Parallel Monte Carlo Simulations of Coarse-Grained Flexible Polymers To Obtain Structural Phase Diagrams

by

Jonathan Gross

(Under the Direction of Michael Bachmann)

Abstract

In this computational study the focus is put on the thermodynamic behavior of coarse-grained models for flexible polymers. Structural transitions of polymer conformations as a function of monomer-monomer interaction range, as well as adsorption at curved surfaces of cylinders of different materials are unraveled. This is accomplished by utilizing the latest available technologies, such as high-end graphics processing units, and advanced general-ensemble Monte Carlo methods. A pseudophase diagram of a free 90-mer is constructed using a microcanonical analysis approach, to demonstrate the effect of interaction range changes to the liquid and solid structural phases. Various adsorbed polymer structures are determined for different types of cylindrical substrates. Complete pseudophase diagrams of a 30-mer adsorbed at 5 different materials are constructed.

Index words: Monte Carlo Simulations, Parallel Tempering, Flexible Polymers, Structural Phase Transitions, Polymer Adsorption, Homopolymer, Nanocylinders

Parallel Monte Carlo Simulations of Coarse-Grained Flexible Polymers To Obtain Structural Phase Diagrams

by

Jonathan Gross

Dipl. Phys., University of Leipzig, Germany, 2010

A Dissertation Submitted to the Graduate Faculty
of The University of Georgia in Partial Fulfillment
of the
Requirements for the Degree

Doctor of Philosophy

Athens, Georgia

2013

Parallel Monte Carlo Simulations of Coarse-Grained Flexible Polymers To Obtain Structural Phase Diagrams

by

Jonathan Gross

Approved:

Major Professor: Michael Bachmann

Committee: David P. Landau
Steven P. Lewis
Shan-Ho Tsai

Electronic Version Approved:

Maureen Grasso
Dean of the Graduate School
The University of Georgia
August 2013

1785

The University of Georgia

Center for Simulational Physics

Parallel Monte Carlo Simulations of Coarse-Grained Flexible Polymers To Obtain Structural Phase Diagrams

Author:

Jonathan Gross

Supervisor:

Michael Bachmann

July, 2013

Acknowledgments

First of all, I would like to acknowledge my advisor Michael Bachmann. He was a great guide and mentor throughout the years, beginning with when we first started working together at Forschungszentrum Jülich in 2009 as part of the Guest Student Programme of the Jülich Supercomputing Centre. Through his recommendation I had my first encounter with graphics cards, which soon should become the main tool for my scientific research in the years to follow. At the same time I met Thomas Neuhaus, who is the second person I would like to give special thanks to. With his years of experience, he easily guided me through the advances in generalized ensemble methods, which he explained in great detail and with great passion. He not only became another important mentor to me, but also a collaborator and friend. The next person I greatly appreciate to have met is David Landau. I was so fortunate to be able take one of his graduate classes covering computer simulations. I highly respect the wide knowledge of Dr. Landau. He is able to put everything into the "bigger picture" and inspires to pursue every detail. I would also like to thank Thomas Vogel, who has become one of my closest collaborators and the "go-to-person" for everyday scientific and technical problems. But not only at the computer we had so much to talk about, but also at the grill while barbecuing. I would like to thank Mike Caplinger and Jeff Deroshia, for keeping our computers running and up to date, also Shan-Ho Tsai from the Georgia Advanced Computing Resource Center for letting me test drive their latest hardware acquisitions. And last but not least I would like to say a big "Thank you!" to my wonderful

family, my wife Nicole, and my two sons, Josua and Jakob, who sacrificed so much, leaving our family in Germany behind, to allow me to continue my research in the USA.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

A growing interest in the structural and dynamical behavior of polymers has promoted multidisciplinary research in recent years. The structure and the biological function of polymers, particularly of proteins, are strongly connected and misfoldings of proteins can cause severe illnesses. Over many decades, most of the scientific work has aimed at macroscopic systems such as single polymers with high polymerization degree or polymer melts. This was mainly due to the fact that on one hand the experimental equipment was not capable of revealing finer structural details on smaller scales and on the other hand, the theoretical treatment of even very simple polymer models was only possible in limits where cooperative effects on mesoscopic scales could be neglected. Computer simulations, however, made it possible to investigate systems strongly influenced by finite-size effects, that are not accessible via experiment. But even with computer simulations many questions remained unanswered, because of the limited available resources and the high requirements for reliable simulational results. Still, computer simulations have contributed substantially to a better understanding of phase transitions in general, including thermodynamic transitions in polymer systems which require mutual interaction of nonbonded monomers such as collapse, aggregation, and adsorption at substrates. Many of these studies were done in the traditional way of thinking that phase transitions only occur in very large systems close to the thermodynamic limit. This brought up the idea of finite-size scaling; a concept which has also successfully been applied to polymer systems [1, 2]. However, it has also turned out to be quite difficult to use this approach for transitions based on nucleation processes, where local effects, including competing effects of monomer arrangement inside the nucleus and at the surface, govern the whole nucleation process. That means, before crystallization can be perceived as a condensation process on macroscopic scales, the system has to pass a series of "subphase" transitions [3, 4], which depend on microscopic details and do not necessarily systematically scale with system size. This has been extensively studied for small atomic clusters [5–7] and, more recently, for polymers of finite length [8–15]. The simulation and analysis of such transitions is very demanding and requires computational methodologies and resources that have only recently become available. These methods, with generalized-ensemble Monte Carlo algorithms in the lead, even enable a different way of statistical analysis on the basis of the density of states or microcanonical entropy [16], which, although already having been known since the foundation of statistical mechanics, has widely been neglected in the long period of analytic studies, because the density of states is hardly accessible analytically. Phase-like transitions can be identified even in finite systems, like the here investigated polymer chains, using a systematic analysis of the inverse microcanonical temperature, as proposed recently [13].

Polymer adsorption at substrates also plays an important role in nanotechnology. Peptide adhesion on semiconductors [17, 18] for example enables new applications in fields like bioelectronics. The capacity of lithium batteries can be improved by coating nanowires with polymers [19]. Biosensors can be built by coating nanotubes with polymers [20]. Adsorbed polymers find applications in photonics [21] as well.

Because of this wide range of possible applications, it is necessary to understand the fundamental processes of structure formation of polymers and the adsorption on surfaces.

The structure of this dissertation is the following. In Chapter 2 the basics of statistical physics and thermodynamics that are necessary to understand the simulations and their analysis are summarized. The Monte Carlo method for computer simulations and associated algorithms and updates are explained in Chapter 3. How Monte Carlo simulations can benefit from employing state-of-the-art technologies is demonstrated in Chapter 4, where we implement replica-exchange (parallel tempering) Monte Carlo simulations for flexible polymers on modern GPUs, as published in [22, 23]. The influence of the interaction range of non-bonded monomers on the structural phase behavior is investigated in Chapter 5, following the publications [24, 25]. In Chapter 6 we study the adsorption of a polymer on nanocylinders with varying radii and surface attraction strengths. A summary of our research is given in Chapter 7.

Chapter 2

Essential thermodynamics

2.1 The microcanonical ensemble

Let us consider an isolated system with fixed energy E, volume V and particle number N. This system with total energy E is composed of two subsystems A and B that are able to exchange energy with the following condition:

$$E = E_A + E_B. \tag{2.1}$$

With a given energy E_A the number of microstates in subsystem A with that energy is given by $g(E_A)$, and $g(E_B) = g(E - E_A)$ is the number of microstates in subsystem B. The total number of microstates is then:

$$g(E_A, E_B) \approx g_1(E_A) \times g_2(E - E_A). \tag{2.2}$$

By applying the natural logarithm to Equation 2.2, the multiplicative relationship can be transformed into an additive one

$$\ln g(E_A, E - E_A) = \ln g_1(E_A) + \ln g_2(E - E_A). \tag{2.3}$$

In the total system the probability is the same for every state, i.e., $1/g(E)$. The probability of the energy states in subsystem A, however, is determined by the energy E_A. To determine the most probable value for $g(E_A)$ we have to find the maximum of $\ln g(E_A, E - E_A)$

$$\left(\frac{\partial \ln g(E_A, E - E_A)}{\partial E_A}\right)_{N,V,E} = 0. \tag{2.4}$$

Considering Equation 2.3 this can be written as

$$\left(\frac{\partial \ln g_1(E_A)}{\partial E_A}\right)_{N_A,V_A} = \left(\frac{\partial \ln g_2(E_B)}{\partial E_B}\right)_{N_B,V_B}, \tag{2.5}$$

with N_A, V_A and N_B, V_B being the number of particles and volumes of subsystem A and B, respectively. I introduce the following abbreviation

$$\beta(N, V, E) = \left(\frac{\partial \ln g(E)}{\partial E}\right)_{N,V}, \tag{2.6}$$

so that Equation 2.5 can be written as

$$\beta_A(N_A, V_A, E_A) = \beta_B(N_B, V_B, E_B). \tag{2.7}$$

If this equation is satisfied, subsystem A and subsystem B are in *thermal equilibrium*. This equilibrium is reached when $\ln g(E)$ is maximal. The second law of thermodynamics states that the entropy of a system with fixed energy E, volume V and particle number N reaches its maximum when the system is in thermodynamic equilibrium. Thus the entropy S of the system is given by

$$S(E) = k_B \ln g(E), \tag{2.8}$$

with k_B being the Boltzmann constant. The quantity $g(E)$ here counts the number of states with energy E. When dealing with continuous systems, $g(E)$ is also called the density of states. With Equation 2.7, or just $\beta_A = \beta_B$, we can derive the *temperature*, because two bodies in thermal contact are in equilibrium if their temperatures are the same. The definition of the microcanonical temperature is then

$$T(E) = \left(\frac{\partial S(E)}{\partial E} \right)^{-1}_{N,V}. \tag{2.9}$$

This also means $\beta = 1/k_B T$. The inverse thermal energy β is also sometimes referred to as *inverse temperature*, since the Boltzmann constant is set to unity in most cases. The microcanonical entropy in Equation 2.8 and its derivatives with respect to energy play an important role in identifying phase transitions, see Section 3.3.

2.2 The canonical ensemble

If the system is now allowed to exchange energy with a heat bath, it will reach equilibrium with the heat bath temperature. But even when the equilibrium is reached there will still be an exchange of energy between the system and the heat bath. The internal energy U of the system is constant for a given temperature T in thermal equilibrium, because the entropy reaches its maximum in equilibrium and $dU = TdS$. The internal energy can be related to the statistical average energy $U = \langle E \rangle = \text{const.}$ in equilibrium. The probability for the system to be in a microstate with energy E_i in the canonical ensemble is given by

$$p_i = \frac{1}{Z_{\text{can}}} e^{-E_i/k_B T}. \tag{2.10}$$

The mean energy then evaluates to

$$U = \langle E \rangle = \frac{1}{Z_{\text{can}}} \sum_i E_i \cdot e^{-E_i/k_B T}, \tag{2.11}$$

with the canonical partition sum

$$Z_{\text{can}} = \sum_i e^{-E_i/k_B T} = \sum_i e^{-\beta E_i} = \sum_E e^{-\beta E} \sum_i \delta_{E,E_i} = \sum_E g(E) e^{-\beta E}. \tag{2.12}$$

That means the canonical energy distribution can be written as

$$p_{\text{can,E}} = \frac{1}{Z_{\text{can}}} g(E) e^{-\beta E}. \tag{2.13}$$

High energy values only contribute significantly at high enough temperatures to the energy distribution. At low temperatures the low energy microstates dominate. The energy of the macrostate is then the single maximum energy peak of the energy distribution in Equation 2.13. (If the system is close to a phase transition with coexistence of phases, the energy distribution has more than one peak.) With increasing system size the energy peak gets more pronounced and in the thermodynamic limit, i.e., infinite system size, those states with energy E_{max} dominate the energy distribution. The mean energy then becomes $\langle E \rangle \approx E_{\text{max}}$ and $\partial S/\partial U \approx \partial S/\partial E_{\text{max}}$. Ultimately, this means that in the thermodynamic limit the microcanonical and the canonical temperatures are the same and the quantitative thermodynamic behavior of the system can be derived from canonical and microcanonical quantities alike. Refer to [26] for a more detailed elucidation.

2.3 Thermal fluctuations

Thermodynamic systems react to changes of their environment. Changes in temperature are reflected by changes in the specific heat, or changes to an external field have effects on the susceptibility of a magnetic system. For the systems investigated in this work the specific heat plays an important role. It measures the change in energy necessary when the temperature of the system is changed

$$c_V(T) = \frac{C_V(T)}{N} = \frac{1}{N}\left(\frac{\partial U}{\partial T}\right)_{N,V}, \tag{2.14}$$

from which

$$c_V(T) = \frac{1}{N}\frac{d\langle E\rangle(T)}{dT} = \frac{1}{Nk_BT^2}\left(\langle E^2\rangle - \langle E\rangle^2\right) \tag{2.15}$$

can be derived. This means the specific heat corresponds to the thermal fluctuations of the energy. Equation 2.15 can be generalized for an arbitrary observable $\mathcal{O}$ that changes with temperature

$$\frac{d\langle\mathcal{O}\rangle}{dT} = \frac{1}{k_BT^2}\left(\langle\mathcal{O}\cdot E\rangle - \langle\mathcal{O}\rangle\cdot\langle E\rangle\right). \tag{2.16}$$

This is especially useful if $\mathcal{O}$ is some kind of order parameter that allows the identification of different phases or structures. The temperature at which Equation 2.16 assumes an extremum, can be used to estimate the actual transition temperature. In systems with finite sizes, the transition temperature estimates do not necessarily coincide perfectly for different fluctuation quantities. In the thermodynamic limit, however, these temperatures converge to the same phase transition temperature. With this in mind, whenever the term “phase transition” is mentioned when referring to a system with finite size, pseudo-phase transition or structural transition would be more appropriate. But for simplicity and ease of writing “phase transition” might be used, knowing real phase transitions only appear in the thermodynamic limit.

Chapter 3

Methods

3.1 Monte Carlo simulations

In Monte Carlo simulations [2, 26] the thermodynamical behavior of a system is investigated by preparing "random" states and calculating ensemble averages of quantities, rather than solving Newton's equations of motion as in molecular dynamics. The process of generating theses random samples to calculate ensemble averages is called a Markov process. A Markov process is a mechanism to transform a system in state $\mathcal{C}$ to a new state $\mathcal{C}'$. This transformation only depends on the original state $\mathcal{C}$ and the transition probability $P(\mathcal{C} \to \mathcal{C}'; \Delta\tau)$ in a "time step" $\Delta\tau$, that means it does not depend on any information about former states the system may have been in. The time step $\Delta\tau$ is not real time, but "simulation time" to perform an update to the system. Such a process is driven by the master equation

$$\frac{\Delta p(\mathcal{C})}{\Delta\tau} = \sum_{\mathcal{C}'} \left(p(\mathcal{C}')P(\mathcal{C} \to \mathcal{C}'; \Delta\tau) - p(\mathcal{C})P(\mathcal{C}' \to \mathcal{C}; \Delta\tau) \right) . \tag{3.1}$$

$p(\mathcal{C})$ is the probability distribution of the system to be in state $\mathcal{C}$, which is independent of time, i.e., a stationary state. In equilibrium, i.e., $\Delta p(\mathcal{C})/\Delta\tau = 0$ the right side of Equation 3.1

has to vanish. Since this theoretically would allow cycling processes, which would not lead to the desired physical statistical equilibrium ensembles, it is demanded that every term of the sum on the right side vanishes individually. This condition is called *detailed balance*

$$\frac{P(\mathcal{C} \to \mathcal{C}')}{P(\mathcal{C}' \to \mathcal{C})} = \frac{p(\mathcal{C}')}{p(\mathcal{C})}. \tag{3.2}$$

A Monte Carlo update would not only have to satisfy detailed balance, but also has to guarantee that every point in phase space can be reached. It should be possible to reach any state of the system from any other state, in a finite number of updates with a non-zero probability. This principle is called *ergodicity.*

3.1.1 Metropolis importance sampling

When simulating physical systems with Monte Carlo techniques, it would be highly inefficient to accept all proposed updates, i.e., set $P(\mathcal{C} \to \mathcal{C}') = 1$, which would be *simple sampling* Instead, since we are interested in the thermodynamical behavior of our investigated systems, we can tune the transition probability in Equation 3.2 so that the important states, i.e., the most probable states, are sampled most often. This is called *importance sampling.* In 1953 Metropolis et al. [27] proposed what today is widely know as the Metropolis method. The probability of a state $\mathcal{C}$ of a system in the canonical ensemble is given by $p(\mathcal{C}) \sim e^{-\beta E(\mathcal{C})}$, with $\beta = 1/k_B T$. To the system in state $\mathcal{C}$ with energy $E(\mathcal{C})$, an update is proposed to transform it into state $\mathcal{C}'$ with energy $E(\mathcal{C}')$. For a polymer system such an update could be the displacement of one monomer or the rotation of a part of the chain. In a spin system it could be the change of one spin or a cluster of spins. The acceptance probability of a proposed update to the system is given by

$$P(\mathcal{C} \to \mathcal{C}') = \min(1, e^{-\beta \Delta E}), \tag{3.3}$$

where $\Delta E = E(\mathcal{C}') - E(\mathcal{C})$. This means that an update where $\Delta E \leq 0$ would always be accepted. If $\Delta E > 0$ a random number $r \in [0, 1)$ is generated. Only if $r < e^{-\beta \Delta E}$ such an update would be accepted; it would be rejected otherwise. A simulation with N updates would generate a time series for an observable $\mathcal{O}$. The mean value calculated from that time series would than be an estimate for the canonical expectation value of that observable

$$\overline{\mathcal{O}} = \frac{1}{N} \sum_{i=0}^{N} \mathcal{O}_i \approx \langle \mathcal{O} \rangle. \tag{3.4}$$

(For an infinitely long simulation ($N \to \infty$): $\overline{\mathcal{O}} = \langle O \rangle$.) There are, however, a few drawbacks to the Metropolis algorithm. At low temperatures the widths of the energy distributions become extremely narrow. Also the Boltzmann factor $e^{-\beta E}$ is very close to zero, which is reflected in a close-to-zero acceptance rate of proposed updates at low temperatures. The simulation might get trapped in a low-energy state and impair the statistics. Another problem with Metropolis arises for simulations near transition temperatures. In case of a first-order transition, the energy distribution has two peaks of the same height at the transition point, representing the energies of the two coexisting phases. Since the energies between the two phases are suppressed entropically, it is very unlikely to sample both peaks with equally good statistics. Instead the simulation will again get trapped in one of the two states. For second-order phase transitions, the specific heat diverges at the critical temperature. Since the specific heat is connected to the energy fluctuations, (compare Equation 2.15), simulations near the transition temperature will suffer from a *critical slowing down.* This slowing down is caused by the fact that even small update to the system might result in a dramatic change in energy and thus a very large value for $\Delta E > 0$ in Equation 3.3, which will lower the acceptance rate significantly.

3.1.2 Parallel tempering

To resolve the drawbacks of Metropolis sampling, several *generalized-ensemble* methods, such as parallel tempering [28–30], simulated tempering [31, 32], multicanonical sampling[33–36], and Wang-Landau sampling [37, 38], have been developed. The most popular one, because of the very easy and straightforward implementation, is parallel tempering, which is also known as replica-exchange Monte Carlo. The idea behind parallel tempering is simulating n_r copies or replicas of the same system

$$\mathcal{C} = \{C_1, C_2, \dots C_{n_r}\} \tag{3.5}$$

at different temperatures $T_1 < T_2 < \dots < T_{n_r}$. Each replica is simulated using the Metropolis algorithm, for a given number of sweeps, completely independent from all other replicas. Commonly a sweep is equal to N single updates for a system with N particles. After this an exchange of temperatures between replica C_i and C_j is proposed. The acceptance probability for such an exchange is given by

$$P(\mathcal{C}_{\text{old}} \to \mathcal{C}_{\text{new}}) = \min[1, \exp(\Delta\beta\Delta E)], \tag{3.6}$$

where $\Delta\beta = \beta_j - \beta_i$ is the difference of the inverse temperatures, $\Delta E = E(C_j) - E(C_i)$ is the energy difference, and $\mathcal{C}_{\text{old}}$ and $\mathcal{C}_{\text{old}}$ are the sets of replicas before and after the exchange, respectively. This equation satisfies detailed balance. The exchange of temperatures between copies allows each copy to be heated up and cooled down throughout the whole simulated temperature interval. This not only decreases autocorrelation time, but also helps a conformation that is trapped in a local low energy state to escape and improves the overall sampling. The most important advantage of parallel tempering is its embarrassingly parallel nature, which makes it especially easy for parallel implementation on modern multicore processors, cluster computers and advanced accelerators, such as graphics processing units. Even though parallel tempering vastly improves the sampling efficiency when compared to Metropolis, similar restrictions apply. The temperature spacing at low temperatures has to be chosen in a way that there is enough overlap in the energy histograms to achieve high enough acceptance rates between low temperature replicas. That being said, parallel tempering is not the optimal method to unravel the lowest temperature or ground-state behavior of a system, but it still has proven to be a robust simulation tool.

3.2 Monte Carlo updates

The one thing all Monte Carlo updates have in common is their random nature. All Monte Carlo moves depend on random numbers. Each update requires multiple random numbers to propose a change to the system, and the acceptance of each Monte Carlo move again depends on a random number. To obtain good statistics to calculate the desired thermodynamical averages and their statistical errors, it is necessary to perform simulations with a vast number of sweeps, often in the order of 10^9. This imposes serious requirements to the random number generator (RNG). The RNG has to have a large periodicity, and correlations between numbers in the sequence and other patterns need to be avoided. For all simulations in this work, a RNG developed by Marsaglia and Zaman [39], which is known as *RANMAR*, was used. This RNG passes all known test suites for the "goodness" of random numbers.

3.2.1 Temperature-dependent single monomer displacement

The most basic and most used Monte Carlo move of all performed simulations is a single monomer displacement. For this update, one monomer is selected randomly and a previously generated displacement vector $\vec{d}$ is added to its coordinates. The displacement vector is created as follows. Three random numbers $r_i \in [0, 1)$ are generated. The Cartesian coordinates of the displacement vector $\vec{d}$ are then calculated like this

$$d_i = (r_i - 0.5) \cdot \alpha(T) \qquad i = \{x, y, z\}. \tag{3.7}$$

If we neglect the factor $\alpha(T)$ for a while, this would create a three dimensional vector, whose coordinates have a value between $-1/2$ and $1/2$. In terms of the polymer models investigated this would be a fairly large move. Considering the change in energy and the dense structures for theses polymers at low temperatures, large updates would results in a low acceptance rate, effectively slowing down the simulation dramatically. This is where the temperature dependent factor $\alpha(T)$ comes into play. It is initially set to 1. During the equilibration phase of each simulation the acceptance ratio of accepted moves versus proposed moves is measured. If the acceptance rate is higher than 0.55, $\alpha(T)$ is multiplied by a factor larger than, but close to 1, and thus making the interval for the coordinates wider. If, on the other hand, the acceptance ratio drops below 0.45, $\alpha(T)$ is multiplied by a factor smaller than, but close to 1. The modification factor $\alpha(T)$ quickly converges to an optimal value so that the acceptance rate of the monomer displacement is roughly 50% at all temperatures. In the production run of each simulation $\alpha(T)$ is then a fixed value for each temperature and is not changed anymore.

3.2.2 Slithering snake moves

The slithering snake update involves removing either the first monomer from the polymer chain and attaching it to the end of the chain, or cutting the last monomer and moving it to the front. The easiest implementation of this update move would be to calculate the vector between first and second monomer, then removing the first monomer and adding the vector to the coordinates of the last monomer. At the newly calculated position the previously cut monomer would be pasted, and vice versa, of course. This would allow a more rapid change to the polymer at fairly high temperatures, compared to the displacement move, but virtually all such moves would be rejected at dense, low temperature conformations. Nevertheless in combination with parallel tempering even the lower temperature conformations can profit from this update, since they are eventually exchanged to higher temperatures, at which slithering snake updates are accepted, and thus improve the overall sampling.

3.2.3 Crankshaft update

In a crankshaft move a portion of the polymer is rotated around an axis between two randomly chosen monomers by a random angle φ. The ends of the polymer are excluded as a possible pair though, because this would lead to a conformation with the same energy due to symmetry. Like the slithering snake update, such an update introduces a more global update to the polymer, but only is efficient at high temperatures. Again, for low temperatures virtually all of such moves are rejected.

