Diplomarbeit

HP-Proteine auf verallgemeinerten Gittern und Homopolymerkollaps

Thomas Vogel

Januar 2004

Betreuer: Prof. Dr. Wolfhard Janke
Dr. Michael Bachmann

Gutachter: Prof. Dr. Wolfhard Janke
Prof. Dr. Ulrich Behn

Institut für Theoretische Physik
Fakultät für Physik und Geowissenschaften
Universität Leipzig

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Einleitung

Watching a Protein Fold

What did the researchers learn from viewing a full simulated microsecond of protein folding? A burst of folding in the first 20 nanoseconds quickly collapses the unfolded structure, suggesting that initiation of folding for a small protein can occur within the first 100 nanoseconds. Over the first 200 nanoseconds, the protein moves back and forth between compact states and more unfolded forms. „If you look at those curves," notes Kollman, „they're very noisy – the structure is moving, wiggling and jiggling a lot." [1]

Proteine gehören zu den wichtigsten biologischen Makromolekülen in Lebewesen. Sie steuern die fundamentale Prozesse in Zellen, speichern genetische Informationen, helfen beim Transport von anderen Molekülen und bilden selbst Gewebe.

Was liegt also näher, als diese vielseitigen Makromoleküle selbst zu untersuchen? Sehr viel wissen wir bereits über Proteine: Aus welchen Komponenten sie bestehen, wie sie gebildet werden, wie sie aussehen, welche Funktionen sie ausfüllen. Ein großes Rätsel bleibt aber weiterhin erhalten: Wie gelangen Proteine zu ihrer Form?

Natürlich wird diese Arbeit diese Frage nicht beantworten, sie beschäftigt sich nicht einmal mit realistischen Proteinen oder der Dynamik der Faltung. Wenn wir wissen wollen, wie sich Proteine falten, und das müssen sie, denn sie entstehen als lineare Kette von Aminosäureresten und erfüllen ihre Funktion durch ihre dreidimensionale Form, müssen wir einfach hinsehen. Nicht zuletzt durch die Experimente von Anfinsen [2] liegt die Vermutung nahe, daß ausschließlich physikalische Wechselwirkungen innerhalb des Proteins sowie zwischen dem Protein und seiner Umgebung den Faltungsprozeß bestimmen. Die experimentelle Analyse dieses Prozesses, das „Hinsehen" also, ist jedoch außerordentlich schwierig und aufwendig.

Wir können daher versuchen, ein Modell zu konstruieren, welches wir dann, fern jeglicher realer Beschränkungen, untersuchen können. Diese Arbeit beschäftigt sich mit einem ein-

fachsten Modell von Proteinen. Das hat zwei Gründe: Erstens sind realistische Modelle von Proteinen so kompliziert (siehe Kap. 1.2), daß die Rechenleistung unserer Computer noch lange nicht ausreichen wird, um mit ihnen gute Ergebnisse für reale Proteine zu erhalten und zweitens wissen wir nicht einmal genau, welches die fundamentalen Wechselwirkungen sind, die tatsächlich eine Rolle spielen. Ein einfachstes Modell wird zwar i.a. keine Aussagen über das Verhalten realer Proteine machen können, aber es ist der erste Schritt zu komplexeren Modellen.

In Kapitel 1 wird eine kurze Einführung in die Biochemie realer Proteine gegeben. Es werden alle wesentlichen Eigenschaften von Proteinen angesprochen, so daß der Leser selbst entscheiden kann, wie stark das ebenso in diesem Kapitel eingeführte Modell tatsächlich von der Realität abstrahiert ist. Es wird ebenso kurz auf realistische Wechselwirkungsmodelle eingegangen.

Das einfachste Modell, welches ich in dieser Arbeit untersuche, ist ein Gittermodell. In Kapitel 2 werden alle Gitter, die ich in der Arbeit benuzt habe, vorgestellt. In Kapitel 3 wird dann das Modell selbst näher untersucht. Polymere (und somit auch Proteine) werden auf dem Gitter dargestellt als *interacting self-avoiding walks*, also nicht selbstüberschneidende Wege mit einer Wechselwirkung zwischen den Punkten des Weges. In diesem Kapitel wird z.B. untersucht, wieviele solcher möglichen Wege es auf den verschiedenen Gittern überhaupt gibt.

Für meine Simulationen benutze ich einen Kettenwachstumsalgorithmus mit Populationskontrolle, d.h. ich lasse ein Protein sequentiell wachsen und kann zu jeder Zeit entscheiden, ob ich es vervielfältigen möchte, weil es mir „gut“ erscheint oder ob ich versuche, es aussterben zu lassen. Solche Strategien heißen „*Go with the winners*“-Strategien. Diese Strategie i.a. und meinen Algorithmus im speziellen stelle ich in Kapitel 4 vor.

Das folgende Kapitel 5 zeigt Eigenschaften ausgewählter Modellproteine sowie meine Ergebnisse aus Simulationen. Es werden die Proteine vorgestellt, die teilweise Modelle realer Proteine sind, andererseits aber auch nur als Modellproteine interessant sind, weil sie etwa besondere Strukturen mit sehr niedriger Energie haben. Ich werde, soweit vorhanden, alle Ergebnisse mit bereits bekannten Ergebnissen vergleichen.

Ein sehr interessantes Thema liegt etwas abseits der rein biologisch motivierten Systeme. In Kapitel 6 werde ich einen Abstecher zu Homopolymeren machen. Homopolymere kollabieren bei einer bestimmten Temperatur. Dieser Kollaps zeigt in 4 Dimensionen starke Anzeichen eines 1. Ordnung Phasenübergangs, obwohl es kein echter Phasenübergang sein kann und im thermodynamischen Limes ein Übergang 2. Ordnung zu erwarten ist. Dieses bekannte Verhalten konnte ich mit meinen Simulationen bestätigen und beschreibe die geometrischen und energetischen Eigenschaften der Proteine in den jeweiligen Phasen.

Die wichtigsten Ergebnisse und die sich aus diesen Untersuchungen ergebenen noch offenen Fragestellungen sind im abschließenden Paragraphen kurz zusammengefaßt.

Kapitel 1

Reale Proteine und HP Proteine

1.1 Biochemie realer Proteine

Proteine sind die am häufigsten vorkommenden Makromoleküle in lebenden Zellen und deren Bestandteilen. Ihr Name leitet sich ab vom griechischen *protos*, was etwa mit „Erster“ oder „Wichtigster“ übersetzt werden kann. Proteine treten in großer Vielfalt auf, in einer einzigen Zelle können sich über tausend verschiedene Klassen von Proteinen befinden. Sie erfüllen die verschiedensten biologischen Funktionen wie z.B. als Katalysator, zum Stofftransport oder als mechanische Stützen. Diese Funktionsvielfalt läßt sich über die Struktur der Proteine auf die chemischen Eigenschaften ihrer Bestandteile zurückführen.

Alle Proteine sind Ketten aus der gleichen Menge von 20 verschiedenen Aminosäuren. Jede von ihnen ist mit ihrem Nachbar kovalent verbunden. Abgesehen von der Funktionsvielfalt von Proteinen sind die verschiedensten biologischen Produkte aus Proteinen aufgebaut: Enzyme, Hormone, Antikörper, Federn, Spinnenfäden, Pilzgifte, das Horn des Rhinozeros.

Die Struktur realer Proteinen soll, in Form einer kurzen Übersicht, das Thema dieses einführenden Kapitels sein, um dem Leser später eine Idee dessen zu geben, was das eigentliche Thema dieser Arbeit, HP Proteine, versucht zu modellieren und wie weit entfernt bzw. wie nah das Modell der realen Welt ist.

Hierbei werden zuerst die Aminosäuren kurz beschrieben und die kovalenten Bindungen, die diese zu Peptiden und Proteinen verbinden, später die Proteine an sich und deren Struktur und Faltung.

1.1.1 Aminosäuren und Peptide

Die erste Aminosäure, die entdeckt wurde, war das Asparagin, im Jahr 1806, die letzte wurde erst 1938 bekannt, das Treonin. Alle Aminosäuren (siehe Tab. 1.1) haben vereinfachte Namen, welche in einigen Fällen sogar auf ihren (historischen) Ursprung bzw. Eigenschaften hindeuten. So wurde Asparagin z.B. zuerst in Spargel gefunden, Tyrosin zum ersten Mal in

Aminosäure	Abkürzung	Symbol	hydropatischer Index	Vorkommen in Proteinen (%)
apolare, aliphatische R-Gruppe				
Glycin	Gly	G	−0.4	7.5
Alanin	Ala	A	1.8	9.0
Valin	Val	V	−4.2	6.9
Leucin	Leu	L	3.8	7.5
Isoleucin	Ile	I	−4.5	4.6
Prolin	Pro	P	−1.6	4.6
aromatische R-Gruppe				
Phenylalanin	Phe	F	2.8	3.5
Tyrosin	Tyr	Y	1.3	3.5
Tryptophan	Trp	W	−0.9	1.1
neutrale, polare R-Gruppe				
Serin	Ser	S	−0.8	7.1
Threonin	Thr	T	−0.7	6.0
Cystein	Cys	C	2.5	2.8
Methionine	Met	M	1.9	1.7
Asparagin	Asn	N	−3.5	4.4
Glutamin	Gln	Q	−3.5	3.9
negativ geladene R-Gruppe				
Aspartat	Asp	D	−3.5	5.5
Glutamat	Glu	E	−3.5	6.2
positiv geladene R-Gruppe				
Lysin	Lys	K	−3.9	7.0
Arginin	Arg	R	−4.5	4.7
Histidin	His	H	−3.2	2.1

Tabelle 1.1: Alle 20 Standardaminosäuren mit den zugehörigen Buchstabenabkürzungen und einigen Eigenschaften. Der hydropatische Index ist eine Skala, die die Hydrophobizität und die Hydrophilität vereint. Negative Werte entsprechen dem hydrophilen Bereich, positive dem hydrophoben [3, 4]. Das prozentuale Vorkommen wurde aus dem Vorkommen der Aminosäuren in über 200 ausgewählten Proteinen bestimmt [3, 5].

Käse (*tyros* griechisch für „Käse“), Glycin hat einen süßlichen Geschmack (*glykos* griechisch für „süß“).

Die 20 Aminosäuren, aus denen die Proteine aufgebaut sind (natürlich gibt es noch sehr viele weitere), heißen „Standard-“, „Primär-“ oder „normale“ Aminosäuren. Sie haben einheitliche 3-Buchstaben Abkürzungen und Buchstabensymbole (siehe Tab. 1.1).

Gemeinsame charakteristische Eigenschaften Alle Aminosäuren bestehen aus einem $\alpha-$Kohlenstoffatom (C), an welches ein Wasserstoffatom (H), eine Carboxylgruppe (COOH), eine Aminogruppe (NH_2) und eine Seitenkette (R-Gruppe) binden (siehe Abb. 1.1). Genau in der Seitenkette unterscheiden sich alle Aminosäuren. Sie haben unterschiedliche Größen

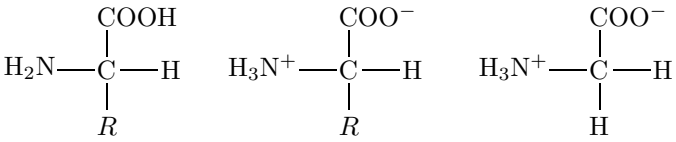


Abbildung 1.1: **Links** Allgemeine Struktur jeder Aminosäure, nicht ionisiert bzw. (**Mitte**) ionisiert, wie sie in Wasser vorkommt. **Rechts** Die „einfachste“ aller Aminosäuren, das Glycin. Hier ist „$R = \mathrm{H}$“.

und Strukturen, sie unterscheiden sich in ihrer elektrischen Ladung und beeinflussen die Wasserlöslichkeit.

Die einfachste Seitenkette ist das einzelne Wasserstoffatom (H). Die entsprechende Aminosäure ist das Glycin (siehe Abb. 1.1). Besitzt die Seitenkette weitere Kohlenstoffatome, werden diese der Reihenfolge nach mit $\beta-, \gamma-, \delta-$ usw. Kohlenstoff benannt.

In allen Aminosäuren (außer dem Glycin) ist das $\alpha-$Kohlenstoffatom von 4 verschiedenen Gruppen umgeben, sie sind asymmetrisch. Aminosäuren sind optisch aktiv, das $\alpha-$Kohlenstoffatom ist ein chirales Zentrum. Jede Aminosäure kann sich in 2 Zuständen befinden, dem linksdrehenden (L-) bzw. rechtsdrehenden (D-) optischen Isomer (Enanthiomer) (siehe Abb. 1.2).

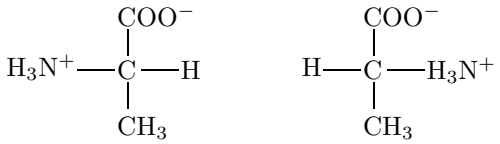


Abbildung 1.2: Die 2 optischen Isomere von Alanin. **Links** Das L-Alanin, **rechts** das D-Alanin.

Proteine bestehen aus L-Aminosäuren Damit Proteine eine eindeutige dreidimensionale charakteristische Struktur (siehe Abschnitt 1.1.3) annehmen können, die ihre biologische Funktion bestimmt, ist es notwendig, daß alle Proteine aus denselben optischen Isomeren aufgebaut sind. Die Natur hat sich hierbei für die L-Aminosäuren entschieden, alle bekannten Proteine sind ausschließlich aus L-Aminosäuren aufgebaut.

Klassifikation der Aminosäuren, Polarität Die chemischen Eigenschaften der Aminosäuren sind bestimmt durch die Eigenschaften der Seitenketten. Eine der wichtigsten Eigenschaften der Aminosäuren ist ihre Polarität[1], d.h. die Tendenz, mit Wasser zu wechselwirken. Die Polarität der verschiedenen Seitenketten variiert stark von total apolar bzw. hydrophob (wasserunlöslich) bis stark polar bzw. hydrophil (wasserlöslich).

Eine Klassifikation der Aminosäuren bezüglich ihrer Eigenschaften kann wie folgt aussehen (und wurde schon in Tab. 1.1 angedeutet). Aminosäuren mit apolarer und aliphatischer

[1]Diese und nur diese Eigenschaft wird später im Modell verwendet, von allen anderen wird abstrahiert.

R-Gruppe[2], aromatischer R-Gruppe[3] (hauptsächlich apolar), polarer neutraler R-Gruppe und mit positiv bzw. negativ geladener R-Gruppe. Die Strukturformeln aller 20 Standardaminosäuren findet man z.B. in [3].

Peptide Aminosäuren können sich kovalent durch sogenannte Peptidbindungen zu Aminosäureketten (Peptiden) zusammenfügen. Bei der Bindung werden der Carboxylgruppe der einen und der Aminogruppe der anderen Aminosäure die Elemente des Wassers entzogen, wonach diese dann aneinander binden (siehe Abb. 1.3)

Warum diese Bindung zustande kommt, ist schwerer zu verstehen. Das thermodynamische Gleichgewicht favorisiert die Ausgangsstoffe, nicht die Reaktionsprodukte, deswegen muß die Carboxylgruppe erst angeregt werden, um ihre Hydroxylgruppe leichter abgeben zu können. Warum und wie das passiert, soll hier nicht besprochen werden [3].

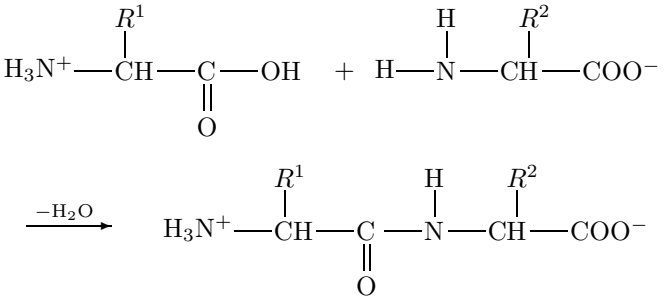


Abbildung 1.3: Bildung einer Peptidbindung zwischen zwei Aminosäuren unter Abgabe von Wasser.

Nomenklatur Verbinden sich 2 Aminosäuren, erhält man ein Dipeptid, analog ergeben sich Tri-, Tetra- und Pentapeptide. Vereinigen sich mehrere Aminosäuren zu einer Kette, erhält man ein Oligopeptid, bei sehr vielen Aminosäuren ein Polypeptid.

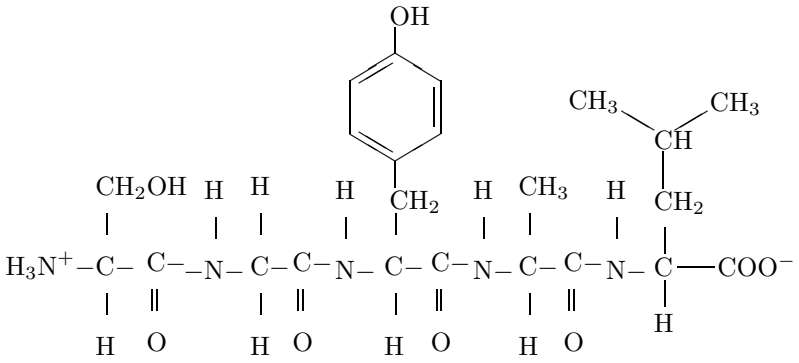


Abbildung 1.4: Strukturformel des Pentapeptids serilglyciltyrosinilalanilleucin. Gemäß den Abkürzungen aus Tab. 1.1 heißt es auch Ser-Gly-Tyr-Ala-Leu bzw. SGYAL. Hervorstechend sind die aromatische Seitenkette des Tyrosin (*sixring*) und die hydrophobe Seitenkette des Leucin (ganz rechts).

