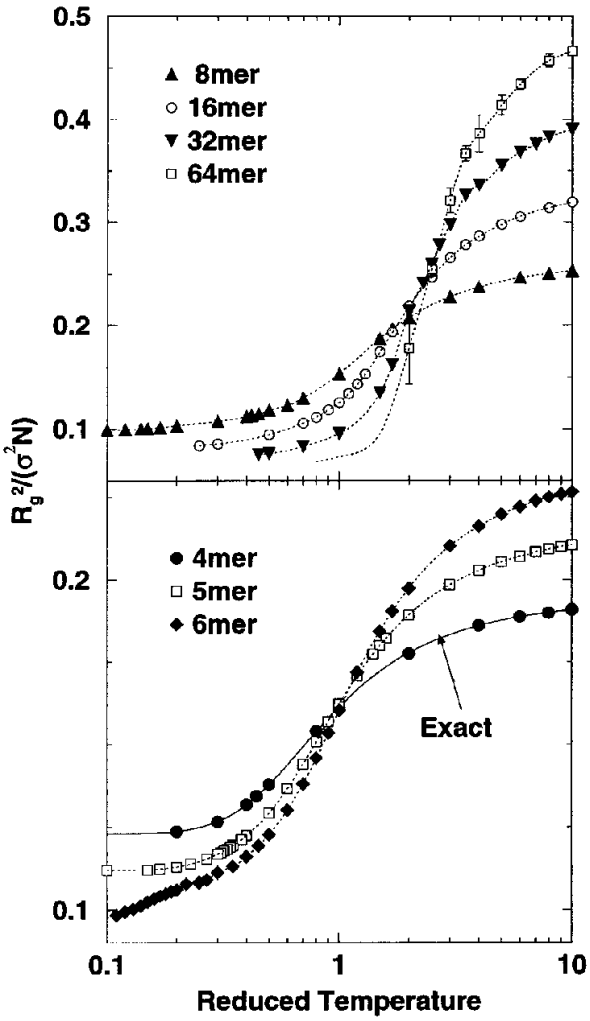

Figure 2.3.: Mean-square radius of gyration for different chain lengths and $\lambda = 1.5$, as calculated by Zhou et al. The solid line represents the analytic result from Taylor and Lipson [4, 5] for the $N = 4$ chain, while the dotted lines are spline fits. Image taken from Ref. [1].

For the $N = 4$ chain, some of the integrals contributing to the intramolecular distribution function have to be determined numerically. The final results, given in Fig. 2.3 (for the $N = 4$ chain), agree well with values calculated in Monte Carlo simulations.

For longer chains, approximations have to be made. This, while conserving the qualitative behavior of the system, results in a disagreement between the analytic values and the results from simulations [5].

2.3. Phase transitions and phase diagram

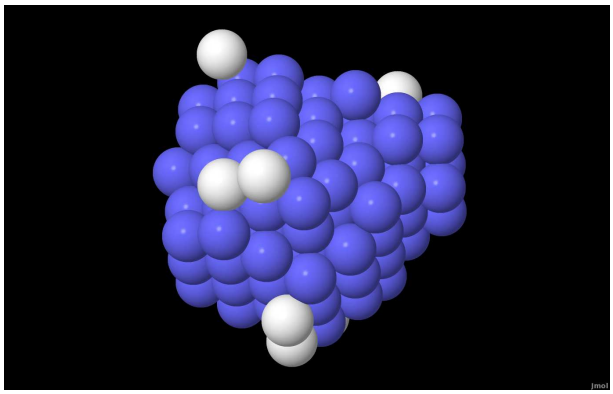

Figure 2.4.: A graphical representation of the crystallite state for the $N = 128$ polymer chain, with $\lambda = 1.04$ and $T = 0.424$. Again, the spheres correspond with the hard cores of the individual monomers.

Taylor, Paul and Binder have determined a (T–λ) phase diagram for a $N = 128$ chain as shown in Fig. 2.5 [2]. Depending on the value of λ, there exist two or three different phases. In all cases, for high temperatures the chain is in the expanded coil phase, and then undergoes one or two phase transitions when the temperature is decreased. For values $\lambda > 1.06$, there exists a second-order phase transition to the compact, but disordered collapsed globule phase. Decreasing the temperature further leads to a first-order transition to the frozen crystallite phase. For $\lambda \leq 1.05$, the expanded coil freezes directly into the crystallite phase. In my work, I concentrated on this discontinuous phase transition. The phase diagram for the $N = 64$ chain shown in Fig. 2.6 looks similar, but is slightly shifted downwards. The $N = 64$ chain freezes directly to the crystallite phase for $\lambda \lesssim 1.04$. These (to my knowledge yet unpublished) results for the smaller chain were provided by Wolfgang Paul.

Taylor et al. [2] have obtained their results using the Wang-Landau sampling method. This Monte Carlo algorithm is able to construct the density of states $g(E_n)$ for a given system by iteration, even though in this case the function covers about 840 orders of magnitude. Then, from the known density of states, one can

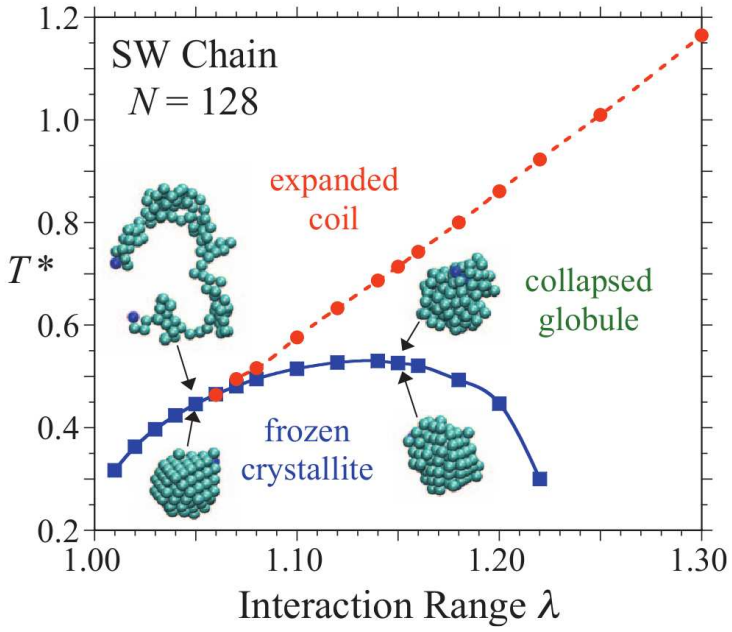

Figure 2.5.: Temperature – interaction range (T–λ) phase diagram for the $N = 128$ polymer chain (taken from Ref. [2]). The dots represent the phase transition temperatures for the respective values of λ, while the lines are interpolations between these points.

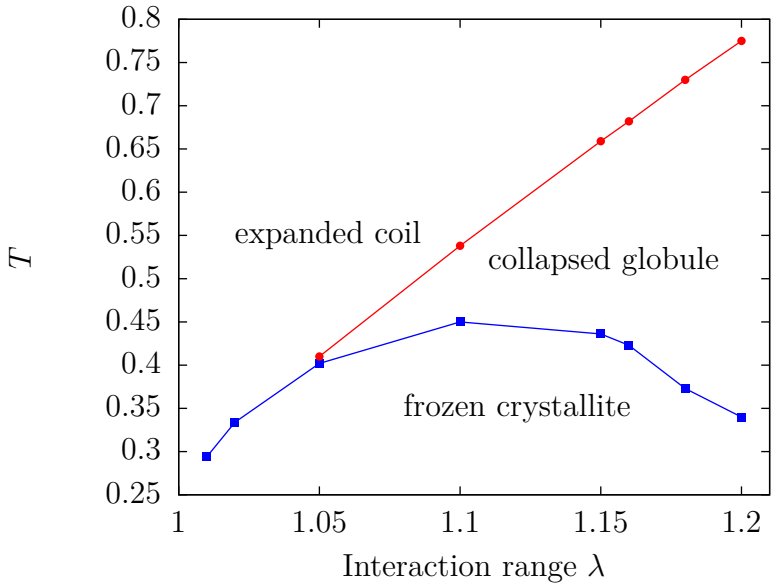

Figure 2.6.: (T–λ) phase diagram for the $N = 64$ polymer chain. Again, the dots represent the phase transition temperatures, while the lines are interpolations. Data provided by Wolfgang Paul, University of Jena.

calculate all desired thermodynamic properties by integration. In particular, the location of phase transitions can be derived from an analysis of the heat capacity of the system [3]. The heat capacity is given as

$$C(T) = \frac{1}{kT^2}\left(\langle E^2\rangle - \langle E\rangle^2\right). \tag{2.9}$$

This result can be obtained by combining the equations of classical thermodynamics and those of statistical mechanics. The expectation values $\langle E^2\rangle$ and $\langle E\rangle$ are easy to calculate if the density of states is known. In particular we have

$$U(T) = \langle E_n\rangle = \frac{1}{Z(T)}\sum_n E_n g(E_n) e^{-E_n/k_BT}, \tag{2.10}$$

with the partition function

$$Z(T) = \sum_n g(E_n) e^{-E_n/k_BT} \tag{2.11}$$

as a normalization constant. Then, for a finite-size system, maxima in $C(T)$ can be related to phase transitions or similar structural rearrangements, as seen in Fig. 2.7. This approach, used by Taylor et al. [2, 3], has the advantage that one has to perform only one simulation, and can afterwards use the obtained results for all temperatures. In Chapter 3, we will discuss the more traditional Metropolis algorithm used in my work, which is only valid for a fixed temperature.

An alternative to the approach discussed so far is to directly make use of the probability density function

$$P(E_n, T) = \frac{1}{Z(T)} g(E_n) e^{-E_n/k_BT} \tag{2.12}$$

at a given temperature. If the system is close to a first-order phase transition, one expects to see a bimodal distribution in the probability density. Directly at the transition, both peaks should be weighted equally, as seen in Fig. 2.8.

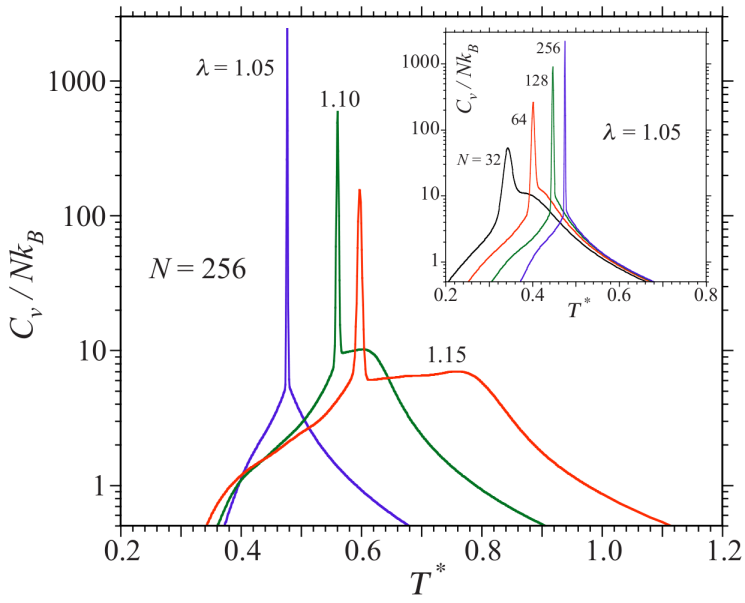

Figure 2.7.: Heat capacity per monomer for the $N = 256$ polymer chain for different values of λ, and, in the inset, for $\lambda = 1.05$ and different chain sizes N. The clear peak in the specific heat function indicates the location of the freezing transition, while the shoulder seen only for bigger values of λ (or smaller N) locates the coil-to-globule transition. Image taken from Ref. [3].

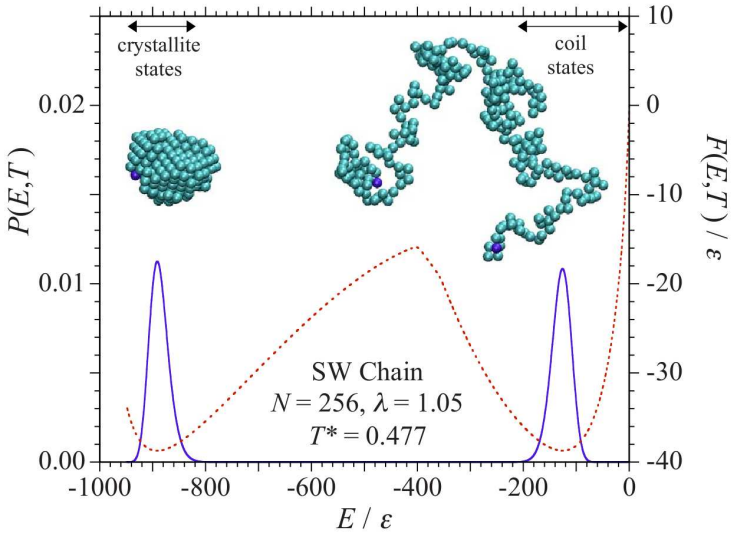

Figure 2.8.: Probability density function for the $N = 256$ polymer chain (solid line), taken from Ref. [3]. Since $T^* = 0.477$ is a transition temperature for the system under investigation, the two peaks in the bimodal distribution have equal area. The dashed line represents the free energy $F(E,T) = -k_B T \ln P(E,T)$ of the system. Note the location of the two phases in the local minima of the free energy, and there is a barrier in between separating the phases.

3. Simulation Methods

In this chapter, we will discuss the numerical methods used in my investigation of the polymer chain. First, I will give a brief overview of the standard Metropolis Monte Carlo algorithm to generate samples out of a given probability distribution. After that, we will learn the basics about transition path sampling. This simulation technique, which is a direct application of the Metropolis algorithm, is able to sample transition pathways as they occur in phase transitions. Then, we will discuss a number of methods used to analyze the structural properties of simulated systems.

3.1. Metropolis Monte Carlo

Let us assume we have some given probability distribution ρ. For example, in the canonical ensemble of statistical mechanics, this distribution is

$$\rho(r^N, p^N) \propto e^{-\beta H(r^N, p^N)}, \tag{3.1}$$

which is a function of all the coordinates r^N and momenta p^N of the physical system. Here, $\beta = 1/k_B T$ is the inverse temperature and $H(r^N, p^N) = U(r^N) + E_{\text{kin}}(p^N)$ is the Hamiltonian, the sum of potential and kinetic energy of the system. The expectation value of some observable A is calculated as

$$\langle A \rangle = \frac{\int dr^N dp^N e^{-\beta H(r^N, p^N)} A(r^N, p^N)}{\int dr^N dp^N e^{-\beta H(r^N, p^N)}}. \tag{3.2}$$

Now suppose, A does not depend on the momenta of the system. Then the integration over p^N cancels out of Eqn. (3.2) and we are left with

$$\langle A \rangle = \frac{\int dr^N e^{-\beta U(r^N)} A(r^N)}{\int dr^N e^{-\beta U(r^N)}}. \tag{3.3}$$

This kind of integral can be approximated by taking a sample out of the distribution and evaluating the value of A at the sample configurations:

$$\langle A \rangle \approx \frac{1}{N} \sum_{i=1}^{N} a_i. \tag{3.4}$$

In that case, one takes the unweighted average of all the values evaluated at the chosen configuration. However, these configurations are distributed according to

the respective probability distribution. Metropolis Monte Carlo, first introduced in Ref. [6], is a way of generating such a sample by performing a random walk in configuration space. This random walk is constrained by rules which ensure the validity of the sample.

3.1.1. Metropolis algorithm

Without further proof, we will just give the details of the Metropolis algorithm, for the special case mentioned above. That is to say, we have a system in the canonical ensemble. This system is described by the number of particles N, the volume V and the temperature T. To specify a configuration of the system, we need to know the positions of all particles.

We start with some arbitrary initial configuration, denoted by r^N. The algorithm then works as follows:

1. Select some random particle i.
2. Generate a trial state: Move that particle by a small random distance $\Delta\vec{r}$, such that $\vec{r}_i \to \vec{r}_i{}' = \vec{r}_i + \Delta\vec{r}$.
3. Evaluate the value of the potential at the new trial configuration and take the difference to the old value: $\Delta U = U(..., \vec{r}_i{}', ...) - U(..., \vec{r}_i, ...)$.
4. If $\Delta U < 0$, accept the trial configuration. Otherwise, only accept the trial configuration with a probability of $e^{-\beta\Delta U}$.
5. If accepted, the trial state becomes you new initial state. If not, just keep the old initial state for the next iteration.
6. Continue from step 1.

It is important to note that $e^{-\beta\Delta U}$ is nothing but the ratio of the probability distribution evaluated at the trial state to that at the initial state. That means, the higher the energy difference ΔU is, the less likely a trial state gets accepted. This acceptance criterion is actually only one of many different ones, all satisfying the detailed balance condition. This condition ensures the validity of the generated random walk as a (correlated) sample of the probability distribution.

In general, the Metropolis algorithm can be used for any system where one wants to generate a sample from a known probability distribution.

3.2. Transition path sampling

Transition path sampling, or TPS, is a simulation technique to sample rare transitions in complex systems. Suppose we have a system which is most of the time in either one of two well-known stable states, denoted by A and B. These states correspond to two regions in the configuration space of the system. An obvious

example would be the polymer chain at the respective phase coexistence temperature, with A and B corresponding to the expanded coil and the frozen crystallite. Discriminating between the two states A and B is easy; however, the transition mechanism taking the system from A to B is unknown. Here, transition path sampling comes into play, since it is able to sample the transition pathways. In the following, I will explain the details of this method. Most of these details are taken from various review articles, such as [7–9].

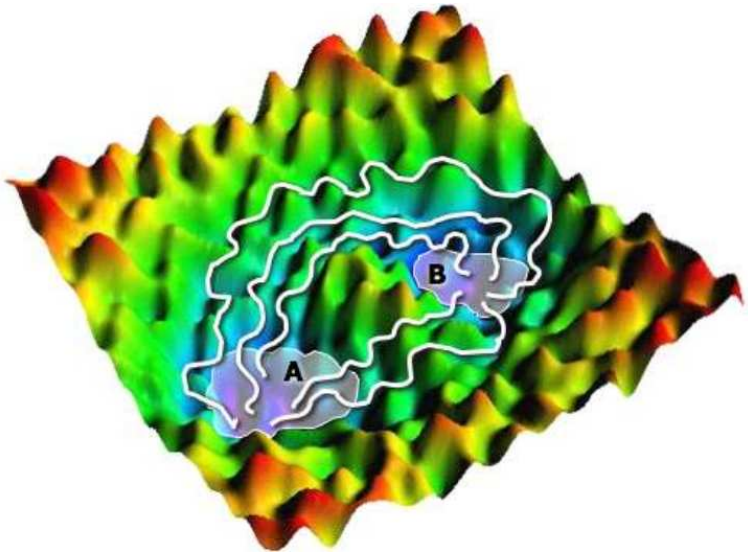

Figure 3.1.: Transition pathways connecting the stable states A and B on a schematic sketch of a rough potential energy landscape. Image taken from Ref. [7].

3.2.1. Transition path ensemble

The basis for TPS is the definition of the transition path ensemble, or in other words, the set of all reactive pathways connecting A with B. Those pathways should start at time 0 in A and end at $\mathcal{T}$ in B. Each path is described by an ordered sequence of microstates, or configurations:

$$x(\mathcal{T}) \coloneqq \{x_0, x_{\Delta t}, x_{2\Delta t}, \ldots, x_{\mathcal{T}}\}. \tag{3.5}$$

x_t, which we also call a time *slice*, is the complete configuration of the system at time t. This time does not necessarily have to be a physical time. As we will see later, TPS can also be applied if the underlying dynamics of the system is a Monte Carlo "time" evolution. Depending on the concrete dynamics, x_t may consist of all the positions and momenta (r^N, p^N) of the particles, or of all the positions (r^N) only. In any case, the underlying dynamics are assumed to be Markovian, meaning the probability of the future time evolution is fully described by the current state of the system and does not depend on previous states.