3.3 Non-iterative multiple histogram reweighting

Parallel tempering or Metropolis simulations at different temperatures T_i usually yield energy histograms $H_i(E)$ as a direct result. These histograms can be combined to obtain the density of states, which makes it possible to perform an inflection-point analysis as proposed in [13]. Each histogram is an estimate for the density of states $g_i(E) \propto H_i(E)\exp(\beta_i E)$ up to an unknown constant which is different for each β_i. For the analysis of the microcanonical entropy and its derivatives, it is convenient to continue working with the ratio

$g_i(E+\Delta E)/g_i(E)$. Entropic differences can then be written as [40, 41]

$$\begin{aligned}\Delta S_i(E) &= S_i(E+\Delta E) - S_i(E) \\ &= \ln[g_i(E+\Delta E)/g_i(E)] \\ &= \ln[H_i(E+\Delta E)] - \ln[H_i(E)] + \beta_i \Delta E.\end{aligned} \tag{3.8}$$

Introducing the following weight

$$w_i(E) = \frac{H_i(E+\Delta E)\cdot H_i(E)}{H_i(E+\Delta E) + H_i(E)}, \tag{3.9}$$

which is reciprocally proportional to the variance of $\Delta S_i(E)$, yields the weighted average over all histograms:

$$\overline{\Delta S}(E) = \frac{\sum_i \Delta S_i(E) w_i(E)}{\sum_i w_i(E)}. \tag{3.10}$$

This result can be used for an approximation of the inverse microcanonical temperature, defined as

$$\beta(E) \equiv T^{-1}(E) = \left(\frac{dS}{dE}\right)_{N,V} \approx \frac{\overline{\Delta S}(E)}{\Delta E}. \tag{3.11}$$

To improve the quality of the estimate we start off with the five-point method[1] for the derivatives instead of the difference quotient used in Equation (3.8). Analogously, we can also directly compute the second derivative[2] of the microcanonical entropy. Inflection-point analysis of $\beta(E)$ [13] allows us not only to locate, but also to classify transitions in the system. In this scheme, a transition is of *first order*, if the derivative of $\beta(E)$ at the inflection point

[1] $f'(x) \approx \frac{-f(x+2\Delta x)+8f(x+\Delta x)-8f(x-\Delta x)+f(x-2\Delta x)}{12\Delta x}$

[2] $f''(x) \approx \frac{-f(x+2\Delta x)+16f(x+\Delta x)-30f(x)+16f(x-\Delta x)-f(x-2\Delta x)}{12\Delta x^2}$

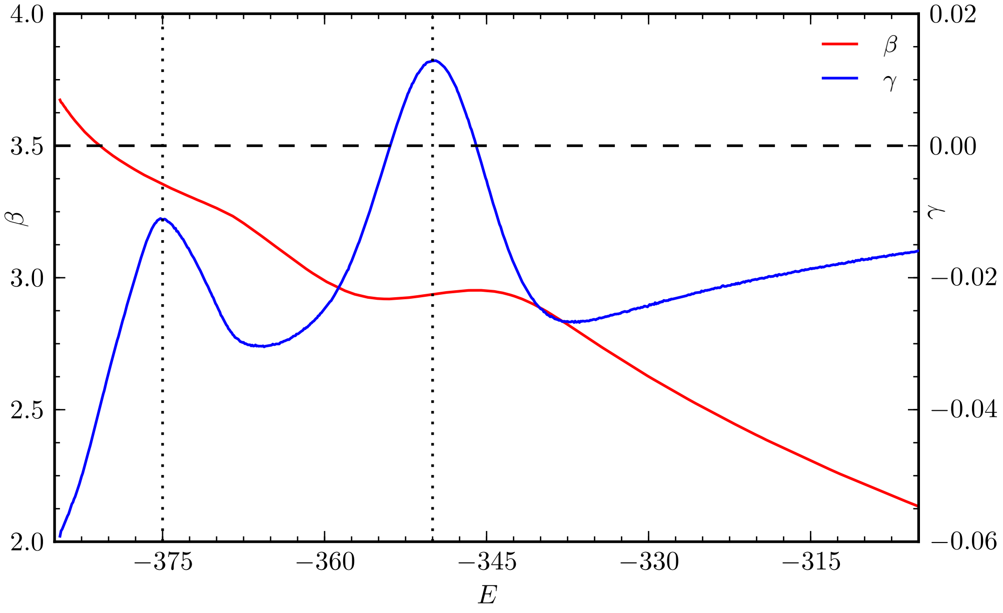


Figure 3.1: Comparison of transition signals in β and γ. The peak in $\gamma(E)$ at $E \approx -350$ corresponds to a first-order-like transition. Another peak in γ at $E \approx -375$ marks a second-order-like transition. Both transitions show up as inflection points in $\beta(E)$ at the respective energies. If the peak value of $\gamma(E)$ for a given transition is positive, we classify the transition as of "first order". In line with this, a "second-order" transition is classified by a negative value for the γ-peak.

has a positive peak value $\gamma(E) = d\beta(E)/dE > 0$. Consequently, an inflection point with a negative peak value corresponds to a *second-order*-like transition.

In the example illustrated in Figure 3.1, $\beta(E)$ has an inflection point at $E \approx -350$ and the corresponding peak in $\gamma(E)$ is positive. The associated transition is, therefore, first-order-like. Another inflection point of $\beta(E)$ is found at $E \approx -375$. The peak in $\gamma(E)$ for this energy is below zero, indicating a second-order-like transition.

Chapter 4

Polymer Simulations on general-purpose GPUs

In recent years, graphics processing units (GPUs) have become very powerful driven by the computer gaming industry. Designed to calculate complex scenes of geometrical bodies in real time, graphic cards have evolved into parallel multithreaded, manycore processors with extremely high memory bandwidth. With the release of NVIDIA's Compute Unified Device Architecture – or *CUDA* for short – in 2007, and with OpenCL two years later, this enormous computing power was accessible for every programmer outside the computer graphics field. It quickly became popular in the scientific computing community, with applications in astronomy [42, 43], medicine [44, 45], financial market analyses [46], molecular dynamics [47, 48], Monte Carlo studies of spin systems [49–51] and Quantum Monte Carlo [52]. This chapter will give a short introduction to the concepts associated with GPU programming and discusses how these can be used to accelerate Monte Carlo simulations of off-lattice polymer models.

4.1 Programming GPUs

4.1.1 GPU architecture

Modern GPUs are typically comprised of a number of streaming multiprocessors (SM). Different GPU generations have different features, called *compute capabilities*. For a complete comparison of the compute capability versions please refer to [53, Chapter 3]. Whenever necessary, the differences in these capabilities are pointed out. These SMs differ from traditional processor design. There are many more arithmetic logical units (ALU) on a GPU than on a CPU, allowing the GPU to process many computation threads at once. The vast number of ALUs comes with the sacrifice of cache and control units. Thus all cores of a SM must perform the same operation at the same time. Code branching has only limited support and is discouraged.

The computing power of the GPU comes from the ability to execute tens of thousands of calculations concurrently, performing the same operation on different data. The *kernel* – the main function of a GPU program – runs the same code on a number of threads in parallel. These threads can be arranged in a grid and block layout to match a decomposition of the investigated system into parts that can be calculated independently.

The grid of independent thread blocks, as shown in Figure 4.1, can be ordered in up to two dimensions for devices with compute level less than 2.0 (i.e. GT200-based architecture) and up to three dimension for compute capabilities greater than 2.0. The grid layout can be changed during the runtime of the program but is immutable for a running kernel. It is possible to launch different kernels in the same program. Since the introduction of the *Fermi* architecture, i.e., compute capability 2.0, the concurrent execution of multiple kernel functions is possible. Before that different kernels had to be executed successively.

Threads, i.e., individual processes, are bundled in thread blocks that can be arranged within the grid in three dimensions with a maximum size of $512 \times 512 \times 64$ (GT200-based

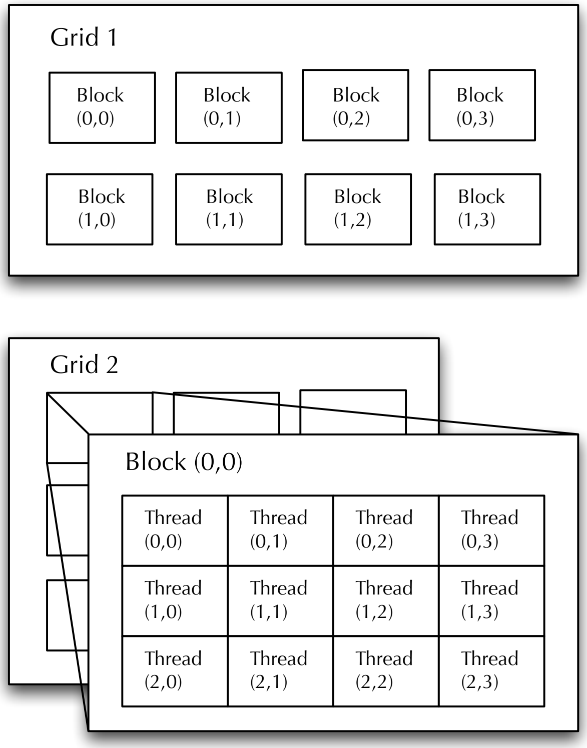


Figure 4.1: Schematic diagram of the grid layout with thread blocks.

cards) or $1024 \times 1024 \times 64$ on Fermi. However, there is also a limit for the maximum number of threads within a block, i.e., 512 threads or 1024 threads for GT200 or Fermi, respectively. Valid block layouts for GT200 cards, for example, are: $512 \times 1 \times 1$, $128 \times 2 \times 2$, $1 \times 512 \times 1$, $1 \times 8 \times 64$, and $2 \times 4 \times 64$. Each thread can be identified by a unique id, that can be calculated from its coordinates within the grid.

At the execution of the kernel, 32 threads are grouped together into a *warp*. These warps are then assigned to a SM, and the group of threads is executed concurrently. The best efficiency is achieved, when all of the threads within a warp have to perform the same operations. As mentioned earlier branching in the execution of the code can lead to efficiency loss, because the operations will be serialized, thus leading to underutilization of the cores.

In practice it is preferable to start many more threads than cores that exist on the device. While the number of SMs and cores per SM is subject to changes within different GPU generations, the warp size is a fixed quantity. This means the same program can run on a wide range of devices with different capabilities, harnessing the maximum potential, because the distribution of the warps is done in hardware on the device. This feature is called *transparent scalability*.

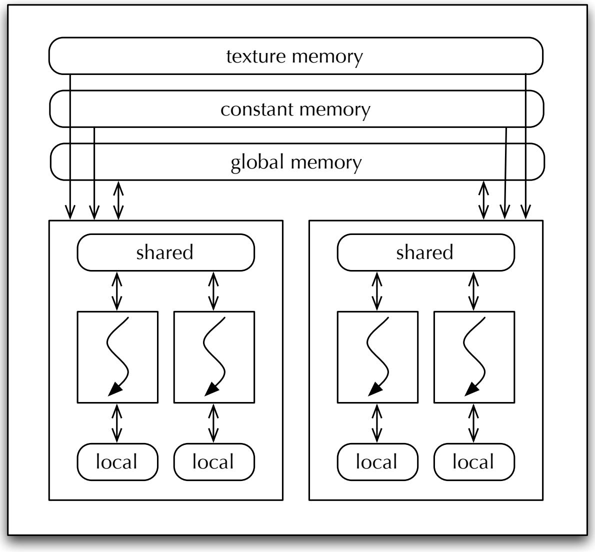


Figure 4.2: Schematic diagram of the memory hierarchy and layout on current GPUs.

Graphics cards are equipped with several types of memory, see Figure 4.2. The largest memory available is the *global memory*, which is typically in the range of 1 to 6 GBs today. The global memory can be read and written by specific functions from the CPU side – also called *host* –, and every thread on the GPU *device* has read-write access to global memory as well. A large memory on the device is necessary since it is not possible for the device to access the RAM of the host. This means all data that need to be processed by the GPU have to be copied to the device for calculation, and the results need to be copied back to RAM for evaluation. One disadvantage of the global memory is its rather high latency. Reading and writing to global memory take of the order of hundreds of clock cycles. The *constant memory* is a special region in global memory. It is initialized on the host and is read-only on the device. During the first read of data in constant memory, its content is cached in lower latency on-chip cache memory, so that consecutive access to the same data is much faster. The size of constant memory is very limited. It is only 64 KB for all card generations. The *texture memory* is another read-only memory, that has to be initialized on the host before kernel execution. Texture memory has its roots in the original purpose of graphics processing and is also used for caching. With increased cache memories since the introduction of Fermi, this memory has become mostly obsolete for GPGPU programming. *Registers* and *shared memory* are very fast, low latency on-chip memories. Access to these memories is usually a lot faster than global memory. Registers are available per thread, and only that one thread they are assigned to has read-write access. Shared memory is allocated to a thread block within the same SM. All threads in that thread block have read and write access to shared memory. The total number of registers per SM is 16384 for GT200-based chips and is twice that number for Fermi. Shared memory sizes are 16 KB on GT200 and 48 KB on Fermi [54]. There is also *local memory*, which is a per thread memory region residing in global memory. The size of local memory available per thread is 16 KB for GT200 and was increased to 512 KB on Fermi. However, access to local memory is not cached, so it suffers the same latency drawbacks as global memory accesses.

CUDA name	OpenCL name
global memory	global memory
constant memory	constant memory
shared memory	local memory
local memory	private memory
grid	NDrange
thread block	work group
thread	work item

Table 4.1: Comparison of CUDA and OpenCL nomenclature.

In principle, the concepts of thread and memory layout are the same for CUDA and OpenCL. Even scheduling and grouping of threads works the same for both approaches. But the nomenclature is different. Table 4.1 summarizes the most important differences. In particular *local memory* depicts different memories in the two nomenclatures and is not to be confused.

4.1.2 CUDA and OpenCL

The CUDA software developer toolkit comes in two variants, a high-level C/C++ interface to GPU functions, the so-called CUDA runtime API, and a lower level programming layer the CUDA driver API. The toolkit contains a set of extensions to the C programming language to accomplish the most common tasks in GPU programming, like memory management and operations, but also new data types for mathematical calculations. For a detailed description of the CUDA programming language see [53, 55]. CUDA runs exclusively on NVIDIA hardware. However an open standard is being developed and supported by all major GPU vendors and other hardware developers. The Open Computing Language[56] (OpenCL) is maintained by the Khronos Group. All the same concepts apply to CUDA and OpenCL. OpenCL is comparable to the CUDA driver API, being closer to the hardware and more explicit. But high level interfaces for various programming languages such as C++, Python and Java exist that make accelerated GPU computing available through comfortable functions. A scheme of the sequence of a typical parallel GPU program is shown in Figure 4.3. The program starts like any other program on the host with the initialization of variables and data. For a GPU program one also has to allocate and initialize all memories that are needed for the calculations on the GPU in addition to those on the CPU. When all data are copied to the device memory, the kernel function is called and the program now spreads into the parallel portion. Multiple threads concurrently perform the calculations on the GPU. The execution of the kernel on the GPU is completely independent from the host, which in

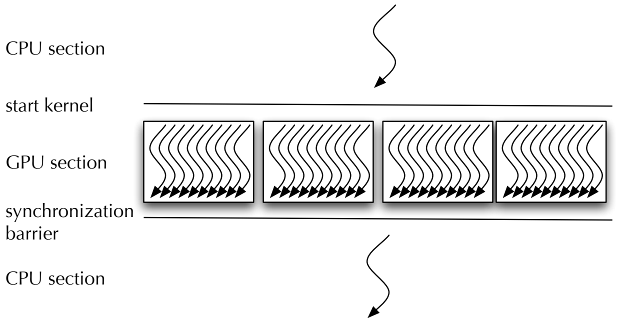


Figure 4.3: Scheme of the sequence in a parallel GPU program.

principle could proceed in its own execution. For this reason a synchronization barrier has to be implemented in the main program to wait for the calculations on the device to finish. After the kernel finishes its execution, the results of the computation need to be copied back to the host for further processing. Since compute capability 2.0, it is possible to have more than one kernel running at the same time.

4.2 Model and implementation

4.2.1 Elastic, flexible polymer model

As an example for a molecular system, we consider the following elastic bead-spring homopolymer chain, as illustrated in Figure 4.4. All monomers interact via a shifted and truncated pairwise Lennard-Jones potential

$$U_{\mathrm{LJ}}^{\mathrm{mod}}(r_{ij}) = U_{\mathrm{LJ}}(r_{ij}) - U_{\mathrm{LJ}}(r_c), \tag{4.1}$$

$$U_{\mathrm{LJ}}(r_{ij}) = 4\epsilon \left[\left(\frac{\sigma}{r_{ij}} \right)^{12} - \left(\frac{\sigma}{r_{ij}} \right)^{6} \right], \tag{4.2}$$

where r_{ij} is the distance between two monomers i and j. The Lennard-Jones parameters are $\epsilon = 1$ and $\sigma = 2^{-1/6} r_0$, the minimum of the potential is at $r_0 = 0.7$ and the cutoff radius is $r_c = 2.5\sigma$. The shift in the potential is done, to avoid a discontinuity at the cutoff radius. To model the bonds between adjacent monomers, we use the finitely extensible nonlinear elastic (FENE) anharmonic potential

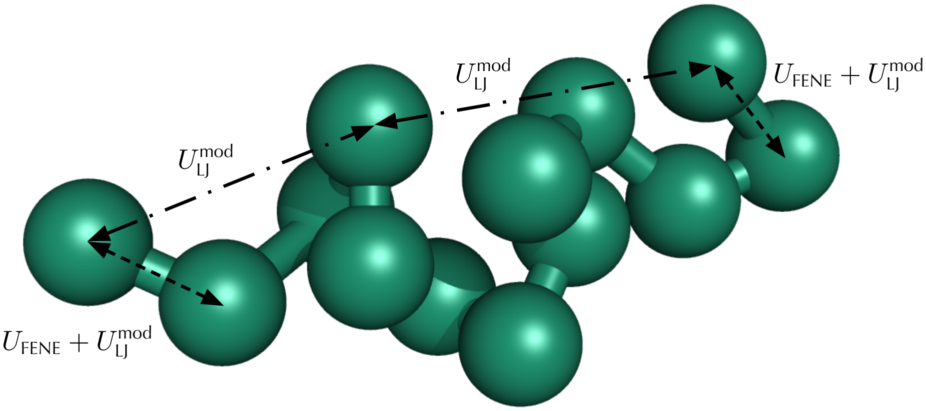

$U_{\mathrm{LJ}}^{\mathrm{mod}}$
$U_{\mathrm{FENE}} + U_{\mathrm{LJ}}^{\mathrm{mod}}$
$U_{\mathrm{LJ}}^{\mathrm{mod}}$
$U_{\mathrm{FENE}} + U_{\mathrm{LJ}}^{\mathrm{mod}}$

Figure 4.4: Rendering of a 13-mer to illustrate monomer–monomer interaction. Bonded neighbors interact via a FENE potential U_{FENE} and a modified Lennard-Jones potential $U_{\mathrm{LJ}}^{\mathrm{mod}}$. Between all non-bonded monomer pairs only the Lennard-Jones potential is effective.

$$U_{\mathrm{FENE}}(r_{ii+1}) = -\frac{K}{2} R^2 \log \left[1 - \left(\frac{r_{ii+1} - r_0}{R} \right)^2 \right]. \tag{4.3}$$

We follow the parametrization found in [9, 10], to be able to compare results. The minimum of the FENE potential is at $r_0 = 0.7$ and the potential diverges for $r \to r_0 \pm R$ with $R = 0.3$. The spring constant K equals 40. The total energy of a conformation $\mathcal{C} = (\vec{r}_1, \cdots, \vec{r}_N)$ for

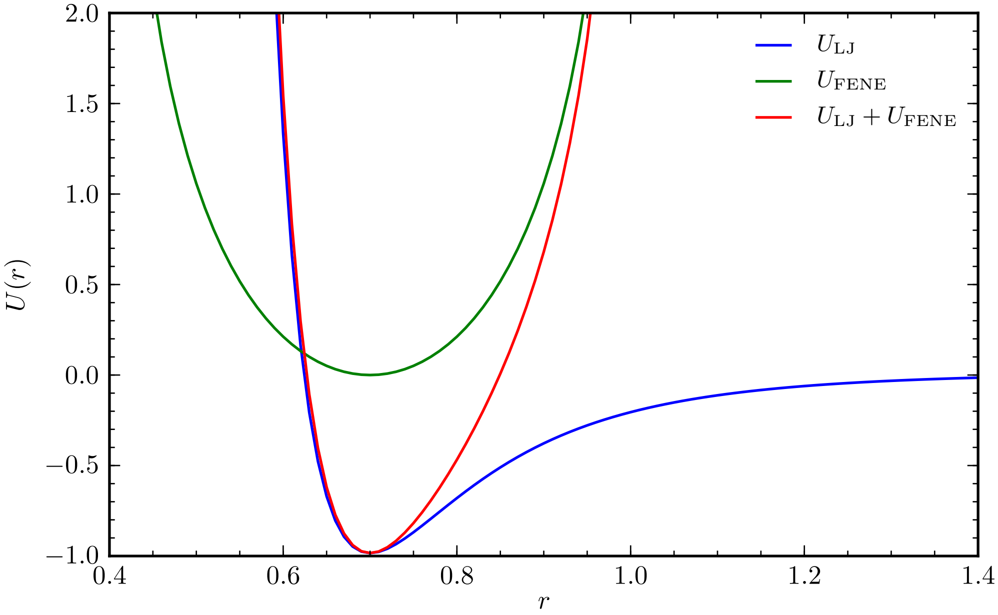


Figure 4.5: The Lennard-Jones potential U_{LJ} describes the interaction between non-bonded monomers. The bonds are modeled with a finitely extensible nonlinear elastic potential U_{FENE} in addition to the Lennard-Jones term. The plot shows the form of the standalone FENE potential and the sum of FENE and Lennard-Jones, which is the effective bond-potential.

a chain with N monomers is then given by

$$E(\mathcal{C}) = \frac{1}{2}\sum_{\substack{i,j=1\\ i\neq j}}^{N} U_{\mathrm{LJ}}^{\mathrm{mod}}(r_{ij}) + \sum_{i}^{N-1} U_{\mathrm{FENE}}(r_{ii+1}). \tag{4.4}$$

The form of the potentials is plotted in Figure 4.5, showing the Lennard-Jones potential for the non-bonded monomers and the effective bond potential, the sum of Lennard-Jones and FENE potential. The simulations were carried out using replica-exchange Monte Carlo as described in Section 3.1.2.

4.2.2 GPU implementation

In this section the focus lies on implementational details. The listings in this section contain CUDA specific syntax emphasizing the most relevant parts of a CUDA program. All calculations are performed using single-precision floating-point operations. Single-precision operations are usually faster by a factor of 8 than double-precision operations, because one double-precision unit is shared among 8 ALUs. For NVIDIAs professional line of Tesla cards the factor between single and double precision is only two.

Listing 4.1: Specification of the thread layout and kernel call within the main program.

```
dim3 dimGrid(NCONFS);
dim3 dimBlock(BS);
run<<<dimGrid,dimBlock>>>(d_confs,d_rnds,d_energies,d_rees,
    d_rgyrs);
cudaThreadSynchronize();
```

Listing 4.1 shows how the main GPU function in invoked from the host. First, one needs to set up the dimension for the grid of threads. In this case, the three-dimensional variable dimGrid is initialized with only one dimension, the number of replicas. This means that every replica of the system runs independently in its own thread block. The size of such a thread block dimBlock is set to a constant BS which depends on the number of beads in the polymer. Again the thread block can be three-dimensional, but only one dimension is used here. Details will be discussed in Section 4.3. After setting up the grid layout, the kernel run is called with the given layout embraced by triple chevrons and a list of arguments. Because a kernel function call returns the control immediately back to the main program, the synchronization barrier cudaThreadSynchronize() instructs the main program to wait until GPU is finished with the execution of the kernel. In our implementation the exchange of replicas is done on the CPU, while the GPU is used for the Metropolis algorithm with the expensive energy calculation, see Figure 4.6.

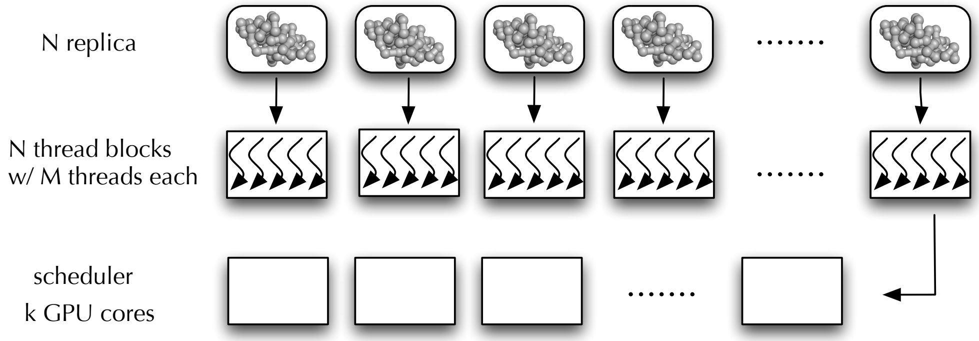


Figure 4.6: Schematic representation of parallel implementation of replica-exchange simulation on GPUs. Each of the N simulated replicas is assigned to one thread block with M threads. These thread blocks are then processed by the k streaming multiprocessors on the GPU. Usually $N > k$ for optimal usage of the GPU resources.

Listing 4.2: The kernel function – showing the usage of shared memory and the main work loop. TX is the unique ID for each thread.

```
__global__ void run
  (Polymer* d_confs, float* d_rnds, float* d_energies,
   float* d_rees, float* d_rgyrs) {
  int id = blockIdx.x;
  __shared__ Polymer ps;
//.. initialization of some variables
  if (TX == 0)
    ps = d_confs[id];
  __syncthreads();
  while (n < NSWEEPS) {
    oneSweep(&ps, rnds, n);
    ener = energy(&ps);
    if (TX == 0) {
```

```
14       Ree = endToEndDistance(&ps);
15       Rgyr = radiusOfGyration(&ps);
16       d_energies[n+offset] = ener;
17       d_rees[n+offset] = Ree;
18       d_rgyrs[n+offset] = Rgyr;
19       n++;}
20     __syncthreads();
21   }
22   if (TX == 0)
23     d_confs[id] = ps;
24   __syncthreads();
25 }
```

A pruned record of the insides of the kernel function is shown in Listing 4.2. In line 5 of that listing, the index of the current thread block is assigned to an integer variable and is used to link this thread block to a specific replica. The shared memory for a local copy of the replica is allocated in line 6. This means that all threads within this block, and only those, have fast access to the copy. The actual process of copying the selected replica to shared memory is shown in lines 5–9, where only one thread is assigned to copy the polymer conformation from the array of conformations d_confs in global memory to shared memory. The barrier __syncthreads() lets all other threads of that block wait for the copying to finish before proceeding with the actual calculation. In line 11 the function oneSweep containing the Metropolis algorithm and the energy calculation is executed by all thread in that block in parallel. The details of the parallel implementation and CUDA specifics are shown in Listings 4.3 and 4.4. Again, only one thread is used in lines 13–20 to collect statistics. The local copy of the polymer is copied back to global memory in line 23 for further evaluation on the CPU.

Listing 4.3: Calculation of pairwise FENE interactions.

```
1 if (TX < N-1) {
2   r = distance(p, TX, TX+1);
3   energy[TX]+=-0.5*K*R*R*__logf(1-((r-r0)/R)*((r-r0)/R));
4 }
5 __syncthreads();
```

Due to the fact that every thread executes the same code, it is possible to insert conditions based on the id of the thread to alter what each thread actually calculates. Since in this model there are only pairwise interactions between monomers, it is possible to parallelize the calculation of the energy. For the FENE part of the potential this is particularly straightforward, because only consecutive monomers are involved, see Listing 4.3. Each thread calculates only one pair of monomers as part of the potential. Line 3 of that listing shows the call to a special math function __logf(). This is a hardware accelerated single precision implementation of the natural logarithm, providing additional speed up.

Listing 4.4: Calculation of pairwise Lennard-Jones potential.

```
 for (int i = 0; i < (N >> 1); i++) {
  if (TX < N) {
   if (TX > i) {
    idx1 = i;
    idx2 = TX;
   }
   else {
    if ( (N & 1) == 0) {
     idx1 = N - 1 - i;
     idx2 = idx1 + TX;
```

```
        }
        else {
          idx1 = N - 2 - i;
          idx2 = idx1 + 1 + TX;
        }
      }
      r = distance (p, idx1, idx2);
      if (r < rc && idx1 != idx2 ) {
         float rs6 = __powf (sigma / (r - ls), 6.0f);
          energy[TX] += 4.0 * epsilon * ( (rs6*(rs6-1)) - E_lj_rc
              );
      }
    ...
__syncthreads();
```

For the Lennard-Jones part of the potential, it is not that trivial. In principle it would be possible to calculate all possible pairs prior to the simulation, but this approach would easily exceed the maximum number of threads per block even for relatively short polymer chains. Instead of calculating and itemizing all possible pairs, indices for pairs are calculated "on-the-fly" and are assigned to threads as shown in Listing 4.4. For both parts of the potential, *one pair* of monomers is assigned to *one thread* to perform the actual calculation of the energy. The results of those calculations are stored in the array energy. When all threads are finished with their part, a *parallel reduction* is performed on this array to obtain the total energy. Instead of using only a single thread and a loop for the summation of the array elements, multiple threads calculate different portions of the sum. A very detailed explanation of parallel reduction on GPUs and different optimization strategies are discussed in Ref. [57].