[2] *aliphatisch* bezeichnet Kohlenstoffverbindungen mit kettenförmiger Anordnung der C-Atome.
[3] *aromatisch* bezeichnet vom Benzol als Grundkörper abgeleitete zyklische Verbindungen.

1.1.2 Proteine

Proteine sind Polypeptide. Wie schon erwähnt, sind Proteine die wichtigsten Makromoleküle in Zellen mit den verschiedensten Funktionen:

- Enzyme: hochspezialisierte Katalysatoren für chemische Reaktionen
- Transportproteine: z.B. Hämoglobin zum Sauerstofftransport
- Proteine zur Ernährung und Reserve: in Pflanzensamen als Nahrung für den Keim
- Kontraktionsproteine: z.B. Actin und Myosin im Muskel
- Strukturproteine: bilden Stützgewebe oder Schutzhüllen
- Proteine zur Verteidigung: Antikörper im Immunsystem
- Regulatoren: z.B. Insulin regelt den Glukosehaushalt
- Exoten: z.B. das Protein, das das Blut einiger Fische der Antarktis vor dem Gefrieren schützt

Wie groß sind die Proteine und warum können sie derart unterschiedliche Funktionen erfüllen? Tabelle 1.2 zeigt die Anzahl der Aminosäurereste, die zu einigen Proteinen beitragen.

Protein	Anzahl der Residuen
Insulin	51
Cytochrome c	104
Ribonuclease A	124
Myoglobin	153
Hämoglobin	574
Immunoglobulin	$\sim 1\,320$
RNA Polymerase	$\sim 4\,100$
Glutamat deshydrogenase	$\sim 8\,300$

Tabelle 1.2: Anzahl der Aminosäurereste spezieller Proteine. Einige Proteine bestehen aus einer einzigen Peptidkette, andere aus mehreren. Das Hämoglobin z.B. besteht aus 4 Peptidketten, von denen jeweils 2 identisch sind und die untereinander nichtkovalent verbunden sind.

Primärstruktur und Funktion Proteine unterscheiden sich nicht nur in der Anzahl ihrer Residuen voneinander, sondern, viel wichtiger und eindeutig, in deren Sequenz. Diese Sequenz heißt Primärstruktur des Proteins. Jede einzelne Aminosäuresequenz faltet sich in eine einzigartige dreidimensionale Struktur, die die Funktion des Proteins bestimmt. Die Primärstruktur legt also eindeutig die Funktion des Proteins fest.

Ist die Primärstruktur gestört, bzw. wird geändert, kann das unterschiedliche Folgen haben. Einerseits sind z.B. über 1400 menschliche genetische Krankheiten bekannt, die sich in fehlerhaften Proteinen äußern, ein Drittel von ihnen ist aufgrund eines einzigen Austauschs

in der Aminosäuresequenz fehlerhaft. Andererseits kann eine Änderung der Primärstruktur trotzdem das „gleiche“ Protein erzeugen (d.h. es wird in seiner Funktion nicht beeinflußt). Solche Proteine heißen polimorph. Etwa 20–30% der menschlichen Proteine sind polimorph.

Proteine haben also oft Bereiche oder Untersequenzen in der Primärstruktur, die essentiell für die Funktion sind und solche, die diese nicht oder nur sehr schwach beeinflussen. Allerdings kann man diese Regionen nicht spezifizieren, sie variieren von Protein zu Protein.

Konkrete Formen, die Proteine annehmen können und den Prozeß der Faltung werden die folgenden Kapitel näher diskutieren.

1.1.3 Die dreidimensionale Struktur von Proteinen

Die räumliche Verteilung der einzelnen Atome eines Proteins heißt Konformation. Eine Konformation kann, z.B. durch eine Drehung um eine Bindung, in eine andere übergehen, ohne daß kovalente Bindungen gebrochen werden. Unter allen möglichen Konformationen gibt es eine mit der geringsten freien Energie, welche im allgemeinen auch die thermodynamisch stabilste ist. Proteine, die sich in dieser, ihrer funktionellen, Konformation befinden, heißen *nativ*.

Neben der schon erwähnten Primärstruktur der Proteine werden drei weitere Niveaus unterschieden: Sekundär-, Tertiär- und Quartärstruktur.

Sekundärstruktur Sekundärstruktur meint die reguläre räumliche Verteilung der Aminosäurereste einer Peptidkette. Hier gibt es einige wenige Grundformen, die wichtigsten sind die $\alpha-$ und die $\beta-$Konformation, auch $\alpha-$Helix und $\beta-$Faltblatt genannt. 1951 sagten Linus Pauling und Robert Corey die Existenz solcher Strukturen vorraus, bevor diese experimentell sichtbar gemacht wurden.

Peptidketten können sich nicht frei um jede beliebige Bindung drehen. Die Peptidbindung ist, bis auf ein Umklappen um 180°, fest. Alle zugehörigen Atome liegen in einer Ebene, der Winkel an der Peptidbindung heißt ω. Die Konformationen, die durch Umklappen der Peptidbindung entstehen heißen *cis*-Form ($\omega = 0\,^\circ$) bzw. *trans*-Form ($\omega = 180\,^\circ$) (siehe Abb. 1.5). Eine freie Rotation ist, im Peptidskelett, nur um die beiden anderen Bindungen (am $C^\alpha-$Atom) möglich. Die Rotationswinkel heißen Φ und Ψ (siehe Abb. 1.6). Ihr Wert ist nach Definition dann gleich Null, wenn die beiden angeschlossenen Peptidbindungsebenen in derselben Ebene liegen. Ebenso ist i.a. die Rotation der Seitenkette möglich sowie Rotationen innerhalb der Seitenkette. Die betreffenden Winkel heißen $\chi^{1,2,\cdots}$, wobei χ^1 der Rotationswinkel der ganzen Seitenkette ist (auch Abb. 1.6).

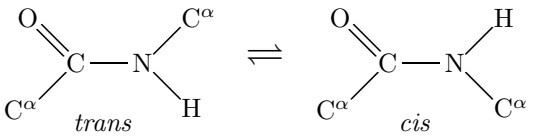


Abbildung 1.5: *trans*- und *cis*-Form (**Links** bzw. **Rechts**) der Peptidbindung. Die *trans*-Form ist energetisch favorisiert [3].

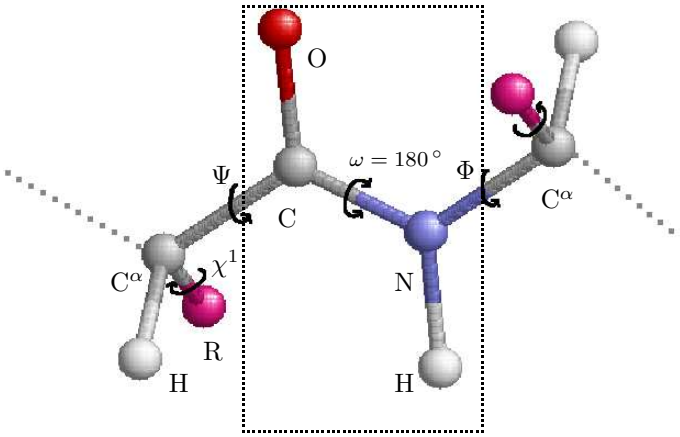


Abbildung 1.6: Planare Geometrie der Peptidbindung in *trans*-Form. Die Ebene, in der die Peptidbindung liegt ist angedeutet (**gestrichelte Box**). Alle Winkel, um die sich die Konformation drehen kann, sind angedeutet.

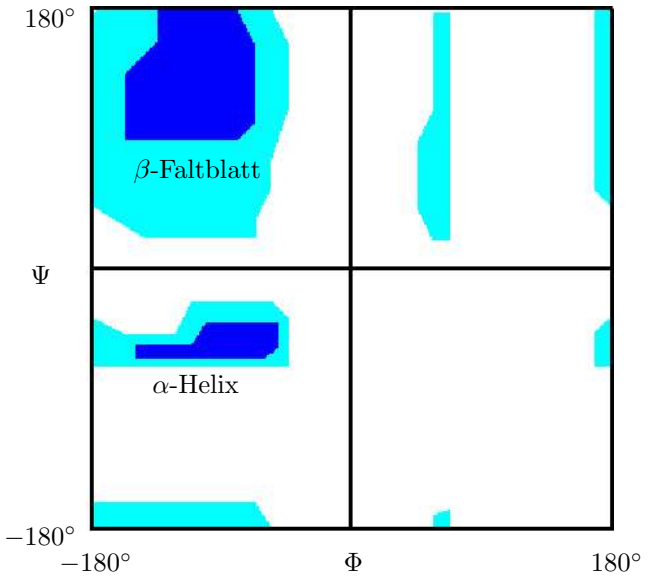


Abbildung 1.7: Ramachandranplot der theoretisch möglichen Winkelpaare (Ψ, Φ) in beliebigen Konformationen von Peptiden. Die dunklen Regionen stehen für Konformationen, die von allen Aminosäuren erreicht werden können, die hellen für alle außer dem Valin und dem Isoleucin. Die weiße Region ist verboten, bzw. wegen Überschneidung der harten Schalen (im Modell) nicht erreichbar. Trotzdem gibt es in einigen Proteinstrukturen Konformationen von Aminosäuren, die hier im weißen Bereich landen, diese sind aber relativ instabil [3].

Jede mögliche Sekundärstruktur kann komplett durch die beiden Winkel Φ und Ψ beschrieben werden. Die erlaubten Werte für Kombinationen der beiden Winkel zeigt Abb. 1.7.

Die einfachste Konformation, die eine Polypeptidkette annehmen kann, berücksichtigt man Wasserstoffbrückenbindungen zwischen den polaren Gruppen $-\mathrm{C}=\mathrm{O}$ und $-\mathrm{N}-\mathrm{H}$, ist eine Helixstruktur, da dann die optimale Nutzung möglicher Wasserstoffbrücken erreicht wird. Dabei ist das Polypeptidskelett spiralförmig angeordnet und von den R-Gruppen umgeben. Der Windungsabstand beträgt dabei $\approx 0.5\,\mathrm{nm}$, die Winkel Φ und Ψ haben Werte von $\Phi = -45°$ bis $-50°$ bzw. $\Psi = -60°$. Jede Windung der Helix beinhaltet im Mittel 3.6 Aminosäuren. In ihrer üblichen Form ist die $\alpha-$Helix rechtsdrehend, sehr selten linksdrehend [3].

Bei der $\beta-$Konformation liegen das Polypeptidskelett planar in einer Ebene in „zick-zack-Ketten“ nebeneinander, die Konformation heißt deswegen auch $\beta-$Faltblatt. Die gegenüberliegenden Ketten liegen etwa $\approx 0.6 - 0.7\,\mathrm{nm}$ auseinander und sind wieder durch

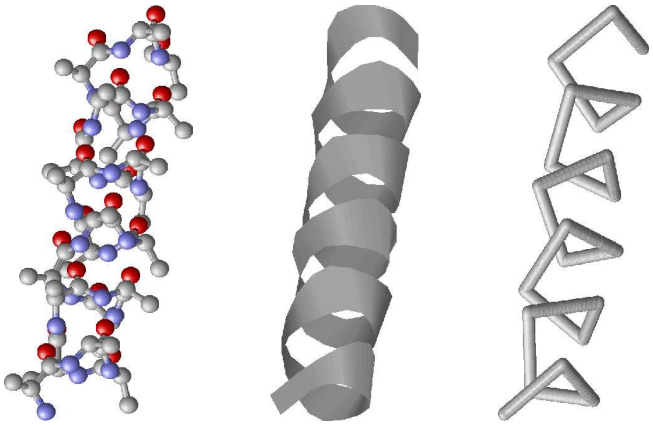


Abbildung 1.8: Struktur der α-Helix in verschiedenen Darstellungen: **Links** *balls and sticks* (ohne Wasserstoffatome), **Mitte** *ribbon* und **Rechts** *backbone*. Darstellungen erzeugt mit *rasmol* [6] (http://www.umass.edu/microbio/rasmol/), Daten von http://broccoli.mfn.ki.se/pps_course_96/ss_960723_6.html.

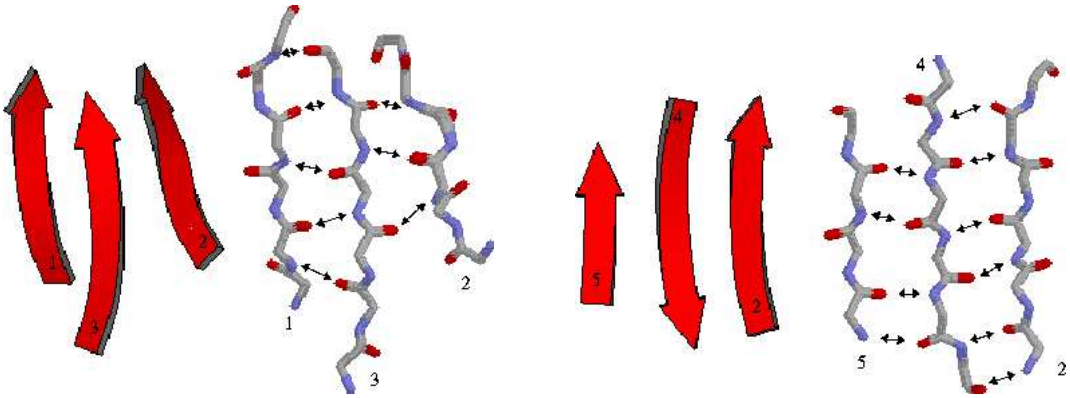


Abbildung 1.9: Parallele (**links**) bzw. antiparallele (**rechts**) β-Faltblattstruktur im Protein Thioredoxin. Die jeweils drei Stränge sind schematisch im *cartoon*- (jeweils links) und *stick*-Format (jeweils rechts, dargestellt sind die Skelettatome N, C^{α}, C und O) gezeigt. Wasserstoffbrückenbindungen zwischen (hier) N und O sind durch Pfeile angedeutet. Die Nummerierung der Stränge gibt deren relative Position in der Polypeptidsequenz an. Bilder und Daten von http://broccoli.mfn.ki.se/pps_course_96/ss _960723_6.html.

Wasserstoffbrücken miteinander verbunden. Die R-Gruppen ragen aus der Ebene in beide Richtungen heraus. Abbildung 1.8 zeigt eine α-Helix in verschiedenen Darstellungen, Abb. 1.9 zwei β-Faltblätter in paralleler bzw. antiparalleler Anordnung.

Nicht alle Aminosäuren finden sich gleich oft in den jeweiligen Konformationen. So befindet sich z.B. das Glutamat mit der größten relativen Wahrscheinlichkeit in einer $\alpha-$Helix, das Glycin mit der geringsten [3](siehe auch [7][4]). Solche Information kann man benutzen, um anhand der Aminosäuresequenz tendenziell Sekundärstrukturen von kurzen Ketten vorhersagen zu können.

Tertiärstruktur Die Tertiärstruktur bezeichnet die dreidimensionale Verteilung aller Atome eines Proteins. Während die Sekundärstruktur durch kurzreichweitige Wechselwirkung

[4]Dort allerdings geringfügig abweichende Angaben. Z.B. trifft man dort Prolin mit der geringsten Wahrscheinlichkeit in einer $\alpha-$Helix an, was der allgemeinen Aussage jedoch nicht widerspricht.

zwischen (nahen) Aminosäureresten entsteht, ist die Tertiärstruktur auch Resultat langreichweitiger Wechselwirkungen in der Aminosäuresequenz. Die meisten Tertiärstrukturen sind sehr komplex und nichtsymmetrisch.

In Abschnitt 1.1.1 wurde festgestellt, daß sich verschiedene Aminosäuren oder besser ihre Seitenketten verschieden gut in Wasser lösen. Das hat natürlich auch auf die Tertiärstruktur Einfluß. Viele Proteine falten sich in wässriger Lösung derart, daß die hydrophoben Seitenketten einen kompakten Kern bilden. Die polaren Seitenketten bilden dann eine äußere Schale, die den Kern so gut wie möglich abschirmt.

Ebenso wurde angedeutet, daß die Primärstruktur, also die Aminosäuresequenz, vollständig die Tertiärstruktur bestimmt. Das wichtigste Indiz für diese Behauptung sind Experimente, die zeigen, daß die Denaturierung einiger Proteine reversibel ist. Das klassische Experiment dazu wurde in den 50er Jahren von Anfinsen durchgeführt [2].

Anfinsen's Experiment Anfinsen benutzte für sein Experiment das Protein Ribonuclease, welches aus 124 Residuen besteht. Seine Tertiärstruktur wird hauptsächlich bestimmt durch 4 Disulfidbrücken zwischen 8 Cysteinresiduen und hydrophobe, nichtkovalente Wechselwirkungen. Gibt man jetzt 2 Substanzen zu, Substanz A, welche die nichtkovalenten Bindungen komplett zerstört und Substanz B, welche die Disulfidbrücken löst, denaturiert die Ribonuclease, d.h. sie nimmt eine willkürliche Gestalt an und verliert ihre chemischen Eigenschaften[5]. Entfernt man Substanz A und B wieder, beobachtet man einen spontanen Rückfall des Proteins in seinen vorherigen, nativen Zustand (Renaturierung). Diesen Prozeß veranschaulicht Abb. 1.10.