The probability density for observing a particular path can be written as the initial probability density $\rho(x_0)$ multiplied with the product of all single time step transition probabilities:

$$\mathcal{P}[x(\mathcal{T})] = \rho(x_0) \prod_{i=0}^{\mathcal{T}/\Delta t-1} p(x_{i\Delta t} \to x_{(i+1)\Delta t}). \tag{3.6}$$

Obviously, both the probability density for the initial state and the transition probabilities for the single time steps depend on the details of the system under consideration and the respective dynamics of the simulation. Now, if we want to obtain the transition path ensemble, we have to restrict the path ensemble of Eqn. (3.6) to paths beginning in A and ending in B. Mathematically, that is done by introducing the characteristic function of the regions,

$$\mathcal{P}_{AB}[x(\mathcal{T})] \coloneqq h_A(x_0)\mathcal{P}[x(\mathcal{T})]h_B(x_\mathcal{T})/Z_{AB}(\mathcal{T}), \tag{3.7}$$

with the characteristic function

$$h_A(x) = \begin{cases} 1 & \text{if} \quad x \in A, \\ 0 & \text{if} \quad x \notin A. \end{cases} \tag{3.8}$$

h_B is defined accordingly for region B. In addition to that, we need the path partition function

$$Z_{AB}(\mathcal{T}) \coloneqq \int \mathcal{D}x(\mathcal{T})\, h_A(x_0)\mathcal{P}[x(\mathcal{T})]h_B(x_\mathcal{T}) \tag{3.9}$$

as a normalization constant. The notation

$$\int \mathcal{D}x(\mathcal{T}) \coloneqq \int \dots \int dx_o dx_{\Delta t} \dots dx_\mathcal{T} \tag{3.10}$$

stands for an integration over all time slices, similar to the path integral formalism of quantum mechanics. The set of all pathways connecting A with B, together with the weight function (3.7), is called the transition path ensemble.

Having done the formal definition, we would now like to create a working implementation of a TPS algorithm, which is able to sample the transition path ensemble. As a first step, we are interested in the form of the single time step transition probabilities, if the system performs a Metropolis Monte Carlo random walk in configuration space. In this case we have [9]

$$p(x_t \to x_{t+\Delta t}) = w(x_t \to x_{t+\Delta t}) + \delta(x_t - x_{t+\Delta t})Q(x_t), \tag{3.11}$$

with the acceptance probability for a trial move from x_t to $x_{t+\Delta t}$

$$w(x_t \to x_{t+\Delta t}) = \eta(x_t \to x_{t+\Delta t}) \cdot \min\left[1, \frac{\rho(x_{t+\Delta t})}{\rho(x_t)}\right] \tag{3.12}$$

and the total rejection probability for a move starting from x_t

$$Q(x) = 1 - \int dx'\, w(x \to x'). \tag{3.13}$$

In other words, if we want to give the full probability (density) for going from some initial state x_t to *any* possible final state $x_{t+\Delta t}$, we must not forget the possibility to stay at x_t in the case of rejection, represented by the second term in Eqn. (3.11). Furthermore, Eqn. (3.12) is nothing but the usual Metropolis acceptance criterion, multiplied with the ad-hoc trial probability $\eta(x_t \to x_{t+\Delta t})$ to generate a trial move from x_t to $x_{t+\Delta t}$.

3.2.2. Generating new trajectories out of existing ones

Let us examine the detailed balance condition for the transition path ensemble in more detail. Formally, it means that the frequency of accepted moves from an old path to a new one is exactly balanced by the frequency of the opposite move [9]:

$$\mathcal{P}_{AB}[x^{(\mathrm{o})}(\mathcal{T})]\pi[x^{(\mathrm{o})}(\mathcal{T}) \to x^{(\mathrm{n})}(\mathcal{T})] = \mathcal{P}_{AB}[x^{(\mathrm{n})}(\mathcal{T})]\pi[x^{(\mathrm{n})}(\mathcal{T}) \to x^{(\mathrm{o})}(\mathcal{T})]. \tag{3.14}$$

Here, the superscripts $^{(\mathrm{o})}$ and $^{(\mathrm{n})}$ stand for the old respectively new trajectory while $\pi[x^{(\mathrm{o})}(\mathcal{T}) \to x^{(\mathrm{n})}(\mathcal{T})]$ is the conditional probability to accept a move to $x^{(\mathrm{n})}$ given the initial path is $x^{(\mathrm{o})}$. As in the case of traditional Monte Carlo, this probability can be expressed as a product of the ad-hoc trial probability and the actual acceptance probability:

$$\pi[x^{(\mathrm{o})}(\mathcal{T}) \to x^{(\mathrm{n})}(\mathcal{T})] = P_{\mathrm{gen}}[x^{(\mathrm{o})}(\mathcal{T}) \to x^{(\mathrm{n})}(\mathcal{T})] \times P_{\mathrm{acc}}[x^{(\mathrm{o})}(\mathcal{T}) \to x^{(\mathrm{n})}(\mathcal{T})]. \tag{3.15}$$

Combining the last two equations, one obtains a condition for the acceptance probability:

$$\frac{P_{\mathrm{acc}}[x^{(\mathrm{o})}(\mathcal{T}) \to x^{(\mathrm{n})}(\mathcal{T})]}{P_{\mathrm{acc}}[x^{(\mathrm{n})}(\mathcal{T}) \to x^{(\mathrm{o})}(\mathcal{T})]} = \frac{\mathcal{P}_{AB}[x^{(\mathrm{n})}(\mathcal{T})]P_{\mathrm{gen}}[x^{(\mathrm{n})}(\mathcal{T}) \to x^{(\mathrm{o})}(\mathcal{T})]}{\mathcal{P}_{AB}[x^{(\mathrm{o})}(\mathcal{T})]P_{\mathrm{gen}}[x^{(\mathrm{o})}(\mathcal{T}) \to x^{(\mathrm{n})}(\mathcal{T})]}. \tag{3.16}$$

Once again, that condition can be satisfied using the Metropolis acceptance criterion

$$P_{\mathrm{acc}}[x^{(\mathrm{o})}(\mathcal{T}) \to x^{(\mathrm{n})}(\mathcal{T})] = \min\left[1, \frac{\mathcal{P}_{AB}[x^{(\mathrm{n})}(\mathcal{T})]P_{\mathrm{gen}}[x^{(\mathrm{n})}(\mathcal{T}) \to x^{(\mathrm{o})}(\mathcal{T})]}{\mathcal{P}_{AB}[x^{(\mathrm{o})}(\mathcal{T})]P_{\mathrm{gen}}[x^{(\mathrm{o})}(\mathcal{T}) \to x^{(\mathrm{n})}(\mathcal{T})]}\right]. \tag{3.17}$$

Now, since the old trajectory is reactive, in other words $h_A[x_0^{(\mathrm{o})}] = h_B[x_{\mathcal{T}}^{(\mathrm{o})}] = 1$, this expression can be simplified to

$$P_{\mathrm{acc}}[x^{(\mathrm{o})} \to x^{(\mathrm{n})}] = h_A[x_0^{(\mathrm{n})}]h_B[x_{\mathcal{T}}^{(\mathrm{n})}] \min\left[1, \frac{\mathcal{P}[x^{(\mathrm{n})}(\mathcal{T})]P_{\mathrm{gen}}[x^{(\mathrm{n})} \to x^{(\mathrm{o})}]}{\mathcal{P}[x^{(\mathrm{o})}(\mathcal{T})]P_{\mathrm{gen}}[x^{(\mathrm{o})} \to x^{(\mathrm{n})}]}\right]. \tag{3.18}$$

Note that here, for the sake of a shorter equation, we have used $x^{(\mathrm{o})} \to x^{(\mathrm{n})} \coloneqq x^{(\mathrm{o})}(\mathcal{T}) \to x^{(\mathrm{n})}(\mathcal{T})$, while $x_0^{(\mathrm{n})}$ and $x_{\mathcal{T}}^{(\mathrm{n})}$ stand for the (single!) configurations at times 0 and $\mathcal{T}$ in the new path.

In the following, we will apply Eqn. (3.18) to a standard Metropolis simulation as the underlying dynamics and thus make the last step towards a full TPS simulation: Obtaining correct algorithms for generating new trial paths out of existing ones. Two such algorithms have proven to be especially useful: Shooting and shifting moves.

3.2.3. Shooting moves

The basic idea for a *shooting* move is to randomly select a time slice within the path, and then (at least in the case of deterministic dynamics) perturb the system slightly at that point. Then you use the perturbed state as a starting point for a forward and backward time evolution of the system, thus generating a new trajectory. This new trial trajectory is then – as in traditional Monte Carlo – accepted with an appropriate acceptance criterion. However, if the underlying dynamics are of a stochastic nature, as it is the case for Monte Carlo dynamics, we can omit the modification step. Instead, it is possible to directly start the new trial trajectory from an unmodified configuration. Furthermore, you only need to generate either the backward or forward trajectory segment in one move, thus saving computational cost.

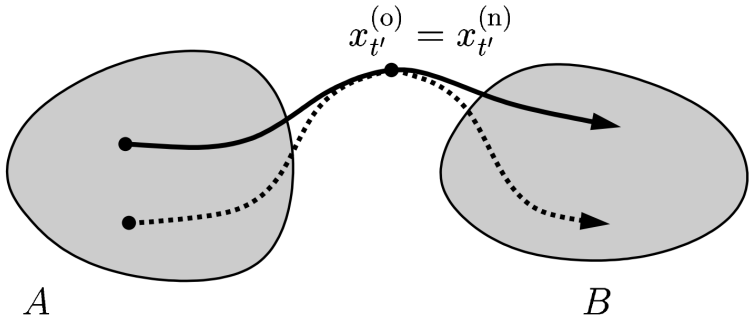

Figure 3.2.: Schematic picture of a shooting move for stochastic dynamics. Due to the stochastic nature of the system, the shooting point need not be modified prior to the forward and backward shooting. Image taken from [9].

For the forward path segment of a trial trajectory, beginning at some time t', the generation probability can be expressed as a product of the single-step transition probabilities [9]:

$$P_{\mathrm{gen}}^{\mathrm{f}}[\mathrm{o} \to \mathrm{n}] = \prod_{i=t'/\Delta t}^{\mathcal{T}/\Delta t - 1} p\left(x_{i\Delta t}^{(\mathrm{n})} \to x_{(i+1)\Delta t}^{(\mathrm{n})}\right), \tag{3.19}$$

where $p(x_t \to x_{t+\Delta t})$ is already known from Eqn. (3.6). A similar expression gives the generation probability for the backward segment of the trial trajectory. In order to obtain the full generation probability for the complete trajectory, one

must also take into consideration the respective probability for the modification of the shooting point. The final generation probability is then given as [9]:

$$P_{\text{gen}}[x^{(\text{o})}(\mathcal{T}) \to x^{(\text{n})}(\mathcal{T})] = p_{\text{gen}}[x^{(\text{o})}_{t'} \to x^{(\text{n})}_{t'}] \prod_{i=t'/\Delta t}^{\mathcal{T}/\Delta t-1} p\left(x^{(\text{n})}_{i\Delta t} \to x^{(\text{n})}_{(i+1)\Delta t}\right) \times \prod_{i=1}^{t'/\Delta t} \bar{p}\left(x^{(\text{n})}_{i\Delta t} \to x^{(\text{n})}_{(i-1)\Delta t}\right), \tag{3.20}$$

with the probability $p_{\text{gen}}[x^{(\text{o})}_{t'} \to x^{(\text{n})}_{t'}]$ to generate the state $x^{(\text{n})}_{t'}$ out of $x^{(\text{o})}_{t'}$ and the single-step transition probability $\bar{p}(x_t \to x_{t-\Delta t})$ for the time-reversed evolution of the system. Now, in order to obtain an explicit expression for the acceptance criterion, you have to take 3.20 and put the fraction

$$\frac{P_{\text{gen}}[x^{(\text{n})}(\mathcal{T}) \to x^{(\text{o})}(\mathcal{T})]}{P_{\text{gen}}[x^{(\text{o})}(\mathcal{T}) \to x^{(\text{n})}(\mathcal{T})]} \tag{3.21}$$

into the second argument of the minimum function in Eqn. (3.18). As shown in more detail in [9], you can make considerable simplifications if you assume that the dynamics of the system conserve a stationary distribution and the distribution of initial states is an equilibrium distribution, identical to the stationary distribution conserved by the dynamics. Further assuming a symmetric generation probability for the shooting point, one obtains in the end

$$P_{\text{acc}}[x^{(\text{o})}(\mathcal{T}) \to x^{(\text{n})}(\mathcal{T})] = h_A[x^{(\text{n})}_0] h_B[x^{(\text{n})}_{\mathcal{T}}] \min\left[1, \frac{\rho(x^{(\text{n})}_{t'})}{\rho(x^{(\text{o})}_{t'})}\right]. \tag{3.22}$$

As already mentioned, in the case of stochastic dynamics, the shooting point need not be modified, thus $x^{(\text{n})}_{t'} = x^{(\text{o})}_{t'}$ and therefore $\rho(x^{(\text{n})}_{t'}) = \rho(x^{(\text{o})}_{t'})$. This leads to a further simplification:

$$P_{\text{acc}}[x^{(\text{o})}(\mathcal{T}) \to x^{(\text{n})}(\mathcal{T})] = h_A[x^{(\text{n})}_0] h_B[x^{(\text{n})}_{\mathcal{T}}]. \tag{3.23}$$

In other words, if you perform your simulation according to the dynamics given by, say, the Metropolis Monte Carlo algorithm, the only thing you have to worry about concerning acceptance is the obvious one: deciding whether the trial path starts in A and ends in B. Furthermore, it is possible to grow and check only either the forward or backward segment separately, thus saving computational cost.

3.2.4. Shifting moves

A *shifting* move is in a certain sense orthogonal to a shooting move. Instead of regrowing the path from some shooting point, in a shifting move the existing path is shifted a certain amount δt forward or backward in time. Afterwards, the then missing fragment is regrown according to the underlying dynamics. A schematic picture of such a move can be seen in figure 3.3.

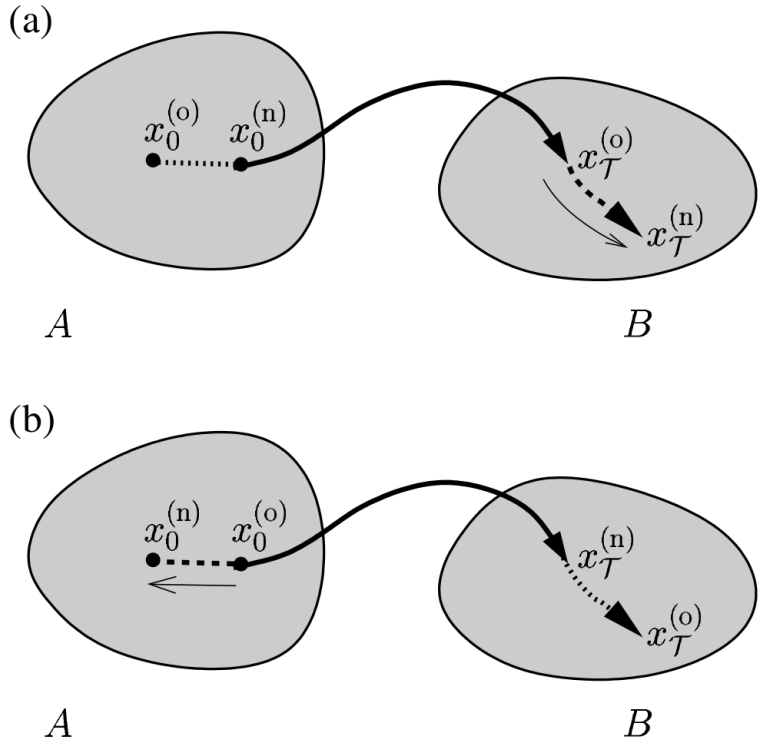

Figure 3.3.: Schematic picture of forward (a) and backward (b) shifting moves. Image taken from Ref. [9].

In the case of a forward shifting move, the time slices from $t' = 0$ to $t' = \mathcal{T} - \delta t$ are identical to a part of the old path [9]:

$$x_{i\Delta t}^{(n)} = x_{i\Delta t+\delta t}^{(o)} \quad \text{for} \quad i = 0, \ldots, (\mathcal{T} - \delta t)/\Delta t. \tag{3.24}$$

Hence, the generation probability for the forward shifting move does only depend on the product of the single time step transition probablities for the new segment of the path:

$$P_{\text{gen}}^{\text{f}}[x^{(o)}(\mathcal{T}) \to x^{(n)}(\mathcal{T})] = \prod_{i=(\mathcal{T}-\delta t)/\Delta t}^{\mathcal{T}/\Delta t-1} p(x_{i\Delta t}^{(n)} \to x_{(i+1)\Delta t}^{(n)}). \tag{3.25}$$

An analogous expression can be given for a backward shifting move. Again, in the end we are interested in an acceptance criterion for both sorts of shifting moves. For that criterion, we require the shifting length δt (which should be an integer multiple of Δt) to be drawn from the same distribution for forward and backward shifting moves. Furthermore, some additional restrictions are applied. They are the same as already known from the discussion of the shooting moves. In particular, it is assumed that the underlying dynamics conserve a stationary distribution and the initial conditions of the system are drawn from the same distribution. With all

those requirements, it is possible to simplify the acceptance probability for forward shifting moves to [9]

$$P_{\mathrm{acc}}^{\mathrm{f}}[x^{(\mathrm{o})}(\mathcal{T}) \to x^{(\mathrm{n})}(\mathcal{T})] = h_A[x_0^{(\mathrm{n})}]h_B[x_{\mathcal{T}}^{(\mathrm{n})}]. \tag{3.26}$$

The same acceptance criterion is valid for backward shifting moves. That means, a shifted path can be accepted whenever it is reactive, i. e. whenever it starts in region A and ends in region B.

3.2.5. Committor analysis and transition state ensemble

Transition path sampling is able to sample otherwise practically non-observable reactive trajectories, taking a system from one (meta-) stable state to another. As a consequence, a typical transition path is expected to consist of many unlikely intermediate states of the system, somewhere in the transition region between A and B. To quantify these states, it is useful to introduce the *committor* of a given state outside A or B. Formally, it is defined as [8]

$$p_B(r,t) \coloneqq \frac{\int \mathcal{D}x(t)\mathcal{P}[x(t)]\delta(r-x_0)h_B(x_t)}{\int \mathcal{D}x(t)\mathcal{P}[x(t)]\delta(r-x_0)}. \tag{3.27}$$

This means that the committor p_B is nothing but the fraction of pathways starting at the configuration r and ending up in region B after a time t, with the committor p_A defined accordingly for region A. Thus, p_B is a statistical value of how *committed* a given initial state is to finally evolve into region B. Accordingly, a committor value of $p_B \approx 0$ means that the system will evolve to region A most of the time, while $p_B \approx 1$ implies the same for region B. In this definition, the committor also depends on time. When actually calculating committor values, this time should be much larger than the typical relaxation time of the system, such that the committor becomes practically time-independent.

However, in any real-world simulation, you can only calculate a finite number of trajectories, check if the end configuration is in B, and thus obtain an approximation for the committor:

$$p_B(r,t) \approx \frac{1}{N}\sum_{i=1}^{N} h_B(x_t^{(i)}) =: p_B^{(N)}(r,t) \tag{3.28}$$

If you now assume that these N trajectories are statistically independent, for large N this $p_B^{(N)}(r,t)$ is a random Gauß variable with a standard deviation of

$$\sigma = \sqrt{\frac{p_B(1-p_B)}{N}}. \tag{3.29}$$

This implies that for reaching a given desired accuracy, the largest number of trajectories is needed for states with $p_B \approx 1/2$. These are, however, the most

interesting states – the transition states of the system. Formally, such a transition state is defined as having equal probability for becoming A or B:

$$p_A = p_B. \tag{3.30}$$

If you then further assume that $p_A + p_B = 1$ (in other words, that there are no other basins of attraction except A and B), it follows immediately that a transition state is characterized by $p_B = 1/2$. The set containing all transition states of the system is called the *transition state ensemble* (TSE).