4.3 Results

4.3.1 Performance comparison

To compare the performance of two different implementations, we define a speed-up factor as follows,

$$S_p = \frac{t_{\mathrm{CPU}}}{t_{\mathrm{GPU}}}, \tag{4.5}$$

where t_{CPU} is the execution time on a single CPU core and t_{GPU} is the runtime on the respective device. All runtimes were measured with cutil-timers, which are wrapper functions to the standard C library call gettimeofday. This ensures a consistent time measurement on a variety of systems, and to measure GPU and CPU times alike. On the CPU side, only the time taken for the actual calculation was measured. No initialization of variables or any file operations were included in the measurements. To consider the extra overhead which comes with GPU computing, the time taken to copy the data hence and forth the device has been included along with the time taken for the sweeps. For the speed comparison, short runs of 400 sweeps per replica were performed. Every 1000 sweeps the replicas were copied back to the host to perform the replica-exchange update. The system size of the benchmark chain was $N = 55$. The hardware specifications of the graphics cards and the reference GPU can be found in Table 4.2.

	reference CPU	GPU1	GPU2	GPU3
name	Xeon E5620	Tesla C1060	GTX285	GTX480
# processors	1	30	30	15
# cores per processor	4 (only 1 used)	8	8	32
RAM	16384MB	4096MB	1024MB	1536MB
clock speed	2.4GHz	1.3GHz	1.48GHz	1.4GHz
max. threads per block	-	512	512	1024
shared memory size	-	16kB	16KB	48kB
registers per block	-	16384	16384	32768

Table 4.2: Specifications of the used hardware for GPU/CPU speed comparison.

First, naive porting of CPU to GPU codes already shows maximum speed-ups around 6–7 for GT200-based cards GPU1 and GPU2 compared to the reference implementation on the CPU. With GPU3, based on the Fermi architecture, the maximum speed-up is about 9. In Figure 4.7 the dependency of the speed-ups on the number of replicas is plotted for the examined GPUs. In this naive approach, each replica of the system was assigned to one thread block containing only one thread. Because of the embarrassingly parallel nature of the parallel tempering algorithm, it is possible to outperform a single CPU when more than 36 replicas of the system are simulated on GPU1, 32 replicas or 13 replicas for GPU2 or GPU3 respectively. This is possible due to the large number of cores available on GPUs, even though their clock speeds are lower than that of modern CPUs. As mentioned in Section 4.1, the size of the thread blocks is limited, and threads on the GPU are bundled to groups called warps. The maximum number of simultaneously active threads in a SM is 1024, i.e., 32 warps, for GT200-based cards (GPU1 and GPU2) and 1536 threads, i.e., 48 warps for Fermi-based GPU3. These warps do not necessarily have to belong to the same thread

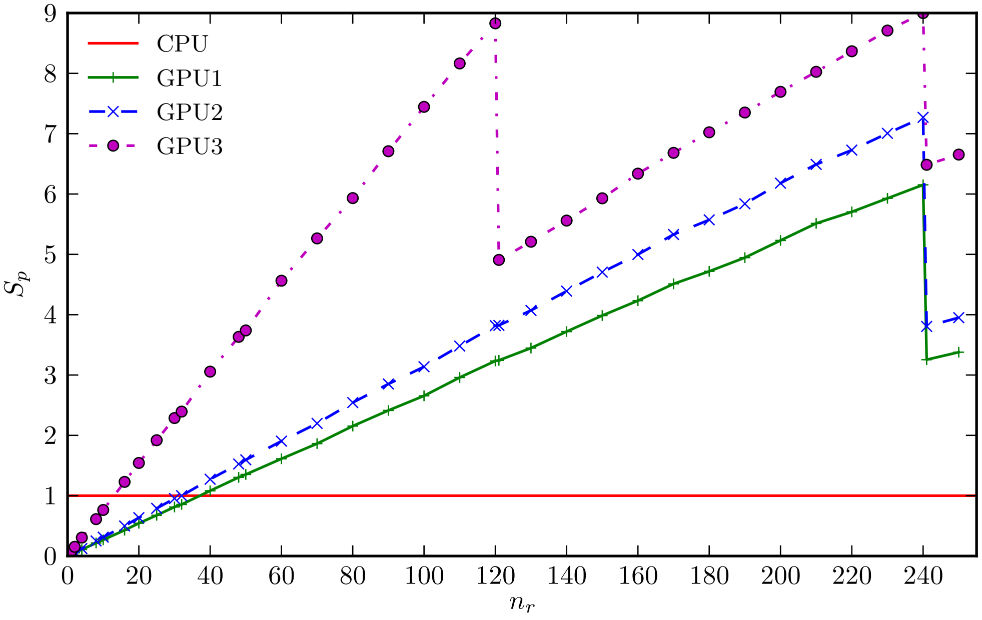


Figure 4.7: Speep-up factor S_p vs. number of replicas n_r for naive porting of CPU code to GPU. Lines are only guides for the eye.

block. Thus also 16 warps from two different thread blocks can be active simultaneously in a single SM, or 3 blocks of 10 warps, or 4 blocks of 8 warps, and so on, up to 8 blocks of 4 warps. This grouping of warps from up to 8 different thread blocks is a limitation of all examined GPUs. The SM is not able to gather as many warps from different thread blocks until its warps or thread limit is reached; explaining the peaks in Figure 4.7. Since there are 30 streaming multiprocessors on GPU1 and GPU2, the maximum number of active warps on the device is reached for 240 thread blocks of 1 thread each. Each SM calculated the single thread of 8 different thread blocks. GPU3, however, has 15 multiprocessors, thus is only capable of executing 120 blocks with 1 thread per block at a time. The maximum speed-up for this thread layout is to be expected at multiples of 240 for GPU1 and GPU2, and multiples of 120 for GPU3. Essentially thread blocks with threads other than multiples of 32 – the warp size – are not recommended at all and should be avoided at all costs.

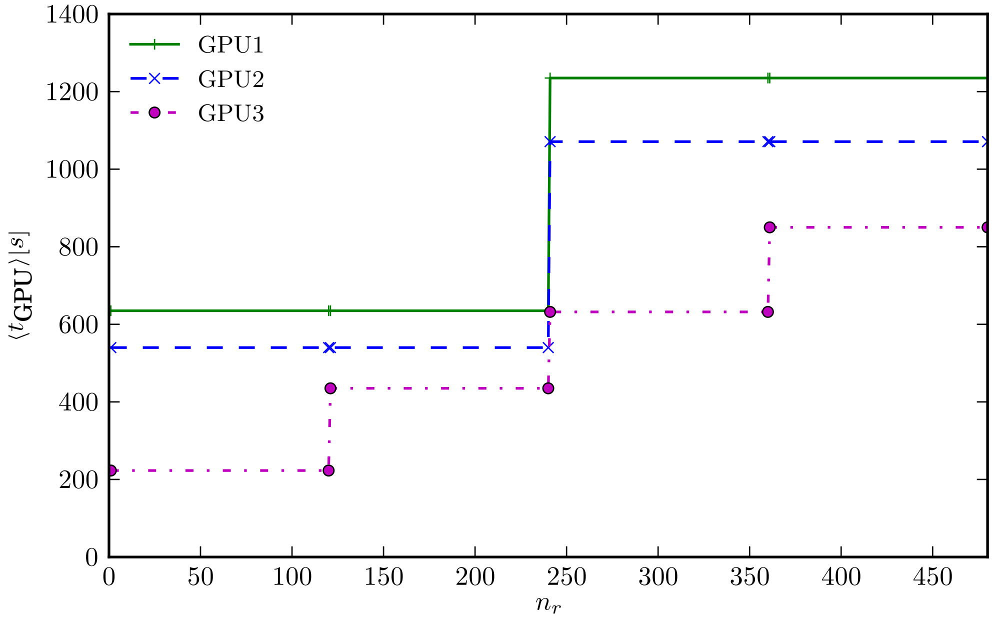


Figure 4.8: Average GPU time $\langle t_{\mathrm{GPU}} \rangle$ in seconds vs. number of replicas n_r for the naive GPU version.

Another interesting finding is that the total runtime of the kernel is the same, whether only one multiprocessor is busy and all the others are idling, or all SMs are equally busy. This leads to a step-like graph, see Figure 4.8, when plotting the kernel runtime versus the number of replicas, i.e., the number of thread blocks. With this in mind, an improved version was implemented with a parallel calculation of the energy function as shown in Listings 4.3 and 4.4. The thread block size was set to 64 – a multiple of the warp size and greater than the number of monomers – each block again holding one replica. The improved implementation also exploits low-latency memory access by using shared memory for storing the coordinates of the monomers. Thus all threads within a block have fast access to them when needed for the calculation of their portion of the energy. Performance is also gained by substituting calls to the standard C library math library functions with optimized hardware accelerated CUDA versions. The hardware accelerated functions usually have a lower precision than their software counterparts, but are executed much faster. As shown in Figure 4.9, this implementation is much faster than the CPU version, when more than 2 replicas are simulated. The maximum speed-up factor for GPU1 is 68, 78 for GPU2 and for GPU3 even 130. Again for the two GT200-based cards, multiples of 240 active thread blocks are a limit for the maximum speed-up. With 240 active thread blocks of 64 threads each, there are 15360 threads running on the GPU. The total number of threads divided by the number of SMs in these cards equals 512. These 512 threads are a collection of 2 warps from 8 different thread blocks. So, with this parametrization the occupancy of the multiprocessors is only at 50%, since 1024 active threads per SM are possible with GT200-based cards. For GPU3, the first maximum is at 120 thread blocks, which complies with a total number of 7680 threads. Even though GPU3 is capable of running 1536 threads in each of its 15 SMs at a time, only 512 threads are active, due to the 8 thread block limitation (this equals to an occupancy of 33%). These occupancy values come from the fact that only 64

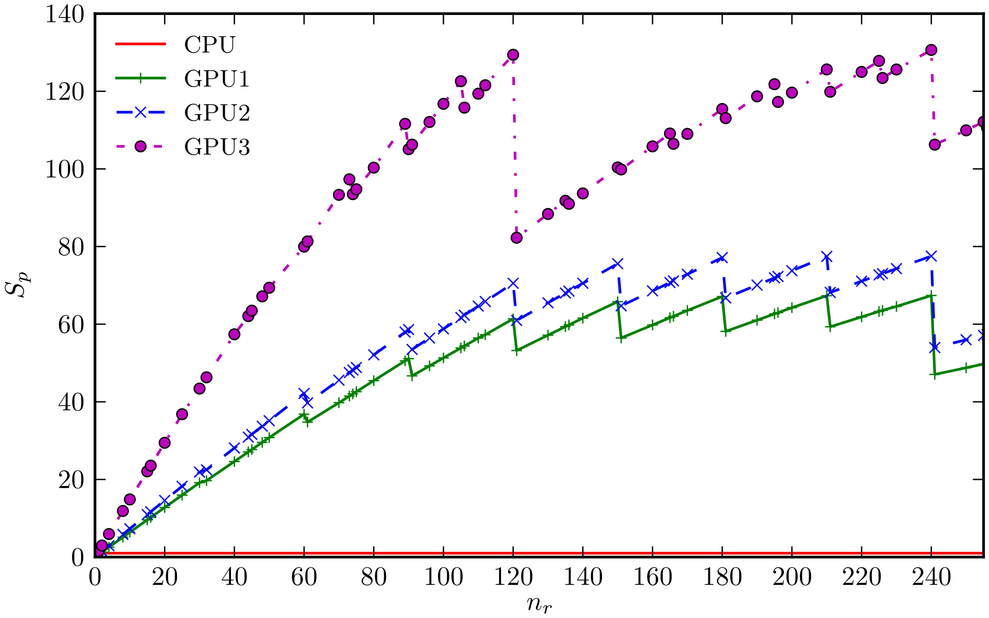


Figure 4.9: Speep-up factor S_p vs. number of replicas n_r for the GPU version with parallelized energy calculation.

threads per replica are used for the energy calculation. A different implementation with more threads or bigger system sizes, which require more threads could increase the occupancy of the multiprocessors and thus increase the efficiency even further. The step-like shape of the curve in Figure 4.10 every 30 replica for GPU1 and GPU2 coincides with the number of SMs, meaning that with every additional thread block the overall speed-up drops until each of the 30 multiprocessors again is equally busy. Also noticeable is that for GPU1 and GPU2 the first maximum of the speed-up factor is reached for 120 thread blocks. That means all cores are equally busy, but apparently there seems to be plenty of latency in memory operations. Thus, the overall speed-up is not affected by adding the same amount of work to each multiprocessor, up to 240 thread blocks in total. For GPU3 the increase in kernel runtime occurs every 15 thread blocks, corresponding to the number of SMs in the given card. Consequently, to maximize the benefit from GPU implementations it is necessary to keep all multiprocessors on the GPU equally busy.

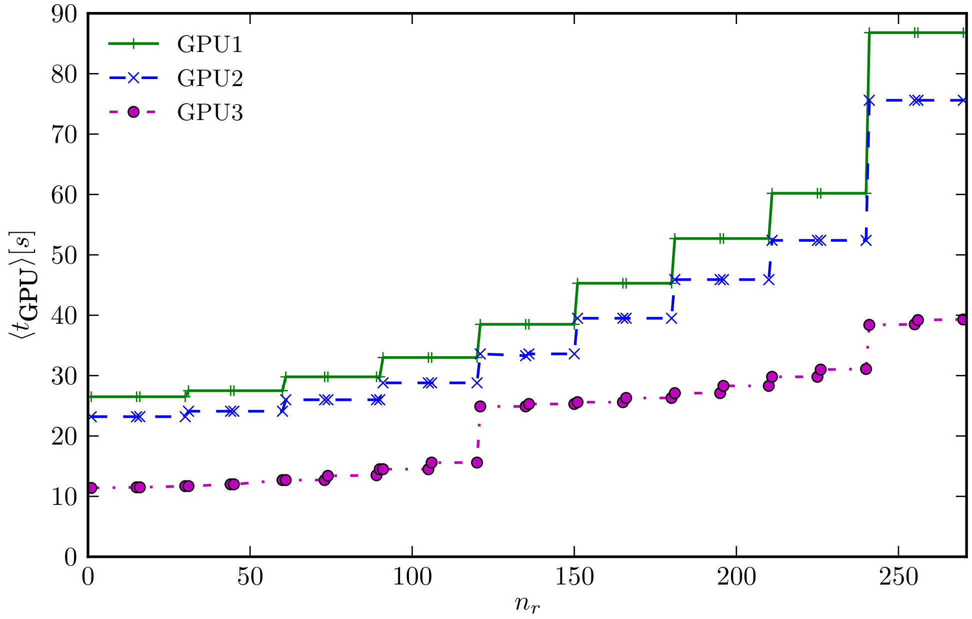


Figure 4.10: Average GPU time $\langle t_{\text{GPU}} \rangle$ in seconds vs. number of replicas n_r for the improved GPU version.

	naive	improved
GPU1	6.1×	68×
GPU2	7.2×	78×
GPU3	9×	130×

Table 4.3: Overview of maximum achieved speed-ups – $\max(S_p(n_r))$ – for the two different GPU implementations, compared to the single-core CPU implementation.

Table 4.3 is a summary of the maximum speed-up factors achieved in our simulations, employing the two different implementations. The ratio of speed-ups from GPU1 and GPU2 is nearly the same as the ratio of their respective clock speeds. Whereas GPU3 with a similar clock speed shows significant speed-ups, which originate from the difference in the chip design of the two GPU generations. To benefit from the new GPU generation the only change necessary to the program was a different layout of the thread blocks with the knowledge of the architecture as shown in Table 4.2. This demonstrates the easy portability of GPU programs to exploit advances in new hardware generations.

4.3.2 Thermodynamics

Let us now briefly review the thermodynamical properties of elastic polymers with 13 and 55 monomers, which non-trivially freeze into icosahedral structures at low temperatures [9, 10]. In Figure 4.11 the specific heat, given by

$$\frac{C_V}{N} = \frac{1}{N}\frac{\partial\langle E\rangle}{\partial T} = \frac{\beta^2}{N}(\langle E^2\rangle - \langle E\rangle^2), \tag{4.6}$$

for the 13-mer and 55-mer, obtained by our parallel tempering simulations at 128 temperatures, is plotted. The very pronounced peak at low temperatures, $T \approx 0.33$ for the 13-mer

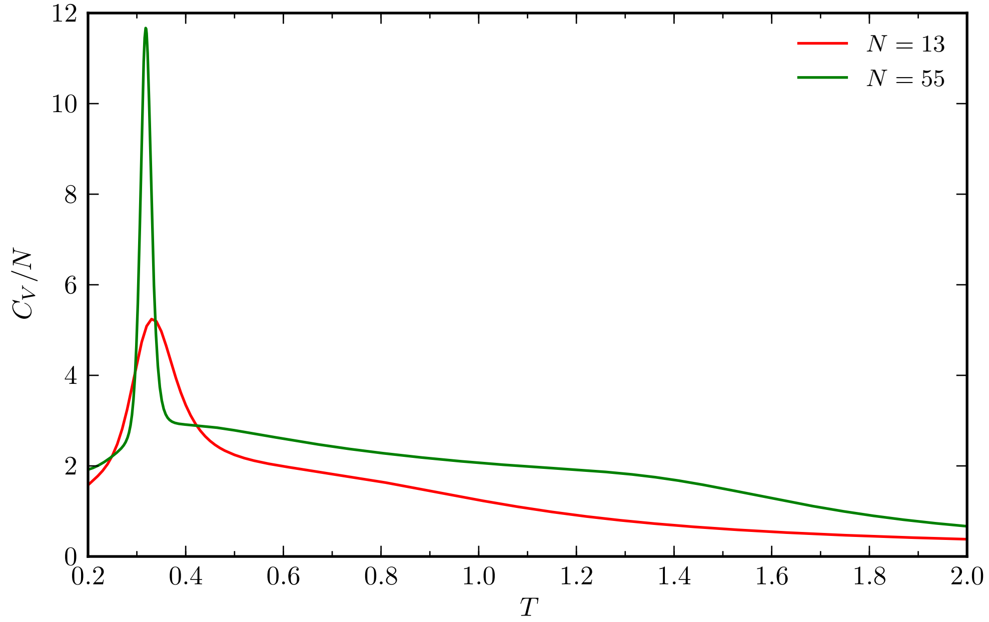


Figure 4.11: Specific heat for homopolymers with 13 and 55 monomers. The distinct peak at low temperatures signals the freezing transition. Below that temperature these two polymers form icosahedral crystals. Error bars are of the size of the line width.

and $T \approx 0.31$ for the 55-mer, indicates the freezing transition. Below these temperatures, the respective polymers have an icosahedral crystal structure. A change in the monotonic behavior of the specific heat curves occurs at higher temperatures, $T \approx 1.0$ for the 13-mer and $T \approx 1.6$ for the 55-mer. To get an insight of what is happening to the polymers at these temperatures it is advantageous to look at another quantity. Figure 4.12 shows the thermal fluctuation of the squared radius of gyration. The following expression,

$$r_{\mathrm{gyr}}^2 = \frac{1}{N}\sum_{k=1}^{N}(\vec{r}_k - \vec{r}_{\mathrm{mean}})^2, \tag{4.7}$$

defines the squared radius of gyration, where $\vec{r}_{\mathrm{mean}}$ is the center of mass of the polymer. The radius of gyration describes the mean distance of every monomer from the center of the

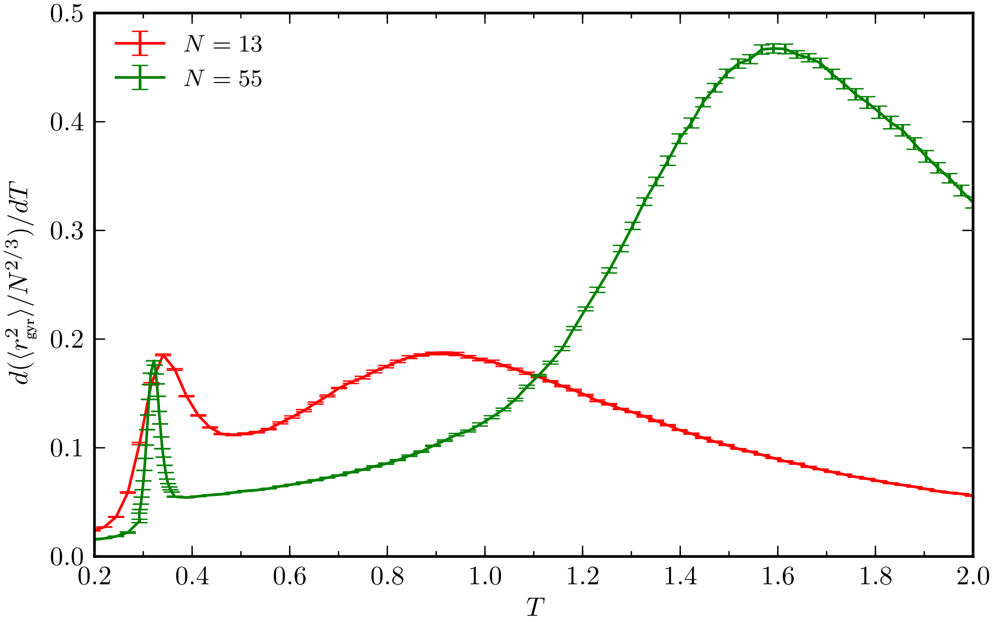


Figure 4.12: Radius of gyration for the 13-mer and 55-mer as a function of temperature. Both the freezing transition at low temperatures and the Θ-collapse at higher temperatures are clearly observable as peaks.

polymer and is thus a spherical approximation of the polymer extension. From its thermal fluctuation

$$\frac{\partial\langle r^2_{\rm gyr}\rangle}{\partial T} = \frac{\langle E \cdot r^2_{\rm gyr}\rangle - \langle E\rangle \cdot \langle r^2_{\rm gyr}\rangle}{T^2}, \tag{4.8}$$

structural transition temperatures can be located very easily. The peaks corresponding to the freezing transition at low temperatures coincide with the respective peaks in the specific heat. The maxima at high temperatures indicate the Θ-transition, where the random coil polymer conformations collapse to very compact, but disordered liquid-like structures. The temperatures are consistent with the signals in the specific heat curves. Table 4.4 gives an overview of all structural "phases" of the two investigated polymer chains.

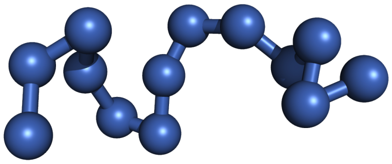
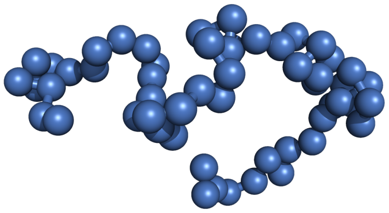
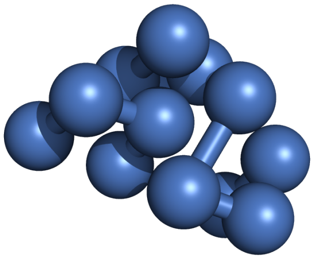
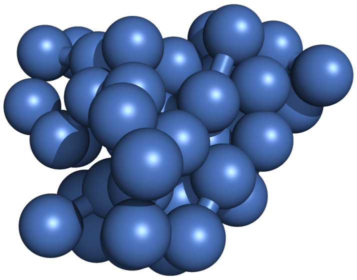
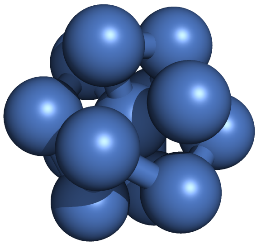
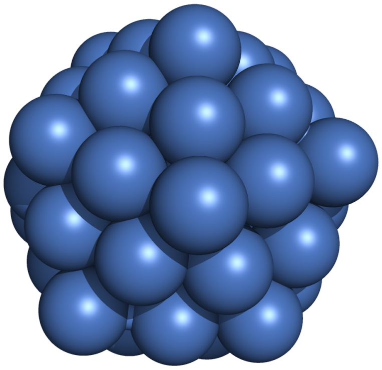
structural phase	13-mer	55-mer
gas		
liquid		
solid		

Table 4.4: Exemplified polymer structures for the 13-mer and 55-mer. The top row shows the coil-like structures the polymer forms at high temperatures. In the middle are the collapsed or liquid structures. The bottom row contains the crystals the polymer forms below the freezing point.

4.4 Conclusions

In this chapter it was shown that replica-exchange Monte Carlo simulations of polymer models can be performed quite efficiently on GPUs. Even for off-lattice models this is a suitable approach. With very simply naive porting of CPU code to the GPU, we find considerable performance gains of factors about 6–9 compared to a single CPU implementation. Utilizing the unique architecture of GPUs with its different memory layers and the ability to schedule a massive amount of threads, we improved the GPU program to attain speed-up factors of around 70 for the GTX285, and even speed-up factors up to 130 for the Fermi-based GTX480 card. It should be noted that our implementation represents a rather basic level of utilizing all the advantages GPUs offer. There is still more potential for further optimizations of the code.

Furthermore, it is possible to access multiple GPUs in a single workstation within the same program. Also nodes of established cluster computers can be equipped with GPUs. A hybrid programming using CUDA/OpenCL and the traditional message passing interface (MPI) would be used in such a scenario. In the last years more and more supercomputers have been equipped with GPUs to deliver cost and energy efficient performance. The future of supercomputing is going to be massively parallel with the need of hybrid programming schemes.

Chapter 5

Influence of the interaction range on flexible polymer structures

5.1 Introduction

In this chapter, we will investigate how structure formation of a single elastic, flexible polymer depends on the range of interaction between nonbonded monomers. The goal is to construct a phase diagram that separates potential structural phases for all classes of flexible polymers under the influence of a thermal environment. In recent years, much work has been dedicated to the identification of structural phases of flexible polymers by using standard representations of generic coarse-grained lattice and off-lattice models for polymers. However, as recent studies of discrete models have shown [8, 58–61], it is also important to understand to what extent the formation of these structural phases is affected by the effective range of the attractive nonbonded interactions competing with excluded-volume effects. One of the most interesting features found in these studies was that, for sufficiently short interaction range, collapse and nucleation are not separate transitions anymore, and a liquid phase does not exist. It is also known that geometric properties of atomic clusters depend sensitively on the range of interaction [5–7, 62].

We here present the results of extensive generalized-ensemble Monte Carlo simulations of a generic Lennard-Jones model for elastic, flexible polymers in continuum in order to reveal the interaction-range dependent phase structure under the influence of finite-size effects. Since the latter essentially affects the behavior in the regime where coil-globule and freezing transitions meet, we also thoroughly compare conventional canonical and more detailed microcanonical analyses of this multiple transition point.