Das ist bemerkenswert, wenn man bedenkt, wie viele andere mögliche Zustände existieren. Entfernt man z.B. nur Substanz B, so daß sich die Disulfidbrücken wieder ausbilden können, mißt man eine relative katalytische Aktivität von $\approx 1\%$. Das entspricht sehr genau der einen Paarung, von 105 verschiedenen Möglichkeiten der 8 freien Cysteine sich zu paaren, die zu katalytischer Aktivität fähig ist.

Daraus und aus anderen Experimenten und Beobachtungen kann das generelle Prinzip, daß die Aminosäuresequenz die Konformation eindeutig bestimmt, abgeleitet werden.

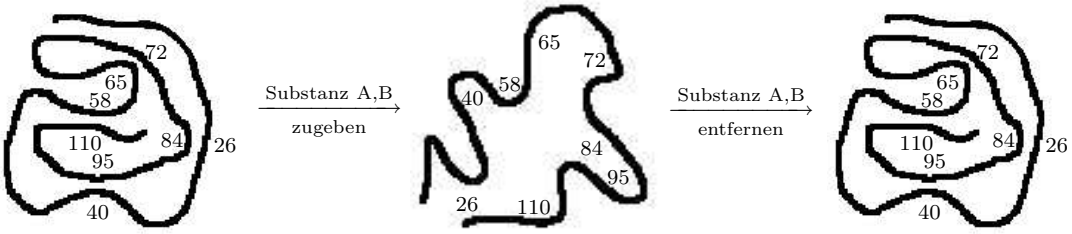


Abbildung 1.10: Denaturierung (**Links, Mitte**) und Renaturierung (**Mitte, Rechts**) von Ribonuklease im Experiment von Anfinsen. Substanz B löst die Disulfidbrücken zwischen Cysteinmolekülen. Diese sind durch die Nummern ihrer Positionen in der Aminosäuresequenz angedeutet.

[5]Die im Experiment konkret gemessene Eigenschaft ist die katalytische Aktivität.

Quartärstruktur Einige Proteine bestehen nicht nur aus einer einzigen Polypeptidkette, sondern aus mehreren, die identisch oder auch verschieden sein können. Das Hämoglobin besteht aus 4 Peptidketten (siehe auch Tab. 1.2), andere aus 12, es gibt Proteine, welche aus bis zu 102 Polypeptidketten bestehen. Die dreidimensionale Verteilung dieser verschiedenen *subunits* heißt Quartärstruktur.

Die Quartärstruktur wird wie auch die Tertiärstruktur von vielen nichtkovalenten Bindungen erhalten. Durch die Bindung von verschieden Polypeptidketten mit verschiedenen Funktionen können diese Proteine z.B. unterschiedliche, aber verwandte oder aufeinanderfolgende Aufgaben ausführen.

Die Abbildungen 1.11 und 1.12 zeigen die Proteine Insulin und Myoglobin. Man sieht sie in ihrer kompletten Darstellung, aufgelöst nach Sekundärstruktur und im Ramachandran-Plot. Beide bestehen aus mehr als einer Polypepdtidkette, was man aufgrund der verwundenen Quartärstruktur aber schwer erkennt.

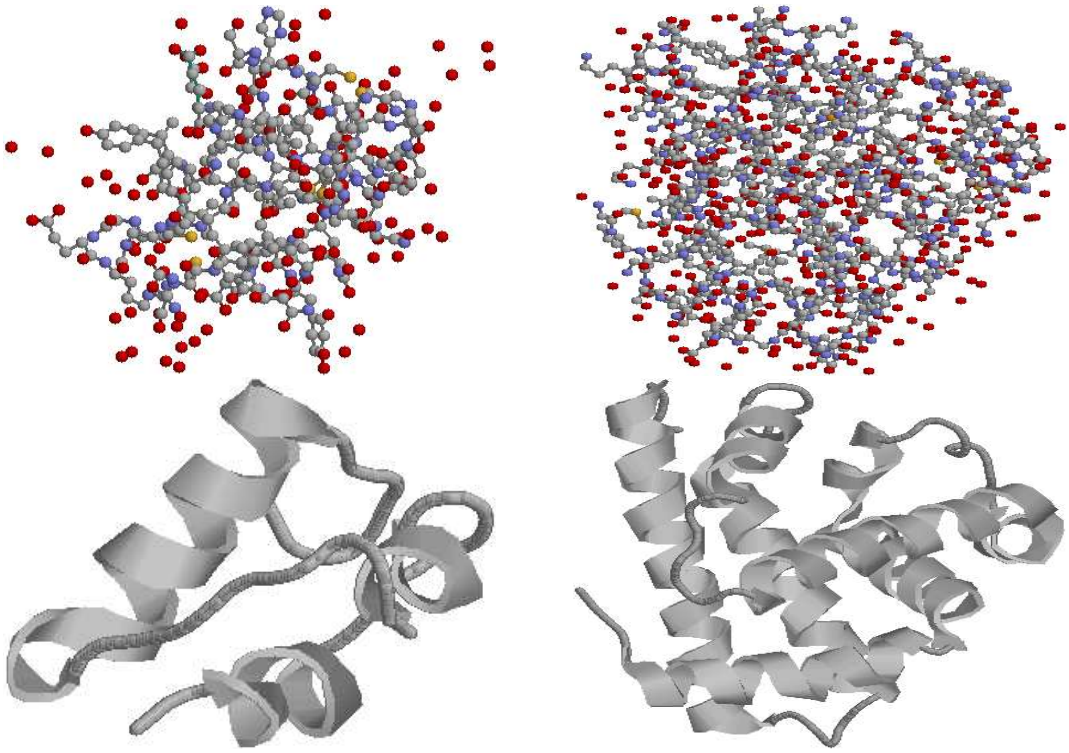


Abbildung 1.11: Die beiden Proteine Insulin (**links**) und Myoglobin (**rechts**). Die Bilder sind von *rasmol* [6] erzeugte Darstellungen (*balls and sticks* bzw. *cartoon*) der .pdb-Originaldateien von der Proteindatenbank (http://www.rcsb.org/pdb/), PDB ID 1B2F und PDB ID 101M. Man erkennt sehr gut die Tertiärstruktur (globular) und die vorherrschende Sekundärstruktur ($\alpha-$Helix).

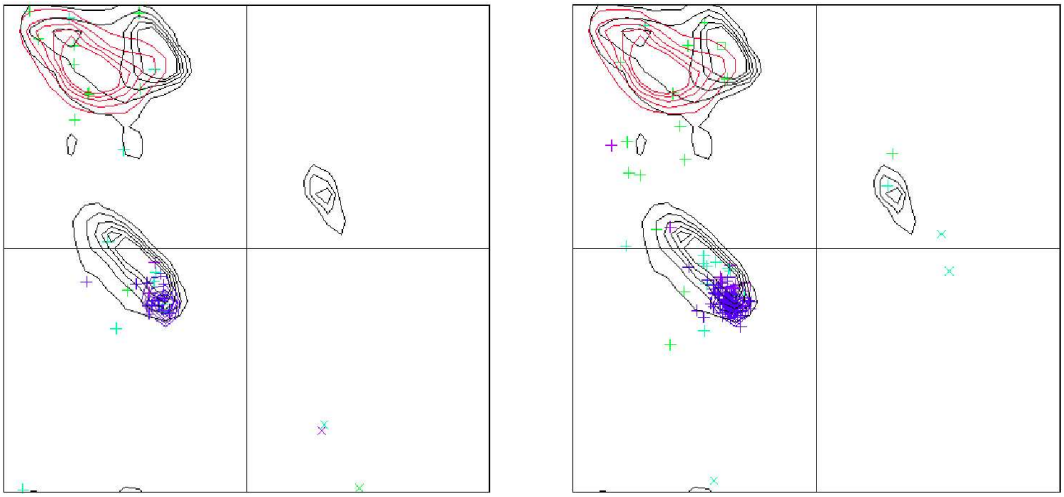


Abbildung 1.12: Hier sind die Ramachandran-Plots der beiden Proteine Insulin (**links**) und Myoglobin (**rechts**) gezeigt, erzeugt von einem Plot-Server (http://www.cmbi.kun.nl/gv/servers /WIWWWI/ramaplot.html) aus den selben .pdb-Originaldateien die für Abb. 1.11 verwendet wurden. Blau bedeuted hier Helix, rot *strand* und grün (bzw. schwarz) *turn* und *loop*. Die äußeren Linien umkreisen das Gebiet, in dem 90% aller Punkte gefunden werden sollten, die inneren das Gebiet, in dem 50% aller Punkte gefunden werden sollten. + bezeichnet „normale“ Residuen, × Glycin und □ Prolin. Man sieht auch hier deutlich, daß, vor allem beim Myoglobin, die meisten Punkte in die Region der $\alpha-$Helicees fallen (vgl. Abb. 1.11 unten).

1.1.4 Proteinfaltung

Bisher sind nur die Vorraussetzungen für die Faltung und deren statischen Zustände betrachtet worden. Denkt man an den Faltungsprozeß an sich, tauchen weitere Fragen auf, z.B.: Was sind die treibenden Kräfte? Wie gelangen Proteine zu ihrer nativen Konformation? In welchen Stufen vollzieht sich die Faltung? Auf welcher Zeitskala spielt sich die Faltung ab? Vorweg: Die meisten Fragen werden Fragen bleiben und sogar noch weitere aufwerfen.

***Blind Watchmaker* Paradoxon** Der erste Schritt ist sicherlich, gehen wir noch einmal ein Stück zurück, die Frage, welche Sequenzen für eine bestimmte Konformation in Betracht kommen. Sicherlich hat die Evolution nicht zufällig Sequenzen ausgewählt, um Proteine für bestimmte Funktionen zu erhalten. Nimmt man nur 100 Aminosäurereste, hat man $20^{100} \cong 10^{130}$ Möglichkeiten, daraus eine Aminosäuresequenz zu erstellen. Der Sequenzraum ist also riesig und fast leer, markiert man nur alle Sequenzen, die tatsächlich in Proteinen vorkommen.

Der Name des Paradoxons kommt wohl sicher von der Idee, daß eine Uhr sehr wohl *designed* ist, d.h. alle Teile bewußt so zusammengefügt sind, daß sie die Funktion einer Uhr erfüllen, genau wie das bei den Proteinen sein soll. Fügt man jetzt aber alle Teile zufällig (blind) zusammen, ist die Wahrscheinlichkeit eine Uhr zu erhalten nahezu Null. Biologische

Proteine können also nicht aus zufälligen Sequenzen entstehen. Dieses Problem löst sich, wenn man nicht nach Sequenzen, sondern vielmehr nach Strukturen sucht. Sucht man unter allen Möglichkeiten die z.B. Isozymsequenz, ist die Chance, sie blind zu finden gleich Null. Sucht man aber eine Struktur, die die Funktion von Isozym erfüllt, ist die Chance etwa 100 Größenordnungen größer, also etwa 10^{-10} bis 10^{-20} [8].

Levinthals Paradoxon Gehen wir nun vom Sequenzraum zum Strukturraum. Auch dieser ist sehr groß. Nimmt man eine feste Sequenz, so kann die zugehörige native Struktur keineswegs durch zufällige Suche im Strukturraum gefunden werden. Schätzungen zufolge würde solch eine zufällige Suche, bzw. das „Abzählen“ aller möglichen Konformationen[6], für eine 100 Aminosäurereste lange Sequenz $\approx 10^{78}$ Jahre dauern, für eine Sequenz mit 40 Residuen immer noch $\approx 10^{18}$ Jahre [9], also immer noch größer als das Alter des Universums. Dieser riesige Unterschied zwischen den geschätzen und den wahren, offensichtlich sehr viel kürzeren Faltungszeiten ist Levinthals Paradoxon.

In der Natur dauert die Faltung eines Proteins mit 100 Residuen bei Körpertemperatur etwa 5 s. Der Faltungsprozeß muß also auf irgendeine Art gesteuert sein, was genau passiert, ist aber nicht bekannt.

Der Faltungsprozess An einigen Proteinen konnte man beobachten, daß der Faltungsprozeß in Etappen verläuft. So ist ein Modell der Faltung das sogenannte $1° \rightarrow 2° \rightarrow 3°$ Modell [8]. Die Primärstruktur führt, aufgrund lokaler Wechselwirkungen, sehr schnell zu Sekundärstrukturen in einzelnen Bereichen (*helix-coil transition*), diese formen sich dann zur Tertiärstruktur. Dies ist eine *backbone*-zentrierte Betrachtungsweise des Problems, die Eigenschaften der Seitenketten wurden hier völlig außen vor gelassen. Die treibende Kraft sind hierbei die Wasserstoffbrücken des Peptidskeletts und die möglichen Winkelpaare (Ψ, Φ). Dieses Vorgehen wurde oft auch in Computersimulationen bevorzugt. Man sucht lokal Eigenschaften, die auf eine bestimmte Sekundärstruktur hindeuten und fügt diese zu einer Tertiärstruktur zusammen.

Eine andere Sicht ist die Seitenkettenzentrierte. Hierbei ist die treibende Kraft die Hydrophobizität. Man geht davon aus, daß die hydrophobe Wechselwirkung die stärkste (in Wasser) ist. „Wirft“ man ein Protein (in beliebiger Konformation) in Wasser, kommt es zu einem spontanen Kollaps (*collapse transition*), bei der der kompakte Endzustand durch die hydrophoben Wechselwirkungen zwischen den unpolaren Seitenketten erhalten wird. Die Ausbildung von Sekundärstrukturen ist dann weniger Folge eines *helix-coil* Übergangs, sondern sie entstehen ebenso durch den Kollaps.

Das „wahre“ Verhältnis dieser Sichtweisen ist nicht bekannt. Computersimulation, die Seitenketteneigenschaften beinhalten, die möglichen (Ψ, Φ)-Winkel aber nicht berücksichtigen, modellieren viele Eigenschaften globularer Proteine gut, verlieren aber jede Informa-

[6] Unter der Annahme, daß jeder der Winkel Φ und Ψ nur 3 Werte annehmen kann(!), bzw. bei jeder Konformationsänderung einer der Winkel sich um $\pm 60°$ ändert und alle anderen Winkel fest sind.

tion über die Sekundärstruktur. Auf der anderen Seite zeigen Proteine, bei denen nur die (Ψ, Φ)-Neigungen simuliert werden und nicht die Seitenketten sehr gute Sekundärstrukturen, aber keine kompakten gefalteten Zustände. Sicherlich hängt die „Wahrheit“ auch stark vom betrachteten Protein ab. Z.B. gibt es Sequenzen, die nicht kollabieren können. Dann kontrollieren die (Ψ, Φ)-Winkelneigungen fast komplett die Struktur, diese besteht dann vielleicht nur aus einer Helix[7].

Die Faltung von Proteinen ist in jedem Fall stark kooperativ, d.h. es gibt keine „halb“ gefalteten Proteine. Gibt man z.B. einer Proteinlösung langsam eine Substanz X zu, die eine Denaturierung bewirkt, kommt man irgendwann an einen Punkt, an dem 50% der Lösung denaturiert ist. Das heißt aber nicht, daß jedes Protein zur Hälfte entfaltet ist, sondern daß 50% komplett denaturiert sind, die anderen aber noch komplett in ihrem nativen Zustand verharren. Eine Übersicht über die beiden Sichtweisen zeigt Tab. 1.3, die aus [8] entnommen ist.

	backbone zentriert	Seitenkettenzentriert
Dominante Kraft	(Ψ, Φ), Wasserstoffbrücken	Hydrophobizität, Wasserstoffbrücken
Übergang	*helix-coil*	Kollaps
Kinetik	Helixformation sehr schnell	Formation des hydrophoben Kerns sehr schnell

Tabelle 1.3: Vergleich der beiden Sichtweisen zur Proteinfaltung in Kernpunkten. Aus [8].