Now, in order to calculate committor values and find transition states with $p_B = 1/2$, you should take configurations out of your sampled pathways as initial configurations. Launch a minimal number of $N_{\min}$ trajectories from each of those configurations. Then, estimate each committor value by equation 3.28 and its statistical error by

$$\sigma^{(N)} = \sqrt{\frac{p_B^{(N)}(1 - p_B^{(N)})}{N}}. \tag{3.31}$$

Continue generating trajectories until the desired accuracy is reached. Now, the configuration is excluded from the transition state ensemble if 0.5 does not lie in the interval $[p_B^{(N)} - \alpha\sigma^{(N)}, p_B^{(N)} + \alpha\sigma^{(N)}]$. Otherwise, the generation of additional trajectories is continued until a maximal number of $N_{\max}$ is reached. The configuration is considered to be part of the TSE if $1/2$ is still within the confidence interval after the $N_{\max}$ iterations. The value of α depends on the desired confidence level; to give an example, a value of $\alpha = 2$ is needed for a confidence level of 95% [9]. That means, with that value the probability that the real committor value is in the interval $[p_B^{(N)} - 2\sigma^{(N)}, p_B^{(N)} + 2\sigma^{(N)}]$, is 95%.

It is also possible to check if $1/2$ is within the confidence interval directly after the $N_{\min}$ trajectories have been calculated, as done for example in Ref. [10]. That will save some computational cost, but at the expense of a lower accuracy for the states that are not members of the TSE.

3.3. Structural analysis

In this section, we will cover the methods to analyze the structural properties of simulated molecular systems, in particular the polymer chain. However, the bond order parameters and connection coefficients discussed in the following are applicable to a wide range of physical systems.

3.3.1. Pair correlation function

The pair correlation function $g(r)$, also known as radial distribution function, describes the mean density at a distance r around a single particle. Formally, it is defined as [11]

$$g(r) = \frac{1}{4\pi r^2 \rho N} \left\langle \sum_{i \neq j} \delta(r - r_{ij}) \right\rangle, \tag{3.32}$$

where N is the number of particles in the system and ρ the average global (number) density. More accurately, we can think of $g(r)$ as the local density around a particle, compared to the average particle density in the system under the assumption of a homogeneous distribution. The pair correlation function is important for several reasons. First of all, it allows an insight into the structure in the vicinity of a particle. Furthermore, for systems with additive pair interactions, the thermodynamic properties are fully determined by $g(r)$. For very long distances, the (undirected) structural correlation between particles vanishes. Therefore, we have

$$\lim_{r\to\infty} g(r) = 1, \tag{3.33}$$

since the density in a certain distance around any particle approaches the average density of the system as this distance approaches infinity.

In the case of a single, isolated system like the polymer chain, it is difficult to define the average particle density ρ. Nevertheless, it makes sense to calculate $g(r)$, even though it is neither normalized for such a system nor is $\lim_{r\to\infty} g(r) = 1$ true anymore. Still, you can obtain useful structural information about your system from the analysis of $g(r)$. For example, the first maximum in $g(r)$ gives the average distance between next neighbors.

3.3.2. Steinhardt bond order parameters

The orientational order parameters described here were first introduced in Ref. [12]. They provide a way of identifying the local structure around a particle in a molecular system. In principle, they do not only allow us to distinguish between liquid and solid structure, but can also classify the type of solid structure.

First, define the complex quantity $q_{lm}(i)$ of particle i as

$$q_{lm}(i) = \frac{1}{N_b(i)} \sum_{j=1}^{N_b(i)} Y_{lm}(\vec{r}_i - \vec{r}_j), \tag{3.34}$$

where the Y_{lm} are the spherical harmonics. The index l is a non-negative integer, and for the index m we have $m = -l, -l+1, \ldots, 0, \ldots, l$. The sum runs over all nearest neighbors of particle i, and hence $N_b(i)$ is the number of these nearest neighbors. Now, based on the q_{lm}, you can define the local bond order parameters, or Steinhardt order parameters, as

$$q_l(i) = \sqrt{\frac{4\pi}{2l+1} \sum_{m=-l}^{l} |q_{lm}(i)|^2}. \tag{3.35}$$

These parameters are sensitive to different crystal structures. Especially q_4 and q_6 haven proven to be useful in distinguishing between cubic and hexagonal structures. One way to obtain such structural information is to plot the parameters for all the particles in the q_4-q_6 plane. Then, depending on the structure of the system, the

parameters will tend to cluster in certain regions of that plane. However, as shown in Ref. [13], thermal fluctuations tend to smear out the parameter distributions considerably, thus making it very hard to distinguish between, say, BCC and HCP structures. For that reason, Lechner and Dellago have introduced an averaged form of the local bond order parameters:

$$\bar{q}_l(i) = \sqrt{\frac{4\pi}{2l+1} \sum_{m=-l}^{l} |\bar{q}_{lm}(i)|^2}, \tag{3.36}$$

where

$$\bar{q}_{lm}(i) = \frac{1}{N_b(i)+1} \sum_{\tilde{N}_b(i)} q_{lm}(k) \tag{3.37}$$

are the locally averaged q_{lm}. The sum, indicated by $\tilde{N}_b(i)$, runs over all neighbors of particle i plus the particle i itself. As clearly visible in Fig. 3.4, using that approach offers a much better distinction between different crystal structures in thermally fluctuating systems.

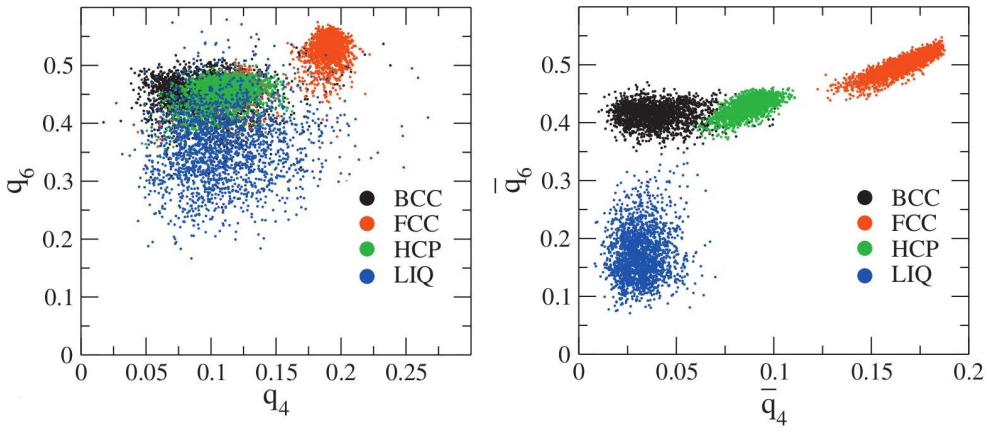

Figure 3.4.: q_4-q_6 plane and $\bar{q}_4$-$\bar{q}_6$ plane for a Lennard-Jones system in three different crystal structures and in the liquid phase. The averaged form of the local bond order parameters allows a much better distinction between the different structures. Image taken from Ref. [13].

Apart from the local parameters described so far, it is also possible to define global bond order parameters of the form [14]

$$Q_l = \left(\frac{4\pi}{2l+1} \sum_{m=-l}^{l} |Q_{lm}|^2 \right)^{1/2}, \tag{3.38}$$

where the Q_{lm} are an averaged version of the $q_{lm}(i)$:

$$Q_{lm} = \frac{\sum_{i=1}^{N} N_b(i) q_{lm}(i)}{\sum_{i=1}^{N} N_b(i)}. \tag{3.39}$$

Again, $N_b(i)$ denotes the number of nearest neighbors of particle i.

3.3.3. Connection coefficients

The connection coefficient d_{ij} between particle i and particle j is defined as the normalized scalar product [15]

$$d_{ij} = \frac{\sum_{m=-6}^{6} q_{6m}(i) q_{6m}^*(j)}{\left(\sum_{m=-6}^{6} |q_{6m}(i)|^2\right)^{1/2} \left(\sum_{m=-6}^{6} |q_{6m}(j)|^2\right)^{1/2}}. \tag{3.40}$$

It can be shown that $d_{ij} = d_{ij}^* = d_{ji}$ and therefore $d_{ij} \in \mathbb{R}$. Furthermore, we have $-1 \leq d_{ij} \leq 1$. What makes the d_{ij} important is that their distribution in a given system strongly depends on the structure of that system. In liquids, the d_{ij} tend to be distributed around 0, while for solids the distribution gets narrower and is shifted towards 1. Two particles i and j can thus be defined to be *connected* when d_{ij} exceeds a certain threshold. In subsequent simulations, 0.5 was chosen as that threshold. In Chapter 6, an example for the usefulness of the connection coefficients will be given, comparing the expanded coil state with the frozen crystallite. Then, a particle can be defined as solid-like if its number of connections exceeds another threshold, say, 7. However, in the case of a system without periodic boundary conditions and thus a non-negligible surface, many particles on the surface will not be recognized as solid-like, since their number of connections is too low, simply because they are surface particles. A refined approach to work around this problem is to make the threshold dependent on the number of next neighbors of the particle. In other words, a particle is considered as solid-like if it has a minimum number of next neighbors (e. g. 4) and is connected to "almost all" of these neighbors, or being more precise, to at least all but one of its neighbors to give some freedom for thermal fluctuations. That way, on the one hand, surface particles can also be detected as solid-like. On the other hand, the dependence of the threshold on the number of next neighbors makes sure that particles deep within the system are not misdetected as solid-like when they are in fact still in a liquid phase, as you would have to expect if you just lowered the threshold globally.

4. Implementation

This chapter deals with the actual implementation of the simulations used to study the polymer chain. First, I will describe the plain Metropolis Monte Carlo simulation. In the second part, we will continue with the TPS simulation based on that.

4.1. Metropolis Monte Carlo simulation

The first step in the actual implementation of a working simulation was the creation of a standard Metropolis Monte Carlo program, written in Fortran 90. My simulation consists of a main program and various auxiliary tools. The scheme how they work together is based on similar programs by Martin Neumann. The programs are the following:

- *mkpolychain* is used to prepare the system, specifying all necessary parameters such as N, T and λ. Furthermore, the technical parameters controlling how many iterations are done have to be entered. By default, the program creates a random, unordered open chain configuration, however, it is also possible to read the initial positions of the chain monomers from an ASCII file. All the information is stored in a binary input file.

- *polychain* is the actual simulation program. It reads the input file, performs the Metropolis algorithm and calculates averages of different quantities of interest, such as the potential energy of the system. Afterwards, the final state of the system and the accumulated averages are written to a binary output file. This output file can be read again by the program, thus directly continuing the simulation. In that way, it is possible to split up long simulation runs into small parts and save all the intermediate results.

- *apolychain* is used to create human-readable output of the results after a simulation. This program reads the output file of the main program, displays all relevant parameters and averages on the screen, and can also create additional PDB files of the system in order to be visualized with other tools.

- *zpolychain* is another auxiliary program designed to read an output file, reset all averages, and write a new input file. It can be used after an initial equilibration procedure of the simulated polymer chain, thus starting the actual simulation run from a more natural state of the system.

The main simulation in the *polychain* program runs according to a simple scheme. The main loop in the code runs a predefined number of *passes*. One pass consists of $N \times n_{\text{tskip}}$ single-particle moves, where n_{tskip} is some integer. After every pass, all the averages are calculated and accumulated. A single particle move is either one of three distinct possibilities, selected at random. These three moves are

1. crankshaft moves, where a random monomer within the chain is rotated through a small angle about the axis defined by the line connecting the two neighboring particles;

2. end moves, where the 1st or N-th monomer is rotated a small angle about a random axis;

3. pivot moves, where a monomer i within the chain is again selected randomly and the whole chain segment $[i+1, N]$ undergoes a random Euler rotation about the i-th monomer.

For the pivot moves, it might look strange at first glance not to implement the "reverse" move, i. e. rotating the segments $[1, i-1]$, and holding the other part fixed. However, that would be completely equivalent, since only the relative positions of the monomers to each other are relevant, and not their absolute positions in some arbitrary frame of reference. Also for that reason, the whole chain is shifted such that the 1st monomer coincides with the coordinate origin again before the main program writes its output file.

On average, of N single-particle moves, one is a pivot move, two are end moves, and the remaining $N-3$ are crankshaft moves.

With the described set of programs, it is already possible to calculate many equilibrium properties of the polymer chain. However, to get a grasp of the actual transition mechanisms taking the system from the expanded coil to the frozen crystallite state, such a simulation is not enough. For that task, we need transition path sampling.

4.2. Transition path sampling

For the TPS simulation, some additional programs had to be written. They use the same subroutines as the plain MMC simulation for the dynamical evolution of the system. However, before actually starting a TPS run, you need some initial path, which is in general somewhat tricky to find. In the case of my simulations, due to the stochastic nature of Monte Carlo, a simple trial-and-error procedure worked well for that task. I started with a $N = 128$ configuration at the coexistence curve ($T = 0.424$ for $\lambda = 1.04$) in the crystallite state, obtained from the standard simulation. This configuration was melted by a massive increase in the temperature of the system, while saving the intermediate configurations at regular MC time intervals. Afterwards, these intermediate states were taken as a basis for many parallel evolutions of the system at the original temperature of $T = 0.424$. By

an observation of the MC time development of the potential energies for all those pathways, I was able to select "promising" trajectory parts. One of those paths was further split up by simply changing the state of the random number generator and then starting parallel MC evolutions. Then, some of those trajectories froze again to the crystallite, while others continued to melt. By further splitting up the "fastest" up and down going trajectories by a change of the random number generator's state, I was able to generate a sufficiently short transition path in the end. Of course, it was also necessary to reverse the "up" (i. e. to the coil state) going trajectory before attaching the "down" going part to it. Fortunately, that is no problem at all since there is no time direction in a Monte Carlo simulation.

Having found an initial path (corresponding to an initial system configuration in the plain simulation), the scheme of the TPS is very similar to that of the normal Monte Carlo program. Again, my simulation actually consists of several smaller programs, namely the following:

- *tpspolychain* reads an initial path (which is again stored in a simple binary input file) and performs the TPS. The final path and the accumulated averages of some quantities are written to a path output file.

- *pathloader* is a multi-purpose program to obtain human-readable results from a path file, such as the development of physical parameters along a given path. In addition to that, the program can do certain modifications to a path, such as reversing or shortening it.

- *selectstates* reads a path file and randomly selects states which are neither A (coil) nor B (crystallite) for committor analysis. Those states are written to standard binary files as used by the main program *polychain*.

- *committorana* is the program performing the committor analysis, as described in chapter 3.2.5 (page 21). It reads one of the binary files produced by *selectstates* and writes the configuration and the results of the committor calculation to another output file of the same format.

Apart from the described programs, there are also some additional smaller ones written for special purposes, especially the creation of an initial path. However, those are not needed once such a path is found. Furthermore, the program *faketps* simulates the procedure of a TPS simulation without actually performing any underlying dynamics, in order to gain insight into the efficiency of a simulation. We will focus on that aspect in the next chapter.

The first simulated transition pathways for the $N = 128$ chain have a length of 40,000 passes, with $n_{\mathrm{tskip}} = 50$, thus leading to a total of $128 \times 40,000 \times 50 = 256,000,000$ single-particle moves per trajectory. On the machines used for my calculations, it takes a CPU time of roughly 10 minutes for these moves to run. In a path file, the system configuration is saved whenever a pass is completed, that means there are 40,000 configurations in one path.

5. Path Sampling Statistics

One of the crucial properties of a TPS simulation is its efficiency in generating new transition pathways. After all, in the end you are interested in sampling as many trajectories as possible in a given amount of CPU time. Therefore, this chapter deals with the efficiency of a TPS simulation in finding new transition pathways. For that, we will also introduce a simple model of a TPS simulation and compare it with real simulation results.

5.1. The path survival function

Concerning the efficiency of the sampling of new transition pathways, we first need to introduce a function that returns the fraction of identical configurations in two paths:

$$f[x^{(\mathrm{k})}(\mathcal{T}), x^{(\mathrm{m})}(\mathcal{T})] \coloneqq \frac{1}{\mathcal{T}/\Delta t + 1} \sum_{i=0}^{\mathcal{T}/\Delta t} \delta(x^{(\mathrm{k})}_{i\Delta t}, x^{(\mathrm{m})}_{i\Delta t}), \tag{5.1}$$

with the Kronecker delta

$$\delta(x_i, x_j) = \begin{cases} 1 & \text{if} \quad x_i = x_j, \\ 0 & \text{if} \quad x_i \neq x_j. \end{cases} \tag{5.2}$$

Now, we are able to define some sort of time correlation function:

$$c(k) \coloneqq \left\langle f[x^{(\mathrm{m})}(\mathcal{T}), x^{(\mathrm{m+k})}(\mathcal{T})] \right\rangle, \tag{5.3}$$

where $x^{(\mathrm{m+k})}(\mathcal{T})$ is the path that evolved from the path $x^{(\mathrm{m})}(\mathcal{T})$ after k TPS moves. Now suppose only shooting moves are performed in the TPS, but no shifting moves. Then $c(k)$ is exactly the function we are looking for: It gives us the average fraction of a path that was already part of that path k TPS iterations ago, or, in other words, the surviving fraction of an initial part after k iterations. Therefore, we could also think of $c(k)$ as the *path survival function*. This function should help us to determine the average time until a completely new path is accepted. Naturally, we expect $c(k)$ to be near 1 for very small k, and that it should practically vanish for very big k. In the simulation, the path survival function is calculated directly by implementing Eqn. (5.3). Fig. 5.1 gives an example for $c(k)$ computed for a real TPS run.

The choice of the optimal path length in a TPS simulation is a somewhat tricky task. On the one hand, it should not be too long, to ensure that the CPU time

required for one single path remains feasible. On the other hand, a too short path length is also not optimal, since it will lower the acceptance rate of trial moves considerably. Thus, in the end the mean time between the generation of completely new pathways might even increase instead of getting shorter.

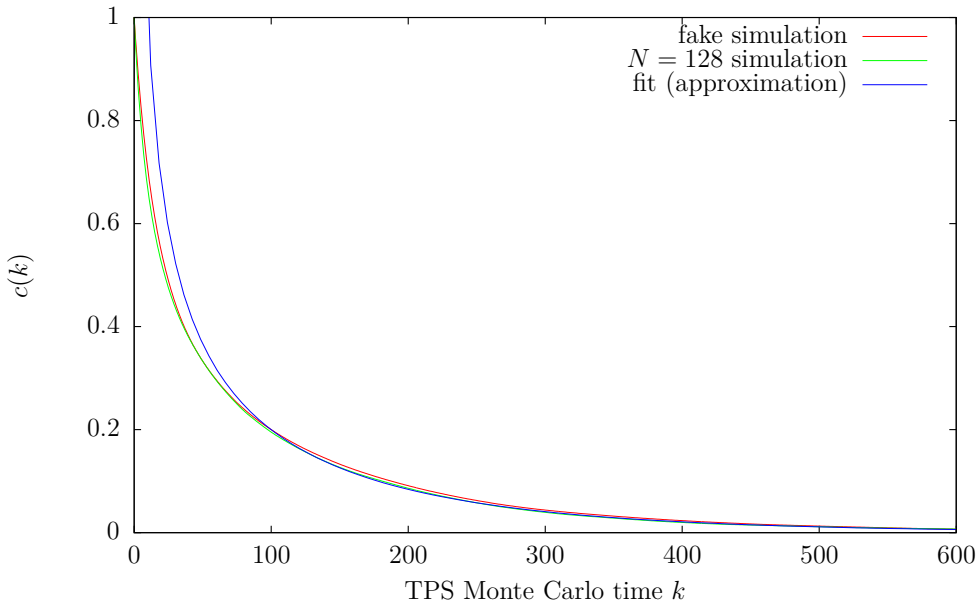

Figure 5.1.: Path survival function $c(k)$ for a typical TPS run with a path length of 40,000 jobs, giving the average fraction of the configurations in the path still present after k TPS iterations. In the case of the "fake" simulation, which is shown for comparison, after selecting the shooting point, the acceptance was simply drawn from a distribution, given in Fig. 5.3. The fit of the simulation data is according to Eqn. (5.15).