5.2 Model and Methods

5.2.1 Model

For this study, we employ a very similar model of a single elastic, flexible homopolymer chain that we used in Chapter 4. A very important difference, however, is an additional parameter in the Lennard-Jones potential to control the effective interaction range. All monomers, bonded and non-bonded, interact via a truncated, shifted Lennard-Jones potential

$$U_{\mathrm{LJ}}^{\mathrm{mod}}(r_{ij}) = U_{\mathrm{LJ}}(r_{ij}) - U_{\mathrm{LJ}}(r_c), \tag{5.1}$$

with

$$U_{\mathrm{LJ}}(r_{ij}) = 4\epsilon\left[\left(\frac{\sigma}{r_{ij}-r_s}\right)^{12} - \left(\frac{\sigma}{r_{ij}-r_s}\right)^{6}\right], \tag{5.2}$$

where the energy and length scales are set to $\epsilon = 1$ and $\sigma = (r_0 - r_s)/2^{1/6}$, respectively. We choose a cut-off radius $r_c = 2.5\sigma + r_s$ such that $U_{\mathrm{LJ}}^{\mathrm{mod}}(r_{ij}) \equiv 0$ for $r_{ij} > r_c$ and $U_{\mathrm{LJ}}(r_c) = (-3983616/244140625)\epsilon \approx -0.016317\epsilon$. The bonds between neighboring monomers are modeled using the anharmonic FENE (finitely extensible nonlinear elastic) potential [63]

$$U_{\text{FENE}}(r_{ii+1}) = -\frac{K}{2}R^2 \log\left[1 - \left(\frac{r_{ii+1} - r_0}{R}\right)^2\right]. \tag{5.3}$$

We locate its minimum at $r_0 = 0.7$, set $R = 0.3$, and choose $K = 40$ [9, 10]. The total energy of a conformation $\mathcal{C} = (\vec{r}_1, \cdots, \vec{r}_N)$ for a chain with N monomers is then given by

$$E(\mathcal{C}) = \frac{1}{2}\sum_{\substack{i,j=1 \\ i\neq j}}^{N} U_{\text{LJ}}^{\text{mod}}(r_{ij}) + \sum_{i=1}^{N-1} U_{\text{FENE}}(r_{ii+1}). \tag{5.4}$$

Within our simulations, the parameter r_s is used to control the width of the potential. The qualitative behavior of the influence of r_s on the shape of the potential is shown in Figure 5.1. While it is convenient to use r_s in the definition of the potential, it is more useful for the subsequent analysis to introduce the potential width δ as a new parameter. For this purpose

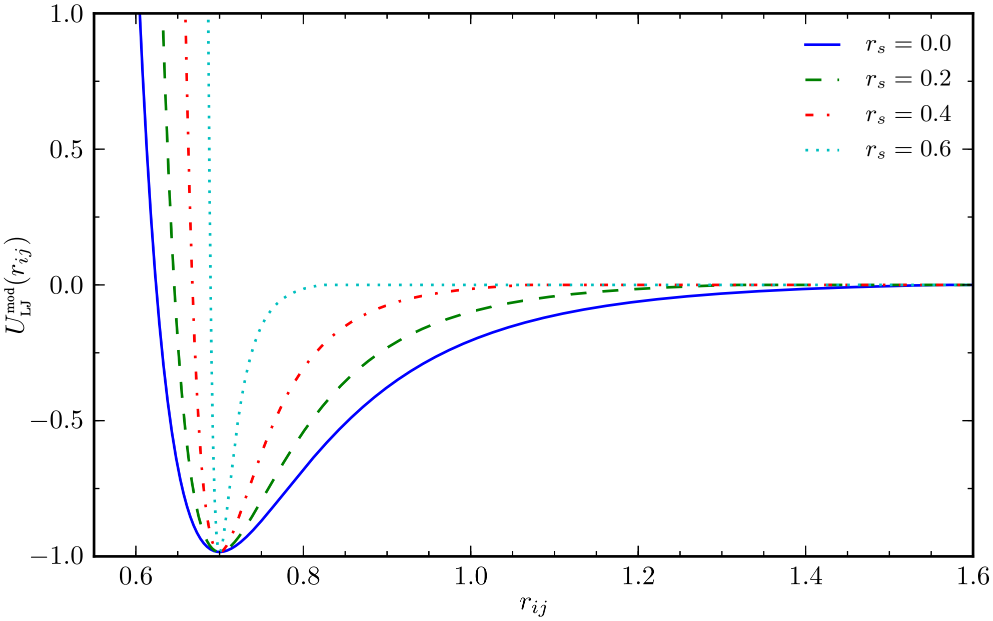


Figure 5.1: Behavior of the modified Lennard-Jones potential for different values of r_s.

only, we define a square well potential

$$U_{\mathrm{sq}}(r) = \begin{cases} \infty & \text{if } r \leq r_1 \\ -\epsilon_{\mathrm{sq}} & \text{if } r_1 < r < r_2 \ , \\ 0 & \text{if } r \geq r_2 \end{cases} \tag{5.5}$$

with the constant $\epsilon_{\mathrm{sq}} = \epsilon/2 + U_{\mathrm{LJ}}(r_c)$ such that $\epsilon_{\mathrm{sq}} = (236173393/488281250)\epsilon \approx 0.483683\epsilon$ and r_1 and r_2 being the radii where $U_{\mathrm{LJ}}^{\mathrm{mod}}(r_1) = U_{\mathrm{LJ}}^{\mathrm{mod}}(r_2) = -\epsilon_{\mathrm{sq}}$, independent of r_s (see Figure 5.2).

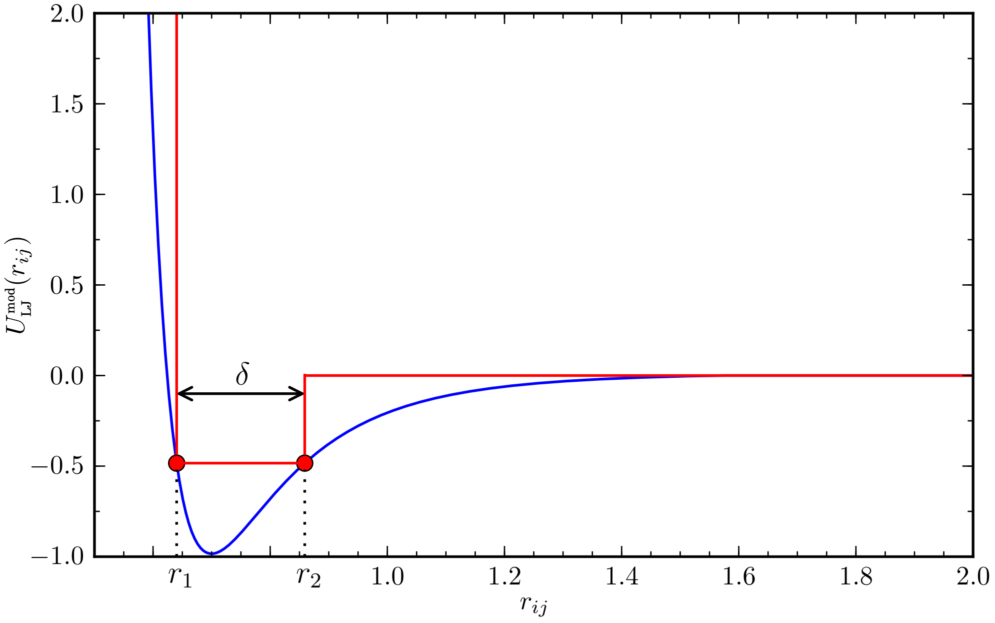


Figure 5.2: The potential width δ is defined by the width of a square well potential of depth $-\epsilon_{sq}$, which is the difference of the two distances r_1 and r_2 where the Lennard-Jones potential equals $-\epsilon_{sq}$.

The relationship between the simulation parameter r_s and the potential width δ is linear:

$$\delta = r_2 - r_1 = \lambda(r_0 - r_s), \tag{5.6}$$

with

$$\begin{aligned}\lambda &= 2^{1/6}\left[\left(1+\sqrt{\frac{1}{2}}\right)^{1/6} - \left(1-\sqrt{\frac{1}{2}}\right)^{1/6}\right]\\ &\approx 0.312382. \end{aligned} \tag{5.7}$$

The maximum value of δ is determined by the unmodified Lennard-Jones term, i.e., for $r_s = 0$, and reads $\delta_{\max} = \lambda r_0 \approx 0.218667$.

5.2.2 Simulation method

For our simulations, we employed the replica-exchange Monte Carlo method known as parallel tempering, see Chapter 3.1.2. One set of simulations was performed on GPUs using either $n_r = 112$ or 128 replicas, depending on the card generation. The calculation of the energy was carried out in parallel by using 128 threads per replica. Consequently there were up to 16384 threads running concurrently on the graphic cards. The advantages of utilizing graphic cards for parallel tempering simulations of polymers have already been discussed in Chapter 4 and have been published in References [22, 23]. This simple scheme can be applied for values of δ as small as about 0.06. For smaller values of δ the freezing transition barrier becomes so strong that an algorithmic improvement is necessary.

Such is made possible by multiple Gaussian modified ensembles (MGME) [64–66], which we will explain later. This Monte Carlo method retains all advantages of parallel tempering, in that it facilitates efficient implementation on parallel computers. At the same time, the sampling of entropically suppressed conformations is increased. The method allows the simulation of strong first-order polymer crystallization for chain lengths up to $N = 147$ at very small interaction width $\delta \approx 0.030$ and for the 90-mer down to $\delta \approx 0.015$. A simulation of all structural phases of these polymers with standard parallel tempering is virtually impossible. We performed MGME simulations on a parallel computer cluster using the message passing interface (MPI).

The basic idea of MGME simulations is to multiply the Boltzmann factor of single canonical ensemble by a Gaussian form centered around some central energy value $E_{G,i}$ and a width ΔE_G, such that $P^{\mathrm{B}}_{\mathrm{MGME},i} \sim e^{-\beta_i E - [(E-E_{G,i})/\Delta E_G]^2}$. In consequence, the probability for a state with energy E to occur in the ith modified ensemble becomes

$$P_{\mathrm{MGME},i}(E) \sim e^{S(E)-\beta_i E - \left[(E-E_{G,i})/(\Delta E_G)\right]^2}, \tag{5.8}$$

where $S(E) = \ln g(E)$ is the microcanonical entropy and $g(E)$ is the density of states. In case of first-order-like transitions, $S(E)$ becomes convex with $d^2S(E)/d^2E > 0$ in a certain finite energy interval, limited by two distinct energies E_+ and E_- with $d^2S(E)/dE^2(E = E_+) = d^2S(E)/dE^2(E = E_-) = 0$, see Figure 5.3. In this case the energy distribution is bimodal.

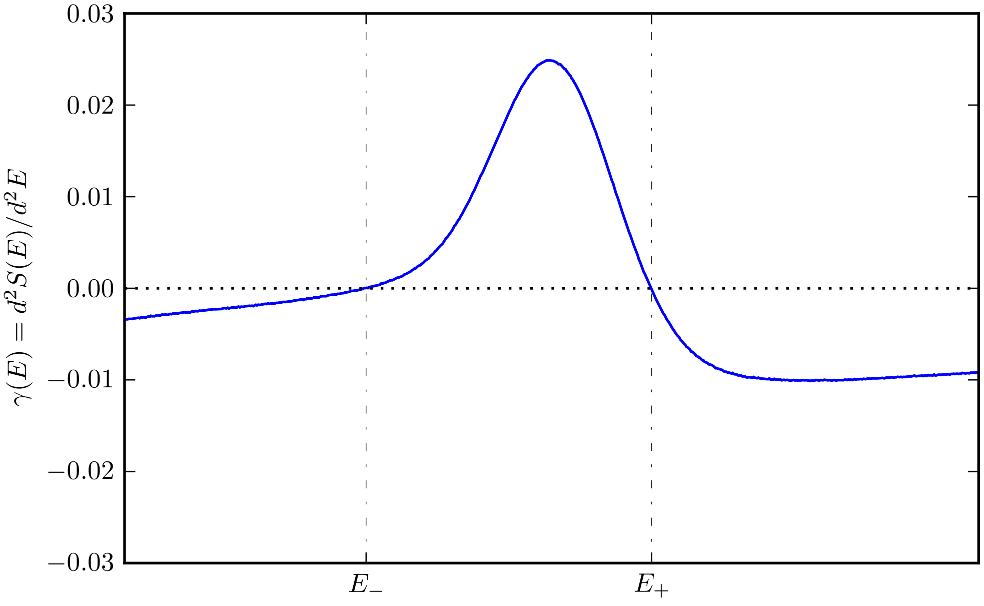


Figure 5.3: Second derivative of the microcanonical entropy $S(E)$ showing a positive peak signaling a first-order like transition. Error bars are the size of the line width.

In MGME simulations the counter term to $S(E)$ in Equation (5.8) of the form $-(E/\Delta E_G)^2$ shifts positive $d^2S(E)/d^2E$ values to negative, provided that ΔE_G is small enough. Thus, energy distribution functions within the single Gaussian ensembles $P_{\mathrm{MGME},i}(E)$ have strictly uni-modal shape. This absence of double-peaked distributions improves the Monte Carlo sampling problem of entropically suppressed regions of state space, while a proper choice of the remaining parameters $E_{G,i}$ and β_i ensures a sufficient overlap between neighboring parallel tempering partitions at i and $i + 1$. Possible algorithmic approaches to the parameter choice are described in Ref. [64]. For illustration, in Figure 5.4 we show a combined energy histogram obtained in actual MGME simulations, the multi-histogram $H_{\mathrm{multi}}(E)$. The simulation covers the entire energy interval of interest for a $N = 90$ polymer at $\delta \approx 0.030$. In addition, neighboring single energy histograms in-between i and $i + 1$ as displayed in the figure have sufficiently large overlap to facilitate swap-updates with reasonable acceptance

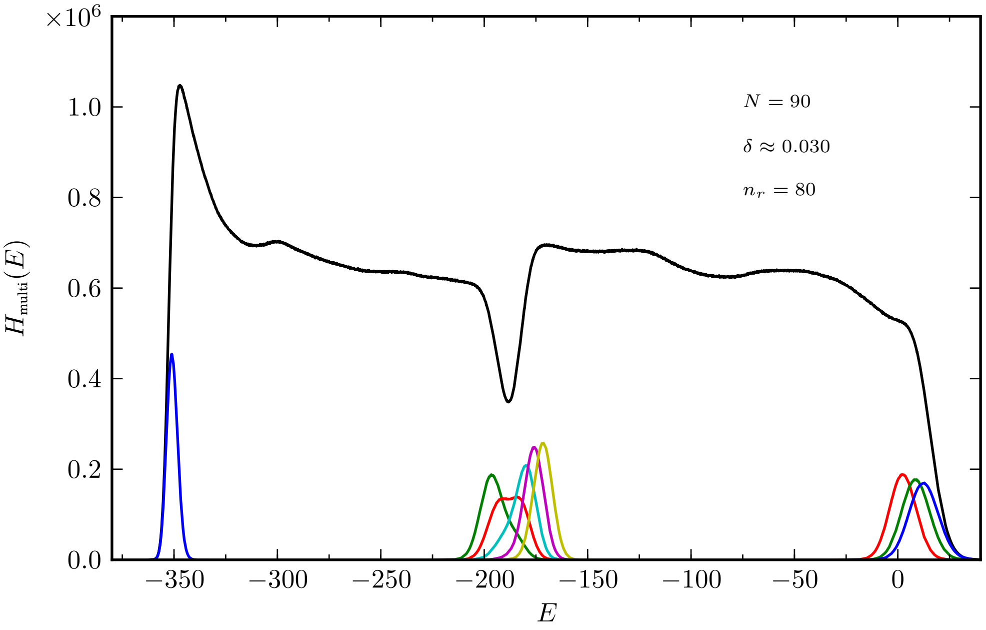


Figure 5.4: Accumulated multi-histogram $H_{\mathrm{multi}}(E)$ from MGME simulations for $N = 90$ and $\delta \approx 0.030$ (upper curve). The parallel tempering partition consists of $n_r = 80$ single Gaussian ensembles (replicas). Some histograms for single Gaussian ensemble are also displayed. They are all of uni-modal shape.

rates. For the given example, the overlap $O = \int \min[P_i(E), P_{i+1}(E)]dE$ in-between neighboring probability distributions of the parallel tempering partition was tuned to a value $O \approx 0.63 \ldots 0.64$. This particular value results in acceptance rates $P_{\mathrm{acc}} \approx 0.5$ for swap updates. We remark that less optimal parallel tempering partitions for MGME simulations can easily be found, and as long as $O > 0.1$ are still considered to be efficient. In our early simulations we actually employed the simple displacement updates for all Cartesian monomer coordinates.

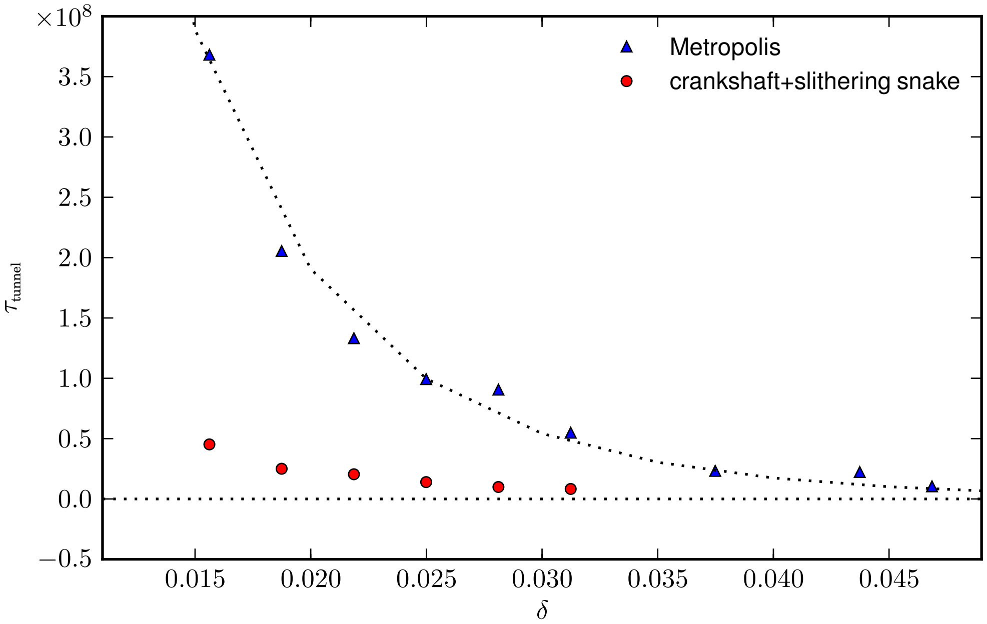


Figure 5.5: Tunneling auto-correlation time as explained in the text as a function of δ at $N = 90$ for simple displacement updates (triangles), improved by slithering snake and crankshaft moves (circles).

For these, we measure the tunneling auto-correlation time τ_{tunnel} in units of sweeps, which counts the time in-between the assignment of a specific conformation (on the parallel tempering partition) to $i = 1$, then to $i = n_r$ and finally to $i = 1$ again. Figure 5.5 displays these times (triangles) for the $N = 90$ polymer as a function of δ. We observe rapidly increasing values for short ranged potentials, which renders simulations of short ranged potentials hard. In typical parallel tempering simulations, we perform $\mathcal{O}(10^9)$ sweeps for each of the n_r copies. Including slithering snake and crankshaft updates (circles) reduces the time scale substantially.

5.3 Results

5.3.1 Comparison with previous studies

In a recent study [60, 61], Taylor et al. investigated a flexible homopolymer chain, where the non-bonded monomers interact via a square-well potential with variable width. Constructing a phase diagram as a function of temperature and potential width, they identified three phases for sufficiently large interaction ranges: expanded coils for high temperatures and crystalline (frozen) structures for very low temperatures, separated by a collapsed-globule phase for intermediate temperatures. The collapse transition was found to be pre-empted by the freezing transition for narrow potentials. While in a canonical analysis approach the signals for the collapse transition vanish, the microcanonical approach is still able to locate the positions of all transitions.

Since the collapse transition point is included in the Maxwell regime of the liquid-solid transition, Taylor et al. concluded that what remains is a first-order-like transition from coil to crystal. This argumentation is fully consistent with the assumption that liquid-solid and coil-globule transitions become indistinguishable in the thermodynamic limit. For the continuum model we used, we can clearly confirm these findings. Figure 5.6 shows how the inflection point associated with the second-order collapse transition enters the Maxwell regime of the liquid-solid transition (dashed lines), if δ is decreased below a threshold value.

However, this argumentation is not sufficiently consequential in light of the microcanonical interpretation of the results obtained for a finite system. First, the Maxwell construction is adapted from the theory of real gases, where it is necessary to get rid of unphysical behav-

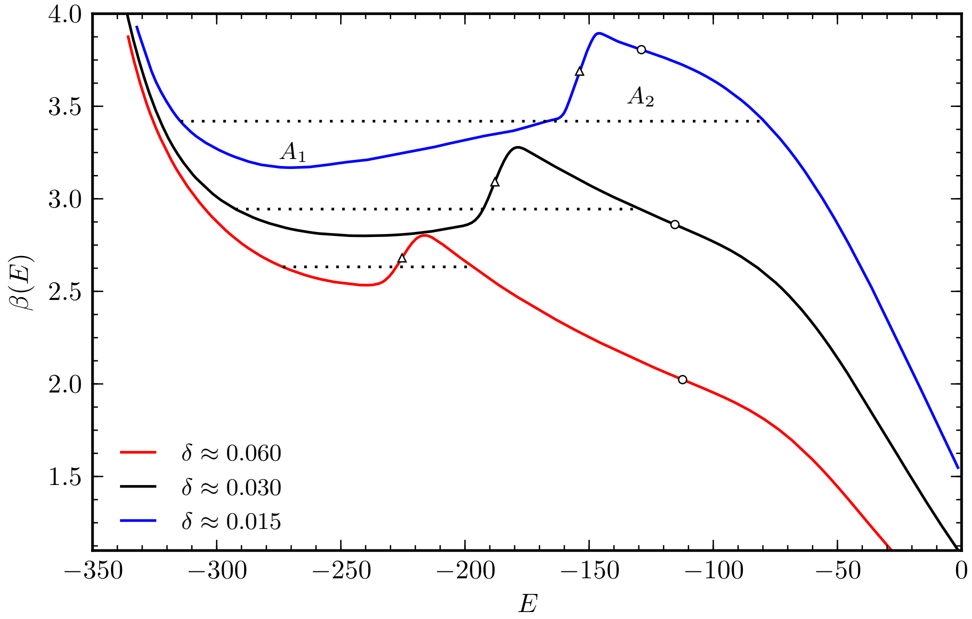


Figure 5.6: Comparison of the microcanonical inverse temperature $\beta(E)$ for three values of δ. The triangles mark the corresponding peak position in the derivative of $\beta(E)$ for the freezing transition. The circles correspond to the maxima in $\gamma(E)$ marking the collapse transition. The dotted lines are located at β-values obtained by Maxwell constructions. Areas A_1 and A_2 enclosed by β curves and Maxwell line coincide.

ior in the infinite system. Here, this is not necessary. The "back-bending effect" smoothly disappears for large systems. Therefore, a Maxwell construction is not needed at all. The analysis of inflection points is sufficient to uniquely identify and classify transitions. Second, in contrast to the canonical "heat bath" temperature, the inverse microcanonical temperature is a well-defined quantity on fundamental statistical grounds. Taking this into account, both transitions remain separate, but microcanonically they cross over. This is a pure finite-size effect. Both transition temperatures will converge to the same transition point in the thermodynamic limit.

In Figure 5.6, the microcanonical temperature curves are shown for three potential widths, $\delta \approx 0.06, 0.03$, and 0.015. Circles mark the transition points for the Θ-collapse and triangles the freezing transition. In addition to the results from the inflection point analysis, the Maxwell line associated with the freezing transition is also included. While for rather broad potentials, i.e., $\delta \gtrsim 0.1$ the temperatures obtained by inflection point analysis and Maxwell construction match, these definitions of the transition temperature differ for narrower potentials.

5.3.2 Interaction range dependency

In the following, we will investigate how the interaction range δ of the potential influences transition points in the system. We have plotted the first and second derivatives of the microcanonical entropy for the 90-mer in Figure 5.7 as well as specific heat curves and thermal fluctuations of the radius of gyration in Figure 5.8.

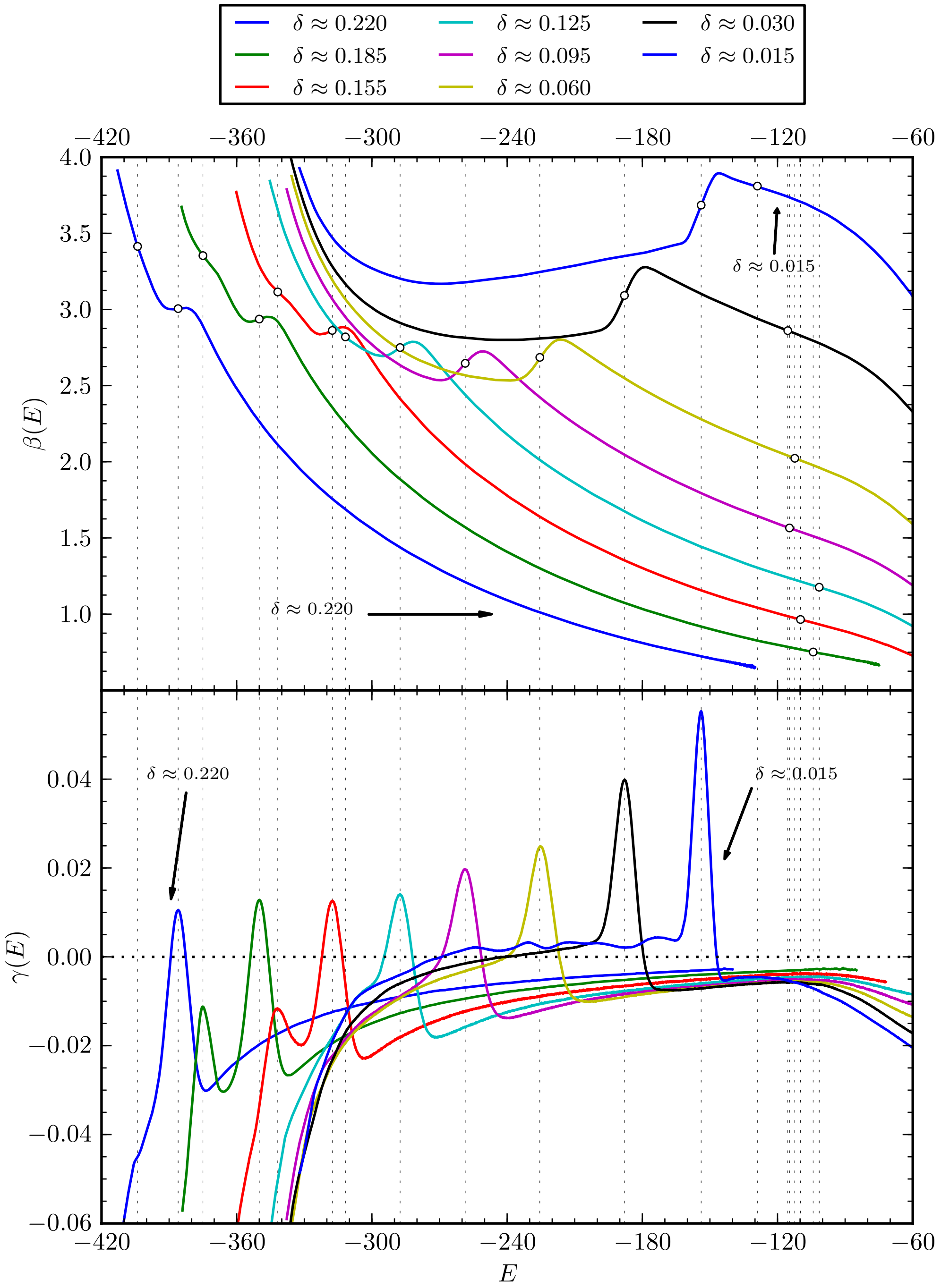


Figure 5.7: (Top) The microcanonical inverse temperature as a function of energy for the 90-mer at different values for the potential width δ. All inflection points of the inverse temperature that signal a transition are marked by a circle. (Bottom) Derivative of the inverse temperature $\beta(E)$ as a function of energy.