1.2 Realistische Modelle und Wechselwirkungen

Wie sehen die realistischen Wechselwirkungen in Proteinen aus? Wenn man diese kennt, sollte es prinzipiell möglich sein, die Tertiärstruktur von Proteinen aus ihrer Primärstruktur vorherzusagen. Eines der Standardmodelle der Energiefunktion realer Proteinsysteme ist das folgende [10]:

$$E_{\text{ges}} = E_{\text{P}} + E_{\text{S}}\,, \tag{1.1}$$

wobei E_{P} die Energie des Proteinmoleküls (in $^{\text{kcal}}/_{\text{mol}}$) selbst beschreibt und E_{S} die Wechselwirkung des Proteins mit seiner Umgebung, i.a. eines Lösungsmittels (hauptsächlich Wasser). E_{P} selbst setzt sich aus mehreren Anteilen zusammen: dem elektrostatischen Coulomb-Term E_{C}, einem Lennard–Jones-Term E_{LJ}, einem Wasserstoffbrücken-Term E_{HB} und dem Torsionsterm für alle Torsionswinkel. Die Potentiale sehen wie folgt aus:

$$E_{\text{C}} = \sum_{(i,j)} \frac{332\, q_i q_j}{\epsilon\, r_{ij}}\,, \tag{1.2}$$

[7] Polybenzyl-L-Glutamat ist z.B. ein klassischer Helixformer.

$$E_{\mathrm{LJ}} = \sum_{(i,j)} \left(\frac{A_{ij}}{r_{ij}^{12}} - \frac{B_{ij}}{r_{ij}^{6}} \right) , \tag{1.3}$$

$$E_{\mathrm{HB}} = \sum_{(i,j)} \left(\frac{C_{ij}}{r_{ij}^{12}} - \frac{D_{ij}}{r_{ij}^{10}} \right) , \tag{1.4}$$

$$E_{\mathrm{Tor}} = \sum_{i} U_i \left(1 \pm \cos(n_i \chi_i)\right) . \tag{1.5}$$

Hier ist r_{ij} der Abstand (in Å) zwischen zwei Atomen i und j, ϵ ist die dielektrische Konstante, q_i die Ladung des Atoms i (in Einheiten der Elektronenladung) und χ^i der Torsionswinkel der chemischen Bindung i. Die Zahl 332 wird für die Umrechnung der Einheiten in $^{\mathrm{kcal}}/_{\mathrm{mol}}$ benötigt. Die geometrischen Parameter, die partiellen Ladungen sowie eine genauere Diskussion der einzelnen Wechselwirkungen und Potentiale findet man in [11].

Eine Möglichkeit, die Wechselwirkung mit dem Lösungsmittel zu berücksichtigen, ist ein Potentialterm proportional zur Proteinoberfläche, die mit der Umgebung in Kontakt ist,

$$E_{\mathrm{s}} = \sum_{i} \sigma_i A_i \, . \tag{1.6}$$

A_i ist die Oberfläche einer jeden funktionellen Gruppe, die Zugang zur Umgebung hat, σ_i ist eine Kopplungsskonstante. Es gibt mehrere Möglichkeiten für die Wahl der funktionellen Gruppen und der Proportionalitätskonstanten. In [10] findet man Zitate für gute Parametermengen. Die Wechselwirkung des Proteins mit der Umgebung besteht aus einer hydrophoben Wechselwirkung, einer Lennard–Jones-Wechselwirkung und einer elektrostatischen Wechselwirkung. Der Term (1.6) berücksichtigt effektiv alle drei Anteile.

Andere Möglichkeiten, die umgebende Lösung einzubeziehen sind eine abstandsabhängige dielektrische Funktion ($\epsilon = \epsilon(r)$), welche allerdings nur die elektrostatische Wechselwirkung berücksichtigt oder das explizite Aufschreiben aller drei Wechselwirkungsterme. Aus der *scaled particle theory* [13] kann man einen expliziten hydrophoben Wechselwirkungsterm gewinnen. Der elektrostatische Term kann durch Lösen der Poisson–Boltzmann-Gleichung erhalten werden, zusammen mit einem Lennard–Jones-Term zwischen Protein und Lösung erhält man daraus eine sehr gute Theorie [10].

Weitere Möglichkeiten (*all atom* Repräsentation der Umgebung, *reference interaction site model*) sind ebenfalls in [10] angedeutet und weiter referenziert.

Aus mehreren Gründen ist die korrekte Vorhersage der nativen Struktur von Proteinen dennoch sehr schwer bzw. nicht möglich. Erstens ist die Einbeziehung der Umgebung eines Proteins nur sehr ungenau möglich, da die Anzahl der in Frage kommenden Moleküle in der Umgebung sehr groß ist. Weiterhin gibt es mehrere mehr oder weniger stark variierende Wechselwirkungsmodelle, das „wahre“ ist wahrscheinlich noch nicht bekannt [12]. Selbst wenn man dies hätte, wäre, wie wir schon gesehen haben, ein „Abzählen aller Konfigurationen“ nicht möglich. Monte-Carlo-Simulationen großer (realistischer) Proteine scheitern z.Z. noch (abgesehen von der Frage des „richtigen“ Modells) an der immensen Rechenleistung, die dazu nötig ist.

1.3 HP Proteine

1.3.1 Das HP-Modell

Nach allem, was in den vorrangegangenen Kapiteln gesagt wurde, ist klar, daß man für Computersimulationen viele Vereinfachungen annehmen muß[8]. Diese Vereinfachungen werden den Sequenzraum, den Strukturraum und die Wechselwirkungen betreffen.

Lau und Dill [14] schlugen in den 80er Jahren ein solches einfachstes Modell für Proteine vor, das HP-Modell, welches von nun an in dieser Arbeit benutzt werden soll. Ein Protein ist im HP-Modell eine lineare Kette von n Aminosäureresten. Im Gegensatz zu den 20 natürlichen Aminosäuren gibt es hier nur 2, eine mit der Eigenschaft, hydrophob zu sein (H), die andere mit der Eigenschaft, polar zu sein (P). Eine Konformation wird repräsentiert durch einen nicht selbstüberschneidenden Weg (*self-avoiding walk*) auf einem diskreten Gitter.

Es muß desweiteren unterschieden werden zwischen „verbundenen“ Nachbarn, solchen die in der linearen Sequenz aufeinanderfolgen, und rein „topologischen“ Nachbarn, die nicht verbunden sind, aber auf dem Gitter auf benachbarten Gitterplätzen liegen. Diese sind in Kontakt. Es wird nun angenommen, daß es genau dann eine Energie $\epsilon < 0$ in Einheiten von $k_{\mathrm{B}}T$ zwischen topologischen Nachbarn gibt, wenn beide hydrophob (H) sind. In allen anderen Fällen (HP-Kontakt, H oder P in Kontakt mit der Umgebung) soll $\epsilon \stackrel{!}{=} 0$ sein.

Die Energie eines HP-Proteins ist also genau dann gleich Null, wenn keine Aminosäure mit einer anderen in HH-Kontakt steht. Der Konformationsraum ist der Raum aller nicht selbstüberscheidenden Wege der Länge $n-1$ auf dem gegebenen Gitter, der Sequenzraum ist der Raum aller Sequenzen der Länge n aus 2 Aminosäuren. Beide sind jetzt also sehr viel kleiner als in der Natur oder in realistischen Modellen wie oben beschrieben.

Sei

$$\mathbf{s} = (s_0, \ldots, s_{n-1}) \quad \text{mit} \quad s_i = \begin{cases} 1 & \text{für H} \\ 0 & \text{für P} \end{cases} \tag{1.7}$$

die Aminosäuresequenz eines HP-Proteins und seien

$$\mathbf{r}_i = (r_x, r_y, r_z) \tag{1.8}$$

die kartesischen Koordinaten der i-ten Aminosäure auf einem kubischen Gitter. Dann kann man eine $n \times n$-Matrix wie folgt definieren:

$$C_{ij}(\mathbf{r}) = \begin{cases} 1 & \text{für} \quad |\mathbf{r}_i - \mathbf{r}_j| = 1 \quad \text{und} \quad |i-j| \neq 1 \\ 0 & \text{sonst.} \end{cases} \tag{1.9}$$

Die Einträge sind also 1, wenn die Amonisäuren i und j in topologischem Kontakt stehen und sonst 0. C_{ij} heißt Kontaktmatrix (ausführlich diskutiert in [7]). In einer Matrix U_{ij} kann

[8] Für Oligopeptide kann man sehr viel realistischere Modelle benutzen als für Proteine, da die Kettenlängen sehr viel kürzer sind und der Konformationsraum daher viel kleiner ist.

man die Wechselwirkung kodieren:

$$U_{ij}(\mathbf{s}) = \begin{cases} \epsilon & \text{für} \quad s_i = s_j = 1 \\ 0 & \text{sonst.} \end{cases} \tag{1.10}$$

In der Matrix gibt es also überall da den Eintrag ϵ, wo die $i$te und $j$te Aminosäure beide hydrophob (H) sind, sonst 0. Es gibt hier auch weitere Möglichkeiten, z.B. kann man sich eine attraktive Wechselwirkung auch zwischen HP-Kontakten denken um zu modellieren, daß sich eine polare Aminosäure „lieber“ an eine unpolare anlagert, als an die Umgebung (siehe z.B. [7]). Ich werde aber im folgenden immer bei dieser Definition von U_{ij} bleiben. ϵ wird immer den Wert $\epsilon = -1$ haben, die Wechselwirkung ist also stets anziehend. Die Gesamtenergie eines Proteins[9] ist wie folgt definiert:

$$E(\mathbf{s}, \mathbf{r}) = \sum_{i=0}^{n-1} \sum_{j=i}^{n-1} C_{ij}(\mathbf{r}) U_{ij}(\mathbf{s}) \,. \tag{1.11}$$

Für die in dieser Arbeit untersuchten Beispiele vereinfacht sich dieser Ausdruck zu

$$E(\mathbf{s}, \mathbf{r}) = -\sum_{\langle i,j \rangle} s_i s_j \,, \tag{1.12}$$

wobei $\langle i, j \rangle$ symbolisieren soll, daß das i-te und j-te Monomer topologische Nachbarn sind, also auf dem Gitter nächste Nachbarn sind, aber in der Sequenz nicht aufeinanderfolgen.

Die negative Energie eines HP-Proteins ist also die Summe aller (topologischen) H-H Kontakte im gefalteten Protein. Die Konformation eines Protein mit der geringsten freien Energie heißt Grundzustand des Proteins. Im allgemeinen ist dieser, durch die Diskretisierung, stark entartet. Ist der Grundzustand eindeutig, d.h. nicht entartet, heißt die Sequenz *designed*. Abbildung 1.13 zeigt ein Beispiel eines sehr kurzen HP-Proteins in seinem Grundzustand.

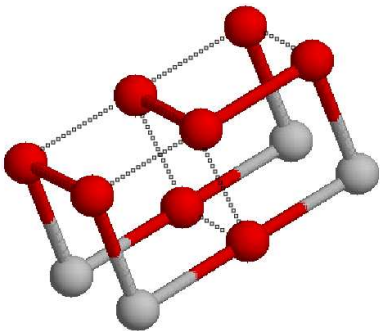


Abbildung 1.13: Ein HP-Protein aus 12 Monomeren mit der Sequenz 111010110101 auf dem kubischen Gitter. Hydrophobe Monomere ($s_i = 1$) sind dunkel (rot), polare ($s_j = 0$) hell (grau) dargestellt. Das Protein befindet sich in seinem Grundzustand mit der Energie $E = -7$. Die HH-Kontakte sind durch eine *gestrichelte Linie* angedeutet.

[9] „Protein“ meint jetzt immer HP-Protein.

1.3.2 Reale Proteine vs. HP Proteine

Allein durch die Bindung an ein diskretes Gitter sollte man nicht erwarten, daß Untersuchungen an HP-Proteinen weitreichende Rückschlüsse auf das Verhalten oder Eigenschaften von realen Proteinen erlauben. Wie man später sehen wird, ist z.B. die Art des Gitters maßgeblich für den Entartungsgrad des Grundzustandes verantwortlich.

Durch die Reduzierung aller Eigenschaften natürlicher Proteine auf die Hydrophobizität (und dabei noch nur auf 2 Arten) der Seitenkette reduziert man sich rein auf die seitenkettenorientierte Sicht der Proteinfaltung. Es ist zu erwarten, daß man den Kollaps, d.h. die spontane Bildung eines hydrophoben Kerns, beobachten kann. Man kann allerdings nicht erwarten, daß man Sekundärstrukturen wie sie in der Natur vorkommen modellieren kann. Diese hängen wie gesehen stark von Eigenschaften und Kräften ab, die im HP-Modell komplett vernachlässigt sind.

Allerdings kann man durch die Einfachheit des Modells mit der derzeit zur Verfügung stehenden Rechenleistung Proteine in realistischer Länge ($n \approx 50 - 150$) gut simulieren. Desweiteren bietet das Modell eine Basis, um Algorithmen zu untersuchen, zu vergleichen oder auf ihre Tauglichkeit im Hinblick auf schwierigere Modelle zu testen und zu optimieren.

Abbildung 1.14 zeigt ein Beispiel in dem ein reales Protein seinem HP-Modell gegenübergestellt ist.

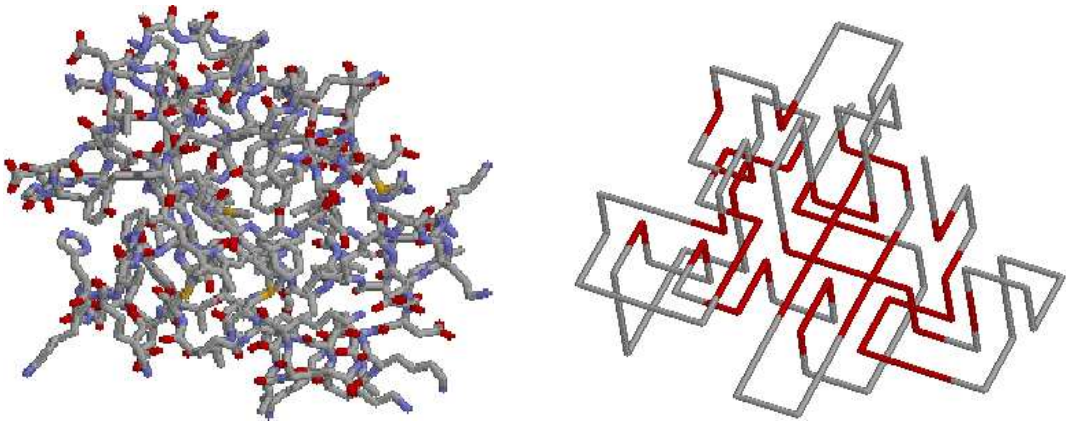


Abbildung 1.14: **Links**: Das reale Protein Cytochrome c, von der Proteindatenbank (http://www.rcsb.org/pdb/index.html), PDB ID 3CYT, dargestellt mit *rasmol*. **Rechts**: Das HP-Modell des Cytochrome c [15] (103 Monomere), in der gleichen Darstellung wie links, auf dem kubischen Gitter. Es befindet sich hier in einem Zustand, der nicht sein Grundzustand, dem aber wahrscheinlich sehr nahe ist.

Kapitel 2

Verallgemeinerte Gitter

Der einfachste (und bisher in allen meinen Ausführungen angenommene) Gittertyp ist das einfache quadratische bzw. das einfache kubische (*simple cubic*) Gitter (sc-Gitter). Seine Basisvektoren sind $\mathbf{a}_1 = (1,0,0)^T$, $\mathbf{a}_2 = (0,1,0)^T$ und $\mathbf{a}_3 = (0,0,1)^T$. Jeder Gitterpunkt hat in 2 Dimensionen 4 nächste Nachbarn, in 3 Dimensionen 6. Die Einheitszelle ist ein Quadrat bzw. Kubus mit einem Gitterpunkt im Zentrum.

2.1 Das 2D Dreiecksgitter

Etwas allgemeiner als das quadratische Gitter (in 2 Dimensionen) ist das Dreiecksgitter, welches folgende Basisvektoren hat:

$$\mathbf{a}_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad \mathbf{a}_2 = \begin{pmatrix} 1/2 \\ \sqrt{3}/2 \end{pmatrix}. \tag{2.1}$$

Jeder Gitterpunkt hat 6 nächste Nachbarn, genau wie das kubische Gitter in 3 Dimensionen. Den Zusammenhang zwischen den Gittern, die Gemeinsamkeiten und Unterschiede werden wir später noch besser erkennen. Abbildung 2.1 zeigt links das Gitter, wie es hier beschrieben ist.

2.1.1 Transformation des Gitters

Das erste reale Problem, welches sich stellt, ist die Darstellung des Gitters in einem Computerprogramm. Sicherlich ist es kompliziert, es genau so abzubilden, wie es oben beschrieben ist. Deswegen werde ich das Dreiecksgitter auf ein quadratisches Gitter in 2 Dimensionen transformieren, wobei jeder Gitterpunkt außer mit seinen 4 nächsten Nachbarn noch mit 2 übernächsten verbunden ist. Anschaulich wird das an Abb. 2.1.

Man verschiebt jede 2. Zeile des Dreiecksgitters eine halbe Gitterkonstante nach links (bzw. jede 2.+1 nach rechts). So ensteht ein Quadratgitter mit Verbindungen zu allen nächsten Nachbarn sowie zu 2 Übernächsten.

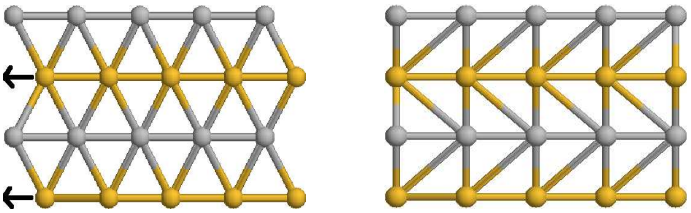


Abbildung 2.1: Transformation des 2D Dreiecksgitters auf das 2D Quadratgitter mit 6 Nachbarn (4 nächste und 2 übernächste).

Jetzt ergibt sich das Problem, daß eine Einheitszelle nicht mehr nur einen, sondern 2 Knoten enthält, die ich separat behandle. Die beiden zusätzlichen nächsten Nachbarn sind jetzt in jeder „geraden“ Zeile (in Abb. 2.1 grau) „links unten“ bzw. „links oben“ in den „ungeraden“ Zeilen (gelb) „rechts oben“ bzw. „rechts unten“.