5.2. Simulating the simulation

One could even think of performing some sort of "fake" TPS simulation in order to test the behavior of $c(k)$ for different scenarios. It turns out that a very crude acceptance model is able to reproduce the real $c(k)$ satisfactorily, as seen in Fig. 5.1. First, as in a real TPS, a random time slice within the "path" is selected, along with a shooting direction. Then, the remaining path length to A (backward shooting) respectively B (forward shooting) is determined. The trial move is then accepted with a certain probability yet to be found. Surely, this probability will depend on the length of the remaining path fraction.

5.3. Analytic calculation of the path survival function

Let us define the function $q(t)$ as the probability that a time slice at time t is changed in the next TPS move, assuming that there are only shooting moves. In that case, the function is given as

$$q(t) = \frac{1}{2}\frac{1}{\mathcal{T}}\int_0^t dt' p_B(t') + \frac{1}{2}\frac{1}{\mathcal{T}}\int_t^{\mathcal{T}} dt' p_A(t'), \tag{5.4}$$

where $p_B(t')$ is the committor of the state at time slice t' in the path, and $p_A(t')$ is defined accordingly. The function $q(t)$ is a sum of two terms: The first term is the probability that a forward shooting move started somewhere before the time t gets accepted and thus the time slice at t is changed. In addition to that, the second term gives the same probability for a backward shooting move started somewhere after t. The factor $1/2$ comes from the fact that you have equal probabilities for forward and backward shooting moves. The second factor, $1/\mathcal{T}$, takes the initial choosing probability for the shooting point into account, which is equally distributed along the path.

In the TPS simulation program, an approximation for $q(t)$ is calculated in two different ways. First, we use a straightforward approach where we simply count how often each slice in the path is updated. Secondly, we implement Eqn. (5.4) numerically, assuming an equal probability for forward and backward shooting moves:

$$q(t) \approx \frac{N_{\mathrm{f,B}}(t)}{N_{\max}} + \frac{N_{\mathrm{b,A}}(t)}{N_{\max}}. \tag{5.5}$$

Here, $N_{\max}$ stands for the total number of forward and backward shooting moves so far. $N_{\mathrm{f,B}}(t)$ is the number of successful forward shooting moves (ending in B), started between 0 and t. $N_{\mathrm{b,A}}(t)$ is defined accordingly as the number of successful backward shooting moves, started between t and $\mathcal{T}$. Basically, this is just a coarse-grained version of the direct approach, so we expect that the two different versions agree very well, and as seen in Fig. 5.2, they do.

Furthermore, we calculate approximations for $p_A(t)$ and $p_B(t)$, denoted by $f_A(t)$ and $f_B(t)$, by bunching a number of adjacent slices together, and count how often a shooting move started within that interval succeeds. We will later see that these approximations prove to be useful for developing an accurate model of a TPS simulation.

If $q(t)$ is the probability that the time slice at t gets updated in one TPS iteration, $1 - q(t)$ is the probability that this does not happen. Accordingly, $[1 - q(t)]^k$ is the probability that the time slice remains unchanged after k TPS iterations. This sounds already very similar to the path survival function $c(k)$, which gives the average unchanged fraction of a path after k TPS iterations. However, in order to get the actual $c(k)$, we have to average $[1 - q(t)]^k$ over all possible time slices:

$$c(k) = \frac{1}{\mathcal{T}}\int_0^{\mathcal{T}} dt[1 - q(t)]^k. \tag{5.6}$$

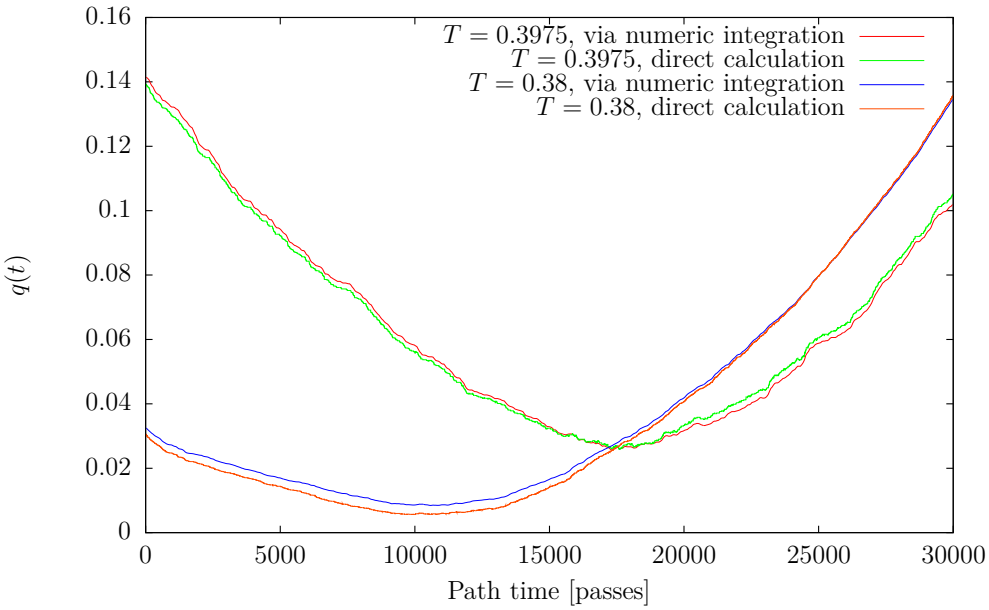

Figure 5.2.: $q(t)$ for $N = 64$ TPS simulations with two different temperatures. The function was calculated both directly (green, orange) as well as via implementing Eqn. (5.4) numerically (red, blue).

Now, let us rewrite that as

$$c(k) = \frac{1}{\mathcal{T}} \int_0^{\mathcal{T}} dt \exp\{\ln[1 - q(t)]^k\} \tag{5.7}$$

$$= \frac{1}{\mathcal{T}} \int_0^{\mathcal{T}} dt \exp\{k \ln[1 - q(t)]\}. \tag{5.8}$$

It is reasonable to assume that $1 - q(t)$ is a function which has a unique maximum around some value t_0 within the path. Therefore, we can do a Taylor expansion of the logarithm around that value:

$$\ln[1 - q(t)] = \ln[1 - q(t_0)] - a(t - t_0)^2 + O(t^3), \tag{5.9}$$

where the first term in the expansion as well as a are constants. Furthermore, the first-order term vanishes since we have assumed that t_0 is the position of the maximum of $1 - q(t)$. Now, we can use the Taylor expansion in the calculation of $c(k)$:

$$c(k) \approx \frac{1}{\mathcal{T}} \int_0^{\mathcal{T}} dt \exp\{k \underbrace{\ln[1 - q(t_0)]}_{=:-b} - ka(t - t_0)^2\} \tag{5.10}$$

$$= \frac{1}{\mathcal{T}} \int_0^{\mathcal{T}} dt \exp\{-bk\} \exp\{-ka(t - t_0)^2\} \tag{5.11}$$

Effectively, we have approximated the original function $[1 - q(t)]^k$ with an attenuation factor e^{-bk} times a Gaussian function $e^{-ka(t-t_0)^2}$. The first factor in the

integral is independent of t. If furthermore k is big enough, we can assume that the Gaussian function is essentially peaked around t_0 and practically vanishes for bigger distances to t_0. In particular, it should be sufficiently small at the integration boundaries 0 and $\mathcal{T}$. In that case, we can substitute the boundaries with minus and plus infinity, thus simplifying the calculation of the Gaussian integral:

$$c(k) \approx \frac{1}{\mathcal{T}} e^{-bk} \int_{-\infty}^{\infty} dt \exp\{-ka(t-t_0)^2\} \tag{5.12}$$

$$= \frac{1}{\mathcal{T}} e^{-bk} \sqrt{\frac{\pi}{ak}}. \tag{5.13}$$

Here, we have used that

$$\int_{-\infty}^{\infty} dx e^{-(x+b)^2/c} = \sqrt{c\pi}. \tag{5.14}$$

In the end, we get a very simple result for $c(k)$, which is of the form

$$c(k) = A \frac{e^{-bk}}{\sqrt{k}}. \tag{5.15}$$

We can now use this function – with parameters A and b – to attempt to fit the calculated values for $c(k)$ from the real simulation as well as those from the simulated TPS run.

However, when performing a simulated TPS, the evolution of the committor values along the path is in general unknown. Therefore, one has to choose some substitute function for p_B and p_A, denoted by f_B and f_A. One possible choice is given in Fig. 5.3, where

$$f_B(t) = \begin{cases} 0 & \text{if} \quad t < 0.48\,\mathcal{T}, \\ t/\mathcal{T} - 0.48 & \text{if} \quad t \geq 0.48\,\mathcal{T}, \end{cases} \tag{5.16}$$

and

$$f_A(t) = \begin{cases} -\frac{8}{63}\frac{t}{\mathcal{T}} + 0.8 & \text{if} \quad t \leq 0.63\,\mathcal{T}, \\ 0 & \text{if} \quad t > 0.63\,\mathcal{T}. \end{cases} \tag{5.17}$$

With this model, the probability for the innermost part of the path to get updated practically vanishes. Nevertheless, this model is able to reproduce the results for $c(k)$ from the real $N = 128$ TPS simulation very well. One could thus speculate that the acceptance function for the real TPS is of a similar form. A numerical approximation of the acceptance function of the real simulation is given in Fig. 5.4.

5.4. Mean time between path updates

Another interesting quantity concerning the efficiency of a TPS simulation is the distribution of the time intervals between two updates for a given time slice. This

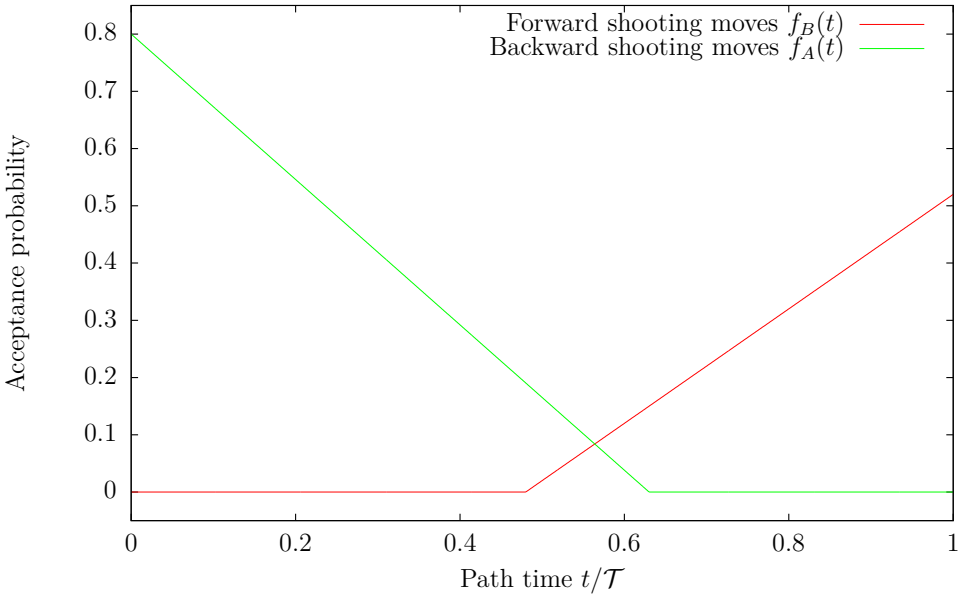

Figure 5.3.: One possible choice for the acceptance function for the simulated TPS run, which was used for the run from Fig. 5.1. This suggests that the acceptance function for the real TPS is similar to the simple model, with a rather low changing probability for the innermost part of the path.

distribution is closely related to $c(k)$. The mean TPS time $T_u(t)$ between two updates is simply the inverse of the update probability $q(t)$:

$$T_u(t) = \frac{1}{q(t)}. \tag{5.18}$$

With this definition, one can calculate a histogram $H(t)$, giving the (not yet normalized) distribution of the mean time between updates over the whole path:

$$H(T) = \int_0^{\mathcal{T}} dt\, q(t)\delta[T - T_u(t)] \tag{5.19}$$

$$= \sum_{i=1}^{n} \frac{q(t_i)}{|\overline{T}'_u(t_i)|}, \tag{5.20}$$

where according to the integration rules for the delta function, the sum runs over all zeroes t_i of

$$\overline{T}_u(t) = T - T_u(t). \tag{5.21}$$

The additional factor $q(t)$ in the integral is necessary since every slice contributes to the histogram with a frequency directly proportional to its update probability. However, having the distribution of the *mean* times between path updates is not enough. In order to obtain the full distribution of the update intervals, we need the

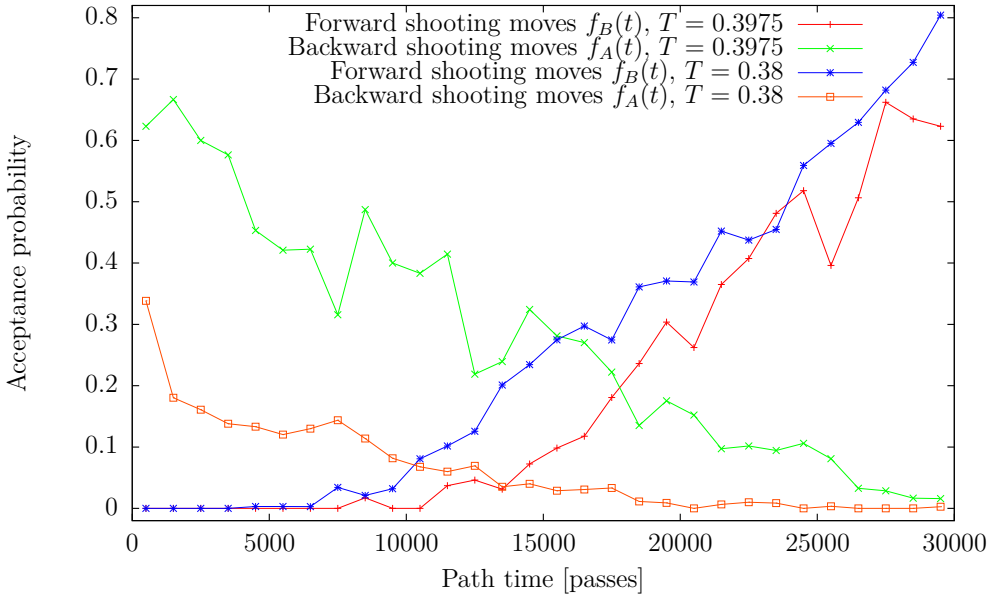

Figure 5.4.: Path acceptance as a function of time for the same two $N = 64$ TPS simulations as used for Fig. 5.2. In this calculation, 1,000 adjacent slices each were bunched together for the averaging procedure. The approximate acceptance value is then the fraction of successful shooting moves started from within these 1,000 slices.

complete information for each slice, in other words, the complete distribution of update time intervals for each slice, and not only the average of that distribution. As already known from Eqn. (5.6), the probability to observe *no* update after k TPS iterations is $[1 - q(t)]^n$. Therefore, the probability P_{k+1} to observe an update of the time slice at t after $k + 1$ TPS iterations is the product of the probability to observe no update in the k iterations before and the actual update probability for one iteration:

$$P_{k+1} = [1 - q(t)]^k q(t). \tag{5.22}$$

Therefore, it is immediately clear that

$$\frac{P_{k+1}}{P_k} = 1 - q(t), \tag{5.23}$$

which is constant for all values of k. Hence, the sequence $\{P_k\}$ is a decreasing geometrical sequence with scaling factor $1 - q(t)$. One can also rewrite Eqn. (5.22) as

$$\begin{aligned} P_{k+1} &= q(t) e^{\ln[1-q(t)]^k} && (5.24)\\ &= q(t) e^{k \ln[1-q(t)]} && (5.25)\\ &=: q(t) e^{-bk}. && (5.26) \end{aligned}$$

Thus, it is possible to determine $q(t)$ at a given time slice from an exponential fit of the distribution of update intervals for that time slice. This average is

$$\bar{k} = \sum_{k=1}^{\infty} kP_k \tag{5.27}$$

$$= \sum_{k=1}^{\infty} k(1-q)^{k-1}q \tag{5.28}$$

$$= q\sum_{k=0}^{\infty}(k+1)(1-q)^k. \tag{5.29}$$

As shown in appendix B, the value of the infinite sum is $1/q^2$. Therefore, we end up with

$$\bar{k} = \frac{1}{q}, \tag{5.30}$$

which is nothing but a discretized way of writing Eqn. (5.18) and thus exactly what we should have expected.

With the result for the distribution of update intervals as a function of the position in the path at our disposal, we are now able to calculate the full distribution of these intervals. We already know that for each time slice t, there is a distribution of update intervals associated with that slice. This distribution is given by Eqn. (5.22), in slightly different notation it is

$$P_t(k) = q(t)[1-q(t)]^{k-1}. \tag{5.31}$$

If we would like to obtain the distribution for the whole path, we simply have to average this function:

$$P_{\text{path}}(k) = \frac{1}{\mathcal{T}}\int_0^{\mathcal{T}} dt\, q(t)[1-q(t)]^{k-1}. \tag{5.32}$$

So, the final distribution can be obtained in a similar way as $c(k)$. As in the case of $c(k)$, we can apply a Taylor expansion to the integrand. Analogous to Eqn. (5.11), we can write

$$P_{\text{path}}(k) \approx \frac{1}{\mathcal{T}}\int_0^{\mathcal{T}} dt\, q(t)\exp\{-b(k-1)\}\exp\{-(k-1)a(t-t_0)^2\} \tag{5.33}$$

$$\approx \frac{1}{\mathcal{T}}\int_0^{\mathcal{T}} dt\, [q(t_0) + \frac{\ddot{q}(t_0)}{2}(t-t_0)^2]\exp\{-b(k-1)\}\exp\{-(k-1)a(t-t_0)^2\}. \tag{5.34}$$

Once again, we have used the fact that t_0 is the position of the minimum of $q(t)$, thus $\dot{q}(t_0) = 0$. If then k is big enough, we can also substitute the integration

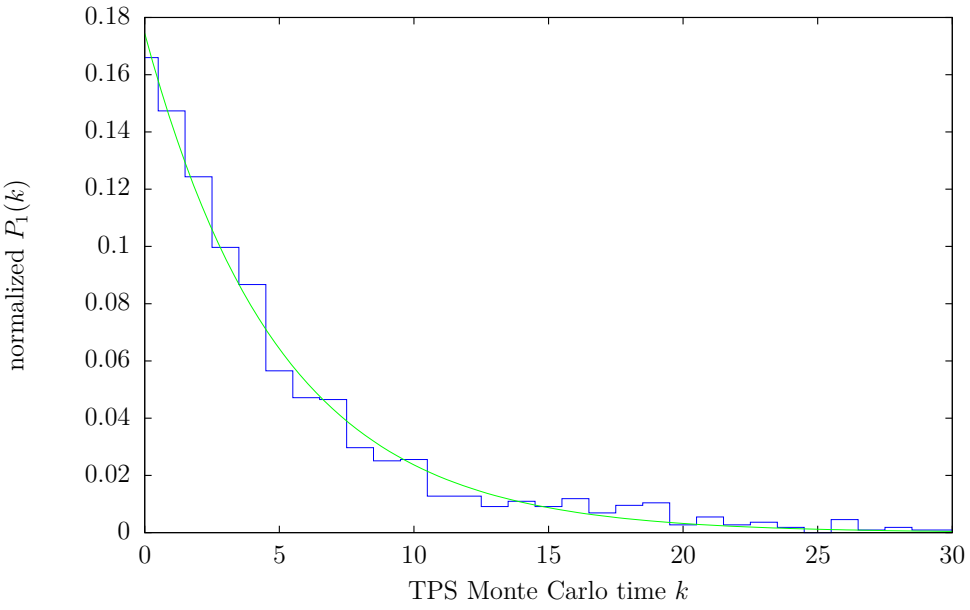

Figure 5.5.: (Normalized) $P_1(t)$, the distribution of update intervals at the beginning of the path, and fit with an exponential function according to Eqn. (5.31). Again, the same $N = 64$ TPS simulation at $T = 0.3975$ as for Fig. 5.2 was used. In order to obtain better statistics, the histogram was actually constructed with the update intervals of the first 100 time slices.

boundaries with minus and plus infinity again:

$$P_{\text{path}}(k) \approx \frac{1}{\mathcal{T}} \int_{-\infty}^{\infty} dt \, [q(t_0) + \frac{\ddot{q}(t_0)}{2}(t-t_0)^2] \exp\{-b(k-1)\} \exp\{-(k-1)a(t-t_0)^2\} \tag{5.35}$$

$$= \frac{q(t_0)}{\mathcal{T}} e^{-b(k-1)} \sqrt{\frac{\pi}{a(k-1)}} + \frac{\ddot{q}(t_0)}{4\mathcal{T}} e^{-b(k-1)} \sqrt{\frac{\pi}{a^3(k-1)^3}}. \tag{5.36}$$

The first term in the integral is simply the approximation for $c(k)$, shifted by one, times another constant. For the evaluation of the second term, we have used that

$$\int_{-\infty}^{\infty} dx x^2 e^{-x^2/c} = \frac{\sqrt{\pi c^3}}{2}. \tag{5.37}$$

In the end, we can write the distribution in the form

$$P_{\text{path}}(k) = e^{-b(k-1)} \left(\frac{A}{\sqrt{k-1}} + \frac{B}{\sqrt{(k-1)^3}} \right) \tag{5.38}$$

in order to apply another fitting procedure to the corresponding data collected in the simulation.