In Figure 5.7 (top) the inverse microcanonical temperature is shown as a function of energy. The unmodified and largest interaction range $\delta = \delta_{\max} \approx 0.22$ corresponds to the leftmost temperature curve. Two effects can be observed as the potential width δ is reduced: The freezing transition marked by the non-monotonic region, also referred to as "back-bending" or "convex intruder" [16, 67], is shifted to higher energies. It also becomes more pronounced for narrower potentials. Another effect of narrowing the interaction length is that the collapse transition, indicated by the shoulders in the $\gamma(E)$-curves, shifts to lower temperatures. The difference in transition temperatures becomes smaller and smaller for shorter interactions range. In the bottom part of Figure 5.7 we show the second derivative of the entropy. With decreasing potential width the freezing transition, signaled by the peak of positive value, shifts to higher energies. The first four curves also show a peak with negative value below the freezing transition, marking the solid-solid transition in the incomplete outer shell of the icosahedron in the core. Note that the solid-solid transitions are second-order-like and occur only for $\delta > 0.12$. At higher energies, above the freezing transition, the curves exhibit a maximum marking the collapse transition. This maximum is shifted to lower energies, as δ decreases.

We also looked at two canonical quantities to identify transitions in the system. A typical quantity that gives insight into the thermodynamic behavior of the system is the specific heat, as shown in Figure 5.8 (top). With decreasing potential width, the signal for the freezing transition, i.e., the pronounced peak at low temperatures, shifts to slightly higher temperatures. For smaller values of the interaction length the freezing temperature drops again. The maxima of the peaks increase with decreasing δ. The solid-solid transition is just visible as a small shoulder below the freezing peak. While the freezing temperature changes only slightly, the collapse temperature undergoes more significant changes. The shoulders indicating the collapse transitions become narrower with decreasing potential width.

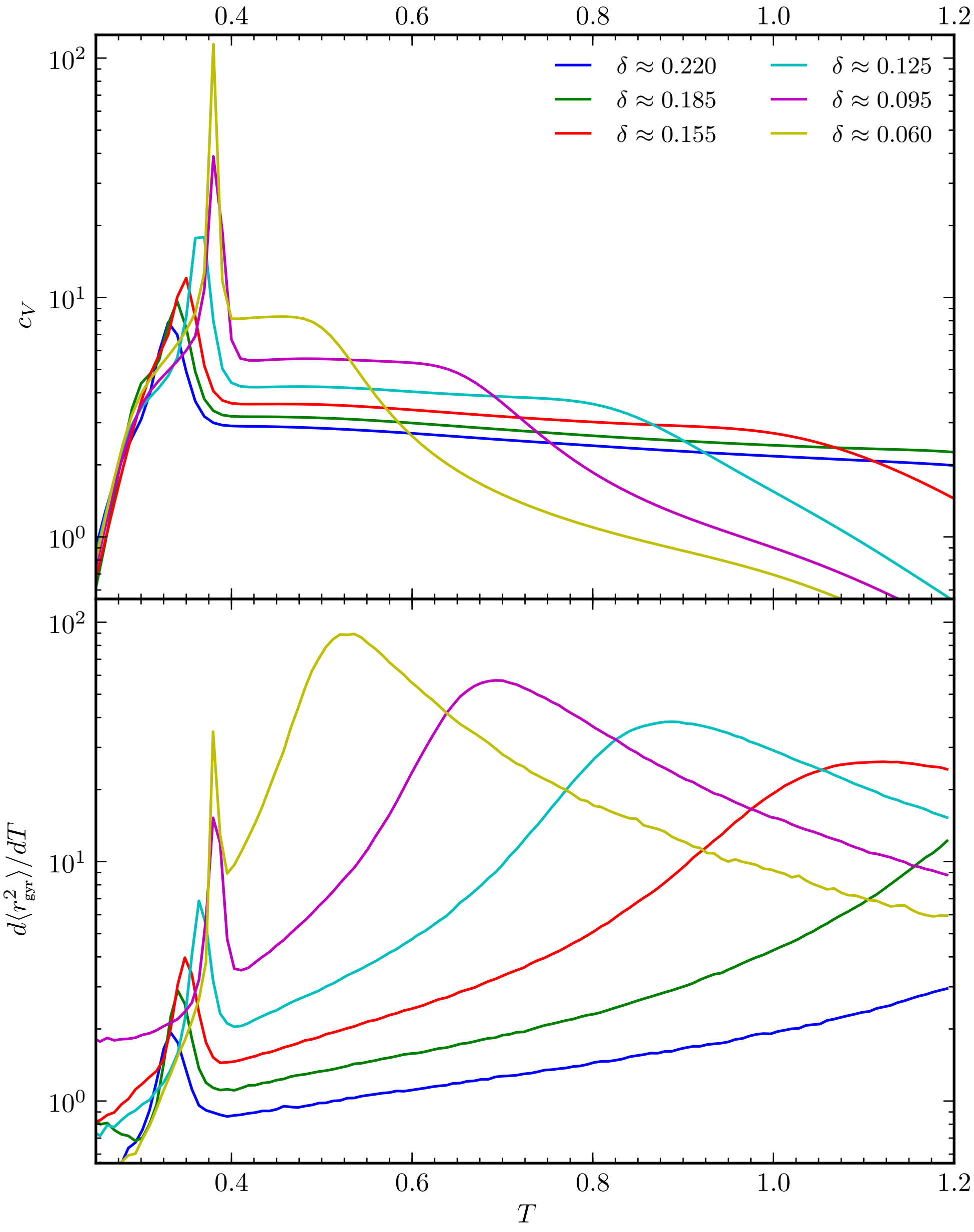


Figure 5.8: (Top) Specific heat and (bottom) thermal fluctuation of the radius of gyration for the 90-mer parametrized by δ.

In this case it is often more advantageous to investigate structural quantities, such as the radius of gyration $r_{\rm gyr}$, which is a measure for the spatial extension of the polymer. Let's discuss the thermal fluctuations of $r_{\rm gyr}$ as shown in Figure 5.8 (bottom). For each δ, there are two prominent peaks. The low-temperature peaks belong to the liquid-solid transition and their locations agree well with those of the respective specific heat peaks. At higher temperatures we find very pronounced peaks that indicate the collapse transition. Again, we can see that with smaller δ the difference between gas-liquid and liquid-solid transition temperatures decreases. The Θ-collapse moves to lower temperatures for short interaction ranges.

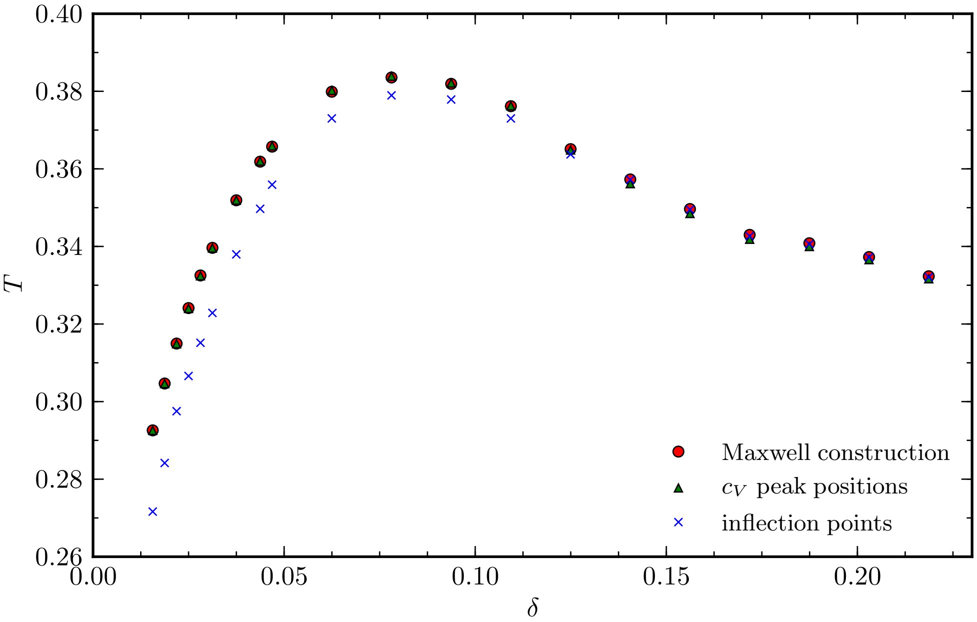


Figure 5.9: Comparison of three different definitions for the freezing transition temperature of the 90-mer. The crosses are transition temperatures indicated by the respective peaks in the first derivative of the microcanonical entropy. Peak positions of the specific heat are represented by triangles. The circles are the transition points obtained by Maxwell construction.

To compare different approaches for transition temperatures, we show the behavior of three definitions for the freezing temperature in Figure 5.9. While specific heat peaks and Maxwell construction agree over the entire δ-range, the values obtained via microcanonical analysis visibly deviate for $\delta \lesssim 0.1$. This is not surprising, because in the case of the specific heat and Maxwell indicators, freezing and collapse signals mix, whereas the inflection points purely indicate the freezing transition only. Therefore, we will construct the structural phase diagram in the following entirely by means of the microcanonical inflection points of the inverse temperature.

5.3.3 Phase diagram for the 90-mer

With all the transition temperatures acquired from the microcanonical analysis, we can construct the structural phase diagram for the 90-mer, parametrized by temperature T and interaction range δ. There are three major phases, see Figure 5.10. In the "gas" phase **G** at high temperatures and short-range interaction, polymer conformations are dominated by expanded coils. For interaction ranges $\delta \gtrsim 0.02$, the "liquid" phase **L** separates the gas phase from distinct solid phases. The red curve in Figure 5.10 is the Θ-transition line, where the expanded coil collapses into disordered, but compact globular states. Reducing the temperature, the polymer structures change from globular to crystalline at the freezing transition line indicated by the green line. With decreasing potential width the liquid phase region becomes smaller, as the collapse transition shifts to lower temperatures. The inset in Figure 5.10 shows the crossover of collapse and freezing at very small interaction ranges. In the microcanonical analysis it is still possible to single out both transition temperatures. At about $\delta = 0.12$ the solid-solid transition (blue line) merges with the freezing transition.

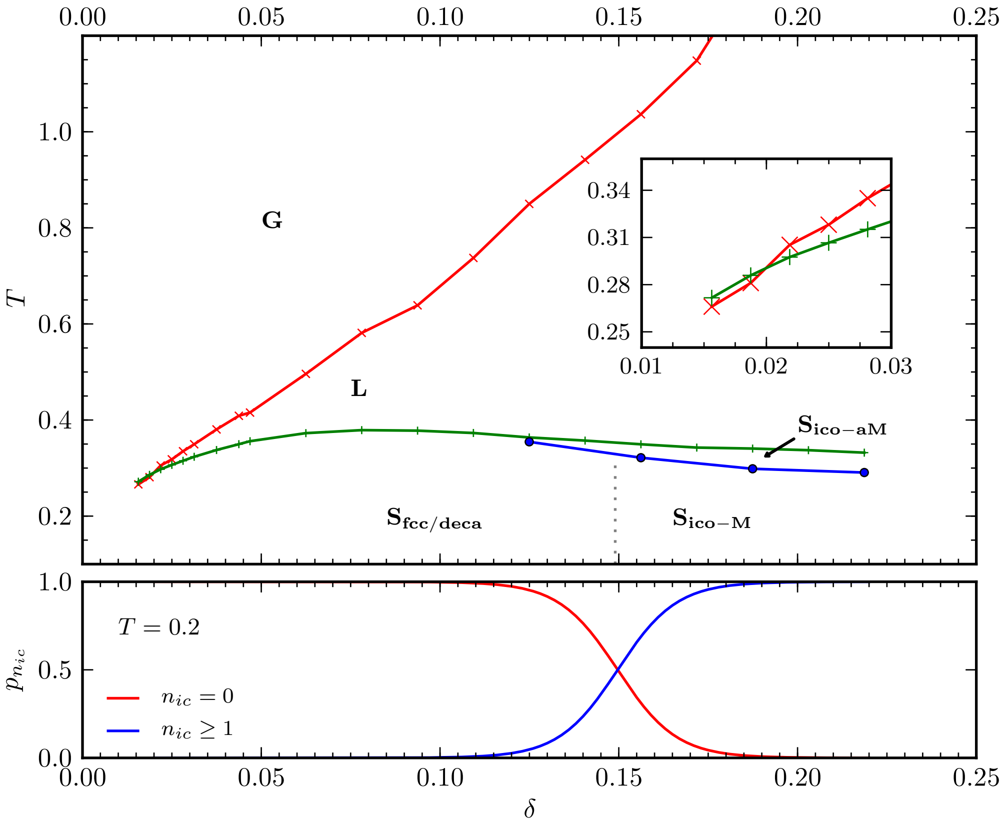


Figure 5.10: (Top) Phase diagram for the 90-mer, obtained by pure microcanonical analysis. The inset shows the crossover of the transition temperatures for collapse and freezing transition. (Bottom) Probability for zero ($n_{ic} = 0$) and nonzero ($n_{ic} \geq 1$) number of icosahedral cores in low temperature structures, cf.[9, 10].

The solid phase $\mathbf{S_{ico-aM}}$ is dominated by structures with at least one icosahedral core and an incomplete outer shell of anti-Mackay type (hcp), see Figure 1 and Figure 2 in Ref. [9, 10]. By reducing the temperature further and passing the solid-solid transition line, the packing is optimized and a Mackay-type fcc layer forms (phase $\mathbf{S_{ico-M}}$). However, the icosahedral interior becomes energetically less optimal for $\delta < 0.15$, and it is replaced by a decahedral arrangement of monomers. These structures can also possess extended fcc-packed fractions ($\mathbf{S_{fcc/deca}}$). Following former studies of atomic cluster models with short-ranged interactions [5–7], one might expect a separate fcc phase to be present at extremely small δ-values and temperatures. We will discuss this crossover in more detail for a simpler example in the following section.

5.3.4 Analysis of low-temperature structures

The characterization of the solid phases in the structural phase diagram is challenging and almost completely determined by surface effects. It is instructive to investigate the low-temperature crystal structures of flexible polymers at different ranges δ of the monomer–monomer interaction potential. As an example, we choose the 55-mer, which forms a perfectly shaped icosahedron for $\delta = \delta_{\mathrm{max}} \approx 0.220$ [9, 10]. In contrast to the 90-mer, its geometric phases are more stable and can be identified clearly. The qualitative behavior however, is similar for longer chains.

As a first step, we pick 10^6 independent conformations from the lowest-temperature partition of the multiple Gaussian modified ensemble for every $0.030 \leq \delta \leq 0.220$ in steps of $\Delta\delta \approx 0.015$. The inverse microcanonical temperatures of those conformations are in the range $4 \leq \beta \leq 5$. For the identification of the solid phases, it is useful to measure the integrated radial distribution function with respect to the particle closest to its center of mass (com), i.e., the total number of monomers inside a sphere of radius r around the center monomer i_{com}:

$$N^{\mathrm{s}}_{i_{\mathrm{com}}}(r) = \sum_{i \neq i_{\mathrm{com}}} \Theta(r - r_{i,i_{\mathrm{com}}}) \,, \tag{5.9}$$

where $r_{i,i_{\mathrm{com}}}$ is the distance between monomer i and the center monomer, and $\Theta(r)$ is the Heaviside function. The results are shown in Figure 5.11, where each individual curve is the average over the data measured for each of the 10^6 conformations. One can clearly differentiate two types of curves for $N^{\mathrm{s}}_{i_{\mathrm{com}}}(r)$. For $\delta = \delta_{\mathrm{max}} \approx 0.220$ we know that the monomer positions correspond to the vertices in two icosahedral layers with radii of circumscribed spheres of 0.67 and 1.33, containing 13 and 55 monomers, respectively. That fact is clearly supported by the corresponding jumps in $N^{\mathrm{s}}_{i_{\mathrm{com}}}(r)$ marked by grid lines at the bottom scale.

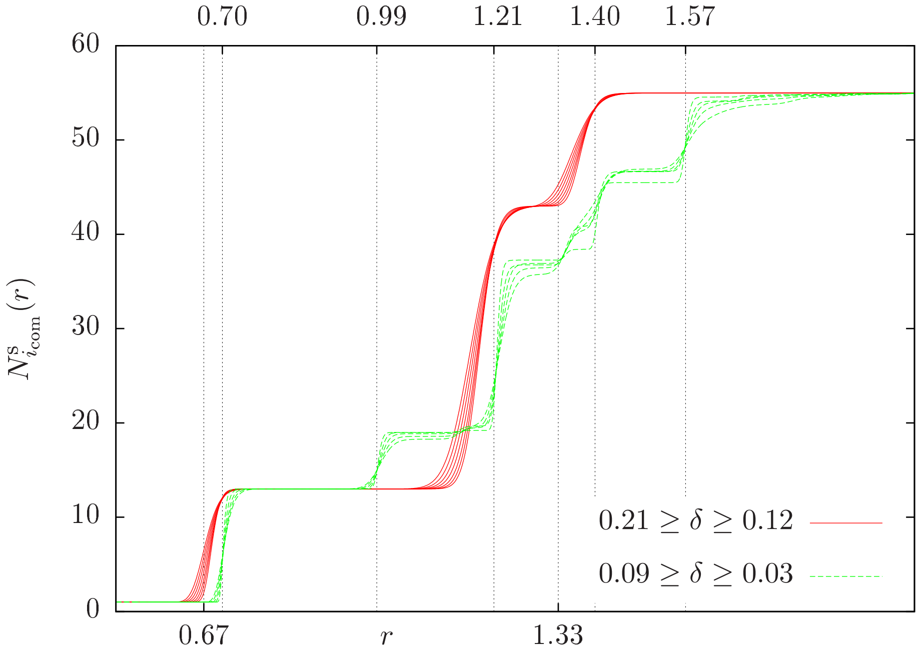


Figure 5.11: Averaged integrated radial distribution functions $N^{\mathrm{s}}_{i_{\mathrm{com}}}(r)$ for the $N = 55$-mer at different values of δ. Data was obtained from many independent low-energy configurations at inverse microcanonical temperatures $4 \leq \beta \leq 5$. Curves are plotted for different values of δ in steps of $\Delta\delta \approx 0.015$. Grid lines and values at top scale correspond to nth nearest neighbor $(1 \leq n \leq 5)$ positions in the fcc-lattice with lattice constant r_0. Grid lines and values at bottom scale correspond to radii of circumscribed spheres of icosahedra with edge lengths r_0 and $2r_0$, cf. Equation (5.3).

For decreasing interaction range, starting at $\delta < 0.12$, the shape of $N^{\mathrm{s}}_{i_{\mathrm{com}}}(r)$ changes qualitatively, indicating that the low-energy states are not icosahedral anymore, which is consistent with Figure 5.10 (bottom). The function now shows jumps at radii corresponding to nth nearest neighbor distances in the fcc lattice (upper scale and grid lines). We emphasize that this crossover picture is very stable, even though we measure at temperatures well above $T = 0$.

In order to unravel the structural details, we now look at the putative ground-state structures and measure their (binned) pair distribution function:

$$g(r) = \sum_j \left(N^{\mathrm{s}}_j(r + 0.5\Delta r) - N^{\mathrm{s}}_j(r - 0.5\Delta r) \right) , \tag{5.10}$$

where we set $\Delta r \leq 10^{-2}$. In other words, we measure distances between all pairs of monomers in the configuration, rather than only the distance of all monomers from a single center monomer as above, and count them in a histogram. We can clearly differentiate three different structural types in different regions of the interaction length. We plot $g(r)$ in Figure 5.12 for three representative values of δ and visualize the corresponding conformations in Figure 5.13. The peaks of the red curves (open squares) in Figure 5.12 correspond to icosahedral structures, which have been discussed above. For very small δ (blue peaks, filled diamonds), i.e., for very short ranged potentials, we find that all peaks coincide with nearest neighbor positions in the fcc lattice. In Figure 5.12, the values at the top scale and the grid lines correspond to the nth-to-nearest neighbor distances ($1 \leq n \leq 19$) in the fcc lattice. All peaks of $g(r)$ of the ground state at $\delta \approx 0.030$ agree very well with these values. However, there are structures in-between (green peaks, filled squares), which are neither icosahedral nor completely fcc structures. Those structures resemble ground states found for atomic, range-dependent Morse clusters [5, 6]. In fact, the ground state at $\delta = 0.110$ corresponds to the decahedral structure "55C" found in [6] (cp. Figure 5.13 b and Figure 7 in [6]). For

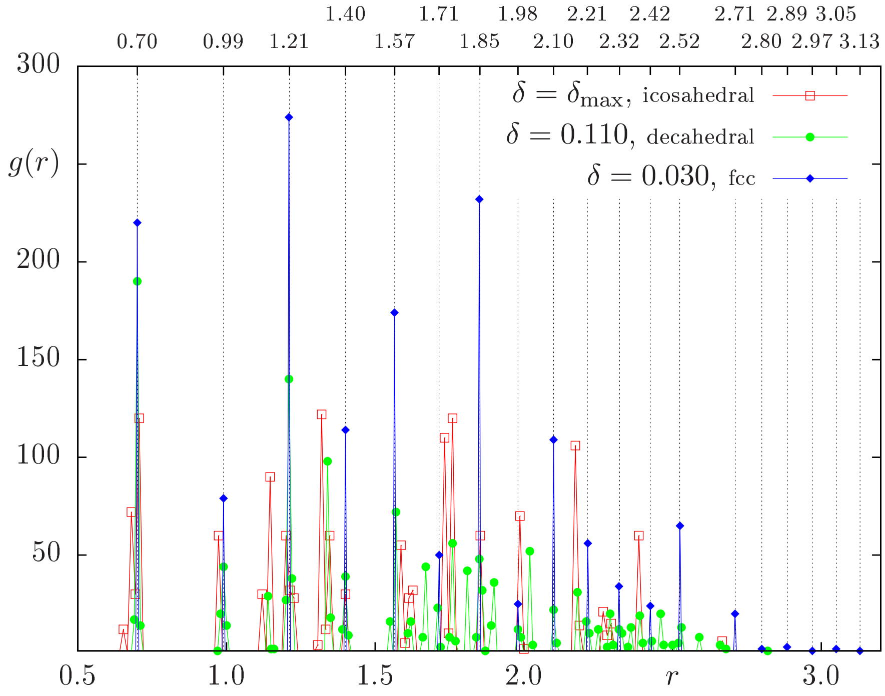


Figure 5.12: Pair distribution function for ground states found at $\delta = \delta_{\max}$ (open squares), $\delta \approx 0.110$ (filled circles) and $\delta \approx 0.03$ (filled diamonds). Grid lines and values at top scale correspond to nth nearest neighbor ($1 \leq n \leq 19$) positions in the fcc-lattice with lattice constant r_0, cf. Equation (5.3).

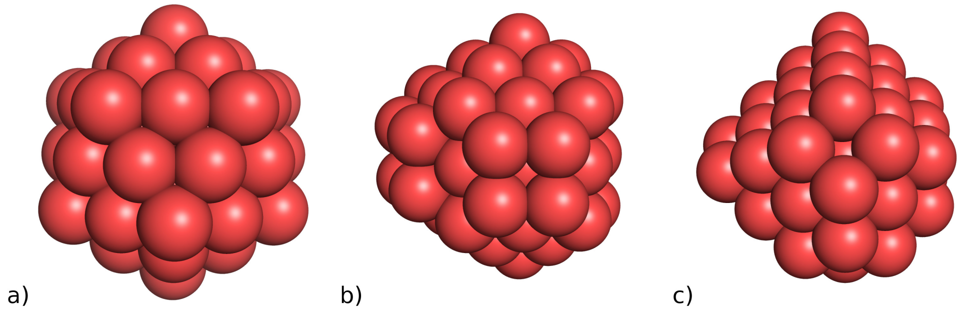


Figure 5.13: Visualizations of ground-state structures corresponding to data plotted in Figure 5.12 a) Icosahedron at $\delta = \delta_{\max}$, cf. [9, 10]; b) $\delta \approx 0.110$, cf. decahedral structure 55C in [6]; c) $\delta \approx 0.030$, all monomers occupy atomic positions at the fcc lattice.

other values of δ close to 0.110, we also find the defective decahedral structures described there.

5.4 Conclusions

We have studied the influence of the interaction length of a Lennard-Jones potential on the structural behavior of an elastic, flexible polymer. We applied advanced simulation methods by using replica-exchange parallel tempering on graphics cards and multiple Gaussian modified ensembles to tackle the strong first-order-like behavior of the freezing transition. We employed the microcanonical inflection-point analysis method [13] that made it possible to construct a structural phase diagram for an elastic flexible polymer with 90 monomers. This analysis of the microcanonical entropy allows to resolve the positions of structural transitions which are much more uncertain in canonical analyses. We are able to precisely locate and also classify transitions by investigating the first and second derivative of the entropy. Both derivatives can be evaluated easily. We find that the liquid phase, separating the extended coil “gas”-like phase from the crystalline solid phases, becomes smaller for shorter interaction ranges. For sufficiently small interaction range, we eventually observe a crossover of transition lines. The crossover point marks the triple point in the thermodynamic limit and thus the direct transition from gas to solid. According to the microcanonical signals, both transitions remain separate for finitely long polymers. Summarizing the structural analysis of the solid phases, we find that the icosahedral ground-state structures identified for the standard Lennard-Jones potential [9, 10] do not survive at smaller interaction ranges. In analogy to former studies of atomic Morse clusters [5–7], we find transitions from icosahedral to decahedral and fcc structures for decreasing interaction range. These transitions are strongly influenced by the repulsive part of the potential as they are mainly triggered by released stresses in the conformation.