Algorithmisch ist dieser Fall identisch dem kubischen Gitter in 3 Dimensionen mit 6 nächsten Nachbarn. Um zu den nächsten Nachbarn zu gelangen, schaut man nur nicht auf die Positionen $x \pm 1$, $y \pm 1$ und $z \pm 1$ (in karthesischen Koordinaten), sondern zu $x \pm 1$, $y \pm 1$, $(x-1, y-1)$ und $(x-1, y+1)$ (bzw. $(x+1, y-1)$ und $(x+1, y+1)$).

2.1.2 Exkurs: Design von HP-Proteinen

Ich möchte hier versuchen zu zeigen, wie ich mit einer sehr naiven Idee versucht habe, Sequenzen so zu designen, daß sie auf dem 2D Dreiecksgitter einen eindeutigen Grundzustand haben (zumindest, was die Lage der Monomere auf dem Gitter betrifft, also die „Tertiärstruktur“). Vorweg bemerkt: Es ist fehlgeschlagen, einige Dinge werden aber durch die folgenden Untersuchungen klarer. Die Ideenskizze:

- Nimm einen Grundzustand eines n-Homopolymers, wie z.B. in Abb. 2.2 links ein 48-Homopolymer. Homopolymer heißt hier, daß ein Protein nur aus hydrophoben Aminosäuren bestehen soll bzw. aus einer einzigen Sorte von Monomeren mit attraktiver Wechselwirkung. Der Grundzustand ist dann die auf dem jeweiligen Gitter kompakteste mögliche Konformation.
- Erzeuge daraus ein n-Heteropolymer, indem genau die Monomere, die die wenigsten Kontakte zu anderen Nachbarn haben (ohne mit denen verbunden zu sein), polar gesetzt werden, alle anderen hydrophob bleiben. In Abb. 2.2 links sind das die Monomere $17, 38, 41$ und 45 mit jeweils einem Kontakt zu anderen Monomeren.
- Führe eine Grundzustandssuche durch[1].

Das 48-Homopolymer hat auf dem 2D Dreiecksgitter eine Grundzustandsenergie $E_{\min} = -73$. Dann ist die entsprechende Erwartung für die Grundzustandsenergie des abgeleiteten 48-Heteropolymers $E = -69$ (wir ziehen die 4 Monomere, die jeweils $E_{\text{local}} = -1$ zur Gesamtenergie beitragen ab). Das zuerst überraschende Ergebnis sieht man in Abb. 2.2: Es gibt einen Zustand für dieses 48-Heteropolymer mit $E = -70$!

[1] Der dazu benutzte Algorithmus wird später noch genau beschrieben werden.

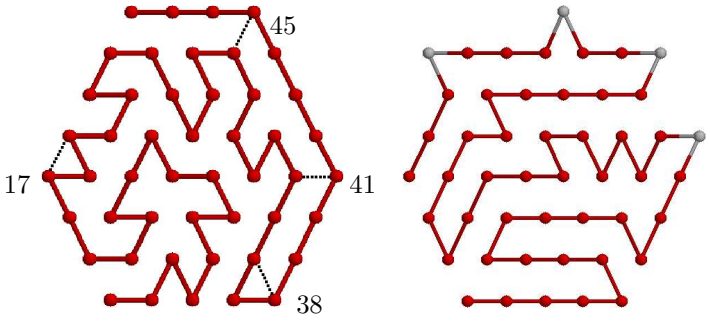


Abbildung 2.2: **Links:** Grundzustand des Homo-48mers auf dem zweidimensionalen Dreiecksgitter. Angedeutet sind die Monomere mit $E_{\text{local}} = -1$. **Rechts** ein Zustand mit der Energie $E = -70$ für das Hetero48mer $H_{16}PH_{20}PH_2PH_3PH_3$.

Es ist nun einzusehen, warum das so ist. Die polaren Monomere werden sich nicht so anordnen, daß die hydrophoben Nachbarn gegenseitig keinen Kontakt bilden können. Vielmehr entspricht der Zustand $E = -70$ dem kompakten Grundzustand eines $(48 - 4 = 44)$ Homo44mers, an den außen noch polare Monomere angelagert sind. So können sogar noch zwischen benachbarten verbundenen Monomeren am Rand Kontakte entstehen. Ändern wir also die Strategie:

- Nimm einen Grundzustand eines n-Homopolymers, wie z.B. in Abb. 2.2 links ein 48-Homopolymer.
- Erzeuge daraus ein $n + x$-Heteropolymer, indem x polare Monomere zwischen hydrophobe und verbundene Randmonomere gesetzt werden.
- Führe eine Grundzustandssuche durch.

Schauen wir uns wieder das obige Beispiel an. Gehen wir von Abb. 2.2 rechts aus und fügen in diesem Fall 12 polare Monomere an (das so entstehende Protein zeigt Abb. 2.3 links). Dadurch entstehen 12 neue Kontakte, die zu erwartende Grundzustandenergie ist also $E = -70 - 12 = -82$. Abbildung 2.3 rechts zeigt einen Zustand des Hetero60mers der durch eine Grundzustandssuche mit der neu entstandenen Sequenz erhalten wurde mit genau dieser Energie.

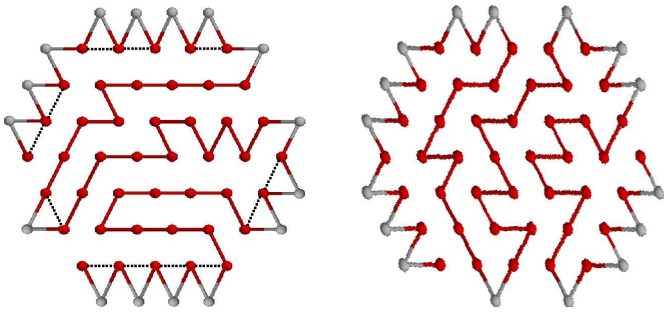


Abbildung 2.3: **Links:** Das durch Hinzufügen von polaren Monomeren aus dem Hetero48mer entstandene Hetero60mer. **Rechts** ein Zustand, der durch eine Computersimulation gefunden wurde. Denkt man sich die Verbindungen zwischen benachbarten Monomeren weg, unterscheiden sich die beiden Konfigurationen nur duch eine Drehung des polaren Außenrings um 3 Plätze.

Die beiden Konfigurationen unterscheiden sich, abgesehen von den Bindungen im hydrophoben Kern, immer noch durch Lage der polaren Hülle relativ zum Kern. Der Grundzustand derart erzeugter Sequenzen ist i.a. also weder eindeutig, noch kann man so eine eindeutige Tertiärstruktur designen.

2.2 Das 3D Tetraedergitter

Aus der Kristallographie sind verschiedene Gitter in 3 Dimensionen bekannt, die eine höhere Koordinationszahl haben, als das sc-Gitter, in unserem Sinne also „allgemeiner“ sind, z.B. das kubisch raumzentrierte (*body centered cubic*) Gitter (bcc-Gitter). Es hat die Basisvektoren [16]

$$\mathbf{a}_1 = (1,1,-1)^T/2\,, \quad \mathbf{a}_2 = (-1,1,1)^T/2 \quad \text{und} \quad \mathbf{a}_3 = (1,-1,1)^T/2\,. \tag{2.2}$$

Jeder Gitterpunkt hat im bcc-Gitter 8 nächste Nachbarn.

Im folgenden soll ein Gitter eingeführt werden, welches 12 nächste Nachbarn hat. Analog zur Verallgemeinerung in 2 Dimensionen, wähle ich dafür hier ein „dreidimensionales (3D) Dreiecksgitter“ bzw. das Tetraedergitter.

2.2.1 Beschreibung des Gitters

Das 3D Dreiecksgitter ist aus folgenden Basisvektoren aufgebaut:

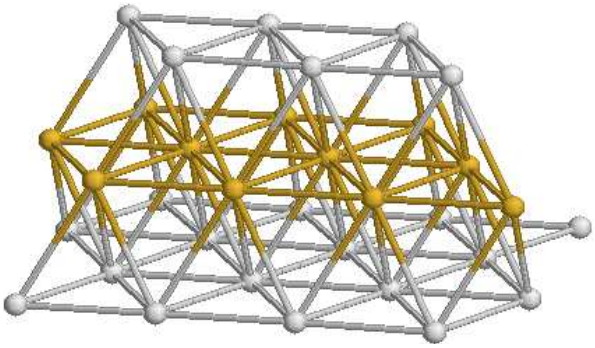
$$\mathbf{a}_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}\,, \quad \mathbf{a}_2 = \begin{pmatrix} 1/2 \\ \sqrt{3}/2 \\ 0 \end{pmatrix}\,, \quad \mathbf{a}_3 = \begin{pmatrix} 1/2 \\ 1/(2\sqrt{3}) \\ \sqrt{2/3} \end{pmatrix}\,. \tag{2.3}$$

Die Basisvektoren $\mathbf{a}_1$, $\mathbf{a}_2$ und $\mathbf{a}_3$ bilden einen Tetraeder. Das so entstehende Gitter hat, wie gewünscht, die Koordinationszahl 12, ist also noch verketteter als das bcc-Gitter. Abbildung 2.4 zeigt das Tetraedergitter und einen Knoten mit seinen 12 Nachbarn.

Eigenschaften

i) Das Tetraedergitter ist identisch dem ebenfalls aus der Kristallographie bekannten kubisch flächenzentrierten (*face centered cubic*) Gitter (fcc-Gitter). Als dessen Basisvektoren werden üblicherweise

$$\mathbf{a}'_1 = \begin{pmatrix} 1/2 \\ 1/2 \\ 0 \end{pmatrix}\,, \quad \mathbf{a}'_2 = \begin{pmatrix} 0 \\ 1/2 \\ 1/2 \end{pmatrix}\,, \quad \mathbf{a}'_3 = \begin{pmatrix} 1/2 \\ 0 \\ 1/2 \end{pmatrix} \tag{2.4}$$



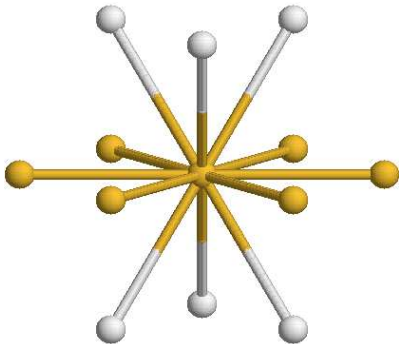


Abbildung 2.4: **Links:** Ein Blick auf das aus den Basisvektoren $\mathbf{a}_1$ bis $\mathbf{a}_3$ aufgebaute Gitter. **Ausschnitt (rechts):** Ein Knoten mit seinen 12 Nachbarn (Ähnliche Abbildungen findet man in [16, 17]).

angegeben [16]. Die Transformationsgleichungen sind die folgenden:

$$\begin{aligned}
\sqrt{2}\,x &= \frac{1}{\sqrt{2}}\,x' + \frac{1}{\sqrt{2}}\,y' + 0\,z'\,, \\
\sqrt{2}\,y &= \frac{1}{\sqrt{3}}\,x' - \frac{1}{\sqrt{3}}\,y' + \frac{1}{\sqrt{3}}\,z'\,, \\
\sqrt{2}\,z &= -\frac{1}{\sqrt{6}}\,x' + \frac{1}{\sqrt{6}}\,y' + \frac{2}{\sqrt{6}}\,z'\,.
\end{aligned} \tag{2.5}$$

ii) Man kann das Gitter aus verschiedenen Richtungen betrachten. Schaut man sich das Gitter z.B. aus der Richtung von $\mathbf{a}_2 - \mathbf{a}_3 = (0, 1/\sqrt{3}, -\sqrt{2/3})^T$ aus an, sieht man, daß die Gitterebenen nur aus 2D Dreiecksgittern aufgebaut sind (siehe Abb. 2.5).

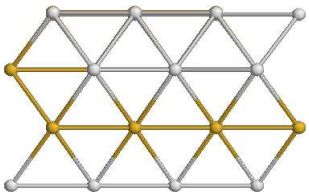


Abbildung 2.5: Blick aus Richtung (0,-1,0) auf das Dreiecksgitter nach einer globalen Drehung $(0, -1/\sqrt{3}, \sqrt{2/3})^T \rightarrow (0, 1, 0)^T$.

2.2.2 Transformation des kubischen Gitters

Analog 2D Aus Abb. 2.5 erkennt man die Vorgehensweise analog der Transformation. Nach der globalen Drehung $(0, -1/\sqrt{3}, \sqrt{2/3})^T \rightarrow (0, 1, 0)^T$ braucht man jetzt, wie in 2 Dimensionen, „nur noch“ jede 2. Reihe (und alle darunterliegenden) eine halbe Gitterkonstante entlang der $x-$Achse zu verschieben (und schliesslich die Gitterkonstante in alle Richtungen auf 1 zu strecken, um auf ganze Zahlen für die absoluten Koordinaten zu kommen). Das Dreiecksgitter und das entsprechende kubische Gitter zeigt Abb. 2.6 (in etwa gleicher Position).

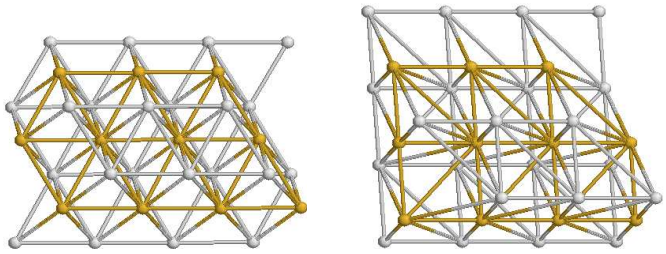


Abbildung 2.6: **Links** Das Dreiecksgitter, **rechts** das entsprechende Gitter auf kubischem Untergrund. Beide Bilder sind in etwa vom selben Beobachterpunkt aus aufgenommen. Am dem Beobachter nächsten Punkt des rechten Bildes wurde dieses jedoch noch etwas nach „oben“ gedreht, um die Tiefe erkennen zu können.

Das fcc-Gitter Die einfachere Lösung ist die Benutzung der Basisvektoren $\mathbf{a}_i'$ für das fcc-Gitter. Das entspricht einer globalen Drehung auf ein kubisches System. Abbildung 2.7 verdeutlicht noch einmal den Zusammenhang zwischen dem Tetraeder- und dem fcc-Gitter und zeigt die Elementarzelle des Gitters.

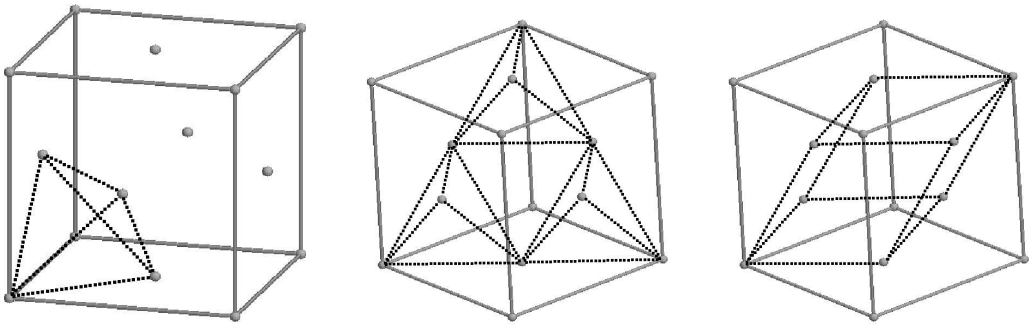


Abbildung 2.7: **Links:** Eine kubische Zelle des fcc-Gitters, die Basisvektoren bilden einen Tetraeder. Die Abbildung in der **Mitte** zeigt eine Lage des Tetraedergitters in einer kubischen Zelle des fcc-Gitters. Ganz **rechts** ist die primitive Elementarzelle des fcc-Gitters und somit auch des Tetraedergitters dargestellt. Die Elementarzelle ist selbst kein Tetraeder.

Kapitel 3

Ketten und Zufallswege

3.1 Abzählen aller Konformationen und das Problem der Gewichte

3.1.1 Beschreibung des Problems

Rosenbluth und Rosenbluth [18] zeigten bereits Mitte des letzten Jahrhunderts, daß es ein Problem mit zufällig wachsenden Graphen auf Gittern gibt, die sich nicht selbst wieder berühren dürfen (*self-avoiding random walks*). Abbildung 3.1 zeigt einen in diesem Sinne erlaubten und einen verbotenen Graphen.

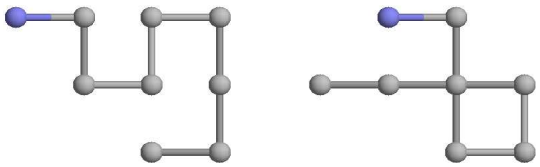


Abbildung 3.1: **Links** ein erlaubter Graph mit $L = 8$ Kanten, **rechts** ein verbotener Graph derselben Länge.