5.5. Tree representation of path acceptance

Another form of visualizing the progress of a TPS simulation (with at least some amount of stochastic dynamics and shooting moves only) is to draw a tree diagram as already done by Juraszek and Bolhuis [16]. In such a tree representation, the x-axis of the diagram corresponds to the time of the transition pathways, while the progress of the TPS iterations is shown on the y-axis, running from top to bottom. A horizontal line is drawn whenever a trial path, or, more precisely, a trial partial path, is accepted. The corresponding shooting point is indicated by a vertical line. A path that is completely independent from the previous path is obtained whenever a forward shot is accepted that starts from a point from within a previously successful backward shooting move, or vice versa. In the diagram, such an event is visualized by an overlap between the respective forward and backward shooting moves. An example for a tree representation is given in Fig. 5.6.

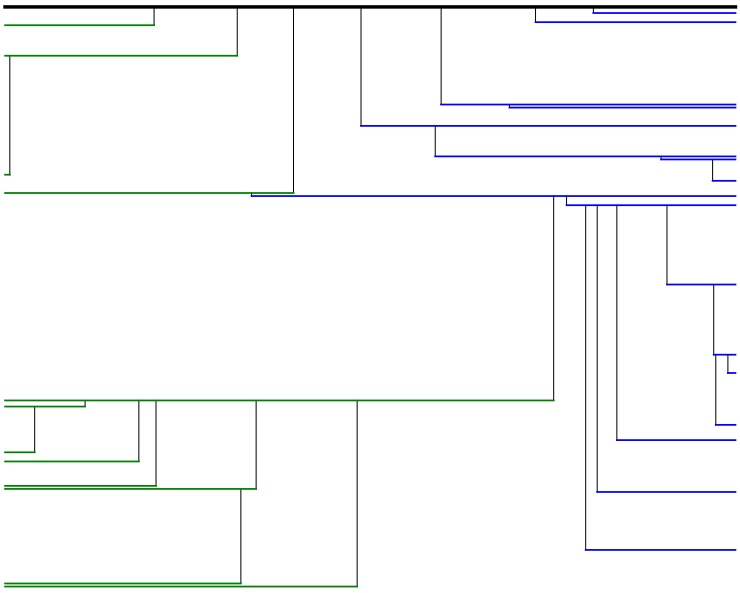

Figure 5.6.: Tree representation of a typical TPS simulation for the $N = 64$ chain with $\lambda = 1.04$ and $T = 0.38$. Successful forward shooting moves are drawn in blue, backward shooting moves in green. Shooting points are indicated by vertical lines. As indicated by the overlap between green and blue lines, an independent path is generated in the lower third of the tree. The diagram covers about 200 TPS iterations.

It is clear that for any simulation run, the tree representation of the path history holds the full information required to calculate the path survival function $c(k)$ and

all the other related quantities discussed in this chapter. For example, the function $q(t)$ can be viewed as the density of tree lines along the path time axis.

5.6. Calculation for a simple acceptance model and comparison with a simulated TPS run

Here, a practical example for the previous calculations should be performed and also checked against a corresponding simulated TPS run. First, we need two acceptance functions for forward and backward shooting moves. For simplicity, let $\mathcal{T} = 1$. The acceptance function we choose for forward shooting moves is

$$f_B(t) = \frac{t}{2}, \tag{5.39}$$

and the symmetrically defined function for backward shooting moves is

$$f_A(t) = -\frac{t}{2} + \frac{1}{2}. \tag{5.40}$$

We can now calculate

$$q(t) = \frac{1}{2}\int_0^t dt' f_B(t') + \frac{1}{2}\int_1^t dt' f_A(t') \tag{5.41}$$

$$= \frac{t^2}{4} - \frac{t}{4} + \frac{1}{8}. \tag{5.42}$$

From this, we can calculate the histogram $H(T)$ of the mean times between path updates. However, since $q(t)$ is symmetric about $1/2$, we can simplify our calculation by integrating only from 0 to $1/2$:

$$H(T) = 2\int_0^{1/2} dt\, q(t)\delta\left[T - \frac{1}{q(t)}\right]. \tag{5.43}$$

For that, we need the roots of the argument of the delta function:

$$T - \frac{1}{\frac{t^2}{4} - \frac{t}{4} + \frac{1}{8}} \stackrel{!}{=} 0. \tag{5.44}$$

After a bit of algebra and solving a quadratic equation, we get

$$t_{1,2} = \frac{1}{2} \pm \sqrt{\frac{4}{T} - \frac{1}{4}}. \tag{5.45}$$

We note that the roots are only real and within $(0, 1)$ if $8 \leq T \leq 16$, which makes perfect sense, since this is the codomain of $1/q(t)$. Furthermore, we can exclude t_1,

as it is greater than 1/2. We also need the derivative of Eqn. (5.44) with respect to t, which turns out to be

$$\frac{d}{dt}\left(T - \frac{1}{q(t)}\right) = q(t)^{-2}\left(\frac{t}{2} - \frac{1}{4}\right). \tag{5.46}$$

Finally, using

$$q[t_2(T)] = \frac{1}{T} \tag{5.47}$$

and doing some additional algebra, we end up with the distribution function

$$H(T) = \frac{8}{T^3\sqrt{\frac{16}{T} - 1}}, \tag{5.48}$$

which is, as already mentioned, only defined for $8 \leq T \leq 16$. A comparison of this distribution function with the result from a "fake" TPS run with the same acceptance model shows very good agreement (Fig. 5.7).

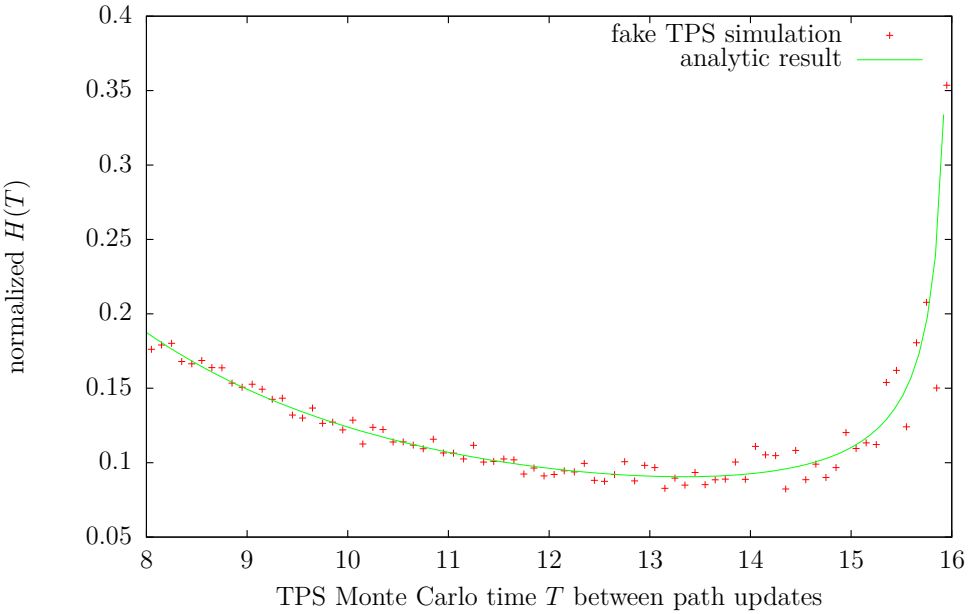

Figure 5.7.: Exact result for $H(T)$, the distribution function of the average times between path updates, for our simple path acceptance model (Eqn. (5.48)) and the result from a "fake" TPS run with the same acceptance function as in the analytic calculation. Both curves are normalized for comparison.

In order to calculate the path survival function

$$c(k) = \int_0^1 dt[1 - q(t)]^k, \tag{5.49}$$

we can either use the above approximations or simply plug the integral into a computer algebra system like *Mathematica*. If we do the latter, we end up with

$$c(k) = \left(\frac{15}{16}\right)^k {}_2F_1\left[\frac{1}{2}, -k; \frac{3}{2}; \frac{1}{15}\right], \tag{5.50}$$

where ${}_2F_1(a, b; c; z)$ is the hypergeometric function. The exact, the numerical and the approximate result are shown for comparison in Fig. 5.8. As expected, the exact and the numerical result from the simulated TPS are practically identical. The approximation is only good for values of k of about 20 or bigger. For smaller values, the function $[1 - q(t)]^k$ is not small enough at the integration boundaries 0 and 1 of Eqn. (5.49), thus the step to substitute them by minus and plus infinity is not really justified.

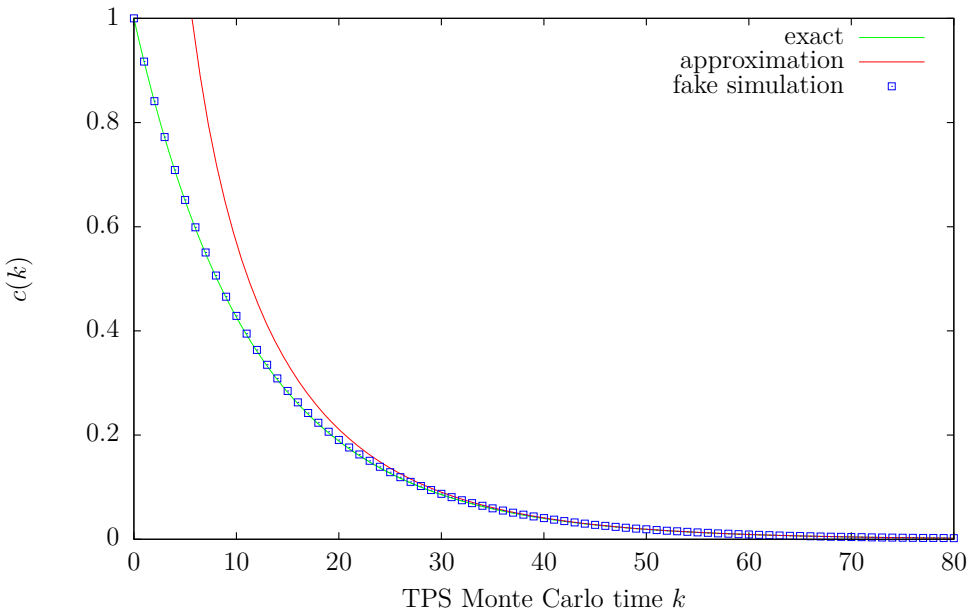

Figure 5.8.: Comparison between the exact result for $c(k)$, the result from the "fake" TPS simulation and the approximation for our simple path acceptance model. Clearly, the approximation is only valid for bigger values of k. In the case of the TPS, a total of $50,000$ TPS Monte Carlo steps were simulated.

The other alternative is to use the approximate form of $c(k)$ as given in Eqn. (5.13).

For that, we first need the values of all the constants in the Taylor approximation:

$$q\left(\frac{1}{2}\right) = \frac{1}{16}, \tag{5.51}$$

$$\ddot{q}\left(\frac{1}{2}\right) = \frac{1}{2}, \tag{5.52}$$

$$a = -\frac{1}{2}\frac{d^2}{dt^2}\ln[1-q(t)]\Big|_{t=1/2} = \frac{4}{15}, \tag{5.53}$$

$$b = -\ln[1-q(1/2)] = \ln\frac{16}{15}. \tag{5.54}$$

This leads us to

$$c(k) = \left(\frac{15}{16}\right)^k \sqrt{\frac{15\pi}{4k}}. \tag{5.55}$$

A last check of the validity of all these calculations is to compare $c(1)$ with a more direct result for the same quantity. $c(1)$ is the average fraction of a path after a single iteration. We can calculate that "by hand" as well. First, the average fraction of a path remaining after a successful shot is

$$\langle x'_{\text{rem}}\rangle = \frac{\int_0^1 dx\, x f_B(x)}{\int_0^1 dx\, f_B(x)}. \tag{5.56}$$

Putting in all the numbers, we end up with

$$\langle x'_{\text{rem}}\rangle = \frac{2}{3}. \tag{5.57}$$

The complete average is now simply the weighted mean of one (for the case, that the trial move is rejected) and $\langle x'_{\text{rem}}\rangle$. As the total acceptance for a trial move is 1/4, we have

$$\langle x_{\text{rem}}\rangle = \frac{3}{4}\times 1 + \frac{1}{4}\langle x'_{\text{rem}}\rangle \tag{5.58}$$

$$= \frac{11}{12}. \tag{5.59}$$

This result turns out to be exactly the same as $c(1) = \frac{15}{16}\,{}_2F_1\left[\frac{1}{2}, -1; \frac{3}{2}; \frac{1}{15}\right]$.

Very similar to $c(k)$ is the distribution function for the update intervals for the complete path,

$$P_{\text{path}}(k) = \int_0^1 dt\, q(t)[1-q(t)]^{k-1}. \tag{5.60}$$

If you perform the calculation with *Mathematica*, you will get

$$P_{\text{path}}(k) = \frac{1}{3}2^{1-3k}7^{k-1}F_1\left[3, 1-k, 1-k, 4; \frac{2}{\sqrt{15}+1}, \frac{-2}{\sqrt{15}-1}\right] \tag{5.61}$$

as a result, where $F_1(a, b_1, b_2, c; x, y)$ is the Appell hypergeometric function. Alternatively, we can also perform the approximation procedure described above.

Note that as $q(t)$ is a second-order polynomial in t, its Taylor approximation up to second order is already exact. Plugging in the already known constants into Eqn. (5.36), we get

$$P_{\text{path}}(k) = \frac{1}{16}\left(\frac{15}{16}\right)^{k-1}\sqrt{\frac{15\pi}{4(k-1)}} + \frac{1}{8}\left(\frac{15}{16}\right)^{k-1}\sqrt{\frac{15^3\pi}{4^3(k-1)^3}}. \tag{5.62}$$

Another comparison, between the exact, the numeric and the approximate result for $P_{\text{path}}(k)$, is given in Fig. 5.9. The result is very similar to that for $c(k)$, again the approximation is only valid for bigger values of k.

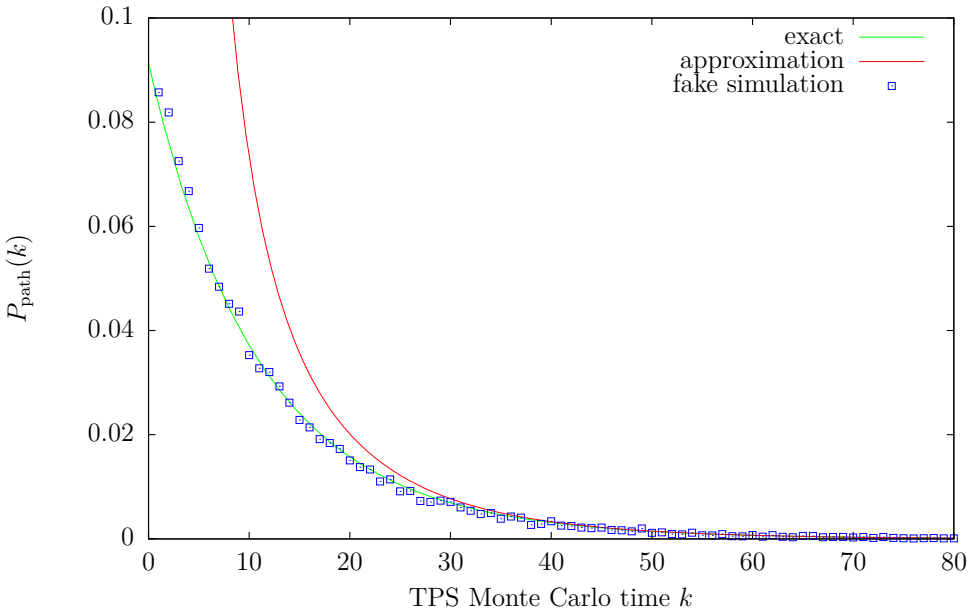

Figure 5.9.: Comparison between the exact result for $P_{\text{path}}(k)$, the result from the "fake" TPS simulation and the approximation for our simple path acceptance model. As in the case of $c(k)$, the approximation is only valid for bigger values of k. The same "fake" TPS simulation as for the calculation of $c(k)$, with a total of $50,000$ simulated TPS Monte Carlo steps, was used.

6. Results and discussion

In this chapter, I will discuss the results of my simulations. I am going to start by comparing my Metropolis Monte Carlo simulation with that by Zhou et al. Then, the validity of the simulation is further checked by comparing the results for a few different temperatures and interaction ranges with those from Taylor et al. Later, we will evaluate the performance and especially the results from the TPS simulation. The transition pathways harvested by the TPS are then used for an extensive committor analysis and a statistical analysis of the found transition states. In particular, we are able to confirm an observation already made by Taylor et al. in Ref. [2]: The typical transition state indeed consists of a crystalline nucleus attached to one or more chain fragments. However, due to our insights in the dynamics of the transition, we can further investigate the structural properties of these transition states.

6.1. Metropolis Monte Carlo results

I have performed a number of simulations with $\lambda = 1.5$ for different temperatures, as already done by Zhou et al. [1]. This was actually the first step I have taken to check the validity of my implementation. As shown in Fig. 6.1, the two simulations agree very well. Since the Zhou et al. results were taken graphically from Fig. 2.3, the data points are not fully accurate. However, it says in the paper that the errors are (except for the $N = 64$ chain) smaller than the size of the points, which is also the expected error from the graphical extraction process. For the $N = 64$ chain, the errors are a bit larger, as seen in Fig. 2.3.