Chapter 6

Polymer adsorption on nanocylinders

6.1 Introduction

In this chapter the thermodynamic behavior of a coarse-grained flexible homopolymer in the presence of an attractive cylindrical substrate is investigated. We are interested in the structural phase behavior of the polymer with changing surface attraction strengths. The effective monomer–surface attraction strength is associated with the radius of a modelled nanocylinder. The objective of this study is the creation of a complete phase diagram indicating all possible structures the polymer can assume as a function of the radius of the nanocylinder and temperature. Previous studies of polymers adsorbed on planar surfaces [68–73] on lattices as well as in continuum, already provide insight in the structural phases of adsorbed chains. Recently also the adsorption on curved surfaces [74–78] was the focus of many studies. We try to connect the findings of adsorption on curved surface and those on planar surfaces, since the limit of a infinite cylinder radius yields the same characteristics of a planar surface. We simulate a 30-mer interacting with nanocylinders of ten different radii and five materials using parallel tempering. Multiple geometrical observables and their thermal fluctuations are analyzed to locate and identify structural phase transitions. Finally we are able to assemble pseudophase diagrams for all 5 nanocylinder materials and identify desorbed and adsorbed polymer conformations.

6.2 Model

The polymer that is interacting with a nanocylinder in this chapter is modelled the same way as in Section 4.2.1. Interaction between all monomers is described by a shifted and truncated Lennard-Jones potential

$$U_{\mathrm{LJ}}^{\mathrm{mod}}(r_{ij}) = U_{\mathrm{LJ}}(r_{ij}) - U_{\mathrm{LJ}}(r_c), \tag{6.1}$$

$$U_{\mathrm{LJ}}(r_{ij}) = 4\epsilon \left[\left(\frac{\sigma}{r_{ij}} \right)^{12} - \left(\frac{\sigma}{r_{ij}} \right)^{6} \right]. \tag{6.2}$$

In addition to the Lennard-Jones term the FENE potential

$$U_{\mathrm{FENE}}(r_{ii+1}) = -\frac{K}{2} R^2 \log \left[1 - \left(\frac{r_{ii+1} - r_0}{R} \right)^2 \right], \tag{6.3}$$

is used to model the bonds between adjacent monomers. The parametrization for both potentials is the same as before, compare Chapter 4.2.1. Investigated was a polymer with $N = 30$ monomers.

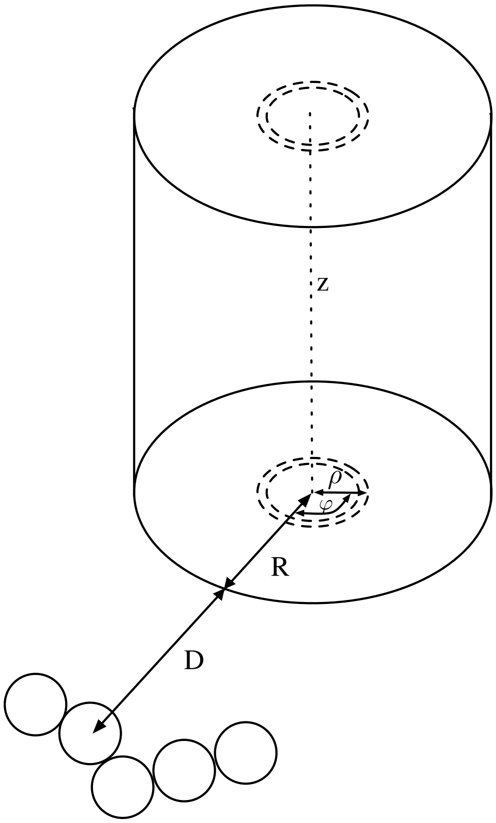


Figure 6.1: Coordinates for integration of Lennard-Jones potential in case of the nanocylinder

For the nanocylinder a homogeneous distribution of atoms is assumed. The potential energy that affects a monomer with distance D from the cylinder surface for a nanocylinder with radius R is calculated as follows:

$$U_{\text{cyl}}(R,D) = \epsilon_{\text{c}}\pi \int_0^{2\pi} d\varphi \int_0^R \rho d\rho \left(\frac{63}{64x^{11/2}} - \frac{3}{2x^{5/2}}\right), \tag{6.4}$$

with

$$x = (D+R)^2 + \rho^2 - 2\rho(D+R)\cos\varphi. \tag{6.5}$$

The potential is essentially a Lennard-Jones like interaction of monomers with the nanocylinder. In the potential of the form as in Equation (6.2), with $\sigma \equiv 1$, the Cartesian coordinates are transformed into cylindrical coordinates. Then an integration over the z-coordinate from

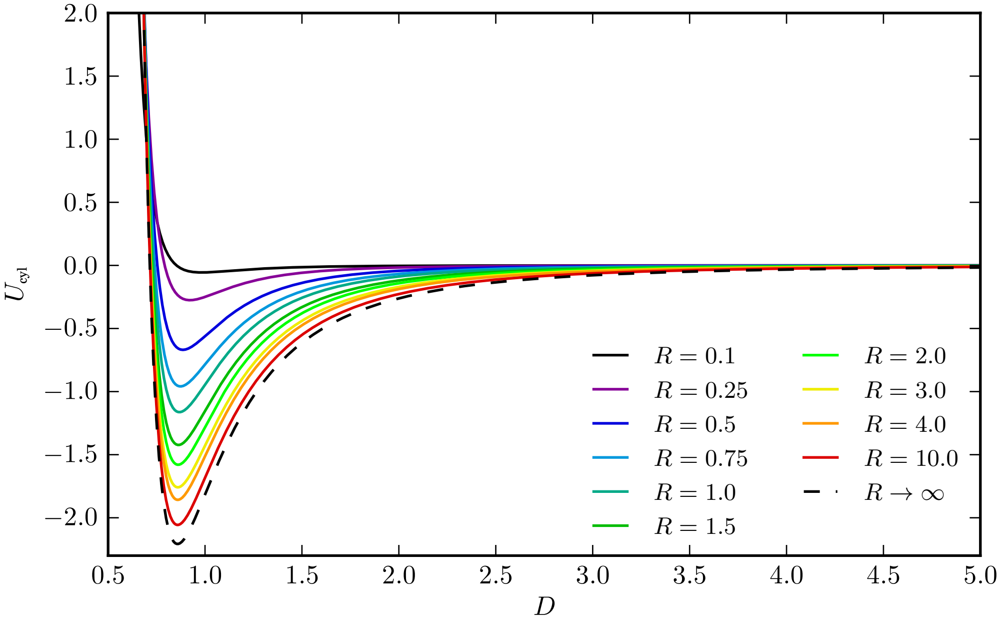


Figure 6.2: Cylinder potential for $\epsilon_c = 1.0$ and different nanocylinder radii R. The dashed line represents the potential for $R \to \infty$, which coincides with the potential of the wall.

negative infinity to infinity is performed along the cylinder axes as shown in Figure 6.1. The resulting potential in Equation 6.4 can only be evaluated numerically. To avoid the compute intensive task of calculating the cylinder potential, which is needed at every Monte Carlo step, the potential is evaluated prior to the simulation and stored in an array. It is restricted to a minimum cutoff distance $D_{\mathrm{min,cut}} = 0.5$ and a maximum cutoff distance $D_{\mathrm{max,cut}} = 5.0$. Below the minimum distance the potential is set to infinity and above the maximum distance it is set to zero. The cylinder potential for $\epsilon_c = 1.0$ for different values of R is plotted in Figure 6.2. Different value for ϵ_c can be understood as different materials the nanocylinder is composed of. In the limiting case $R \to \infty$ the cylinder potential coincides with a wall potential, that could be obtained by integrating the Lennard-Jones potential over the half-space for $y < 0$[1] The wall potential has the form

$$U_{\text{wall}} = 4\pi\epsilon_{\text{c}} \left(\frac{1}{45y^9} - \frac{1}{6y^3} \right). \tag{6.6}$$

This allows at least a qualitative comparison of structures to previous studies of polymer adsorption at walls [40, 73, 79]. For a given conformation $\mathcal{C} = (\vec{r}_1, \ldots, \vec{r}_N, R)$ for a polymer of length N and a nanocylinder with radius R, the total energy is

$$E(\mathcal{C}) = \frac{1}{2} \sum_{\substack{i,j=1 \\ i \neq j}}^{N} U_{\text{LJ}}^{\text{mod}}(r_{ij}) + \sum_{i=1}^{N-1} U_{\text{FENE}}(r_{ii+1}) + \sum_{i=1}^{N} U_{\text{cyl}}(R, D_i). \tag{6.7}$$

Simulations were performed in a simulation box, with steric walls in $x-$ and $y-$direction. The size of the box in both $x-$ and $y-$direction is set to $R + 2N$. This restriction keeps the polymer from moving too far away from the cylinder. A restriction of the $z-$direction is not necessary, since the cylinder spans over the whole $z-$axis. Simulations were performed using parallel tempering at 80 temperatures for 5 different nanocylinder materials, i.e., for $\epsilon_{\text{c}} = \{1.0, 2.0, 3.0, 4.0, 5.0\}$. For each temperature, each value of ϵ_{c}, and each radius, $5 \cdot 10^7$ sweeps were performed to gather sufficient statistical data for the canonical quantities with reasonable error bars. The error bars were obtained using the binning method[80–82].

6.3 Results

In the following section we look at various observables and their thermal fluctuations to locate and identify structural transitions of a 30-mer interacting with nanocylinders composed of 5 different materials. For each material 10 different radii $R = \{0.1, 0.25, 0.5, 0.75, 1.0, 1.5, 2.0, 3.0, 4.0, 10.0\}$ were simulated. The effective attraction of the

[1] Integration over the half-space $x < 0$ is also possible. Since the cylinder stretches along the z-axis, this would mean that either the y-z-plane or the x-z-plane would represent the wall boundary.

cylinder to the polymer grows with the cylinder radius, as more material is added to the cylinder, as plotted in Figure 6.2. The information of all observables, i.e., the peak positions, minima, and maxima, is combined to construct structural phase diagrams for all nanocylinder materials as a function of the respective cylinder radius.

6.3.1 Thermodynamics

Specific heat c_V

The first observable we want to analyze is the specific heat. Figure 6.3 shows the complete set of specific heat plots for all radii and values of ϵ_{c}. For $\epsilon_{\mathrm{c}} = 1.0$ (Figure 6.3 top left) the specific heat plots show two peaks for most radii. A stable peak for all radii at $T \sim 0.35$ signals the freezing or liquid-solid transition. Below that temperature all polymers, adsorbed or desorbed, have a well defined crystalline structure. For $R = 10.0$ the freezing temperature is slightly shifted to a lower temperature. The reason for this becomes apparent later, when the type of crystal that is formed is investigated. The second peak visible for all radii $R > 0.25$ identifies the adsorption transition. With increasing cylinder radius the adsorption temperature increases as well. For $\epsilon_{\mathrm{c}} \geq 2.0$ again the freezing transition is a stable peak at $T \sim 0.35$. Deviations from the freezing temperature are caused by an overlap of multiple structural changes, e.g. for $R = 0.25, \epsilon_{\mathrm{c}} = 2.0$ the freezing and adsorption transitions are so close to each other, that they are visible as only one peak. The adsorption temperatures again are shifted to higher temperatures with increasing cylinder radius. Since our main interest is to study the adsorbed structures, the temperature interval is limited to $T \in [0.1, 2.5]$. That means for those radii that do not show a peak at high temperatures in the specific heat anymore, the adsorption temperature is above $T = 2.5$.

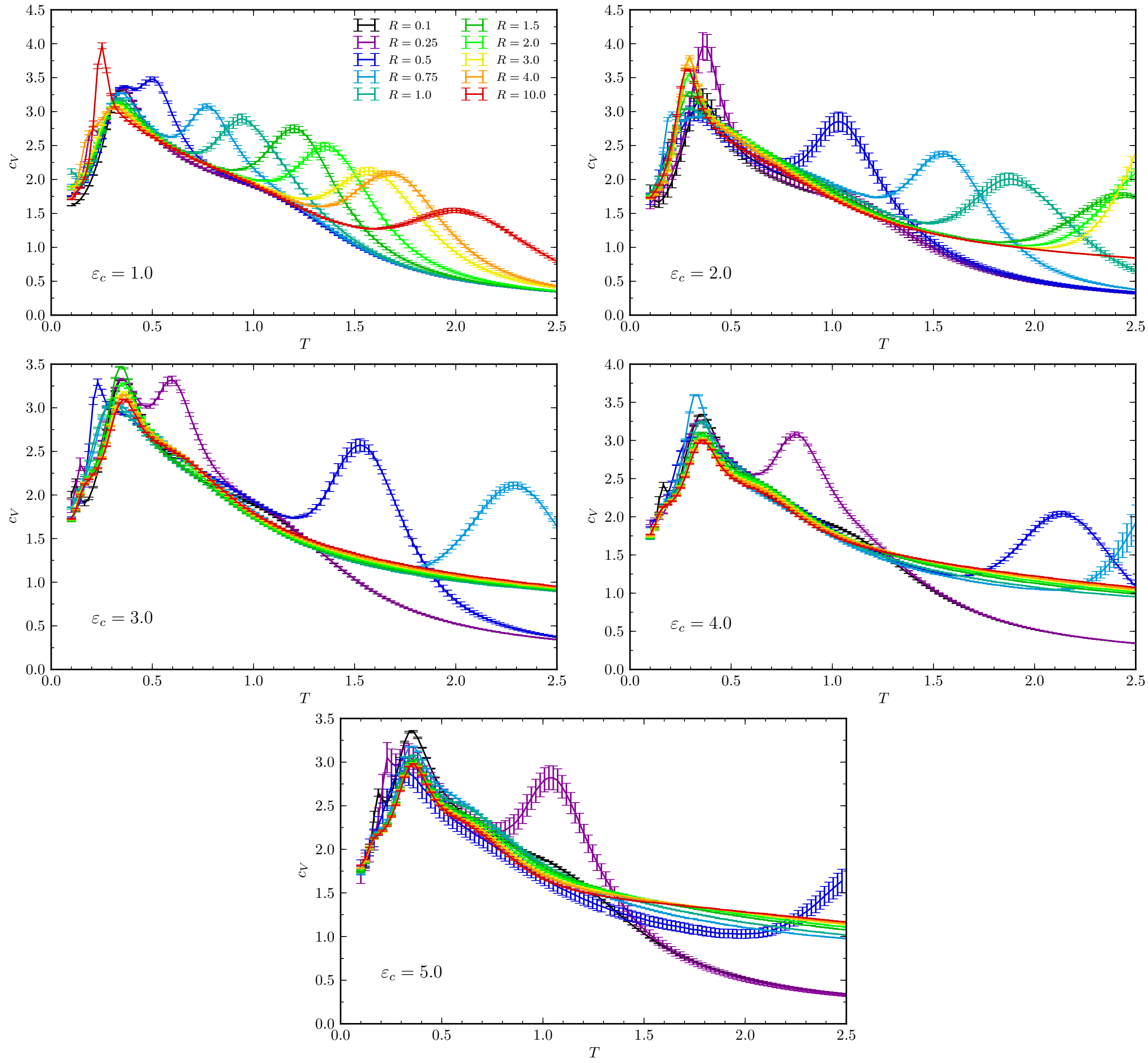


Figure 6.3: Specific heat curves for the interaction of a 30-mer with nanocylinders at 10 different radii, plotted for $\epsilon_c = 1.0$ (top left), 2.0 (top right), 3.0 (middle left), 4.0 (middle right), and 5.0 (bottom).

Radius of gyration and its thermal fluctuations

As in the chapters above the squared radius of gyration r^2_{gyr} is a geometrical observable that helps identifying structural phases of the polymer. In Figure 6.4 is the complete set of squared radius of gyration plotted versus temperature. An interesting feature of those plots is the spreading of r^2_{gyr} at the high temperature end for $\epsilon_{\mathrm{c}} \geq 2.0$. This is caused by the fact, that for stronger effective attraction the polymer gets adsorbed at high temperatures, where the polymer would assume elongated coil-like structures and the cylinder surface restricts the monomer movement. The thermal fluctuations of r^2_{gyr} calculated according to Equation 2.16, shown in Figure 6.5, exhibit peaks at those temperatures where the change of r^2_{gyr} with temperature is maximal. For $\epsilon_{\mathrm{c}} = 1.0$ and 2.0 the thermal fluctuations of r^2_{gyr} show two pronounced peaks. The peak at low temperatures $T \sim 0.35$ indicates the freezing transition. For $\epsilon_{\mathrm{c}} = 1.0$ and $R \geq 3.0$ the fluctuation of the radius of gyration show minima around the freezing temperature. The crystalline structures for these polymers have a larger radius of gyration than the liquid structures. The polymer flattens out on the cylinder surface, resulting in an increase of r^2_{gyr}. The peak at high temperatures signals the Θ-transition. The polymer collapses from the coil-like structures at high temperature to compact structures below. Another interesting feature of the collapse transition is a shift to lower temperatures for all adsorbed polymers. The adsorption itself can not be identified by looking at the radius of gyration alone, but information from other quantities allows this conclusion. The collapse temperature is $T \sim 1.35$ for desorbed polymers and drops below $T = 1.0$ for $\epsilon_{\mathrm{c}} \geq 2.0$ where adsorption appears at higher temperatures. For $\epsilon_{\mathrm{c}} \geq 3.0$ there are multiple smaller peaks and shoulders at temperatures below $T \sim 0.5$, which correspond to the freezing/flattening of the polymer in the presence of the nanocylinder surface.

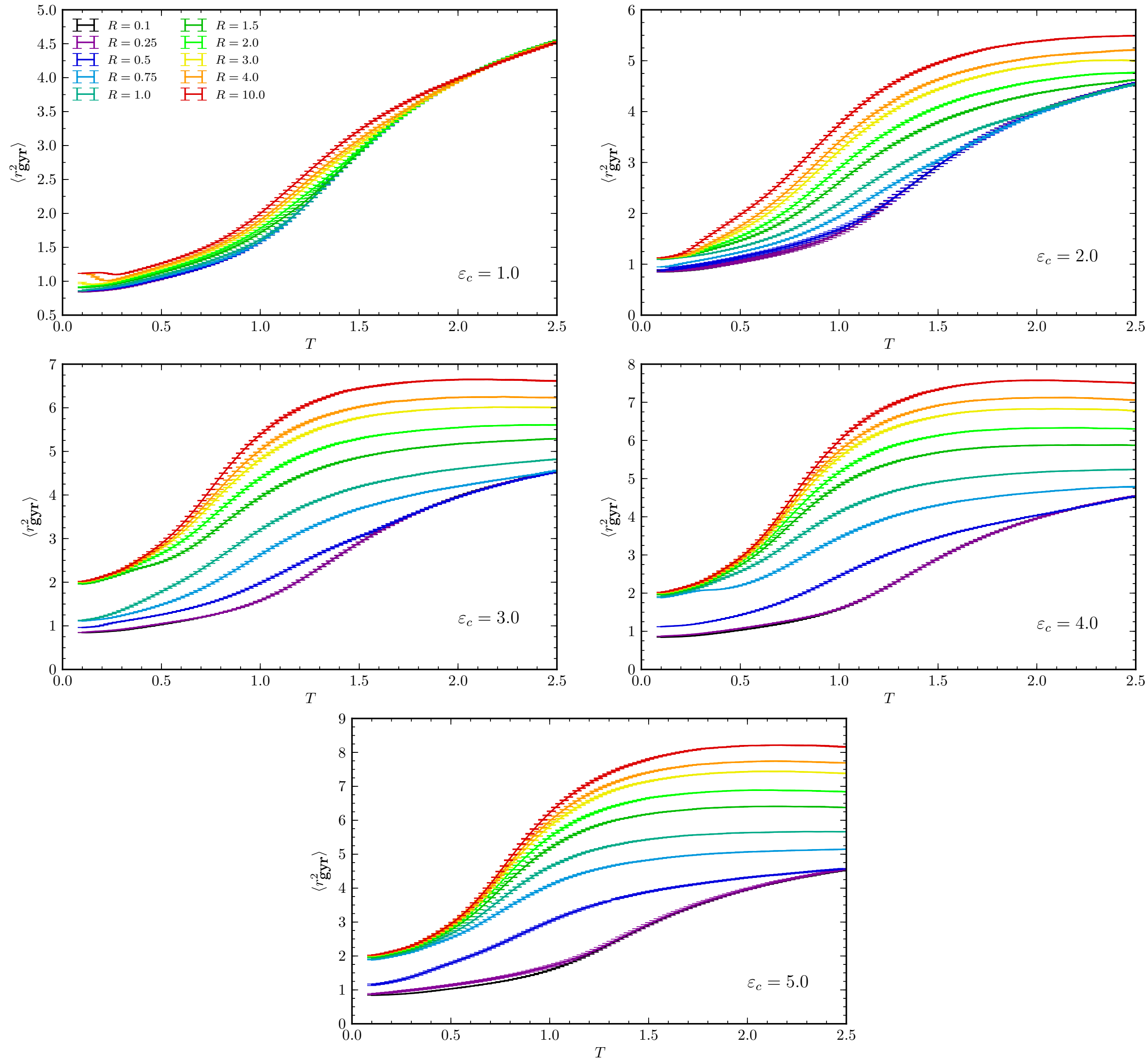


Figure 6.4: Squared radius of gyration as a function of temperature for the 30-mer.

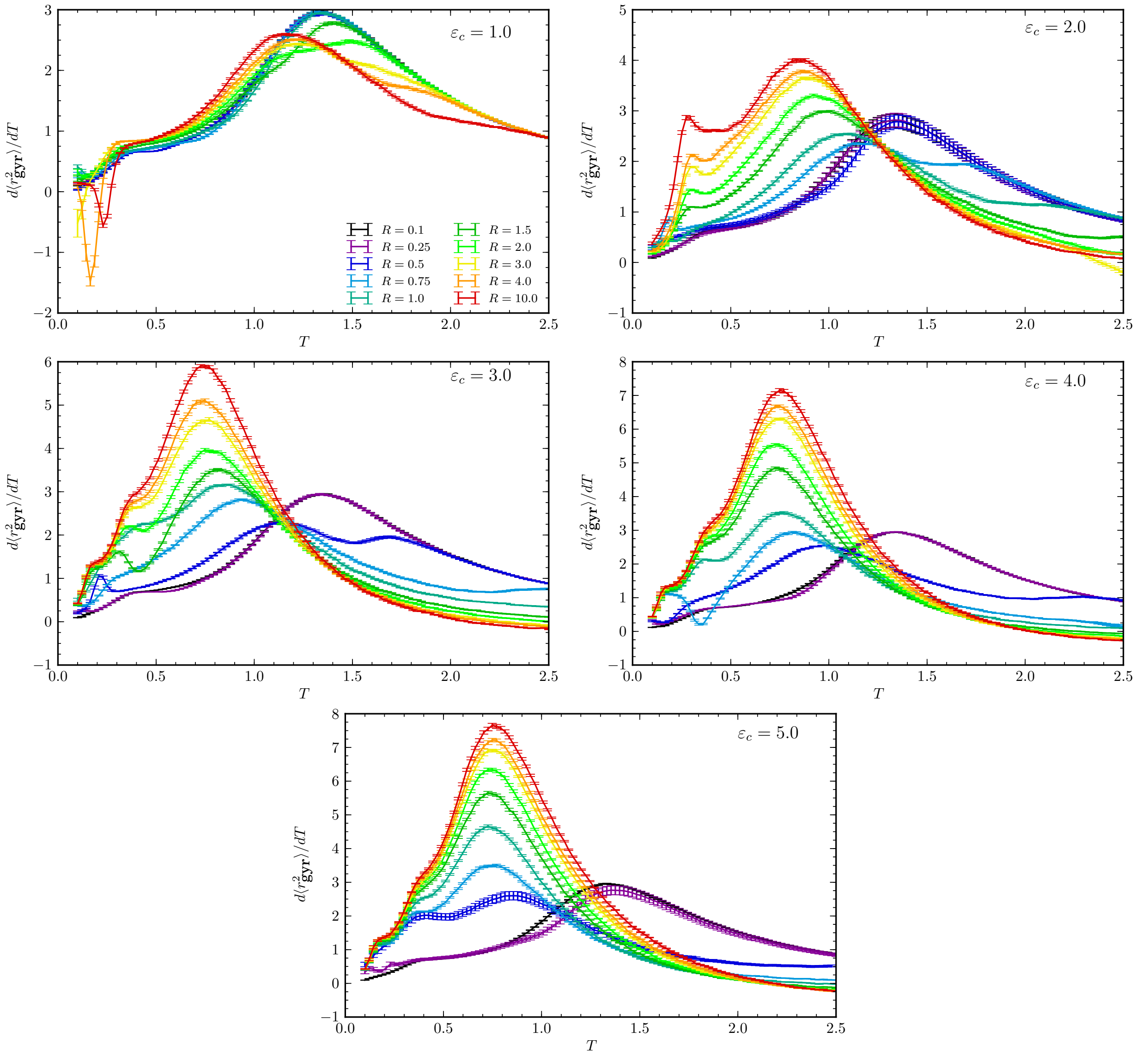


Figure 6.5: Fluctuation of the squared radius of gyration vs. temperature for the 30-mer in presence of cylinders with 10 different radii and 5 different values of ϵ_c.

Polymer extension perpendicular to the cylinder surface

To obtain information mainly about the low temperature structures of the polymer, I introduce a new observable $r_{\mathrm{perp}} = D_{\mathrm{max}}(\mathcal{C}) - D_{\mathrm{min}}(\mathcal{C})$ that measures the difference of distance between the monomer farthest away from the cylinder $D_{\mathrm{max}}(\mathcal{C})$ and the monomer that is closest to the cylinder surface $D_{\mathrm{min}}(\mathcal{C})$. In Figure 6.6 this quantity is plotted for all simulated materials and radii. The value of r_{perp} at the lowest temperature gives insight about the structures of the polymers. For $r_{\mathrm{perp}} \approx 0$ all monomers have about the same distance from the cylinder surface, i.e., the polymer forms one layer on the surface. $r_{\mathrm{perp}} \approx r_0$ means the polymer assumes a two layered structure. Three layers are associated with $r_{\mathrm{perp}} \approx 2r_0$. Due to the length of the polymer $N = 30$ higher level of layering is not observed and higher values of r_{perp} at low temperatures can not be connected with a layered structure anymore. The thermal fluctuations of this quantity, as shown in Figure 6.7, show two peaks for each radius. The peaks at higher temperatures correspond to the adsorption transition and are in good agreement with the adsorption temperatures signaled in the specific heat. The low temperature peaks $T \leq 0.4$ correspond to the freezing transition. A positive peak value means in this case a decrease in r_{perp} when going from higher to lower temperatures. If the thermal fluctuations show a peak with a negative value, that means the distance of monomers from the surface increases for the crystal when compared to the unordered liquid structure.