Nehmen wir die Wahrscheinlichkeit, daß ein Graph $\mathcal{G}$ durch einen Zufallsweg mit obiger Einschränkung entsteht. Wir können an jedem Knoten des Graphen würfeln, wo die nächste Kante hinführen soll. Die Anzahl der Möglichkeiten, $\mathcal{G}$ von einem Knoten aus zu verlängern, ist gleich der Anzahl der freien Nachbarplätze dieses Knotens.[1] (In Abb. 3.2 sind 3 Graphen dargestellt, die für die Platzierung des nächsten Knotens noch alle 3, noch 2 bzw. nur noch eine Möglichkeit haben. Die Wahrscheinlichkeiten, daß der nächste Knoten an einen bestimmten Nachbarplatz kommt, sind entsprechend 1/3, 1/2 bzw. 1.) Die Wahrscheinlichkeit, daß $\mathcal{G}$ so entsteht, ist das Produkt aller Einzelwahrscheinlichkeiten der Kanten. Da diese, wie gezeigt, von der Anzahl der freien Nachbarplätze an jedem Knoten abhängt, haben i.a. nicht alle Graphen gleicher Länge auch die gleiche Wahrscheinlichkeit zu entstehen.

[1] Ohne die Einschränkung der Nichtberührung wäre die Anzahl der Möglichkeiten auf einem quadratischen Gitter ($d = 2$) natürlich immer gleich 4 bzw. 3, wenn man nicht direkt zurücklaufen kann.

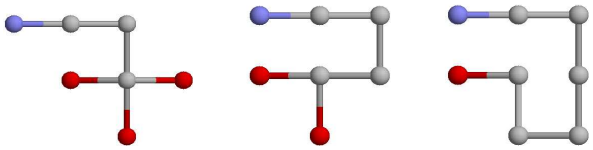


Abbildung 3.2: Ein Graph mit 3 Möglichkeiten (dunkel bzw. rot), die nächste Ecke zu setzen (**links**), mit 2 Möglichkeiten (**Mitte**) und ein Graph mit nur einer möglichen Verlängerung (**rechts**).

In Abb. 3.3 sind alle möglichen Graphen, die ich im Folgenden auch Ketten oder Konformationen nennen werde, der Länge $L = 4$, die entstehen können (in $2d$, bis auf Rotation der Kette am Ursprung, d.h. vom Ursprung aus wird per Definition immer zuerst nach rechts gegangen) mit den jeweiligen Wahrscheinlichkeiten aufgeführt.

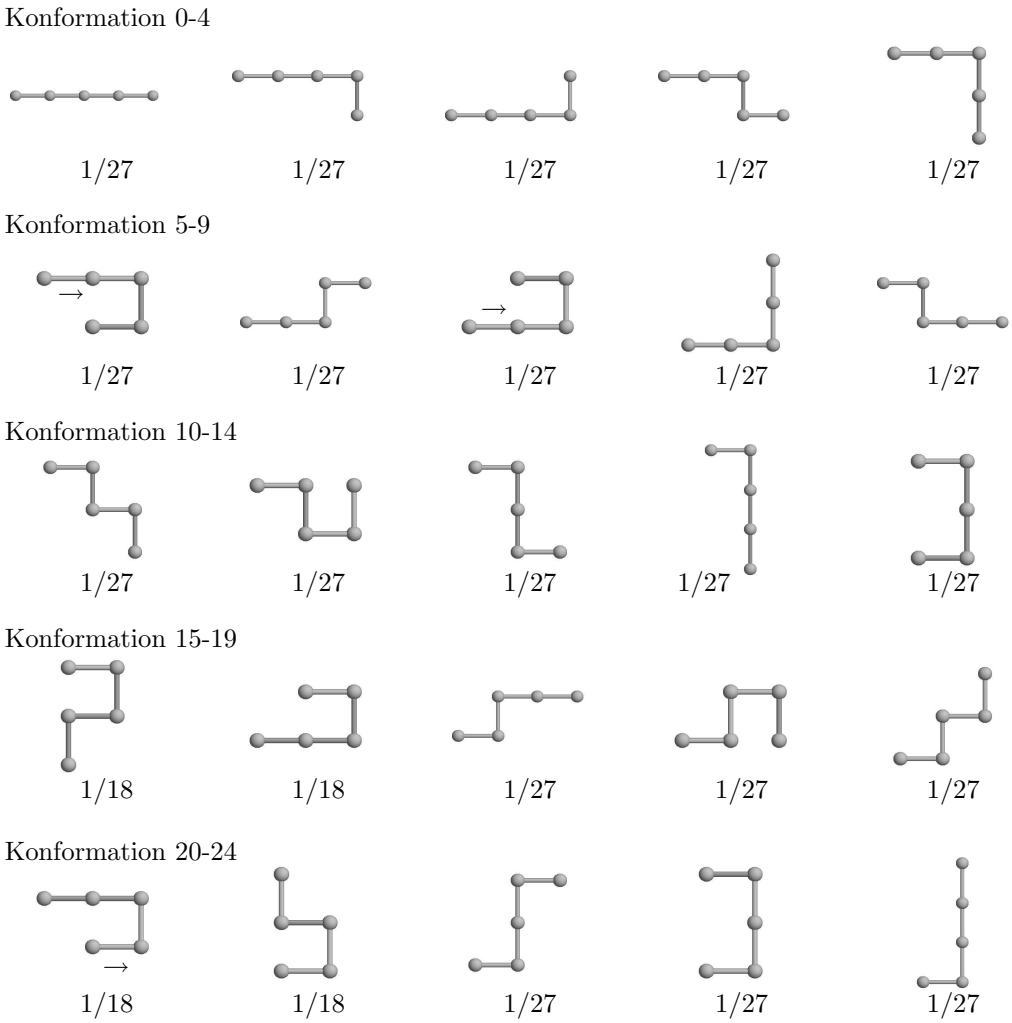


Abbildung 3.3: Alle möglichen Konfigurationen von Ketten aus 5 Knoten und die dazugehörigen Wahrscheinlichkeiten. Der Pfeil markiert die Richtung des Wachstums, wenn dies notwendig ist. So wird zum Beispiel klar, warum Konfiguration 5 eine andere Wahrscheinlickeit hat, zu entstehen, als Konfiguration 20, obwohl diese sich sonst nicht unterscheiden.

Abbildung 3.4 zeigt, mit den jeweiligen Wahrscheinlichkeiten, alle möglichen Ketten der Länge $L = 5$ (in $2d$), bis auf Rotation um den Ursprung und Spiegelung an der $x = 0$-Achse.

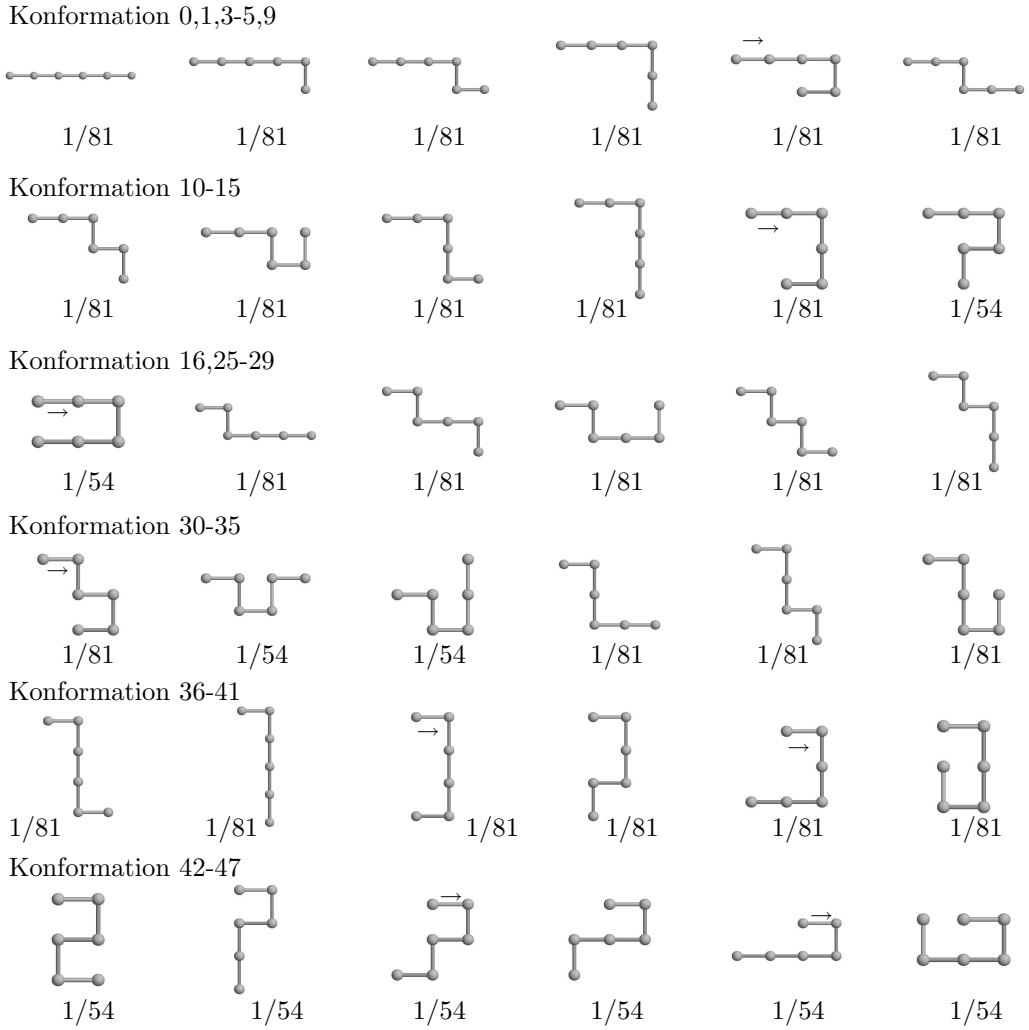


Abbildung 3.4: Alle möglichen Konformationen von Ketten aus 6 Knoten. (Dargestellt bis auf diejenigen, die durch Spiegelungen an der horizontalen Achse aus anderen schon gezeigten hervorgehen. So ist z.B. Konformation 2 nicht gezeigt, da sie durch Spiegelung aus Konformation 1 hervorgeht (siehe auch Abb. 3.3 Konformation 1 und 2) sowie z.B. Konformation 6, die aus 3 hervorgeht.) Der Pfeil hat hier die gleiche Bedeutung wie in Abb. 3.3.

3.1.2 Simple-Sampling-Experiment

Ich habe eine Simulation durchgeführt, in der zuerst alle möglichen Konformationen mit einer bestimmten Länge erzeugt werden. Die Ergebnisse sind z.T. schon in Abb. 3.3 und 3.4 gezeigt, für größere Längen kann man nur noch schwer alle bildlich darstellen. In Tab. 3.1 sind daher die Anzahlen der möglichen Konformationen zu den jeweiligen Knotenanzahlen[2] n in 2 Dimensionen auf dem Quadratgitter angegeben, sowie die relative Rechenzeit, die benötigt wurde, um diese alle zu finden. Die erhaltenen Werte aus Tab. 3.1 stimmen mit Literaturwerten [19] überein[3].

n	#Konformationen	Rechenzeit[a]
5	25	0
6	71	0
7	195	0
8	543	0
9	1 479	0
10	4 067	0
11	11 025	0
12	30 073	0
13	81 233	0
14	220 375	0
15	593 611	0
16	1 604 149	1
17	4 311 333	1
18	11 616 669	5
19	31 164 683	12
20	83 779 155	35
21	224 424 291	98
22	602 201 507	270
23	1.61114×10^9	718
24	4.31665×10^9	1979
25	1.15366×10^{10}	5457
26	3.08703×10^{10}	15001

[a] bezüglich der Rechenzeit für $n = 16$

Tabelle 3.1: Anzahl aller möglichen Konformationen (*self-avoiding random walks*) zu gegebener Anzahl der Knoten n in $2d$.

[2] Etwas verwirrend ist, daß machmal die Anzahl der Knoten oder die Anzahl der Kanten als Länge einer Kette zu lesen ist. Die Anzahl der Knoten n ist gleich der Anzahl der Kanten $L - 1$.

[3] Jedoch wurden dort nur die Konformationen gezählt, die nicht durch Spiegelung um eine Achse entstehen, das sind also (bis auf die 0. Konformationen) die Hälfte der hier gezählten.

$n=5$				$n=6$					
Nr.	Anzahl	Nr.	Anzahl	Nr.	Anzahl	Nr.	Anzahl	Nr.	Anzahl
0	3700933	13	3706048	0	1233836	25	1235935	37	1235417
1	3706276	14	3701399	1	1232121	26	1233616	38	1235088
2	3703372	15	5553242	3	1236630	27	1234683	39	1232798
3	3702664	16	5552276	4	1233987	28	1235858	40	1235288
4	3704079	17	3704925	5	1234973	29	1233008	41	1232552
5	3699766	18	3704179	9	1235207	30	1234384	42	1850632
6	3704680	19	3703776	10	1234306	31	1853035	43	1849776
7	3705127	20	5555525	11	1235364	32	1855065	44	1852574
8	3703724	21	5554842	12	1235226	33	1234075	45	1851191
9	3707912	22	3703046	13	1235228	34	1234004	46	1851851
10	3703218	23	3705204	14	1235647	35	1233637	47	1852328
11	3703701	24	3706614	15	1851170	36	1236303		
12	3703472			16	1852083				

Tabelle 3.2: Anzahl der zufälligen Entstehungen der jeweiligen Konformationen in $2d$. Die Nummern stimmen mit denen aus Abb. 3.3 und 3.4 überein.

Als nächstes habe ich jeweils 100 000 000 Konformationen mit der Knotenanzahl $n = 5$ ($L = 4$) und $n = 6$ ($L = 5$) zufällig wachsen lassen und mitgezählt, wie oft jede Konformation dabei entstanden ist. Die Ergebnisse zeigt Tab. 3.2. Die Nummern stimmen mit denen aus Abb. 3.3 und 3.4 überein. Es zeigt sich sehr deutlich, daß bestimmte Konformationen häufiger entstehen und zwar auch genau die, die nach den Überlegungen von oben eine höhere Wahrscheinlichkeit haben sollten, erzeugt zu werden. Für $n = 5$ (siehe Abb. 3.3) ist das Verhältnis der Wahrscheinlichkeiten $18/27 = 2/3$, für $n = 6$ ebenso $(54/81 = 2/3)$. Durch die Simulation wurde dieser Wert bestätigt: $370/555 = 2/3$ bzw. $123/185 \simeq 2/3$. Damit sind also obige Überlegungen bestätigt.

3.2 Zufallswege

Gehen wir jetzt zu sehr großen n, so daß man unmöglich noch alle möglichen Konfigurationen auszählen kann, so wie wir es oben getan haben.

Entsprechend den vorherigen Überlegungen zur Anzahl der Wege, die (unter bestimmten Bedingungen) auf einem quadratischen Gitter entstehen können, gilt für die Anzahl der Zufallswege auf beliebigen Gittern (mit fester Anzahl nächster Nachbarn) $c_n = z^n$, wenn n die Länge des Weges ist und jeder Gitterpunkt z Nachbarn hat, zu denen der Weg ohne jegliche Beschränkung fortgesetzt werden kann. Gibt man jetzt die Regel vor, dass ein Weg nicht wieder sofort dorthin zurücklaufen darf, wo er gerade hergekommen ist, gilt natürlich $c_n = z \cdot (z-1)^{(n-1)}$.

3.2.1 *Self-avoiding random walks*

Wie sieht nun c_n für *self-avoiding random walks* aus, d.h. daß sich der Weg auch niemals selbst überschneiden darf? Allgemein wird für diesen Fall folgender Ansatz gewählt [20, 21],

$$c_n \sim \mu^n n^{\gamma-1}\,, \tag{3.1}$$

wobei μ die Konnektivitätskonstante ist. Sie gibt die Anzahl der im Mittel freien nächsten Nachbarplätze an. Den Wert von z kann man jetzt nicht mehr so einfach sehen, deswegen muß er, neben γ, numerisch bestimmt werden. Dazu führt man zuerst folgende Größe ein:

$$r_n = \frac{c_n}{c_{n-1}} = \mu\left(\frac{n}{n-1}\right)^{\gamma-1}\,. \tag{3.2}$$

Die Taylorentwicklung um $n = \infty$ bzw. $n^{-1} = 0$ führt auf

$$r_n = \mu\left[1 + (\gamma-1)\,n^{-1} + \frac{1}{2}\,\gamma(\gamma-1)n^{-2} + \mathcal{O}\left(n^{-3}\right)\right]\,. \tag{3.3}$$

Führt man mit dieser Funktion Extrapolationen durch, kann man μ (ohne bias) und γ (mit bias) ablesen.

Es gibt eine zweite Herangehensweise an dieses Problem. Diese führt über Symmetriebetrachtungen. Die Zahl der *self-avoiding random walks* kann man wie folgt darstellen:

$$c_n = \left(c_n^{\text{linear}} s^{\text{linear}} + \sum_i c_{n,i}^{\text{planar}} s_i^{\text{planar}} + \sum_j c_{n,j}^{\text{räumlich}} s_j^{\text{räumlich}}\right)\,. \tag{3.4}$$

$c_n^{(\cdot)}$ bezeichnet hier die Anzahl der nicht durch Symmetrietransformationen ineinander überführbaren linearen Konformationen (linear), dieser Konformationen in der Ebene (planar) bzw. dieser Konformation, die sich in 3 Dimensionen ausbreiten (räumlich). $s^{(\cdot)}$ bezeichnet den entsprechenden Symmetriefaktor, d.h. die Zahl der Symmetrietransformationen (mal der jeweiligen Zähligkeit der entsprechenden Rotation), die man mindestens braucht, um alle Konfigurationen, die durch irgendwelche Symmetrietransformationen auseinander hervorgehen können, zu erzeugen. Die Summen gehen über alle Klassen von Konformationen mit unterschiedlichen Symmetriefaktoren. An den Beispielen für die jeweilen Gitter wird das klarer werden.