For an estimation of the errors in my own simulation, I have calculated block averages of the simulation results and used them as de facto statistically independent estimates for the quantities in question. 20 blocks, consisting of 200,000 passes each, were used, where each pass consists of 50×128 single-particle moves. One example is taken from the simulation of the $N = 64$ chain at $\lambda = 1.5$ and $T = 1.0$, in that case we have

$$\langle R_g^2 \rangle / N = 0.073033 \pm 0.00009, \tag{6.1}$$

$$\langle U \rangle / N = -2.949 \pm 0.002. \tag{6.2}$$

So, the statistical errors are that small that they are actually well within the dots in Fig. 6.1. Another calculation for the same system, but at the higher temperature

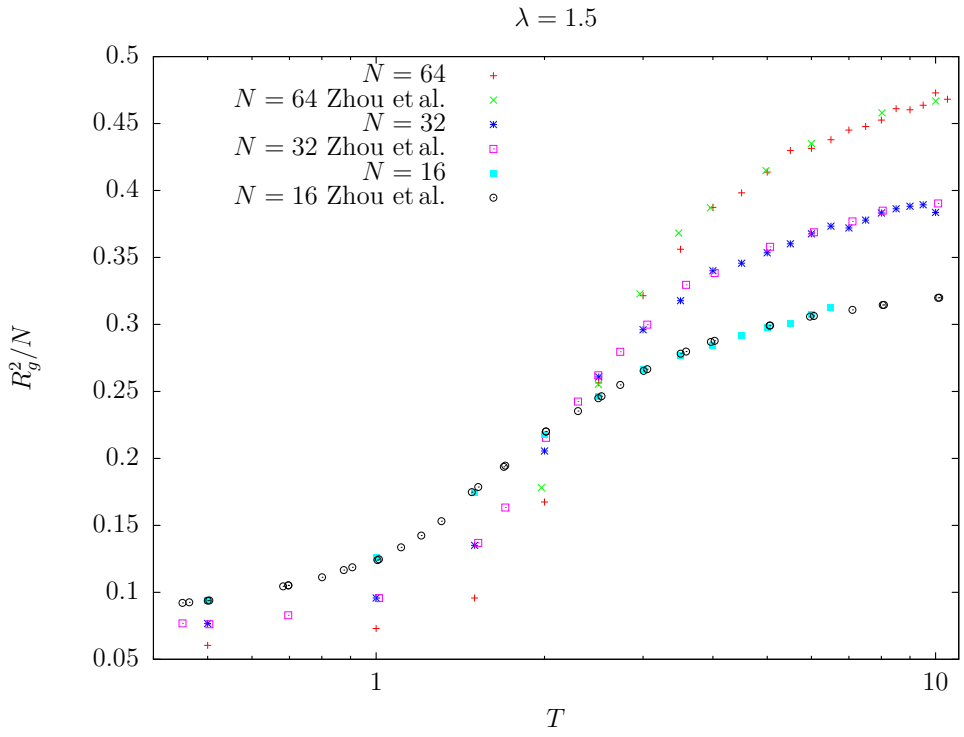

Figure 6.1.: Normalized mean-square radius of gyration versus temperature, comparision between my own Metropolis Monte Carlo simulation results and those of Zhou et al., as published in Ref. [1]. Their data points were extracted graphically from Fig. 2.3.

of $T = 10$ yields

$$\langle R_g^2 \rangle / N = 0.467 \pm 0.002, \tag{6.3}$$

$$\langle U \rangle / N = -0.59432 \pm 0.0009. \tag{6.4}$$

Again, also for this higher temperature, the error in the mean-square radius of gyration is well within the dots in the diagram. As a last test, I have estimated the error for a temperature of $T = 2$, as the error in the results of Zhou et al. is biggest for that temperature. In that case, we have

$$\langle R_g^2 \rangle / N = 0.164 \pm 0.001, \tag{6.5}$$

$$\langle U \rangle / N = -1.554 \pm 0.004. \tag{6.6}$$

Thus, also for this temperature, the error for the mean-square radius of gyration is within the dots in the diagram.

As expected, the pair correlation function $g(r)$ (Fig. 6.2) for the $N = 128$ chain looks very different for the expanded coil and the crystalline phase. For the coil, $g(r)$ has only one maximum at $r = 1$, which is due to the fixed bond length

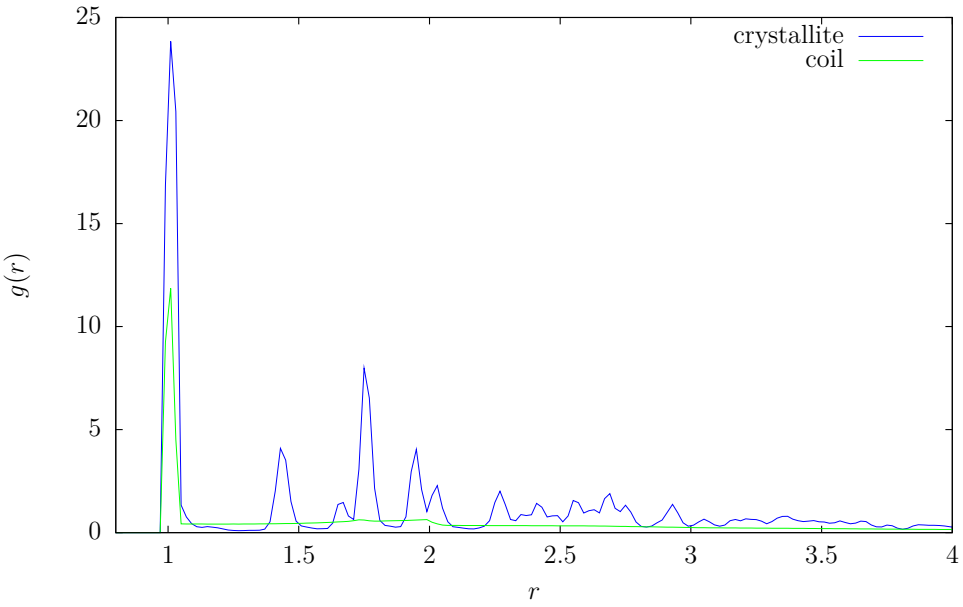

Figure 6.2.: Pair correlation function $g(r)$ for $N = 128$, $\lambda = 1.04$, $T = 0.424$, coil and crystalline phase. Due to the lack of a useful definition of the density for a finite system like the polymer chain, this $g(r)$ is not normalized.

between neighboring chain segments. In contrast, the pair correlation function for the crystalline phase shows many distinct maxima due to the very ordered structure of the chain. In both cases, $g(r) = 0$ for $r < 1$ as a consequence of the hard cores of the monomers.

Also the distribution of $\bar{q}_4$ and $\bar{q}_6$, or, more precisely, the particle's locations in the $\bar{q}_4$-$\bar{q}_6$ plane, are considerably different for the two phases of our system (see Fig. 6.3). To gain an insight into the structure of the system, one can compare the diagram for the polymer chain with the result from Lechner and Dellago [13]. For the crystallite state, the location of the dots in the $\bar{q}_4$-$\bar{q}_6$ plane suggests that the system is mainly in a hcp structure. That makes perfect sense since due to the form of the square-well potential, one would expect that in a crystalline phase, the particles tend to be packed as closely as possible. Due to the finite nature of the system, especially the distribution of the coil particles in the q_4-q_6 plane (without the averaging over next neighbors) looks rather distorted (Fig. 6.4). The difference to the distribution of the averaged parameters is, however, much smaller for the crystallite state.

Another interesting quantity is the overall Monte Carlo acceptance rate as a function of the particle position within the chain. As seen in Fig. 6.6, there is as expected a rather strong difference between the coil and the crystalline phase. In the unordered coil phase, where the chain can move rather freely, the overal acceptance rate is quite high and practically constant for all positions in the chain.

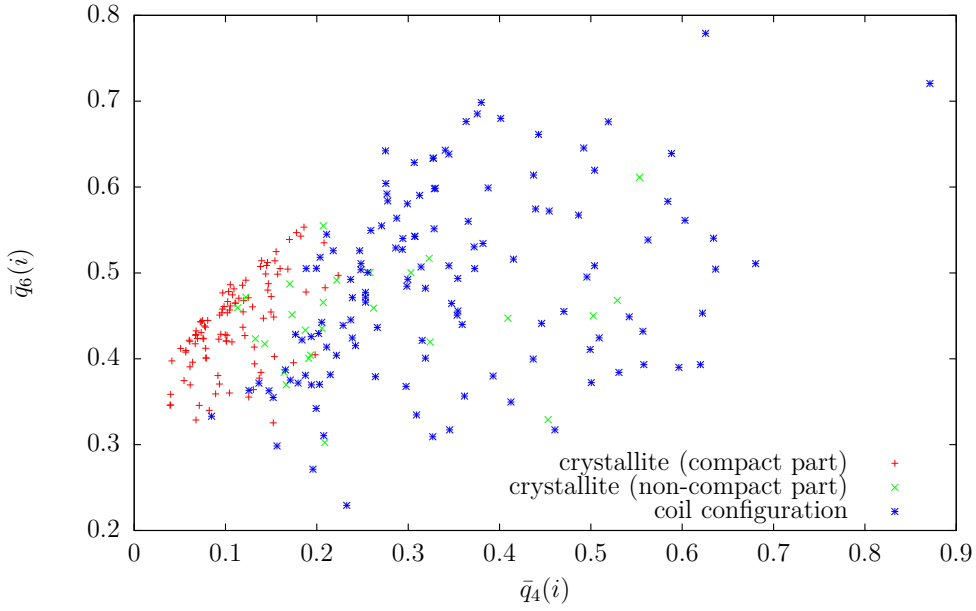

Figure 6.3.: $\bar{q}_4$-$\bar{q}_6$ plane for two typical sample configurations from the system with $N = 128$, $\lambda = 1.04$, $T = 0.424$. For the crystallite state, we further distinguish between compact and non-compact particles.

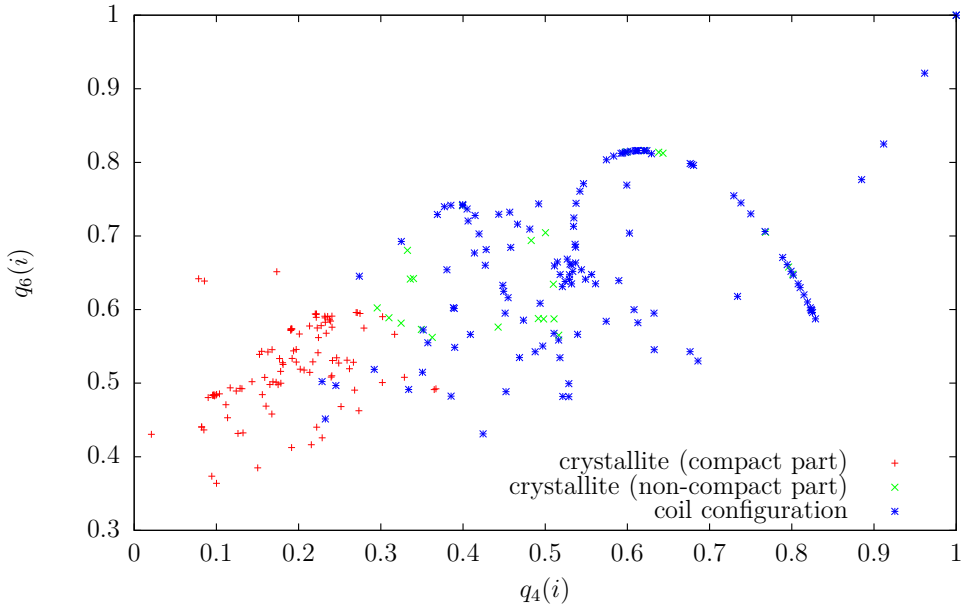

Figure 6.4.: q_4-q_6 plane for two typical sample configurations from the system with $N = 128$, $\lambda = 1.04$, $T = 0.424$. For the crystallite state, we further distinguish between compact and non-compact particles.

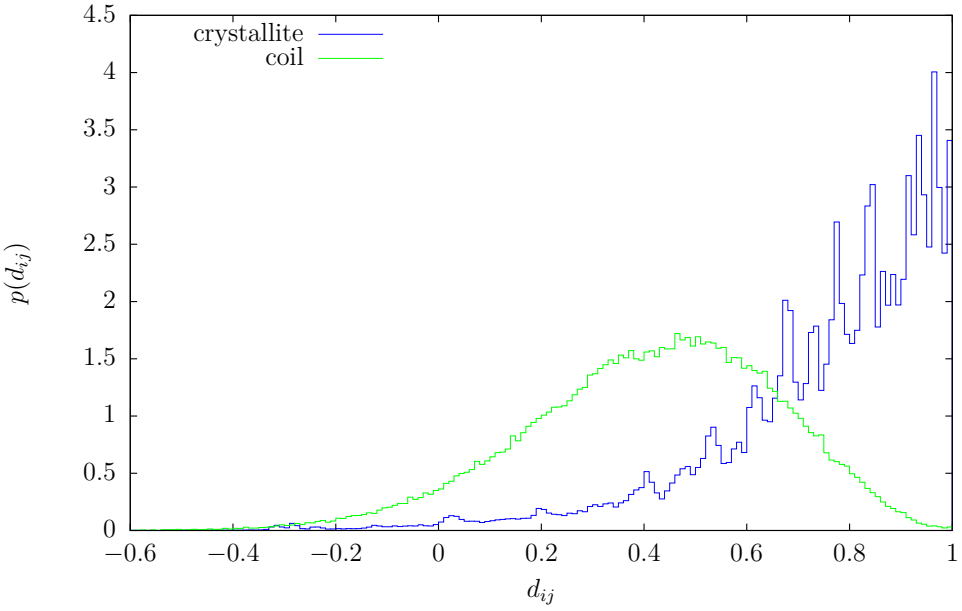

Figure 6.5.: Distribution of the connection coefficients d_{ij} for $N = 128$, $\lambda = 1.04$, $T = 0.424$. The histograms were collected during a simulation run of the system in the coil respectively crystallite state and are normalized for comparison.

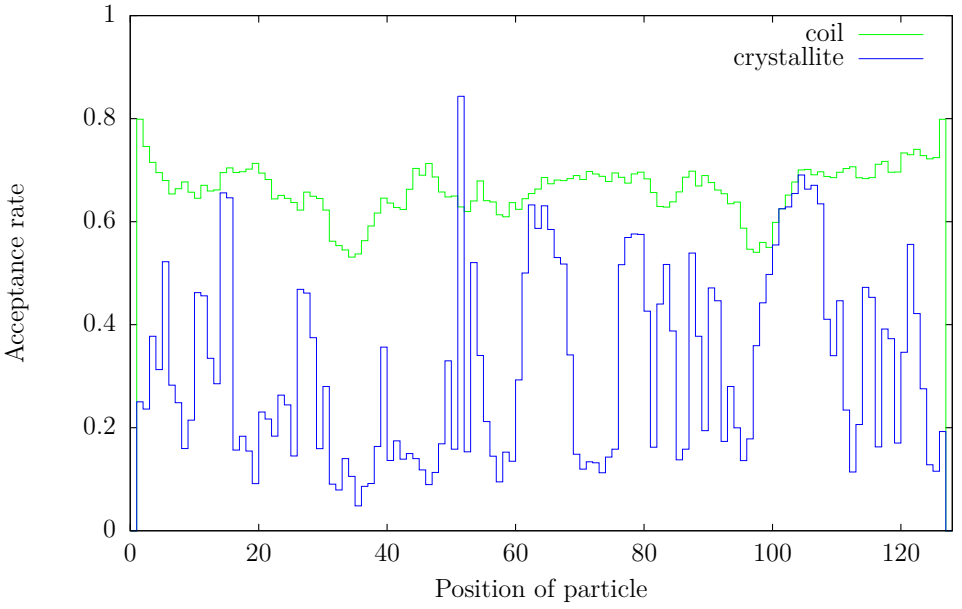

Figure 6.6.: Acceptance rate of crankshaft moves as a function of particle position within the chain, for the $N = 128$, $\lambda = 1.04$, $T = 0.424$ system.

In contrast to that, the acceptance rate in the densely packed crystalline phase is much lower. Furthermore, it is strongly dependent on the particle position within the chain. This indicates that especially particles very deep within the nucleus are unlikely to change there positions once they have crystallized.

In order to perform a TPS simulation to study a phase transition, one needs to know the coexistence temperature for the given system. For the $N = 128$ chain, this information is already available in the form of a complete phase diagram (see Fig. 2.5 on page 8) for a wide range of values for λ, as published by Taylor et al. [2]. For the $N = 64$ chain and the same value of $\lambda = 1.04$ as for the simulation of the longer chain, an approximate transition temperature of $T = 0.38$ was determined by interpolation of the data provided by Wolfgang Paul (Fig. 2.6).

Also, for the TPS one has to be able to distinguish between the two different phases of the system. In our case, the chain is considered to be in crystallite or B state if its number of solid-like particles is above a certain threshold $N_{\min}$ and its potential energy is below a certain threshold $U_{\max}/N$. The coil or A state is defined in the same way, of course with opposite thresholds $N_{\max}$ and $U_{\min}/N$. These thresholds are different for different chain sizes. Table 6.2 gives an overview of the thresholds used in our TPS simulations.

		$N = 64$	$N = 128$
coil	$U_{\min}/N$	-0.8	-0.8
	$N_{\max}$	2	6
crystallite	$U_{\max}/N$	-2.1	-2.1
	$N_{\min}$	35	60

Table 6.2.: Thresholds used to distinguish between coil and crystallite state.

6.2. Committor analysis

After harvesting about 70 independent pathways (for $N = 128$, $\lambda = 1.04$, $T = 0.424$) with our TPS simulation, the next logical step was a committor analysis of the states in these transition paths. An interesting property of such a transition pathway is the development of the committor value p_B along the time of the path, which corresponds to a "perfect" reaction coordinate. One example for such a time development is given in Fig. 6.7, along with the time development of the global order parameter Q_6 of the whole chain. It is observed that instead of directly evolving from region A to region B in a steady way, the system undergoes a very rough transition. This already indicates that the transition mechanism is of a rather complex nature. Also the time development of the committor for a smaller system (Fig. 6.8) looks very similar. However, the actual transition takes considerable less Monte Carlo time, which has to be expected for a smaller system.

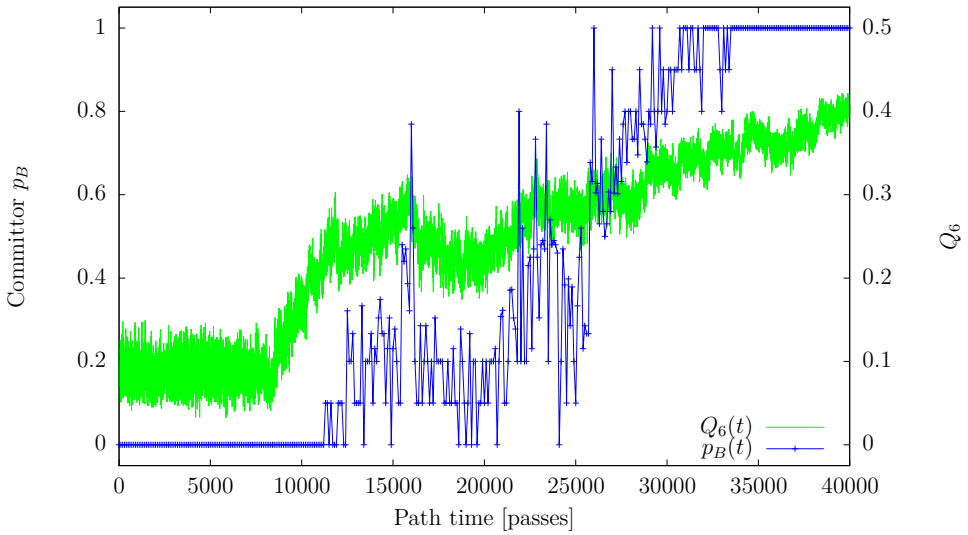

Figure 6.7.: Time development of the committor p_B (blue) and the global order parameter Q_6 (green) for a typical transition path. The simulation was performed with $N = 128$, $\lambda = 1.04$, $T = 0.424$. One pass corresponds to $N \times 50$ single-particle moves.

In Figs. 6.9 and 6.10, we have plotted the time development of the potential energy per particle and the number of solid-like particles for the same two transition pathways. These are the quantities which are used to distinguish between the two states of the system.