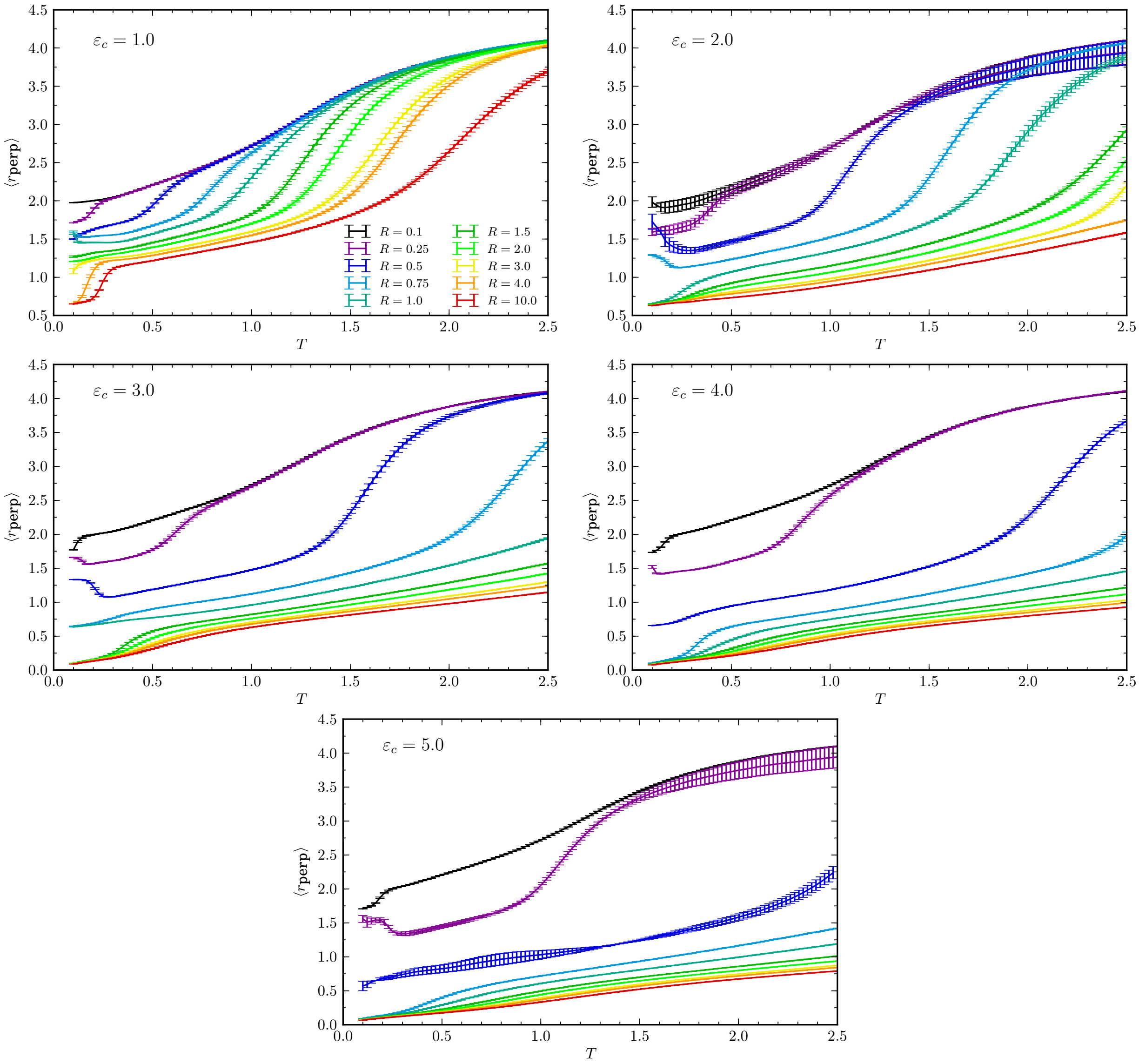


Figure 6.6: Extent of the 30-mer perpendicular to the cylinder surface r_{perp} plotted against temperature.

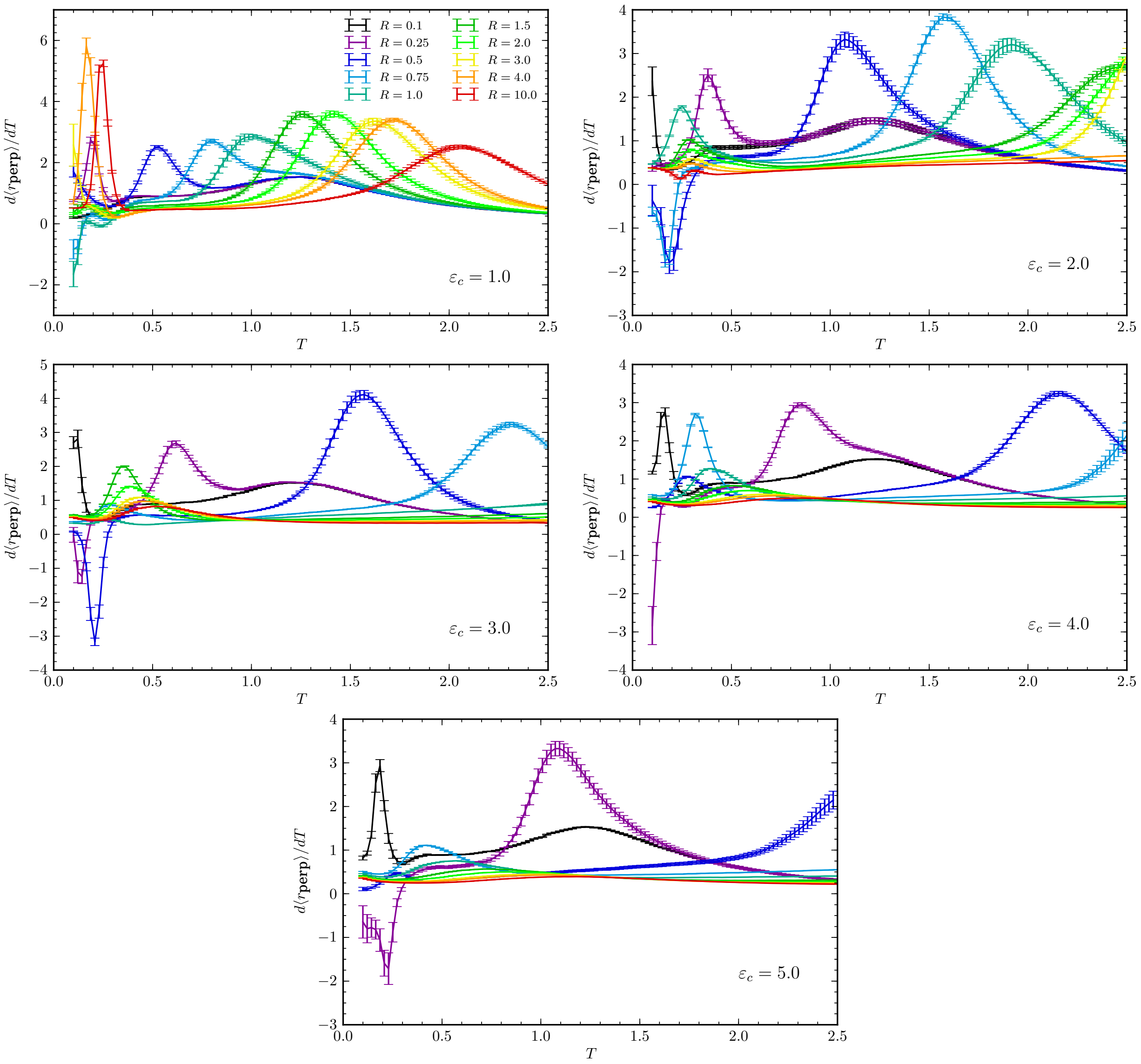


Figure 6.7: Thermal fluctuations of r_{perp} as a function of temperature.

Number of monomer–surface contacts

The last quantity that was investigated is the number of monomer–surface contacts n_c. A monomer is considered to be in contact with the nanocylinder when its distance from the cylinder is $D \leq 0.95$, which is slightly above the distance where the cylinder potential becomes minimal. In Figure 6.8 the number of contacts for the investigated 30-mer is plotted as a function of temperature for all simulated values of R and $\epsilon_{\rm c}$. The number of contacts provides information about the adsorption of the polymer as well as information about the low temperature structures. If $n_c \sim 30$, i.e., all monomers are in contact with the cylinder, the polymer forms one layer on the surface. For $n_c \sim 15$ around half the monomers are in contact with the cylinder surface. At the lowest temperature this corresponds to a two layered crystal structure. Three layers can be related to $n_c \sim 10$. This interpretation agrees with the information obtained from the analysis of $r_{\rm perp}$ and by visualizing the corresponding polymer structures. Analyzing the thermal fluctuations of this quantity, see Figure 6.9, gives additional information about the adsorption transition. $d\langle n_c \rangle/dT$ shows a peak where the number of surface contacts increases, going from high temperatures to low temperatures. The peak positions are in agreement with the corresponding specific heat peaks and the peak positions of the thermal fluctuations of $r$perp. One exception is observed at $R = 10.0$ and $\epsilon_{\rm c} = 2.0$. This positive peak suggests that the number of surface contacts is smaller at lower temperatures. When looking at the actual number of contacts of that parameter set in Figure 6.8 the number indeed decreases from 21 to 20 contacts, which leads to the peak in the fluctuation quantity. This is no fundamentally new or different transition, but it is merely caused by the definition of the number of contacts. The one monomer causing the peak in $d\langle n_c \rangle/dT$ has a distance very close to the threshold and is not counted anymore in the crystal at low temperatures.

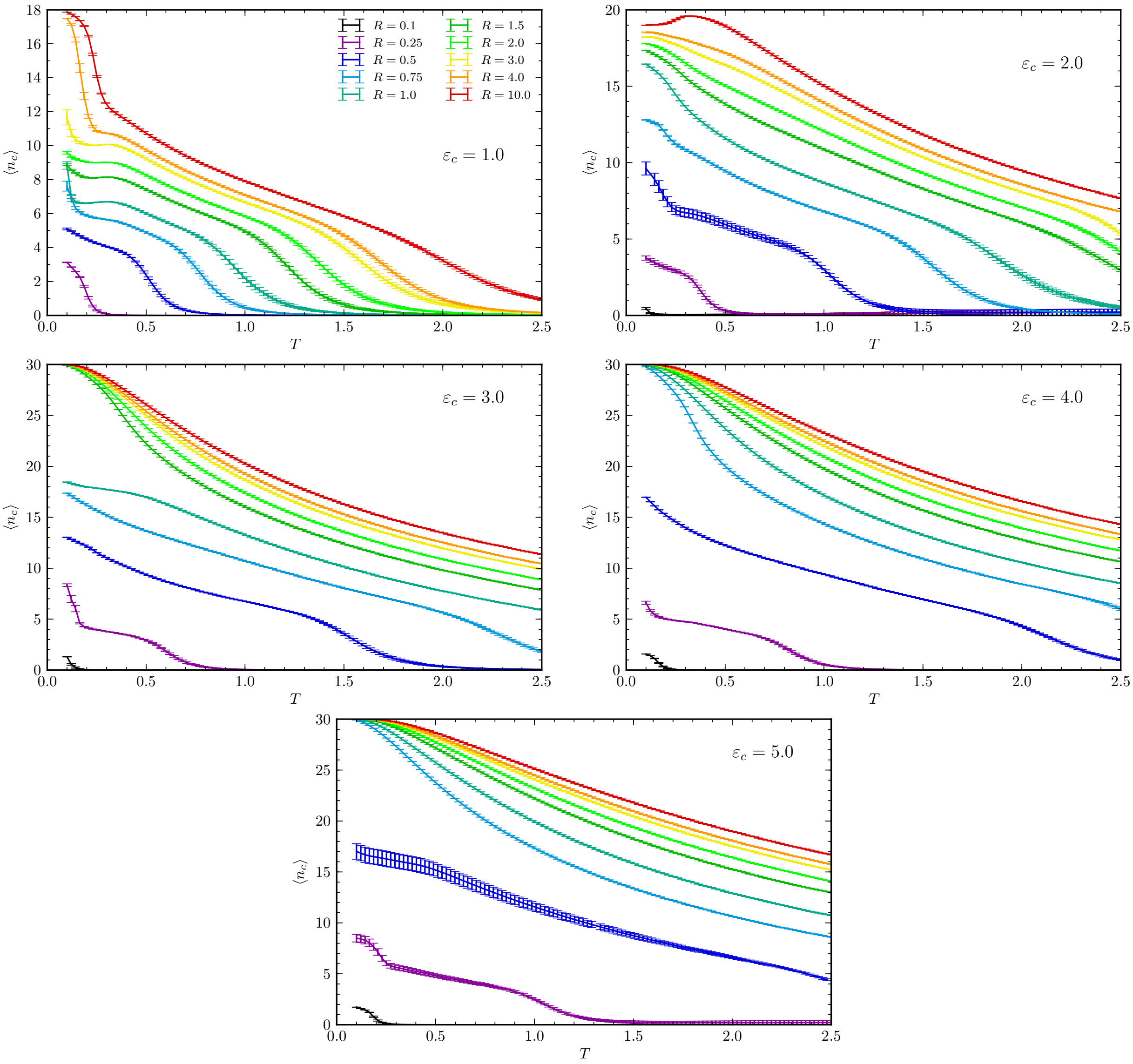


Figure 6.8: Number of contacts of a 30-mer with nanocylinders of different radii vs temperature.

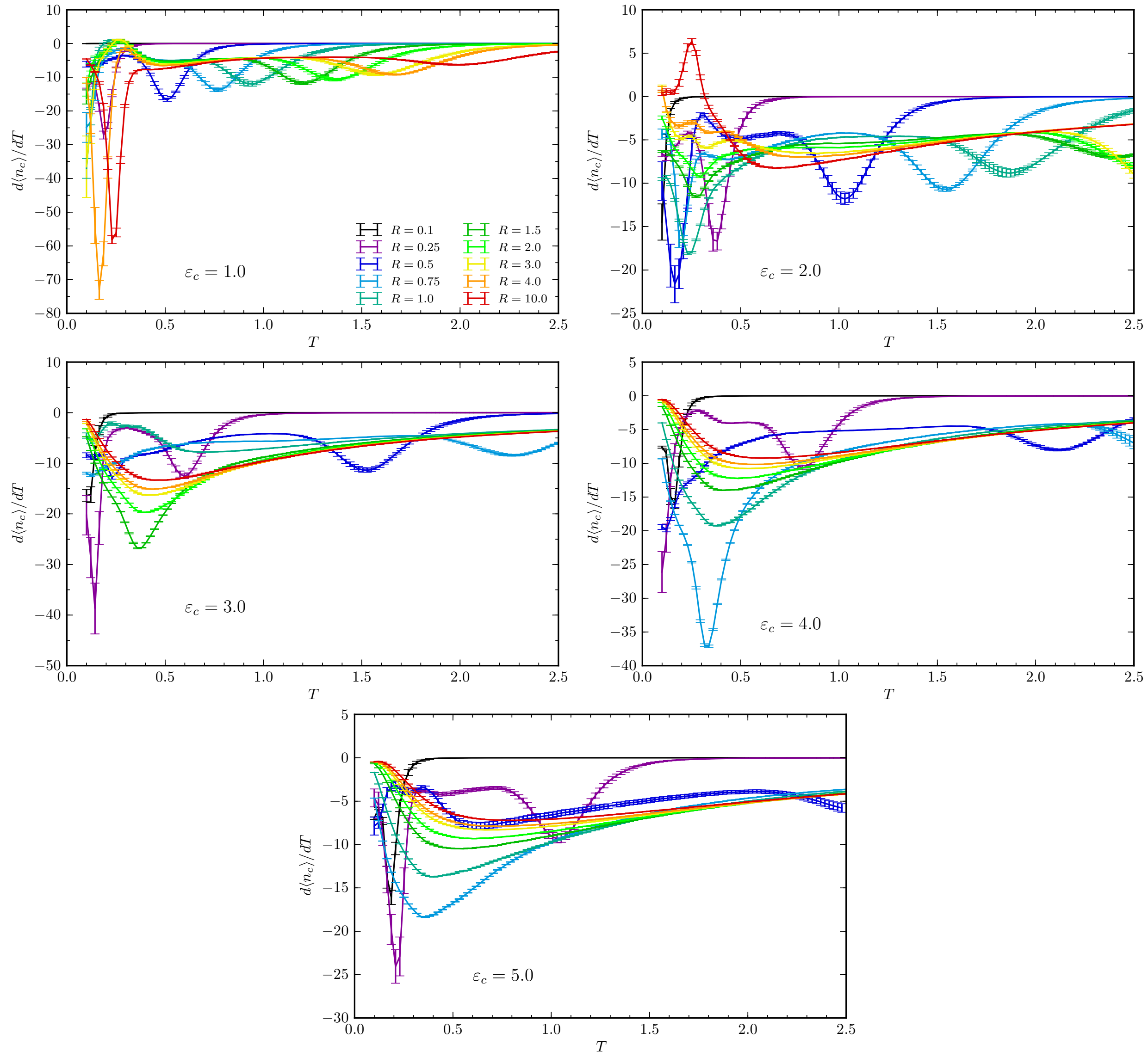


Figure 6.9: Fluctuations of the number of contacts as a function of temperature.

6.3.2 Pseudophase diagrams

The information from all canonical quantities about transitions and structural changes to the polymer is accumulated and forms the foundation of the construction of the pseudophase diagrams in Figures 6.10 ($\epsilon_c = 1.0$), 6.11 ($\epsilon_c = 2.0$), 6.12 ($\epsilon_c = 3.0$), 6.13 ($\epsilon_c = 4.0$), and 6.14 ($\epsilon_c = 5.0$). The thickness of the lines, marking the transitions, indicates that precise transition temperatures are difficult to ascertain, since the canonical quantities for a finite system, such as the short polymer chain investigated here, have slightly different peak positions.

From the high temperature specific heat peaks and the monomer–surface contact numbers and their fluctuations, the separation into two main parts of the phase diagrams can be derived. Three desorbed phases can be identified: desorbed elongated (DE) at high temperatures, the desorbed globule phase (DG), and the desorbed compact or crystal phase (DC) at low temperatures. Representative conformations of the desorbed phases are depicted in Table 6.1. The three main adsorbed phases, (AE) for adsorbed elongated, adsorbed globule (AG), and adsorbed crystal (AC), can further be subcategorized according to the number of layers the polymer forms at the cylinder surface. For weak surface attraction, i.e., small cylinder radii and low ϵ_c, the polymer conformations have large similarities with the desorbed crystal. The crystallized polymer is adsorbed at the nanocylinder, forming the CC phase, where the monomer–monomer attraction is stronger than the monomer–surface attraction. The crystal is in contact with the surface, but number of surface contacts is appreciably smaller than for the AC3 phase. Hence, a flattening of the polymer on the surface does not happen in this region of the phase diagram. In the AC3 phase the polymer is attached to the nanocylinder in three layers, with around a third of its monomers forming bottom layer. With increasing cylinder radius, and thus effective surface attraction, there is a transition to two-layered crystals (AC2); and, finally, for strong surface–monomer attraction the polymer is completely flattened to a single layer in the AC1 phase. For $\epsilon_c = 4.0$ and 5.0 the phase diagrams show a distinct specialty in the adsorbed globule phase (AG1), for radii larger than $R \approx 1.0$ or $R \approx 0.75$ respectively, the surface attraction becomes strong enough that all monomers are in contact with the cylinder even before the polymer crystallizes at lower temperatures. In the case of $\epsilon_c = 5.0$, the strongest simulated surface attraction, single layered polymer conformations can be found even before the chain collapses (AE1). Example structures for all adsorbed phases can be found in Table 6.2. An interesting feature for all studied nanocylinder materials is, that the freezing transition temperature is the most stable at $T \sim 0.35$ within small deviations, for all radii. The adsorption temperature line moves to smaller radii with increasing surface attraction. Another interesting discovery is the crossover of collapse and adsorption temperature. With increasing surface attraction the crossover moves to smaller radii, while the temperature at which the crossover appears remains almost the same $T \sim 1.3$. The collapse transition temperature is lower for all adsorbed phases. It is nearly constant for the desorbed polymers and drops to lower temperatures for adsorbed polymers. The decrease in the collapse temperature becomes more pronounced with increasing surface attraction. This is caused be the conformational restrictions, the polymer suffers from in the adsorbed phase.

The phase diagrams for all values of ϵ_c have one characteristic in common. There is the desorbed region for small radii, i.e. weak surface attraction, and on the other hand the adsorbed region for strong surface attraction at large cylinder radii. The structures we have identified for the desorbed phases are essentially the same as for the free polymer [9, 10, 22], see also Chapter 4 and Table 4.4. Since the cylinder radius can be associated with the effective surface attraction, it seems obvious to compare our findings with a recent study of polymer adsorption at a planar surface with varying attraction strength [73, 79]. In that study a semi-flexible polymer was adsorbed on an attractive wall with a potential of the form in Equation 6.6. The pseudophase diagram for a chain with $N = 20$ monomers was constructed as a function of surface attraction and temperature. Essentially the phase diagram has the same features, namely, desorbed phases DE, DG and DC for weak surface attraction and various adsorbed phases for strong attraction. For the adsorbed phases the notation of the layered structures differs marginally, but they also find complete adsorbed single-layered conformations (AC1) and structures with more layers (AC2a and AC2b), see Figure 3 in [73]. The similarities found are undeniable, since for very large radii, in particular for $R \to \infty$, the cylinder potential and the wall potential coincide. For small radii the single-layered structures start wrapping around the cylinder. Because of the short chain length we do not find completely wrapped structures as in a study of polymer adsorption on nanowires [78]. In the case of the adsorption on a nanowire, the polymer forms highly ordered monolayers covering the wire. These structures strongly depend on the effective radius of the nanowire and only appear for large attraction strengths. We might find similar structures in our model for strong enough surface attraction and longer polymer chains. This will be of interest in possible future studies.

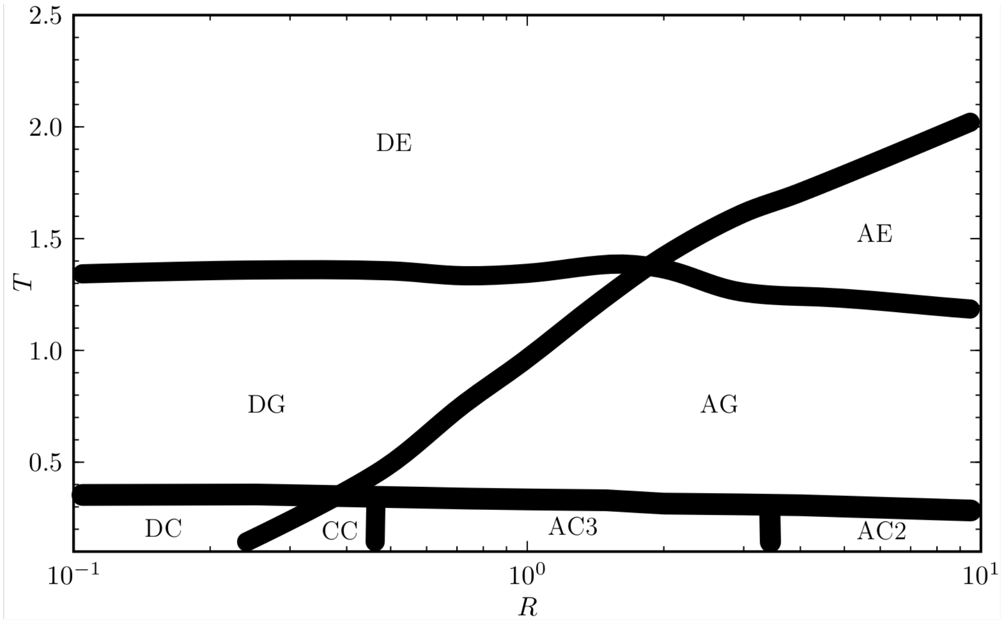


Figure 6.10: Pseudophase diagram of the 30-mer in the presence of a nanocylinder with $\epsilon_c = 1.0$. The thickness of the lines represents the uncertainties of the exact transition temperatures. The names of the structure are described in the text and representative structures are printed in Table 6.1 and Table 6.2.

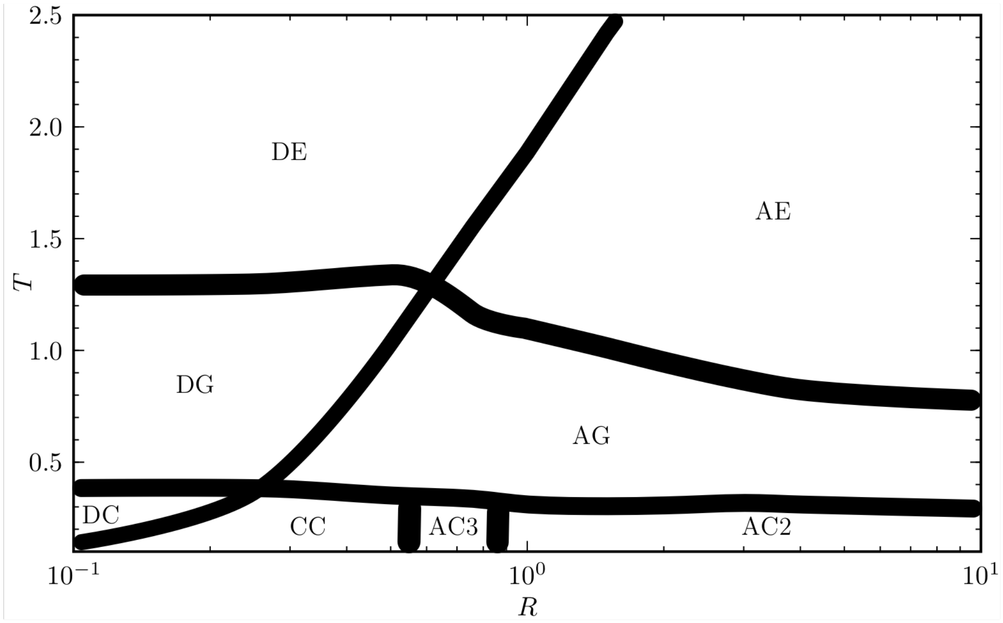


Figure 6.11: Pseudophase diagram of the 30-mer in the presence of a nanocylinder with $\epsilon_c = 2.0$.