Zunächst kann man aber feststellen, daß $c_n^{\text{linear}} = 1$ ist für alle Gitter, s^{linear} ist genau die Anzahl der nächsten Nachbarn k.[4] Weiterhin sind die globalen Rotationen (Zähligkeit genau k) in allen Symmetriefaktoren enthalten. Man kann somit also schreiben

[4] Eine lineare Kette kann man genau k mal am Ursprung drehen, so daß unterschiedliche *walks* entstehen. Durch etwaige Spiegelungen entstehen keine zusätzlichen *walks*.

$$c_n = k \left(1 + \sum_i c_{n,i}^{\text{planar}} s_i'^{\text{planar}} + \sum_j c_{n,j}^{\text{räumlich}} s_j'^{\text{räumlich}} \right) . \tag{3.5}$$

$s'^{(\cdot)}$ enthalten dann, in 2 Dimensionen, z.B. nur noch Spiegelungen.

In Kap. A.2 werden für einige Gitter die Symmetriefaktoren ausführlich bestimmt und veranschaulicht, hier möchte ich erst einmal die Ergebnisse für verschiedene Gitter zeigen:

$$\text{(2d Quadrat)} \quad c_n = 4\,(1 + 2\,c_n^{\text{planar}})\,, \tag{3.6}$$

$$\text{(2d Dreieck)} \quad c_n = 6\,(1 + 2\,c_n^{\text{planar}})\,, \tag{3.7}$$

$$\text{(3d sc)} \quad c_n = 6\,(1 + 4\,c_n^{\text{planar}} + 8\,c_n^{\text{räumlich}})\,, \tag{3.8}$$

$$\text{(3d fcc)} \quad c_n = 12\,(1 + 4\,c_{n,1}^{\text{planar}} + 2\,c_{n,2}^{\text{planar}} + \sum_j s_j'^{\text{räumlich}} c_{n,j}^{\text{räumlich}})\,. \tag{3.9}$$

3.2.2 Ergebnisse auf verschiedenen Gittern

Die Analyse der *self-avoiding walks* auf dem d-dimensionalen einfachen hyperkubischen Gitter ($d = 2, \ldots, 5$) (und auch auf dem 2D Wabengitter) wurde ausführlich in [22] gemacht, für das einfache kubische Gitter in 3D ebenso in [7] und [20]. Die dort angegebenen Ergebnisse liefern für μ und γ (3D sc):

$$\mu \approx 4.684\,, \quad \gamma \approx 1.16\,. \tag{3.10}$$

Ich werde hier die Ergebnisse einer einfachen Analyse mit Gl. (3.3) am n-dimensionalen Dreiecksgitter ($d = 2, 3$) zeigen.

Analog zu Tab. 3.1 zeigt Tab. 3.3 die Anzahl aller *self-avoiding random walks* mit den jeweiligen Knotenanzahlen n auf dem 2D Dreiecksgitter[5]. Macht man einen Fit gemäß Gl. (3.3) an die Meßwerte aus Tab. 3.3 erhält man für das 2D Dreiecksgitter folgende Ergebnisse:

$$\begin{aligned} &\mu \approx 4.124\,, \quad \gamma \approx 1.44 \quad \text{bei Fit bis 1. Ordnung}\,, \\ &\mu \approx 4.164\,, \quad \gamma \approx 1.27 \quad \text{bei Fit bis 2. Ordnung}\,. \end{aligned} \tag{3.11}$$

Auch für das Tetraedergitter habe ich wieder die Anzahl der möglichen nicht selbstüberschneidenden Wege bestimmter Längen gezählt. Das Ergebnis zeigt auch Tab. 3.3 (vgl. auch [24]) im Vergleich zu der Anzahl aller solcher Wege auf einem kubischen Gitter in 3 Dimensionen (vgl. [7, 23]) und ist identisch mit den Ergebnissen in [24]. Die Fits ergeben für das Tetraeder- bzw. fcc-Gitter:

$$\begin{aligned} &\mu \approx 9.90\,, \quad \gamma \approx 1.31 \quad \text{bei Fit bis 1. Ordnung}\,, \\ &\mu \approx 10.019\,, \quad \gamma \approx 1.07 \quad \text{bei Fit bis 2. Ordnung}\,. \end{aligned} \tag{3.12}$$

Der zweite Wert von μ stimmt gut mit dem in [24] überein, er ist dort mit $\mu = 10.036$ angegeben.

[5] Wobei, wie vorn, die globalen Rotationen nicht gezählt wurden.

n	#Konformationen quadratisches Gitter	#Konformationen Dreiecksgitter	#Konformationen kubisches Gitter	#Konformationen Tetraedergitter
3	3	5	5	11
4	9	23	25	117
5	25	103	121	1 225
6	71	455	589	12 711
7	195	1 991	2 821	131 143
8	543	8 647	13 565	1 347 679
9	1 479	37 355	64 661	1.380809×10^{7}
10	4 067	160 689	308 981	1.411478×10^{8}
11	11 025	688 861	1 468 313	1.440161×10^{9}
12	30 073	2 944 823	6 989 025	1.467206×10^{10}
13	81 233	12 559 201	3.313882×10^{7}	1.492879×10^{11}
14	220 375	53 455 781	1.573291×10^{8}	
15	593 611	227 131 875	7.448186×10^{8}	
16	1 604 149	9.636276×10^{8}	3.529191×10^{9}	
17	4 311 333	4.082888×10^{9}	1.668698×10^{10}	
18	11 616 669	1.727899×10^{10}	7.895504×10^{10}	
19	31 164 683		3.729539×10^{11}	
20	83 779 155			
21	224 424 291			
22	602 201 507			
23	1.61114×10^{9}			
24	4.31665×10^{9}			
25	1.15366×10^{10}			
26	3.08703×10^{10}			

Tabelle 3.3: Anzahl aller möglichen Konformationen (*self-avoiding random walks*) zu gegebenen Längen n (Anzahl Knoten) in 2D auf quadratischem Gitter und Dreiecksgitter, sowie in 3 Dimensionen auf kubischem Gitter und Tetraedergitter. Der globale Symmetriefaktor k wurde jeweils schon dividiert.

Die genauen Fits und Regressionsanalysen sowie die daraus ablesbaren Zahlen für alle angegebenen Ergebnisse findet man in Kap. A.1.

Tabelle 3.4 zeigt einige der Eigenschaften der verschiedenen Gitter in einer Zusammenfassung. Ich habe den Quotient μ/k eingeführt, also die relative Konnektivität. An ihr kann man abschätzen, wie stark die Bedingung der Nichtüberschneidung ist. Auf dem 2D Quadratgitter ist die mittlere Anzahl der freien Nachbarplätze nur wenig größer als die Hälfte der insgesamt existierenden Nachbarplätze. Auf dem 3D Tetraedergitter sind es schon über 80%.

	2D Quadrat	2D Dreieck	3D Kubisch	3D Tetraeder
Koordinationszahl k	4	6	6	12
μ	2.638*	4.164	4.684*	10.019
γ	1.34*	1.27	1.16*	1.07
μ/k	0.592	0.694	0.780	0.835
maximal ausgezählte n	31*	18	24*	13
Anzahl Konform.($n_{\max}$)	1.6×10^{13}*	1.7×10^{10}	5.2×10^{15}*	1.5×10^{11}

Tabelle 3.4: Zusammenfassung wichtiger Eigenschaften der betrachteten Gitter. Werte mit * aus [22].

Die Konnektivität μ ist eine Eigenschaft der jeweiligen Gitter, wohingegen γ ein kritischer Exponent ist. Er ist grundsätzlich verschieden in verschiedenen Dimensionen, aber gleich für alle Gitter in einer jeweiligen Dimension. In 2 Dimensionen ist γ exakt bekannt [25], in 3 Dimensionen bisher noch nicht [24]:

$$\gamma_{2d} = 43/32 = 1.34375 \quad \text{(in 2 Dimensionen)}\,, \tag{3.13}$$

$$\gamma_{3d} \approx 7/6 \approx 1.17 \quad \text{(in 3 Dimensionen)}\,. \tag{3.14}$$

Man sieht durch Tab. 3.4 u.a. auch deutlich den Unterschied zwischen dem 2dimensionalen Dreiecksgitter und dem 3dimensionalen kubischen Gitter. Obwohl beide die gleiche Koordinationszahl haben, unterscheiden sie sich in der Konnektivität. Hieraus ergiben sich interessante Fragen: Sind zwei Gitter mit gleicher Koordinationszahl, gleicher Konnektivität und gleichem kritischen Exponenten γ zwangsläufig identisch? Reicht die Gleichheit zweier dieser Zahlen, um zu schlußfolgern, daß zwei Gitter identisch sind? Diese Fragen können hier nicht beantwortet werden, jedoch vermute ich, daß die Antwort zumindest auf die erste positiv ist.

Kapitel 4

Pruned-Enriched Rosenbluth Method (PERM)

4.1 „*Go with the Winners*“-Strategie

Die „*Go with the Winners*“-Strategie ist eine allgemeine Strategie, um einen Konfigurationenraum mit einer gegebenen Verteilung von Konfigurationen abzutasten. Im Gegensatz zur z.B. *Metropolis*-Strategie liegt ihr allerdings kein Markow-Prozess zugrunde.

Bei der „*Go with the Winners*“-Methode wird ein künstliches *bias* eingeführt. Z.B. kann man anstelle gleichverteilter Zufallszahlen anders gewichtete Zufallszahlen ziehen. Man kann sich etwa vorstellen, daß bei einer einfachen Simulation, bei der sich ein Räuber und ein Beutetier zufällig zueinander bewegen, bei der Beute künstlich der Hang zur Entfernung vom Räuber zugefügt wird, damit diese länger überlegt. Die Beute geht also zu einem Zeitpunkt t_i nicht mit der Wahrscheinlichkeit $p_i = 1/2$ auf den Räuber zu, sondern vielleicht nur mit $p_{i,\mathrm{bias}} = 1/3$. In der Tat ist das oft wünschenswert, damit man überhaupt hinreichend lange Statistiken (in denen die Beute überlebt) aufnehmen kann. Dieser Hang muß jedoch wieder durch eine Gewichtung der gegangenen Wege (entstandenen Population, Konformation, ...) ausgeglichen werden, sonst kommt es zu falschen Ergebnissen. Die Gewichte w_i werden so gesetzt, daß $w_i = p_i/p_{i,\mathrm{bias}}$ ist. Dann kompensieren diese Gewichte im Mittel genau den Fakt, daß die möglichen Wege der Beute nicht mit derselben Wahrscheinlichkeit gegangen werden.

Sind die Gewichte nicht optimal[1], kann man eine Populationskontrolle einführen. Wege mit sehr großen Gewichten, d.h. die stark zur Verteilung beitragen, werden vermehrt, Wege

[1] Die Gewichte sind optimal, wenn sie im Mittel alle gleich sind. Diesen Effekt kann man an dem einfachen Beispiel des Räubers und der Beute schwer erkennen. Später werden wir sehen, daß z.B. bei Gitterpolymeren die Gewichte, die dort nicht nur vom *bias* abhängen, bei einer bestimmten Temperatur im Mittel nicht variieren und eine Populationskontrolle, obwohl vorgesehen, kaum vorkommt.

mit sehr niedrigem Gewicht sterben aus. Äquivalente Begriffe sind hierzu *cloning* und *killing* oder *enrichment* und *pruning*.

4.1.1 *Cloning*

Um zu entscheiden, ob das Gewicht $W(t) = \prod_{i=0}^{t} w_i$ einer Population (eines Weges, einer Konformation, ...) an einem bestimmten Punkt t „sehr groß“ ist, legt man eine obere Schranke $W^{>}(t)$ fest. Übersteigt das Gewicht $W(t)$ einer Population diese Schranke an einem beliebigen Punkt t, werden identische Kopien der gesamten Population angelegt. Um die Verteilung richtig zu erhalten, muß das Gewicht der Population auch auf die Kopien verteilt werden. Wird eine Kopie angelegt, bekommt diese das Gewicht $W(t)/2$, das Original verbleibt ebenso mit dem Gewicht $W(t)/2$. Werden mehrere Kopien angelegt, verteilen sich die Gewichte analog. Wieviele Kopien man tatsächlich anlegt ist hier erst einmal egal. Die Anzahl der Kopien hat nur Einfluß auf die Effizienz des Algorithmus und kann sich z.B. nach dem Verhältnis $W(t)/W^{>}(t)$ richten.

4.1.2 *Killing*

Ebenso wie $W^{>}(t)$ legt man eine untere Grenze $W^{<}(t)$ für das Aussterben von Populationen mit zu geringen Gewichten fest. Damit kann man vermeiden, daß Populationen, die sowieso nur wenig zur Verteilung beitragen, übermäßig viel Rechenaufwand kosten. Allerdings kann man solche Populationen nicht einfach rigoros auslöschen, da damit deren, wenn auch kleines, Gewicht verloren gehen würde. Man läßt diese deswegen nur mit einer bestimmten Wahrscheinlichkeit, z.B. $p = 1/2$, sterben. So können Populationen, die dennoch überleben, die Gewichte der aussterbenden Populationen „aufsammeln“. Eine Population die überlebt bekommt also die Gewichte aller im Mittel für sie ausgelöschten Populationen mit. Ihr Gewicht ist dann $p^{-1}W(t)$. [2]

Ähnlich der Anzahl der Kopien, die man anlegen kann, hat auch hier die Wahl der Überlebenswahrscheinlichkeit keine prinzipielle Bedeutung. Durch eine günstige Wahl kann man i.a. aber die Effizienz beeinflussen.

4.1.3 Die Wahl der Grenzen

Die tatsächliche Wahl der Grenzen $W^{>}(t)$ und $W^{<}(t)$ ist eines der zentralen Probleme der beschriebenen Methode. Prinzipiell kann man deren Wert beliebig wählen, ohne an der Statistik etwas falsch zu machen. In der Tat werden die Grenzen sogar dynamisch während

[2] In dem Moment, in dem eine Kette stirbt bzw. überlebt, geht deren Gewicht verloren bzw. kommt zusätzlich noch einmal hinzu. Dadurch wird in dem Moment die Verteilung geändert. Im statistischen Mittel aber, d.h. wenn hinreichend viele Ketten überlebt haben, wird dieser Effekt durch den beschriebenen Mechanismus wieder ausgeglichen und die richtige Verteilung bleibt erhalten. *Killing* ist also kurzfristig nicht richtig, wird es jedoch im statistischen Mittel.

der Simulation angepaßt. Die Effizienz des Algorithmus reagiert jedoch sehr sensibel auf diese Wahl. Man kann zwei grundsätzliche Forderungen stellen [26]:

1. Das Verhältnis $W^{>}(t)/W^{<}(t)$ sollte nicht zu groß sein. So werden zu starke Fluktuationen der Gewichte vermieden. In meinen Simulationen verwende ich immer $W^{<}(t) = 0.2\,W^{>}(t)$. Wie man später sehen wird, ist das manchmal aber recht unpraktikabel.

2. $W^{>}(t)$ sollte so gewählt werden, daß die Anzahl der durch Klonen entstandenen Populationen (Wege, Konformationen, ...) pro Zeitschritt unabhängig von der Zeit selbst ist. Damit verhindert man einerseits, daß die meiste Rechenzeit für Populationen bei kleinen Zeiten verwendet wird und kaum Populationen bei großen Zeiten zur Statistik beitragen, andererseits, daß viele Populationen bei großen Zeiten aus sehr wenigen Populationen bei kleinen Zeiten hervorgehen und deswegen alle sehr stark korreliert sind.

Ich habe mich an den Vorschlag aus [27] gehalten, der dies realisiert: $W^{>}(t) \sim \hat{Z}(t)$, wobei $\hat{Z}(t)$ ein Schätzer für die Zustandssumme sein soll.

4.1.4 Anwendung auf Gitterpolymere

Ein *chain growth* Algorithmus zur Simulation flexibler Gitterpolymere wurde 1997 von Grassberger entwickelt [27]. Dieser ist ein „*Go with the Winners*“-Algorithmus. Er kombiniert die Rosenbluth–Rosenbluth Methode mit Populationskontrolle durch Klonen und Aussterben von Konfigurationen, die hier die Rolle der Population einnehmen. Wie wir bereits in Kap. 3.1.1 gesehen haben, wird das *bias* hier durch die Art der Erzeugung der Konformationen als *self-avoiding walk* induziert. Bestimmte Konformationen enstehen dadurch wahrscheinlicher als andere, was prinzipiell unerwünscht ist. Die Gewichte, die dies ausgleichen, sind die Rosenbluth-Gewichte $w_{\mathrm{Rose}} = 1/p_i$, wenn p_i die Wahrscheinlichkeit ist, daß eine Konformation aus einer anderen zur Zeit t_i hervorgeht. Wie wir auch gesehen haben, entspricht $1/p_i$ genau der Anzahl freier Nachbarplätze m_i des Monomers, an dem das Gitterpolymer fortgesetzt wird.