6.2.1. Finding a reaction coordinate

A reaction coordinate is a one-dimensional function of a system's configuration which allows to track the progress of a transition. Obviously, we would like to find a suitable reaction coordinate for the coil to crystallite transition of the polymer chain. In classical nucleation theory, the size of the biggest crystalline cluster is used as a reaction coordinate [15]. Since our system is only one finite object, there is normally also only one crystalline nucleus in a transition, and we could try to use its size as a reaction coordinate. One way to test if some quantity can be used as reaction coordinate is to take many different configurations and plot the value of the proposed reaction coordinate against the calculated committor in a diagram for each of them.

In Fig. 6.11, we have such a plot for the size of the crystalline nucleus as a possible reaction coordinate. The system is again the $N = 128$ chain with $\lambda = 1.04$ at the transition temperature $T = 0.424$. From each of the harvested transition paths, 200 random configurations which are neither A nor B were selected. Size means here simply number of crystalline particles in the nucleus. A particle is defined as crystalline if it has more than four connections (see section 3.3.3, page 25) and is

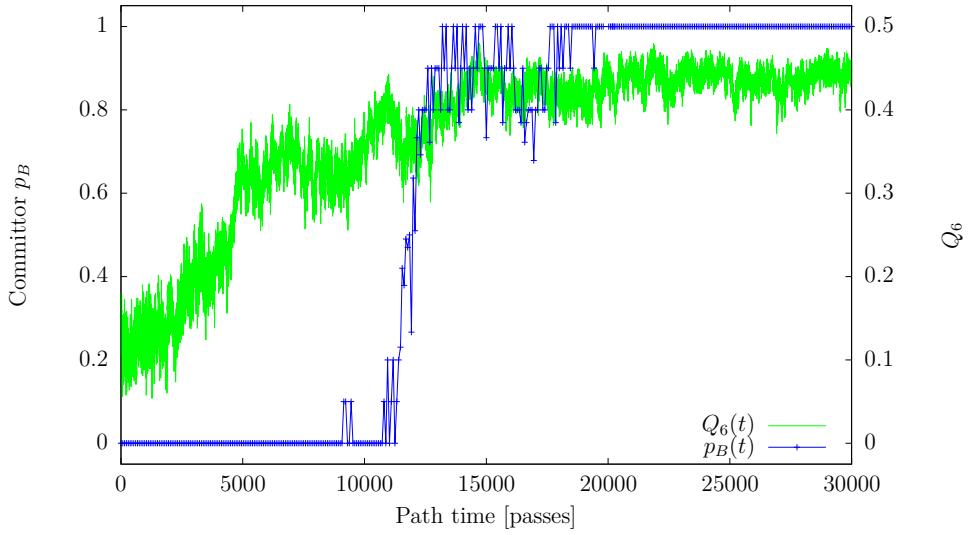

Figure 6.8.: Time development of the committor p_B (blue) and the global order parameter Q_6 (green) for a typical transition path of the smaller system with $N = 64$, $\lambda = 1.04$, $T = 0.38$. One pass corresponds to $N \times 50$ single-particle moves.

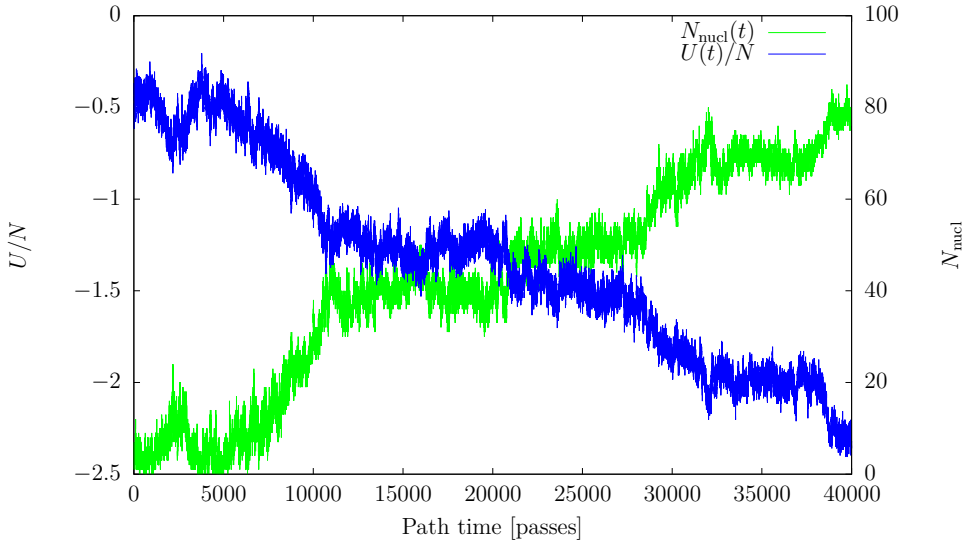

Figure 6.9.: Time development of the potential energy per particle (blue) and the number of particles in the crystalline nucleus (green) for the same $N = 128$ transition path as in Fig. 6.7.

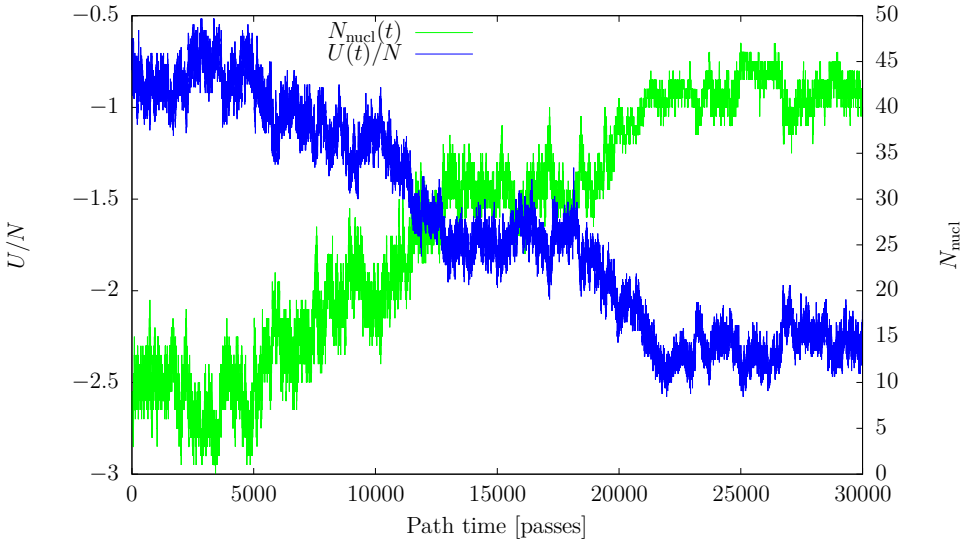

Figure 6.10.: Time development of the potential energy per particle (blue) and the number of particles in the crystalline nucleus (green) for the same $N = 64$ transition path as in Fig. 6.8.

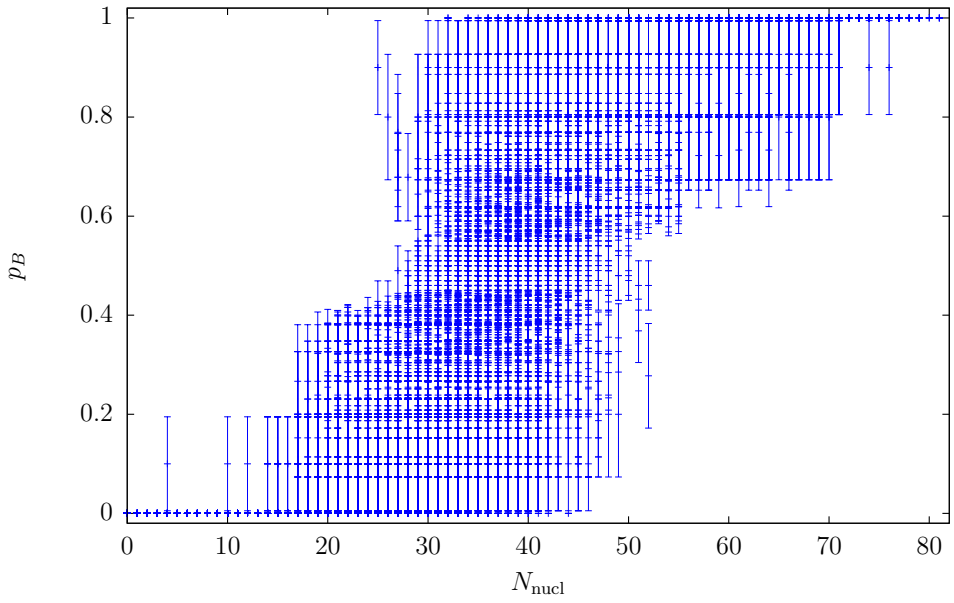

Figure 6.11.: Committor p_B as a function of the number of particles in the crystalline nucleus for random configurations from the $N = 128$ transition pathways. The error bars indicate the error approximation (Eqn. (3.31)) for the calculated committors.

furthermore connected to at least all but one of its next neighbors. For a good reaction coordinate, we would expect a sigmoidal shape of $p_B(r)$, where r is the proposed coordinate. However, as seen in the diagram, there is a huge range of nucleus sizes where every committor value between 0 and 1 is possible. We can thus conclude that the nucleus size is not sufficient to describe the progress of the coil to crystallite transition of the system.

One approach to obtain a better reaction coordinate is to perform a likelihood maximization as first described by Peters and Trout [17]. The proposed coordinate is modeled as

$$q = c_0 + c_1 p_1 + c_2 p_2 + \ldots, \tag{6.7}$$

where the p_i are physical parameters of the system and the c_i are the coefficients to optimize. That way, the reaction coordinate is basically just a linear combination of different physical parameters. However, by introducing parameters like, for example, $p_3 = p_1 p_2$, it is also possible to include non-linear contributions.

Clarion Tung has provided me with his implementation of the optimization procedure. With the calculated committor values of the $N = 128$ polymer chain, I have tried a number of different physical parameters for the model reaction coordinate. Naturally, the number of particles in the crystalline nucleus and the potential energy per particle are expected to be promising candidates for such physical variables that play an important role in the transition. Other parameters I have tried include – but are not limited to – the anisotropy (6.9) of the system, the principal moments of inertia and the number of chain-like loops in a configuration and global order parameters like Q_4 and Q_6. Unfortunately, none of these variables was able to considerably improve the result compared to the number of crystalline particles or the potential energy per particle alone.

One example for such an unsuccessful trial is shown in Fig. 6.12. In that case, in addition to the potential energy per particle, a second physical parameter was used for the optimization procedure. It is defined as

$$A = N_{\text{nucl}}^{2/3}, \tag{6.8}$$

where N_{nucl} is the number of particles in the crystalline nucleus, which was already used in Fig. 6.11. For a more or less sphere-like nucleus, A is expected to be proportional to the surface area of the nucleus.

6.2.2. Transition state ensemble

Another important goal of the committor analysis is to find a sample from the transition state ensemble. This is done by searching for configurations with a committor value of 1/2, up to statistical errors. The procedure described in section 3.2.5 (page 21) was used to obtain that sample, with $N_{\text{min}} = 10$ and $N_{\text{max}} = 100$. After performing the 10 initial time evolutions, configurations where 1/2 was not within $[p_B^{(N)} - 2\sigma^{(N)}, p_B^{(N)} + 2\sigma^{(N)}]$ were excluded from the further selection. In the end, a configuration was considered part of the TSE if 1/2 was within

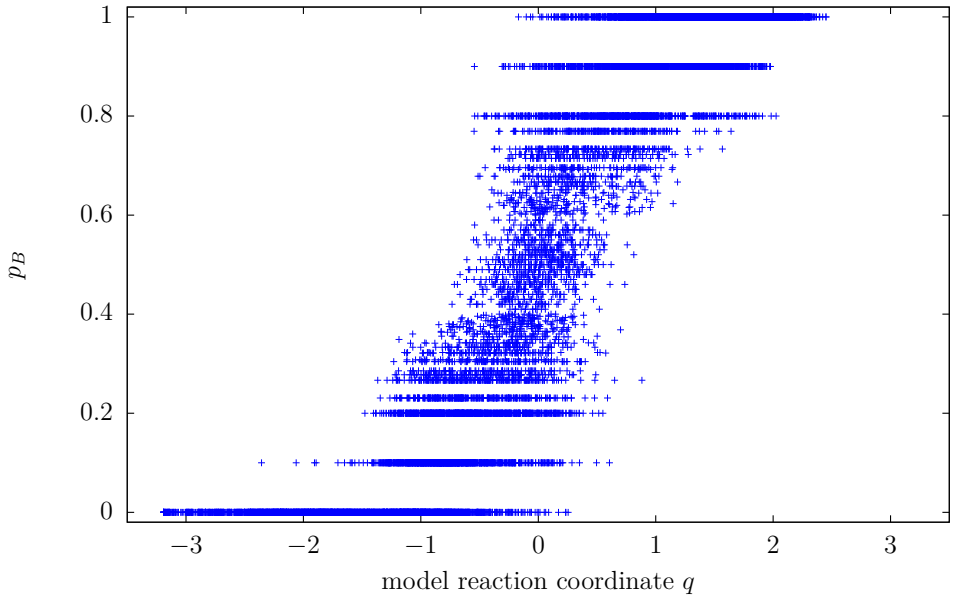

Figure 6.12.: Committor p_B as a function of a model reaction coordinate for the same states as used for Fig. 6.11.

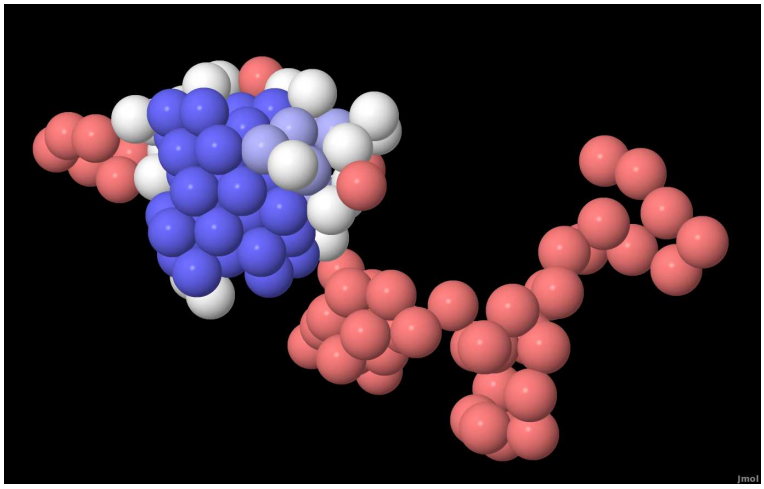

Figure 6.13.: Typical transition state for the $N = 128$ polymer chain, with $\lambda = 1.04$ and $T = 0.424$. The observation of Taylor et al. [2] can be confirmed via committor analysis: The typical transition state indeed consists of a crystalline nucleus attached to one or more chain fragments. Dark blue indicates crystalline particles; particles in light blue are in a compact, but not yet crystalline structure; red marks chain-like particles and white means that neither of these criteria applies.

$[p_B^{(100)} - \sigma^{(100)}, p_B^{(100)} + \sigma^{(100)}]$. Out of the transition paths for the $N = 128$ chain, a little less than 500 transition states were harvested with that approach. A typical example from these states is shown in Fig. 6.13. Already a first visual analysis of the configurations in the TSE confirmed the observation of Taylor et al. [2] concerning the structure of transition states. They concluded that the typical transition state consists of a crystalline nucleus attached to one or more extended chain fragments. However, their result was only drawn from a visual analysis of states which are energetically in between the high-energy coil and the low-energy crystalline phase. Thanks to the new results from the committor analysis, the initial observation could be confirmed using a more rigorous approach.

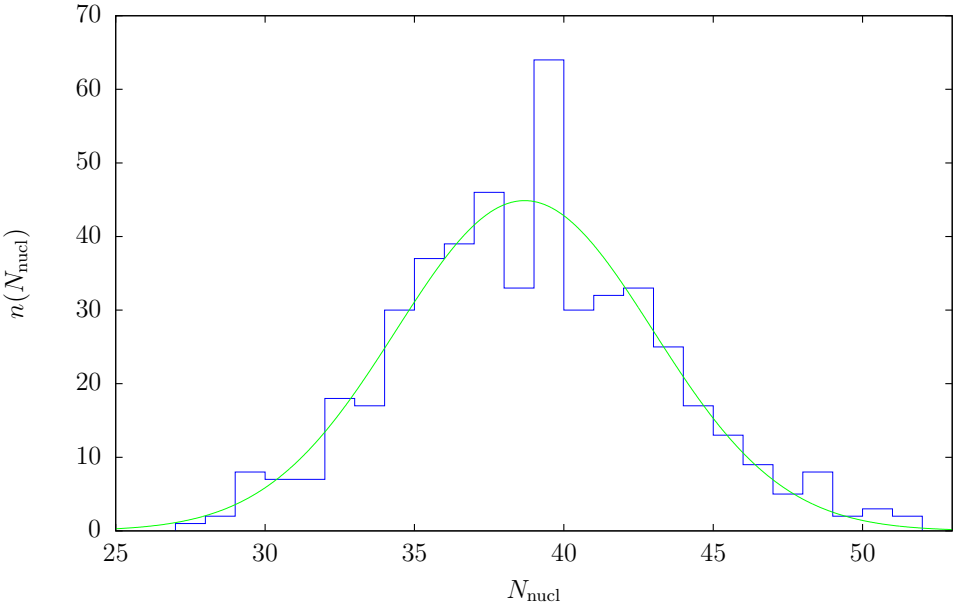

Figure 6.14.: Histogram of the number of particles in the crystalline nucleus for configurations within the $N = 128$ TSE, and fit with a normal distribution.

In Fig. 6.14, we have a histogram of the number of particles in the crystalline core for configurations that are part of the transition state ensemble. As has already become clear from the dependence of the committor on the nucleus size, this distribution is rather broad. However, most of the transition states are clustered around a nucleus size of about 38, with a standard deviation of 4. The fit with a normal distribution yields the same result. The distribution of the energy per particle for the same configurations (Fig. 6.16) looks rather similar. This comes not at all unexpected as the biggest contribution to the system's potential energy comes from the region where the particles are packed very densely – in other words, the crystalline core.

In Fig. 6.15, the distribution of nucleus sizes for the smaller $N = 64$, $\lambda = 1.04$ system is shown for comparison. For this system, a much shorter committor calcu-

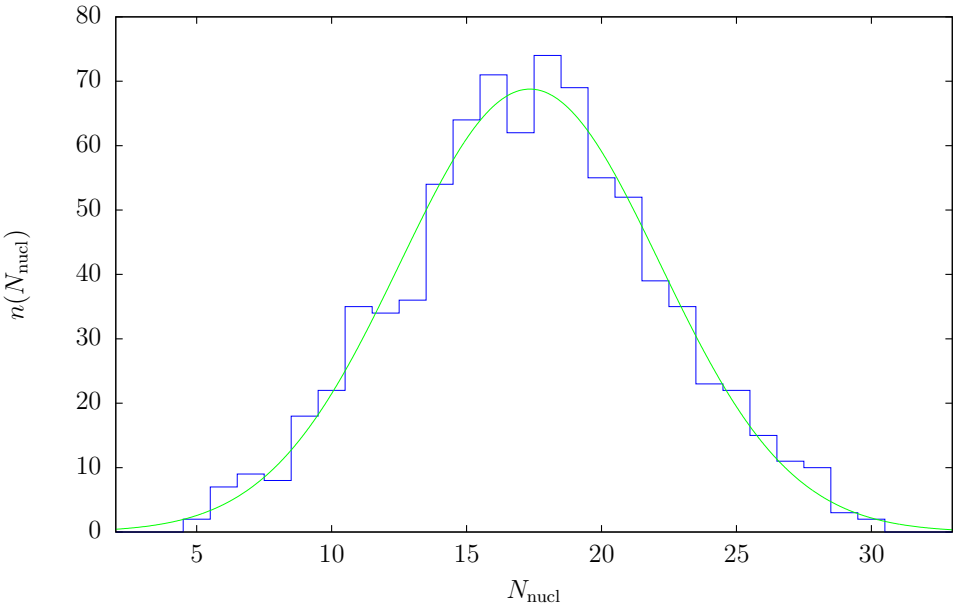

Figure 6.15.: Histogram of the number of particles in the crystalline nucleus for configurations within the $N = 64$ TSE, and fit with a normal distribution.