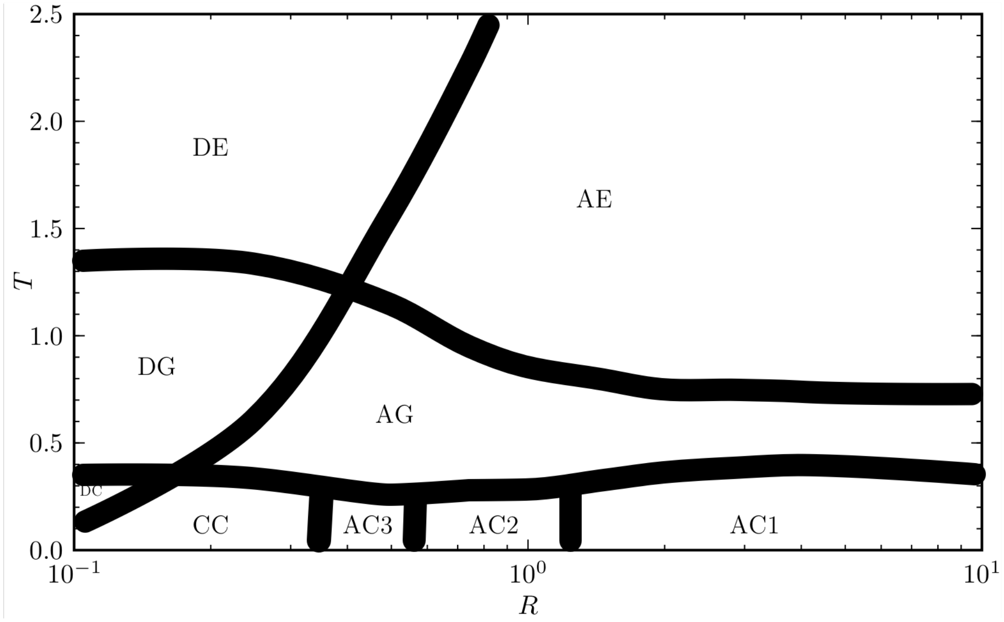


Figure 6.12: Pseudophase diagram of the 30-mer in the presence of a nanocylinder with $\epsilon_c = 3.0$.

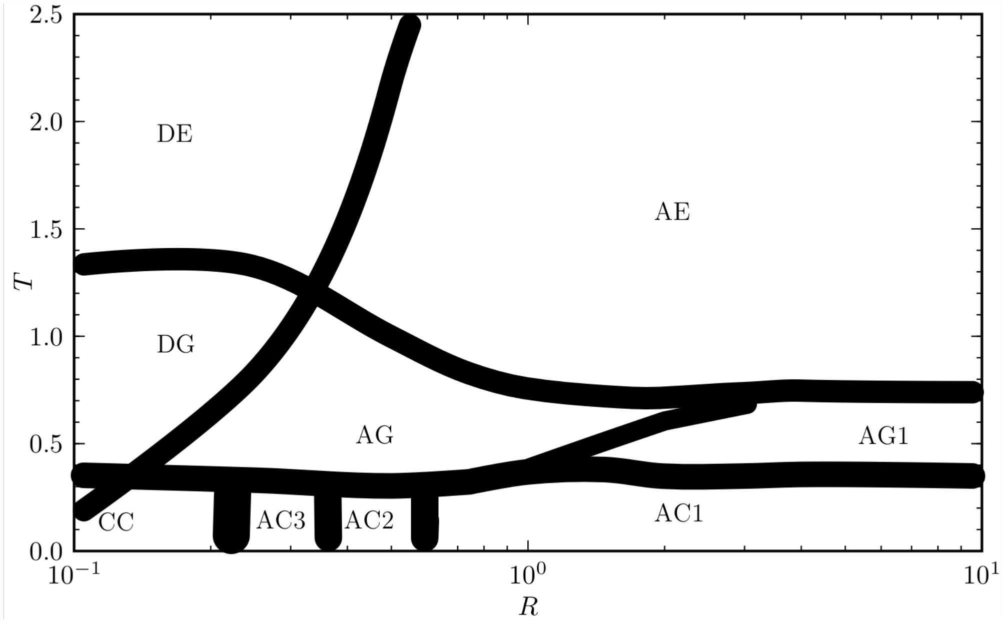


Figure 6.13: Pseudophase diagram of the 30-mer in the presence of a nanocylinder with $\epsilon_c = 4.0$.

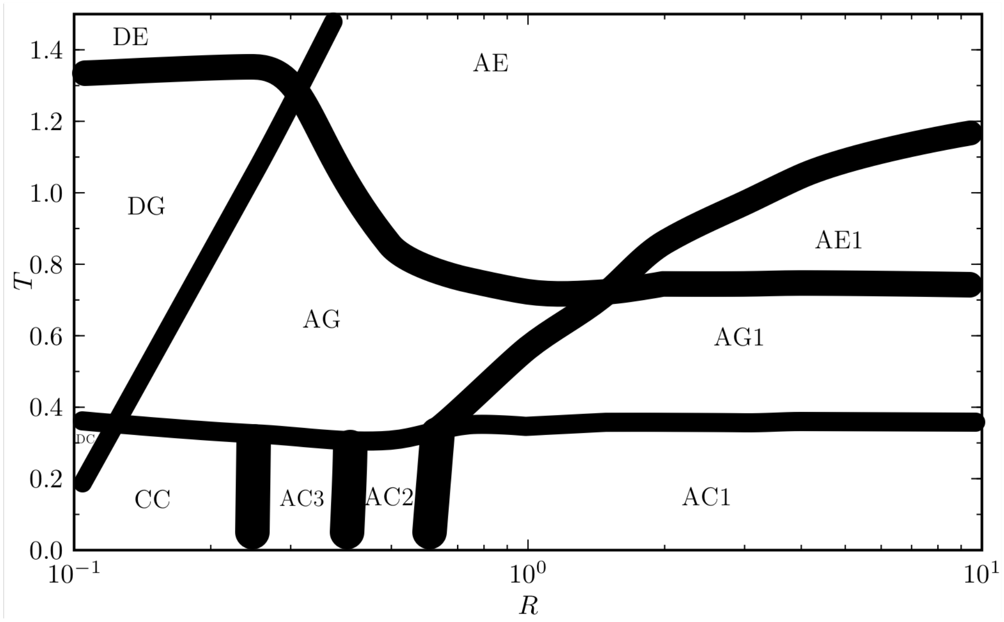


Figure 6.14: Pseudophase diagram of the 30-mer in the presence of a nanocylinder with $\epsilon_c = 5.0$.

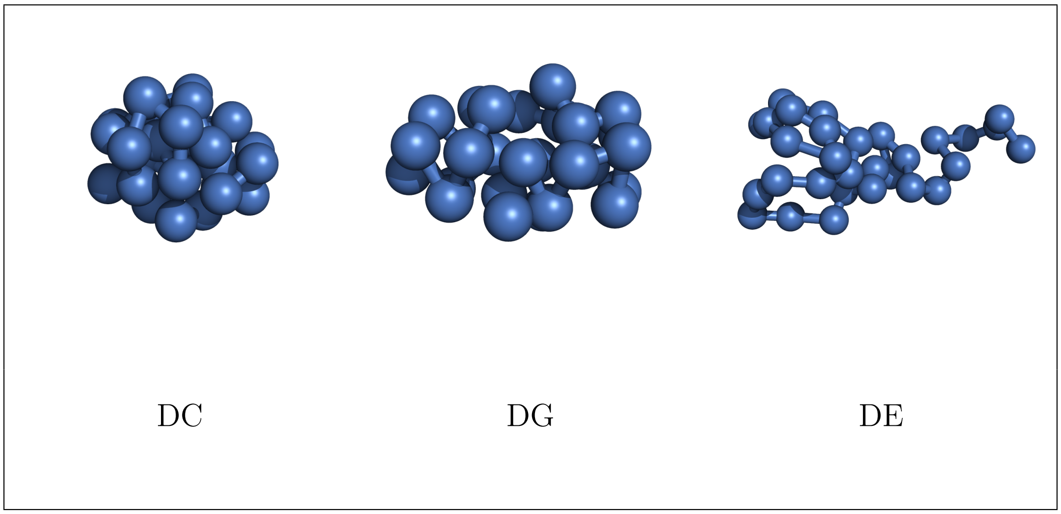


Table 6.1: Typical conformations of all desorbed pseudophases (DC, DG, DE) for the 30-mer.

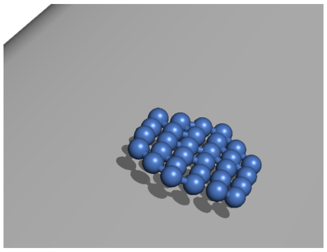
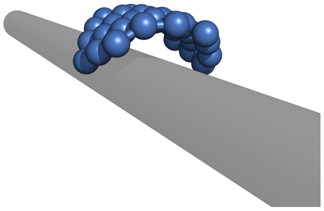
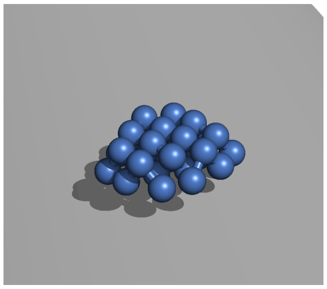
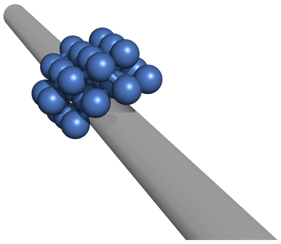
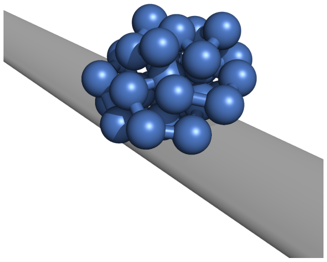
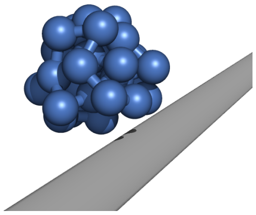
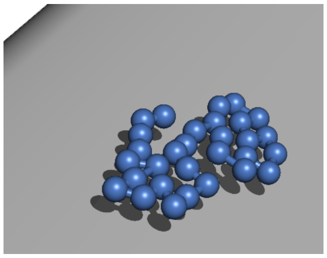
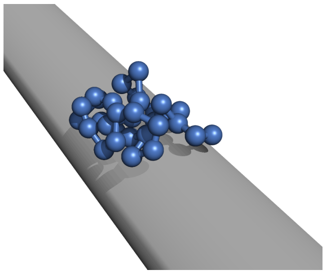
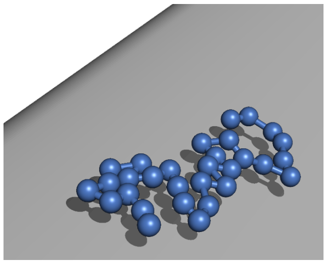
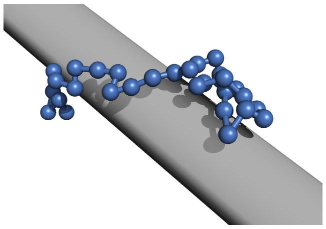
AC1 (large radius)			AC1 (small radius)
AC2 (large radius)			AC2 (small radius)
AC3			CC
AG1			AG
AE1			AE

Table 6.2: Characteristic conformations of all adsorbed pseudophases (AC1, AC2, AC3, CC, AG1, AG, AE1, AE) for the 30-mer as described in the text.

6.4 Conclusions

In this chapter we have investigated the thermodynamic behavior of a coarse-grained flexible homopolymer with 30 monomers in the presence of an attractive cylindrical surface. The monomer–surface attraction strength increases with the cylinder radius. In the limit of an infinite cylinder radius the surface is comparable to a planar surface. Parallel tempering simulations for 5 different nanocylinder materials with 10 different radii each at 80 temperatures enabled us to create pseudophase diagrams. By carefully analyzing various observables, like the specific heat and geometrical quantities, such as the radius of gyration, number of monomer–surface contacts and polymer extension perpendicular to the cylinder surface, we were able to locate and identify structural transitions in the system. We find desorbed polymer structures for cylinders with weak surface attraction or at high temperatures, that are similar to simulations of free polymers. Various absorbed pseudophases are identified as well. For very strong attraction we find adsorbed elongated polymers at high temperatures. Globular polymer droplets on the cylinder surface are found and also highly ordered crystalline polymer layers on the surface. Depending on the cylinder radius and the attraction strength we find one, two or three layers for the adsorbed crystals. Higher levels of layers are not observed because of the rather short length of the polymer. The structural phases found in our study are comparable to structures found for a similar polymer model [73] adsorbed at a planar surface.

Chapter 7

Summary and Outlook

The main interest of this thesis was the thermodynamic behavior of flexible polymers. The polymer is represented by a coarse-grained bead-spring model, where a group of atoms that forms a monomer is regarded as a single bead. Bonds in the polymer chain are represented by a flexible spring. Structural transitions of the polymer in an implicit dilute solution were studied using cutting-edge technology and sophisticated simulation techniques.

For free polymers in continuum, the efficiency of using modern graphics processing units (GPUs) for speeding up Monte Carlo simulations was found to be impressive. Previous findings regarding the thermodynamic behavior and structure formation of the investigated flexible polymer were confirmed. Employing a highly parallel implementation of replica-exchange Monte Carlo, yields about a factor of 130 of increase in efficiency, when compared to serial execution on a CPU. This allows us to gather much more statistics in a much shorter time. Even with main stream graphics card primarily designed for demanding computer games like the consumer cards of Nvidia's Geforce series, we are able to speed up our simulations by factors of around 100. The main vendors of graphics cards recognized the potential of this new application and started to develop and release special hardware for scientific applications based on the same design as their consumer cards. These professional cards offer higher memory bandwidth and faster double-precision performance. Since the performance per core seems to have reached a plateau for traditional CPU computing, the future of scientific computing will rely on hybrid architectures. Even the world largest supercomputers today are built using a heterogeneous design, using GPUs alongside traditional CPU clusters. With the knowledge gained in our investigations, we feel prepared for the challenges of this new generation of scientific computing.

Further we have studied the influence of the interaction range of the non-bonded monomers on the structural phases of a flexible polymer. We employed parallel tempering simulations to calculate the density of states for polymer, where we altered the interaction length of the Lennard-Jones potential effective between monomers. In addition to that we needed to apply multiple Gaussian modified ensembles to deal with the strong first-order-like behavior of the freezing transition for the polymer at low temperatures. The energy histograms of our parallel tempering simulations were reweighted and combined evaluate the microcanonical entropy and its first and second derivative. By analyzing the derivatives we were able to precisely locate and classify transitions of the system. A pseudophase diagram of the investigated 90-mer was constructed as a function of interaction length and temperature. The liquid phase that separates the extended coil or "gas"-like phase from the compact crystalline structures becomes smaller for shorter interaction ranges. It eventually vanishes for very small interaction length and we observe a direct transition from gas to solid. In the microcanonical view we actually see a crossover of collapse and freezing transition, which is a finite size effect and will disappear in the thermodynamic limit. The interaction range also has an influence on the crystal structure found at very low temperatures. The icosahedral structures in the solid phase are not sustained for shorter interaction length. We find evidence for the transition to fcc structures for very short interaction ranges via decahedral crystals at intermediate interaction lengths.

Finally we investigated the adsorption of a flexible homopolymer on nanocylinders.

Again, we applied the replica-exchange Monte Carlo method to single polymer chain interacting with cylinders of different materials and radii. In our model the effective surface attraction of the nanocylinder increases with the radius of the cylinder. For infinitely large radii, the potential of the cylinder matches that of a attractive planar wall. The thermodynamic information of several structural observables was gathered to construct pseudophase diagrams for 5 different cylinder materials as a function of the cylinder radius and temperature. Aside from the already known desorbed phases, that are similar to our previous studies of free polymers, we find various adsorbed polymer structures. For small radii and weak surface attraction at low temperatures, the already compact polymer crystal is adsorbed on the nanocylinder. With increasing radius, the surface attracts the monomers more strongly and we observe a flattening of the polymer on the surface. Due to the relatively short chain length of only 30 monomers, in our study, we see three, two or one layer of adsorbed compact polymers. For very strong surface attraction we find evidence for single-layered structures in the globular and even the coil phase, at higher temperatures, as well. An interesting continuation of this study would investigate longer chains. Higher layers of adsorbed crystals are to be expected as well as complete wrapping of the nanocylinder for small radii and strong surface attraction. Also of interest would be the polymer interaction with a more realistic representation of a nanotube, modelling the tube on coarse-grained particle based level instead of a homogeneous body.

Bibliography

[1] P.-G. de Gennes, *Scaling Concepts in Polymer Physics* (Cornell, Ithace, 1979).

[2] D. P. Landau and K. Binder, *A Guide to Monte Carlo Simulations in Statistical Physics*, 3rd ed. (Cambridge, New York, 2009).

[3] C. Junghans, M. Bachmann, and W. Janke, Phys. Rev. Lett. **87**, 218103 (2006).

[4] C. Junghans, W. Janke, and M. Bachmann, Comp. Phys. Commun. **182**, 1937 (2011).

[5] J. P. K. Doye, D. J. Wales, and R. S. Berry, J. Chem. Phys. **108**, 4234 (1995).

[6] J. P. K. Doye and D. J. Wales, J. Phys. B: At. Mol. Opt. Phys. **29**, 4859 (1996).

[7] L. Cheng and J. Yang, J. Phys. Chem. A **111**, 5287 (2007).

[8] T. Vogel, M. Bachmann, and W. Janke, Phys. Rev. E **76**, 061803 (2007).

[9] S. Schnabel, T. Vogel, M. Bachmann, and W. Janke, Chem. Phys. Lett. **476**, 201 (2009).

[10] S. Schnabel, M. Bachmann, and W. Janke, J. Chem. Phys. **131**, 124904 (2009).

[11] D. Seaton, T. Wüst, and D. P. Landau, Comp. Phys. Commun. **180**, 587 (2009).

[12] D. Seaton, T. Wüst, and D. P. Landau, Phys. Rev. E **81**, 011802 (2010).

[13] S. Schnabel, D. T. Seaton, D. P. Landau, and M. Bachmann, Phys. Rev. E **84**, 011127 (2011).

[14] T. Vogel, T. Neuhaus, M. Bachmann, and W. Janke, Europhys. Lett. **85**, 10003 (2009).

[15] T. Vogel, T. Neuhaus, M. Bachmann, and W. Janke, Phys. Rev. E **80**, 011802 (2009).

[16] D. H. E. Gross, *Microcanonical Thermodynamics* (World Scientific, Singapore, 2001).

[17] K. Goede, M. Bachmann, W. Janke, and M. Grundmann, in *Proceedings of the 28th Conference on the Physics of Semiconductors*, AIP Conference Proceedings, Vol. 893, edited by W. Jantsch and F. Schäffler (AIP, New York, 2007) p. 611.

[18] M. Bachmann, K. Goede, A. Beck-Sickinger, M. Grundmann, A. Irbäck, and W. Janke, Ang. Chem. Int. Ed. **49**, 9530 (2010).

[19] F. S. Gittleson, J. Hwang, R. C. Sekol, and A. D. Taylor, J. Mater. Chem. A **1**, 7979 (2013).

[20] M. Gao, L. Dai, and G. G. Wallace, Electroanalysis **15**, 1089 (2003).

[21] T. Hasan, Z. Sun, F. Wang, F. Bonaccorso, P. H. Tan, A. G. Rozhin, and A. C. Ferrari, Adv. Mater. **21**, 3874 (2009).

[22] J. Gross, W. Janke, and M. Bachmann, Comp. Phys. Commun. **182**, 1638 (2011).

[23] J. Gross, W. Janke, and M. Bachmann, Phys. Proc. **15**, 29 (2011).

[24] J. Gross, T. Neuhaus, T. Vogel, and M. Bachmann, J. Chem. Phys. **138**, 074905 (2013).

[25] J. Gross, T. Neuhaus, T. Vogel, and M. Bachmann, Phys. Proc. , in press (2013).

[26] M. Bachmann, *Thermodynamics and Statistical Mechanics of Macromolecular Systems* (Cambridge University Press, to be published Spring 2014).

[27] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, J. Chem. Phys. **21**, 1087 (1953).

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

[29] C. J. Geyer, in *Computing Science and Statistics: Proceedings of the 23rd Symposium on the Interface*, edited by E. M. Keramidas (Interface Foundation, Fairfax, Virginia, 1991) pp. 156–163.

[30] K. Hukushima and K. Nemoto, J. Phys. Soc. Jpn. **65**, 1604 (1996).

[31] E. Marinari and G. Parisi, Europhys. Lett. **19**, 451 (1992).

[32] A. P. Lyubartsev, A. A. Martsinovski, S. V. Shevkunov, and P. N. Vorontsov-Velyaminov, J. Chem. Phys. **96**, 1776 (1992).

[33] B. A. Berg and T. Neuhaus, Phys. Lett. B **267**, 249 (1991).

[34] B. A. Berg and T. Neuhaus, Phys. Rev. Lett. **68**, 9 (1992).

[35] W. Janke, Physica A **254**, 164 (1998).

[36] B. A. Berg, *Markov Chain Monte Carlo Simulations* (World Scientific, Singapore, 2004).

[37] F. Wang and D. P. Landau, Phys. Rev. Lett. **86**, 2050 (2001).

[38] F. Wang and D. P. Landau, Phys. Rev. E **64**, 056101 (2001).

[39] G. Marsaglia and A. Zaman, Stat. Probabil. Lett. **9**, 35 (1990).

[40] M. Möddel, M. Bachmann, and W. Janke, Macromolecules **44**, 9013 (2011).

[41] M. K. Fenwick, J. Chem. Phys. **129**, 125106 (2008).

[42] E. B. Ford, New Astronomy **14**, 406 (2009).

[43] C. Harris, K. Haines, and L. Staveley-Smith, Experimental Astronomy **22**, 129 (2008).

[44] C. Men, X. Gu, A. Majumdar, Z. Zheng, K. Mueller, and S. B. Jiang, Phys. Med. Biol. **54**, 6565 (2009).

[45] X. Gu, D. Choi, C. Men, H. Pan, A. Majumdar, and S. B. Jiang, Phys. Med. Biol. **54**, 6287 (2009).

[46] T. Preis, W. Paul, and J. J. Schneider, Europhys. Lett. **82**, 68005 (2008).

[47] J. A. van Meel, A. Arnold, D. Frenkel, S. F. P. Zwart, and R. G. Belleman, Molecular Simulation **34**, 259 (2008).

[48] M. S. Friedrichs, P. Eastman, V. Vaidyanathan, M. Houston, S. Legrand, A. L. Beberg, D. L. Ensign, C. M. Burns, and V. S. Pande, J. Comp. Chem. **30**, 864 (2009).

[49] J. Yin and D. P. Landau, Phys. Rev. E **80**, 051117 (2009).

[50] M. Weigel, Comp. Phys. Commun. **182**, 1833 (2011).

[51] T. Preis, P. Virnau, W. Paul, and J. J. Schneider, J. Comp. Phys. **228**, 4468 (2009).

[52] J. S. Meredith, G. Alvarez, T. A. Maier, T. C. Schulthess, and J. S. Vetter, Parallel Computing **35**, 151 (2009).

[53] S. Cook, *CUDA Programming - A Developer's Guide to Parallel Computing with GPUs* (Morgan Kaufmann, New York, 2013).

[54] NVIDIA, *Whitepaper - NVIDIA's next generation CUDA compute architecture: Fermi* (http://www.nvidia.com/content/PDF/fermi_white_papers/NVIDIA_Fermi_Compute_Architecture_Whitepaper.pdf, 2009).

[55] NVIDIA, *CUDA Programming Guide 3.0* (http://docs.nvidia.com/cuda/index.html, 2010).

[56] B. R. Gaster, L. Howes, D. R. Kaeli, P. Mistry, and D. Schaa, *Heterogeneous Computing with OpenCL* (Morgan Kaufmann, New York, 2012).

[57] M. Harris, *Whitepaper - NVIDIA optimizing parallel reduction in CUDA* (http://developer.download.nvidia.com/compute/cuda/1.1-Beta/x86_website/projects/reduction/doc/reduction.pdf, 2009).

[58] F. Rampf, W. Paul, and K. Binder, Europhys. Lett. **70**, 628 (2005).

[59] W. Paul, T. Strauch, F. Rampf, and K. Binder, Phys. Rev. E **75**, 060801(R) (2007).

[60] M. P. Taylor, W. Paul, and K. Binder, J. Chem. Phys. **131**, 114907 (2009).

[61] M. P. Taylor, W. Paul, and K. Binder, Phys. Rev. E **79**, 050801(R) (2009).

[62] G. A. Vliegenthart, J. F. M. Lodge, and H. N. W. Lekkerkerker, Physica A **263**, 378 (1999).

[63] R. B. Bird, C. F. Curtis, R. C. Armstrong, and O. Hassager, *Dynamics of Polymeric Liquids*, 2nd ed. (Wiley, New York, 1987).

[64] T. Neuhaus and J. S. Hager, Phys. Rev. E **74**, 036702 (2006).

[65] M. S. S. Challa and J. H. Hetherington, Phys. Rev. Lett. **60**, 77 (1988).

[66] M. S. S. Challa and J. H. Hetherington, Phys. Rev. A **38**, 6324 (1988).

[67] W. Janke, Nucl. Phys. B. **63A-C**, 631 (1998).

[68] E. Eisenriegler, K. Kremer, and K. Binder, J. Chem. Phys. **77**, 6296 (1982).

[69] A. Milchev and K. Binder, J. Chem. Phys. **114**, 8610 (2001).

[70] J. Krawczyk, A. L. Owczarek, T. Prellberg, and A. Rechnitzer, Europhys. Lett. **70**, 726 (2005).

[71] M. Bachmann and W. Janke, Phys. Rev. Lett. **95**, 058102 (2005).

[72] M. Bachmann and W. Janke, Phys. Rev. E **73**, 041802 (2006).

[73] M. Möddel, M. Bachmann, and W. Janke, J. Phys. Chem. B **113**, 3314 (2009).

[74] A. Milchev and K. Binder, J. Chem. Phys. **117**, 6852 (2002).

[75] I. Gurevitch and S. Srebnik, Chem. Phys. Lett. **444**, 96 (2007).

[76] I. Gurevitch and S. Srebnik, J. Chem. Phys. **128**, 144901 (2008).

[77] S. Srebnik, J. Polym. Sci., Part B: Polym. Phys. **46**, 2711 (2008).

[78] T. Vogel and M. Bachmann, Phys. Rev. Lett. **104**, 198302 (2010).

[79] M. Möddel, W. Janke, and M. Bachmann, Phys. Chem. Chem. Phys. **12**, 11548 (2010).

[80] W. Janke, in *Proceedings of the Euro Winter School Quantum Simulations of Complex Many-Body Systems: From Theory to Algorithms*, NIC Series, Vol. 10, edited by J. Grotendorst, D. Marx, and A. Muramatsu (John von Neumann Institute for Computing, Jülich, 2002) p. 423.

[81] D. P. Landau, Phys. Rev. B **13**, 2997 (1976).

[82] W. W. Wood, *Physics of Simple Liquids* (Wiley, Ney York, 1968).