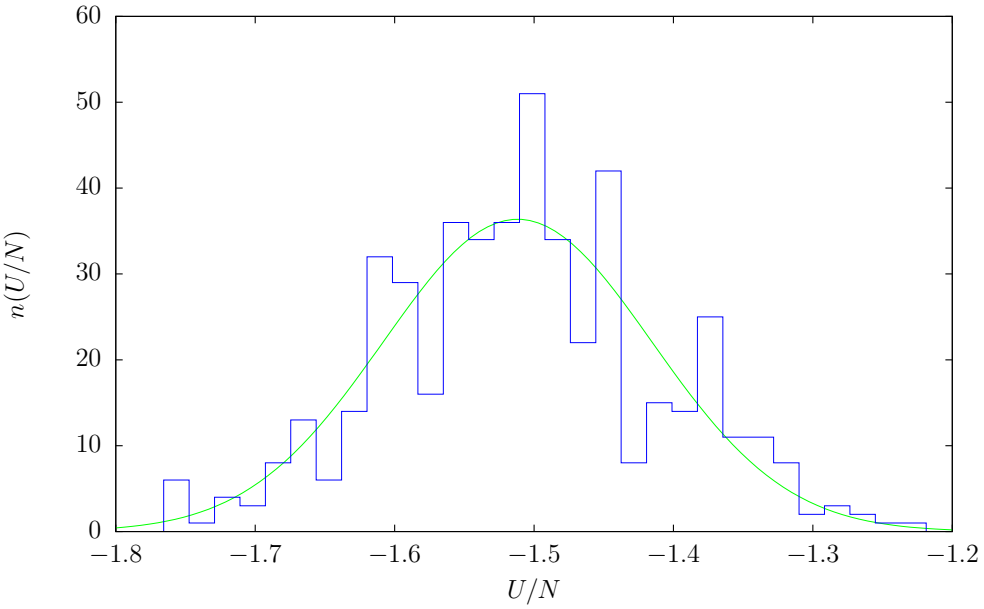

Figure 6.16.: Histogram of the energy per particle for configurations within the $N = 128$ TSE, and fit with a normal distribution.

lation has yielded about 800 transition states. The transition states are clustered around a nucleus size of about 17 particles, again with a standard deviation of 4.

The anisotropy

$$a = \frac{I_{\max}}{I_{\min}} - 1, \tag{6.9}$$

where $I_{\max}$ and $I_{\min}$ are the largest respectively smallest principal moments of inertia, has a different distribution within the TSE (Fig. 6.17). Both the distribution of the anisotropy of the whole system and of the crystalline core only are shown. While the distribution for the whole chains is rather broad and has a long tail towards big anisotropy values, the distribution for the crystalline cores is sharply peaked around 0.5. Again, this confirms the observation that the presence of a more or less spherical or at least not too anisotropic, crystalline nucleus is important. However, the overall shape of the complete system, mostly determined by the positions of the open chain segments, does not play an important role in the transition.

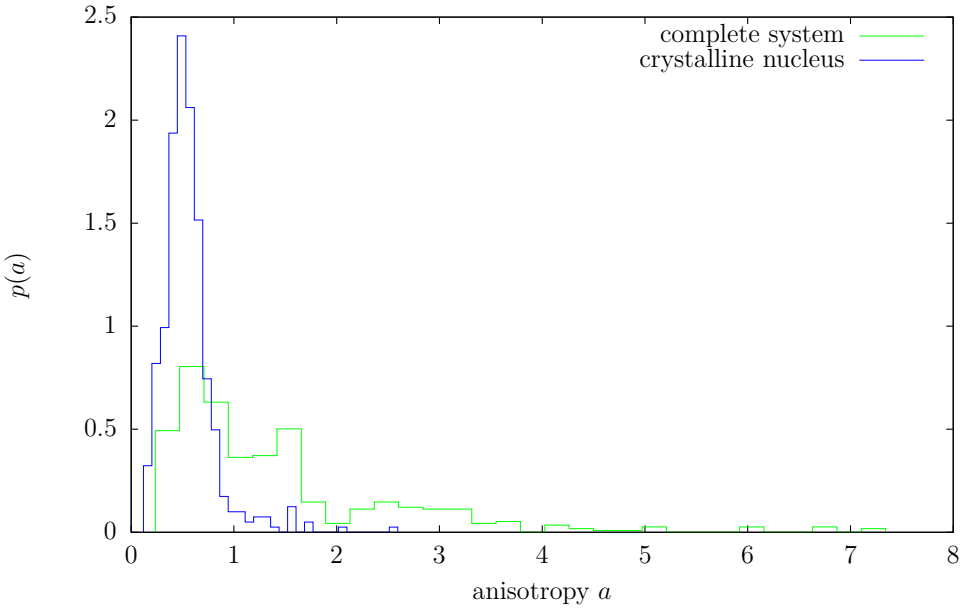

Figure 6.17.: Histogram of the anisotropy of both the complete system and the crystalline core only for configurations within the $N = 128$ TSE. The histograms are normalized for comparison.

7. Final Remarks

We have studied the first-order freezing transition of a single, flexible homopolymer chain with square-well interactions. Using the transition path sampling simulation technique and an evolution according to Metropolis Monte Carlo dynamics, we were able to sample the transition state ensemble of this system. We performed our TPS simulations with an interaction range of $\lambda = 1.04$ for two different chain lengths of $N = 64$ and $N = 128$. Prior to that, we have validated the implementation of the Metropolis Monte Carlo simulation by comparing our simulations with results from the literature for $\lambda = 1.5$ and different chain lengths of 16, 32 and 64 monomers.

In addition to the TPS simulations of the polymer chain, we have performed a statistical analysis of the efficiency of TPS in finding new transition pathways. The path survival function introduced in this thesis can be used to find out how much simulation time it takes on average to find a completely new transition path. Furthermore, we have also calculated the distribution of update times along a path. In all cases, the predictions derived from our analytical calculations have agreed very well with the results from the actual TPS simulations.

Our committor analysis of the harvested transition pathways has yielded a sample from the transition state ensemble. A statistical analysis of the TSE confirmed the previous observations concerning the structure of the typical transition states. The states in the TSE have a single crystalline nucleus, while the rest of the system is still in a chain-like configuration. The nucleus sizes are normally distributed around an average value. The crystalline nucleus is typically already in the hcp structure, which is also the preferred structure of the fully crystallized system. Furthermore, it is evident that the anisotropy distribution of the nuclei alone is much sharper than that of the whole transition states. This indicates that, typically, the nuclei in the transition states show less deviation from a spherical shape than the whole system, with its often extended open chain fragments. However, it remains an open question how the structural properties of the nucleus and the chain fragments contribute to a reaction coordinate. It is clear that due to the complex constraints of the system, a reaction coordinate cannot be determined by the size of the crystalline nucleus alone. Instead, one could speculate that also the spatial arrangement of the chain plays an important role in the further progress of the folding transition. The challenge for further research is thus to identify the relevant degrees of freedom involved in the chain arrangement. Possibly, the application of machine learning algorithms can help solving this complex puzzle.

A. Calculation of the mean-square radius of gyration

It will be shown that Eqns. (2.4) and (2.6) are equivalent. From Eqn. (2.4), when dropping the brackets for averaging, we have

$$R_g^2 = \frac{1}{N}\sum_{i=1}^{N}(\vec{r}_i - \vec{r}_{\text{mean}})^2 \tag{A.1}$$

$$= \frac{1}{N}\sum_i \left(\vec{r}_i - \frac{1}{N}\sum_j \vec{r}_j\right)^2 \tag{A.2}$$

$$= \frac{1}{N}\sum_i \left[\vec{r}_i^2 + \left(\frac{1}{N}\sum_j \vec{r}_j\right)\left(\frac{1}{N}\sum_k \vec{r}_k\right) - 2\vec{r}_i\frac{1}{N}\sum_j \vec{r}_j\right] \tag{A.3}$$

$$= \frac{1}{N}\sum_i \vec{r}_i^2 + \left(\frac{1}{N}\sum_j \vec{r}_j\right)\left(\frac{1}{N}\sum_k \vec{r}_k\right) - \frac{2}{N^2}\sum_i \vec{r}_i \sum_j \vec{r}_j \tag{A.4}$$

$$= \frac{1}{N}\sum_i \vec{r}_i^2 - \frac{1}{N^2}\sum_i \vec{r}_i \sum_j \vec{r}_j \tag{A.5}$$

$$= \frac{1}{2N^2}\sum_i \sum_j \left[\vec{r}_i^2 + \vec{r}_j^2 - 2\vec{r}_i\vec{r}_j\right] \tag{A.6}$$

$$= \frac{1}{2N^2}\sum_i \sum_j (\vec{r}_i - \vec{r}_j)^2 \tag{A.7}$$

$$= \frac{1}{2N^2}\sum_{i\neq j} (\vec{r}_i - \vec{r}_j)^2 \tag{A.8}$$

$$= \frac{1}{N^2}\sum_{i<j} (\vec{r}_i - \vec{r}_j)^2 = \frac{1}{N^2}\sum_{i<j} r_{ij}^2. \tag{A.9}$$

This is exactly Eqn. (2.6), of course again without the averaging.

B. Calculation of the average time between path updates

In order to obtain the average time between path updates at a given time slice with update probability q, we need to calculate

$$\bar{k} = q \sum_{k=0}^{\infty} (k+1)(1-q)^k. \tag{B.1}$$

The difficult part is the infinite sum

$$\sum_{k=0}^{\infty} (k+1)(1-q)^k = \frac{1}{q^2}. \tag{B.2}$$

In this calculation, the crucial step is to note that in some sense, this sum is nothing but an infinite collection of geometric series. Let us first consider the partial sum

$$\sum_{k=0}^{m} (k+1)(1-q)^k. \tag{B.3}$$

For the sake of simplicity, let us introduce $p \coloneqq 1-q$, where $p \in [0,1)$. We can now visualize the sum as

$$\begin{aligned} \sum_{k=0}^{m} (k+1)p^k \quad = \quad & p^0 \\ & + \; p^1 \; + \; p^1 \\ & + \; p^2 \; + \; p^2 \; + \; p^2 \\ & + \; \dots \\ & + \; p^m \; + \; p^m \; + \; p^m \; + \; \dots \; + \; p^m. \end{aligned} \tag{B.4}$$

This is now a geometric series up to index m, plus the same series without the first element, plus the same series without the first two elements, and so on. We know that for a geometric series

$$\sum_{k=0}^{m} p^k = \frac{1-p^{m+1}}{1-p}. \tag{B.5}$$

Hence, we can write

$$\sum_{k=0}^{m}(k+1)p^k = (m+1)\frac{1-p^{m+1}}{1-p} - \sum_{n=0}^{m-1}\frac{1-p^{n+1}}{1-p} \tag{B.6}$$

$$= (m+1)\frac{1-p^{m+1}}{1-p} - m\frac{1}{1-p} + \sum_{n=0}^{m-1}\frac{p^{n+1}}{1-p} \tag{B.7}$$

$$= \frac{1}{1-p} - \frac{(m+1)p^{m+1}}{1-p} + \sum_{n=1}^{m}\frac{p^n}{1-p} \tag{B.8}$$

$$= \frac{1}{1-p} - \frac{(m+1)p^{m+1}}{1-p} + \frac{1}{1-p}\left[\frac{1-p^{m+1}}{1-p} - 1\right] \tag{B.9}$$

$$= \frac{1-p^{m+1}}{(1-p)^2} - \frac{(m+1)p^{m+1}}{1-p}. \tag{B.10}$$

The last remaining step is to take the limit as m approaches infinity:

$$\sum_{k=0}^{\infty}(k+1)(1-q)^k = \lim_{m\to\infty}\sum_{k=0}^{m}(k+1)(1-q)^k \tag{B.11}$$

$$= \lim_{m\to\infty}\frac{1-p^{m+1}}{(1-p)^2} - \frac{(m+1)p^{m+1}}{1-p} \tag{B.12}$$

$$= \frac{1}{(1-p)^2} \tag{B.13}$$

$$= \frac{1}{q^2}. \quad \square \tag{B.14}$$

Bibliography

[1] Yaoqi Zhou, M. Karplus, J. M. Wichert, and C. K. Hall. *Equilibrium thermodynamics of homopolymers and clusters: Molecular dynamics and Monte Carlo simulations of systems with square-well interactions.* J. Chem. Phys. **107**, 10691 (1997).

[2] Mark P. Taylor, Wolfgang Paul, and Kurt Binder. *Phase transitions of a single polymer chain: A Wang-Landau simulation study.* J. Chem. Phys. **131**, 114907 (2009).

[3] Mark P. Taylor, Wolfgang Paul, and Kurt Binder. *All-or-none proteinlike folding transition of a flexible homopolymer chain.* Phys. Rev. E **79**, 050801 (2009).

[4] Mark P. Taylor. *Configurational statistics for isolated square-well chain molecules: exact results for short chains.* Mol. Phys. **86**, 73 (1995).

[5] Mark P. Taylor and J. E. G. Lipson. *Collapse of a polymer chain: A Born-Green-Yvon integral equation study.* J. Chem. Phys. **104**, 4835 (1996).

[6] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller and Edward Teller. *Equation of state calculations by fast computing machines.* J. Chem. Phys. **21**, 1087 (1953).

[7] Christoph Dellago and Peter G. Bolhuis. *Transition Path Sampling and Other Advanced Simulation Techniques for Rare Events.* In *Advanced Computer Simulation Approaches for Soft Matter Sciences III*, vol. 221 of *Advances in Polymer Sciences*, pp. 167–233. Springer, Berlin (2009).

[8] Christoph Dellago, Peter G. Bolhuis, and Philip L. Geissler. *Transition path sampling methods.* In M. Ferrario, G. Ciccotti, and K. Binder (editors), *Computer Simulations in Condensed Matter Systems: From Materials to Chemical Biology*, pp. 1–38. Springer, Berlin (2006).

[9] Christoph Dellago, Peter G. Bolhuis, and Philip L. Geissler. *Transition path sampling.* In *Advances in Chemical Physics*, vol. 123 of *Advances in Chemical Physics*, pp. 1–78. John Wiley & Sons, New York (2002).

[10] Elisabeth Schöll-Paschinger and Christoph Dellago. *Demixing of a binary symmetric mixture studied with transition path sampling.* J. Chem. Phys. **133**, 104505 (2010).

[11] Jean-Pierre Hansen and Ian R. McDonald. *Theory of Simple Liquids*. Academic Press, London, 3rd ed. (2006).

[12] P. J. Steinhardt, D. R. Nelsen, and M. Ronchetti. *Bond-orientational order in liquids and glasses*. Phys. Rev. B **28**, 784 (1983).

[13] Wolfgang Lechner and Christoph Dellago. *Accurate determination of crystal structures based on averaged local bond order parameters*. J. Chem. Phys. **129**, 114707 (2008).

[14] Daniele Moroni. *Efficient Sampling of Rare Event Pathways*. Ph.D. thesis, University of Amsterdam (2005).

[15] P. R. ten Wolde. *Numerical Study of Pathways for Homogeneous Nucleation*. Ph.D. thesis, University of Amsterdam (1998).

[16] J. Juraszek and P. G. Bolhuis. *Sampling the multiple folding mechanisms of Trp-cage in explicit solvent*. Proc. Natl. Acad. Sci. USA **103**, 15859 (2006).

[17] Baron Peters and Bernhardt L. Trout. *Obtaining reaction coordinates by likelihood maximization*. J. Chem. Phys. **125**, 054108 (2006).

Abstract

We investigate the crystallization of a single, flexible homopolymer chain using transition path sampling (TPS). The chain consists of N identical spherical monomers evolving according to Metropolis Monte Carlo dynamics. While neighboring monomers have a fixed distance, the non-neighboring monomers interact via a square-well potential. For a sufficiently small interaction range λ, the system undergoes a first-order freezing transition from an expanded, unordered phase to a compact crystalline state. TPS and committor analysis are used to study the transition state ensemble of the chain and to search for possible reaction coordinates. Earlier observations concerning the structural properties of the transition states are confirmed by our calculations: The typical transition states indeed consist of a crystalline nucleus attached to one or more chain fragments. Furthermore, we investigate the statistical properties of TPS simulations and introduce the *path survival function* $c(k)$. This quantity gives the average unchanged fraction of a transition path after k TPS iterations.

Zusammenfassung

Wir verwenden die Simulationstechnik Transition Path Sampling (TPS), um die Kristallisation einer einzelnen, flexiblen Polymerkette zu untersuchen. Die Kette setzt sich aus N identischen, kugelförmigen Monomeren zusammen, wobei die zeitliche Entwicklung des Systems durch die Metropolis-Monte-Carlo-Dynamik bestimmt wird. Während aneinander angrenzende Monomere einen fixen Abstand haben, wechselwirken nicht angrenzende Teilchen mit einem Kastenpotential. Solange die Reichweite λ des Potentials klein genug ist, vollzieht das System einen Phasenübergang erster Ordnung von einem ausgedehnten, ungeordneten Zustand zu einem kristallinen, kompakten Zustand. TPS und Committor-Analyse werden verwendet, um diesen Phasenübergang zu studieren und nach etwaigen Reaktionskoordinaten zu suchen. Frühere Beobachtungen betreffend der Struktur der Übergangszustände können bestätigt werden: Tatsächlich besteht ein typischer Übergangszustand aus einem kristallinen Kern, der an ein oder mehrere kettenartige Fragmente angeheftet ist. Außerdem untersuchen wir die statistischen Eigenschaften von TPS-Simulationen und führen die *Path Survival Function* $c(k)$ ein. Diese gibt uns den durchschnittlichen ungeänderten Anteil eines Übergangspfades nach k TPS-Iterationen an.

Christian Leitold

Contact

E-Mail christian.leitold@univie.ac.at

Personal information

Date and place of birth November 12th, 1986, Judenburg, Austria

Education

1993 - 1997	Primary school
1997 - 2005	Secondary school: BG/BRG Knittelfeld, Austria
January 2003	Participant at THIMUN 2003 (The Hague International Model United Nations) in The Hague, The Netherlands
January 2004	Participant at THIMUN 2004
May 2004	Participant at MUNOL 2004 (Model United Nations of Lübeck) in Lübeck, Germany
2005	Matura (leaving certificate) at BG/BRG Knittelfeld, with honor
2006 - 2011	Study of physics at University of Vienna
September 2007	Summer student in the group for Quantum Optics, Quantum Nanophysics and Quantum Information, University of Vienna
2008	Academic stage 1 (*1. Studienabschnitt*) completed, with honor
2010	Academic stage 2 (*2. Studienabschnitt*) completed, with honor
October 2010	Start of work for diploma thesis in computational physics, group of Christoph Dellago
2011	First part of academic stage 3 (*3. Studienabschnitt*) completed, with honor

Teaching experience

2010 - 2011	Tutor for *Physikalisches Praktikum für Biologen* (physics lab courses for biologists), University of Vienna

Winter term 2010 Tutor for *Übungen zur Einführung in die physikalischen Rechenmethoden I + II* (basic math exercises for first-term physics students), University of Vienna

Work experience

August - September 2009 1st short time employment at the Stefan Meyer Institute for Subatomic Physics, Vienna; task: scientific programming

September 2010 2nd short time employment at the Stefan Meyer Institute for Subatomic Physics

Languages

German	Mother tongue
English	Fluent
Spanish	Basic knowledge

Miscellaneous

2005 *Fachbereichsarbeitspreis der Österreichischen Gesellschaft für Astronomie und Astrophsik* (award from the Austrian Society for Astronomy and Astrophysics for my written work about space probes as part of my Matura in physics)

2005 - 2006 Alternative civilian service at the Austrian Red Cross in Knittelfeld