Hinzu kommen bei PERM (P*runed*-E*nriched* R*osenbluth* M*ethod*) noch die Boltzmann-Gewichte $w_{\mathrm{Boltz}} = \mathrm{e}^{-E_i/k_{\mathrm{B}}T}$, die die Energie der Konformation zum Zeitpunkt t_i nach (1.12) und die Temperatur[3] enthalten. Das Gewicht eines wachsenden Gitterpolymers ist also zum Zeitpunkt t

$$W(t) \sim \prod_{i=0}^{t} w_i = \prod_{i=0}^{t} \left(w_{\mathrm{Rose},i} \cdot w_{\mathrm{Boltz},i}\right) , \tag{4.1}$$

$$w_{\mathrm{Rose},i} \cdot w_{\mathrm{Boltz},i} = m_i \cdot \mathrm{e}^{-E_i/k_{\mathrm{B}}T} . \tag{4.2}$$

[3] Die Temperatur kann hier auch als Güte des Lösungsmittels in dem sich ein Polymer befindet interpretiert werden. Hohe Temperaturen entsprechen einem guten Lösungsmittel, tiefe Temperatuen einem schlechten.

4.1.5 Probleme

Zwei gewichtige Nachteile der Methode sollen hier aufgeführt werden:

1. Die Methode erzeugt viele stark korrelierte Populationen. Diese können nicht quantitativ kontrolliert werden. Die stark korrelierten Populationen sind genau diejenigen, die aus einem Original als Kopien hervorgehen. Man muß deswegen sehr genau die Parameter kontrollieren, mit denen man die Anzahl der Kopien einer Originalpopulation steuern kann. Ob und wie das gelingen kann, werden wir sehr viel später sehen.

2. Die Effizienz des Algorithmus ist sehr stark abhängig von Parametern wie z.B. $W^{>}(t)$. Sind diese ungünstig eingestellt, kann es sein, daß der Konfigurationsraum sehr einseitig abgetastet wird und man große Teile dessen verliert oder Fluktuationen auf großen Zeitskalen „übersehen“ werden.

Auf der anderen Seite können sich diese Nachteile auch in Vorteile umkehren. Durch die große Anzahl von Freiheitsgraden in der Implementation kann der Algorithmus prinzipiell auf jedes Problem sehr spezifisch und hoch optimiert werden. Gelingt dies, kann der Effizienzgewinn gegenüber z.B. metropolisartigen Verfahren enorm sein.

4.2 Der Algorithmus

Hier soll nun der ursprüngliche PERM-Algorithmus (urPERM) sehr detailliert beschrieben werden. Es wird auch auf die konkrete Umsetzung eingegangen und erste Varianten, ihn zu modifizieren. In späteren Kapiteln werden eine neuere Version von PERM (nPERM) sowie weitere Modifikationen beschrieben werden. Ausgewählten Quellcode dazu findet man in Kap. A.3.

Der Algorithmus wurde von mir als rekursive Tiefensuche implementiert. Die Gitterpolymere entstehen durch Aneinanderreihung der einzelnen Monomere (Aminosäuren). Nach jedem Anfügen wird kontrolliert, ob das Gewicht der bisherigen Aminosäurekette die obere Grenze übersteigt, die untere Grenze unterschreitet, oder einfach alles „innerhalb normaler Parameter“ verläuft. Dann wird entweder eine Kopie der Kette angelegt, die Kette stirbt aus oder es geht einfach weiter mit dem Wachstum.

Etwas intuitiver kann man vielleicht formulieren: Lasse das Polymer (Protein) wachsen und kontrolliere ständig, ob ein „gutes“ Resultat zu erwarten ist[4]. Falls ja, lege viele Kopien an, falls nein, verwerfe die Kette mit einer gewissen Wahrscheinlichkeit und fange von vorn an bzw. arbeite an vorherigen Kopien weiter. Warum das äquivalent zu obiger Formulierung ist, sollte bald klar werden.

[4]Bei der Suche nach dem energieärmsten Zustand (dem Grundzustand) heißt „gut“ etwa, daß die Energie der aktuellen Kette schon sehr niedrig ist.

Das Kettenwachstum beginnt mit der Initialisierung eines Anfangspunktes. Mit dieser Initialisierung beginnt eine sogenannte *tour*. Eine *tour* endet, wenn alle Kopien bei maximaler Kettenlänge angelangt bzw. ausgestorben sind. Dann beginnt eine neue *tour* mit der Initialisierung des Anfangspunktes. Die Grenzen $W^{<,>}$ werden von der vorhergehenden *tour* übernommen, in der sie i.a. aktualisiert werden.

4.2.1 Initialisierung

Während des Wachstums der ersten Konfiguration sollen weder Kopien angelegt werden, noch die Konfiguration aussterben. Die untere Grenze für die erste *tour* setze ich somit $W^{<}(l) = 0$, die obere (idealerweise $W^{>} = \infty$) z.B. $W^{>}(l) = 10^{1000}$ für alle l. Damit wird sichergestellt, daß die erste Konfiguration ein Zufallsweg ist. Solange sich dieser Weg nicht selbst in eine Sackgasse (sog. *attrition points*) führt, entsteht in der ersten *tour* genau eine Konfiguration, sonst keine.

Als nächstes initialisiere ich z.B. das Gitter[5] und setze das erste Monomer (Startpunkt) an einen beliebigen Gitterpunkt. Das Gewicht w_0 wird 1 gesetzt.

Weiterhin lege ich einen Zähler $c(l)$ an, welcher zählt, wie oft die Kettenlänge l jemals erreicht worden ist und Einfluß auf die Dynamik der Grenzen $W^{<,>}$ haben wird. $c(0)$ wird nun auf 1 gesetzt.

4.2.2 Rekursion

Das Wachstum geschieht durch eine Funktion (`step(l)`) die rekursiv aufgerufen wird und jeweils ein Monomer an die bestehende Kette anfügt, die Grenzen $W^{<,>}$ aktualisiert, sowie Kopien anlegt bzw. das Wachstum stoppt. Man befindet sich zu einem beliebigen Zeitpunkt an einem bestimmten Gitterpunkt zur Länge l, bei Rekursionstiefe 0 ist das der gerade initialisierte Startpunkt $l = 0$. Nun läuft das Wachstum wie folgt:

- Ermittle die freien Nachbarn des aktuellen Gitterpunktes und wähle zufällig einen davon aus.
- Gehe zu diesem Punkt und berechne E_{l+1}, w_{l+1} sowie $W(l+1)$ unter der Annahme, daß das Monomer $l+1$ sich schon dort befindet.

a) $W(l+1) > W^{>}(l+1)$:

   - Setze das Monomer $l+1$.
   - Erhöhe den Zähler $c(l+1)$.
   - Berechne den neuen Schätzer für die Zustandssumme wie folgt:

$$\hat{Z}_{\text{neu}}(l+1) = \hat{Z}_{\text{alt}}(l+1) + W(l+1)\,. \tag{4.3}$$

[5] Später werde ich mich von einem statisch implementierten Gitter lösen, da es für große Ketten zu viel Speicher benötigt.

– Aktualisiere die Grenzen $W^{<,>}$ wie folgt:

$$W^{>}(l+1) = \frac{\hat{Z}(l+1)}{\hat{Z}(0)} \left(\frac{c(l+1)}{c(0)}\right)^2 ,$$
$$W^{<}(l+1) = 0.2\, W^{>}(l+1) . \tag{4.4}$$

– Anlegen der Kopien:

- Lege die Anzahl der Kopien k fest.
- Rufe `step(l+1)` k mal auf.

– Mit dem Original fortfahren[6]: Rufe `step(l+1)` auf.

b) $W(l+1) < W^{<}(l+1)$:

– Ziehe Zufallszahl `zufall` $\in [0,1]$.

– Ist `zufall`< 0.5 mache nichts, Kette stirbt aus.

– Ist `zufall`> 0.5 fahre fort wie in a) ohne den Schritt, in dem Kopien angelegt werden.

c) $W^{<}(l+1) < W(l+1) < W^{>}(l+1)$:

– Mache alles wie in a), ohne Kopien anzulegen.

Für die Wahl der neuen Grenzen $W^{<,>}$ kann prinzipiell jede beliebige Funktion gewählt werden. In [27] wird z.B. anstelle von Gl. (4.4) $W^{>}(l+1) = c^{>}\, \hat{Z}(l+1)$ verwendet, wobei $c^{>}$ eine Konstante der Ordnung 1 ist. Die Anzahl der Kopien ist hier zunächst 1. Sie kann aber z.B. auch (const. $\cdot\, W/W^{>}$) oder $\min[W/W^{>}, m_i]$ gewählt werden.

4.2.3 Abbruch

Wird `step(l)` mit $l = n$ aufgerufen, hat die Kette ihre komplette Länge erreicht. In dem Fall wird die Rekursion nicht weiter fortgesetzt. Nun kann man alle interessanten Observablen der entstandenen Konformation messen, etwa die Gesamtenergie des Proteins, den geometrischen Abstand des ersten Monomers vom letzten usw. Ebenso habe ich jeweils die Koordinaten der einzelnen Monomere gespeichert, um die Konformation später visualisieren zu können.

Danach wird eine neue *tour* initialisert, d.h. alle Parameter wie z.B. $c(l)$ oder $\hat{Z}(l)$ (außer $W^{<,>}$) werden zurückgesetzt und ein neuer Anfangspunkt gesetzt, und gestartet. Die gesamte Simulation kann nach einer vorgegebenen Anzahl von Touren, nach einer vorgegebenen Zeit oder z.B. nach einer vorgegebenen Anzahl der Wiederkehr der Energie des vermuteten Grundzustandes abgebrochen werden.

[6] Hier wird wegen der besseren Beschreibung noch zwischen Kopien und Original unterschieden. Im Algorithmus werden Original und Kopien natürlich völlig gleich behandelt.

4.3 Erste Erkenntnisse

Für erste Versuche mit urPERM benutzte ich ein HP-Protein aus [19] mit $n = 25$ auf dem Quadratgitter und eindeutigem Grundzustand (*designing sequence*) mit der Energie $E_{\min} = -13$:

$$\text{PHPHPHPHPPHPHPHPPHPPHHHHH}\,. \qquad (\text{Seq}\,25_1)$$

Der Grundzustand dieser Sequenz ist in Abb. 4.1 (**links**) dargestellt. Nun ist es ein Unter-

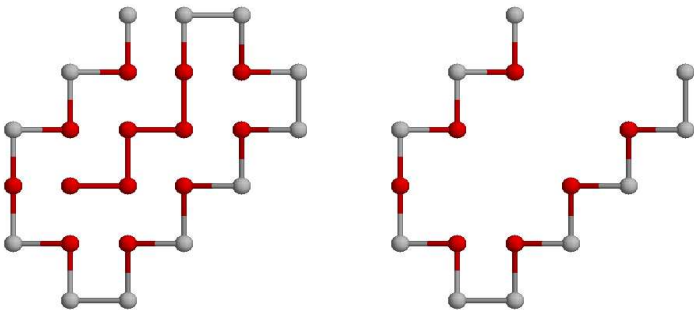


Abbildung 4.1: **Links:** Eindeutiger Grundzustand der Sequenz $(\text{Seq}\,25_1)$, $E = -13$. **Rechts:** Läßt man $(\text{Seq}\,25_1)$ von links wachsen, muß sie auf dem Weg zum Grundzustand die dargestellte Konformation durchlaufen. Das ist äußerst unwahrscheinlich.

schied, ob man bei der Simulation die Sequenz $(\text{Seq}\,25_1)$ von links oder von rechts wachsen läßt. Bei ersterer Variante muß die Sequenz bei ihrem Wachstum einen äußerst instabilen Zustand durchlaufen, will sie zum Grundzustand gelangen. Diese ist in Abb. 4.1 (**rechts**) zu sehen. Das Gewicht des Zustanden ist aufgrund der fehlenden Kontakte so gering, daß er (fast) immer während seiner Entstehung ausstirbt. Die Simulationsdaten zeigt Tab. 4.1. Man sieht, daß wenn das Wachstum rechts beginnt, der Grundzustand in weniger als 20 000

Wachstum von links			Wachstum von rechts		
Zeit	$E_{\min}$	Touren	Zeit	$E_{\min}$	Touren
0	-1		0	-1	
0	-2		0	-2	
0	-3		0	-3	
0	-4		0	-4	
0	-5		0	-5	
0	-6		0	-6	
0	-7		2	-7	
1	-8		8	-8	
6	-9		11	-9	
43	-10	$< 10\,000$	13	-10	
			15	-11	$< 10\,000$
			63	-13	$< 20\,000$
3 000	—	$> 500\,000$	Grundzustand erreicht		

Tabelle 4.1: Die Tabelle zeigt die relative Zeit, die benötigt wurde, um einen neuen Zustand niedrigerer Energie zu finden (Bei der Angabe 0 war die Zeit mit der gegebenen Genauigkeit nicht meßbar). Rechts wird der Grundzustand mit $E = -13$ gefunden, links nicht.

Touren gefunden wird. Läßt man die Sequenz von links wachsen, wird der Zustand mit der Energie $E = -10$ (siehe Abb. 4.2) in der gleichen Zeitgrößenordnung und weniger als 10 000 Touren gefunden. Der Grundzustand wurde nach etwa 50facher Zeit, die für das Finden des Grundzustandes beim Wachstum von rechts benötigt wird, und über 500 000 Touren immer noch nicht gefunden. Dies bestätigt die Vermutung von oben.[7]

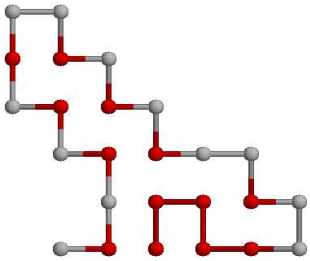


Abbildung 4.2: Der Zustand mit der Energie $E = -10$, der beim Wachstum von links in etwa gleicher Zeit gefunden wird wie der Grundzustand beim Wachstum von rechts (sieh Tab.4.1). Der Zustand befindet sich in einem bzgl. des Algorithmus' sehr stabilen lokalen Energieminimum, so daß der wahre Grundzustand innerhalb der nächsten 500 000 Touren nicht mehr gefunden wurde.

4.4 nPERM

Ein großer Nachteil der oben beschriebenen Implementation von PERM ist das Verhalten der Kopien bzw. Klone bei niedrigen Temperaturen. Es zeigt sich, daß Kopien von Ketten sich sehr oft in die gleiche Richtung weiterentwickeln wie ihre Originale. Der Vorteil durch das *enrichment* der Konfigurationen im Speicher durch Klonen wird dadurch bei niedrigen Temperaturen fast zunichte gemacht[8].

Der hauptsächliche Unterschied zwischen der neuen Version von PERM (nPERM) und der ursprünglichen ist nun der, daß bei Anlegen der Kopien jede Kopie schon weiß, in welche Richtung sie weiterwachsen soll. So kann sichergestellt werden, daß alle Kopien in unterschiedliche Richtungen weiterwachsen.

Als Kriterium für das Klonen oder Terminieren von Ketten wird nicht mehr das aktuelle Gewicht verwendet, sondern ein vorhergesagtes (*predicted*) Gewicht $W_{\mathrm{pred}}(l+1)$ für den nächsten Schritt, welches mit $W^{<,>}$ verglichen wird.

$$W_{\mathrm{pred}}(l+1) = W(l) \cdot m_i \,, \tag{5}$$

wobei m_i wieder die Anzahl der freien nächsten Nachbarn des aktuellen Monomers ist.

Die Anzahl der Kopien k richtet sich nach dem Verhältnis des vorhergesagten Gewichts zur oberen Schranke, ist jedoch maximal die Anzahl der freien Nachbarplätze. Das „Original“ wird im folgenden wie eine Kopie behandelt, da es sich von diesen in keinem Punkt unterscheidet,

$$k = \min\left[m_i, \frac{W_{\mathrm{pred}}(l+1)}{W^{>}(l+1)}\right] . \tag{6}$$

[7] Eine Zeiteinheit ist hier real eine Sekunde. Die Simulation liefen auf einem PII 300MHz PC.

[8] Bei hohen Temperaturen ist der Effekt vernachlässigbar, da die Vielfalt der Konformationen durch thermische Fluktuationen erzeugt wird. Kopien entwickeln sich schon dadurch in eine andere Richtung als ihr Original.

Das nächste Monomer wird nun für jede Kopie auf einen anderen freien Nachbarplatz gesetzt und `step(l+1)` aufgerufen. Da die Anzahl der Kopien bzw. die Anzahl der Ketten mit denen fortgefahren wird, niemals größer als die Anzahl freier Nachbarplätze ist, werden beim Klonen jetzt niemals identische Ketten erzeugt.

4.4.1 Erste Ergebnisse

Angewendet habe ich nPERM zuerst auf die zehn 48mere aus [28]. Das erste von ihnen hat die Sequenz

$$\text{HPHHPPHHHHPHHHPPHHPPHPHHHHPHPHHPPHHPPPHPPPPPPPPHH} \quad (\text{Seq}\,48_1)$$

und die Grundzustandsenergie $E = -32$ auf dem kubischen Gitter in 3 Dimensionen. Alle in [28] angegebenen Grunzustände werden von nPERM ohne Probleme nach relativ kurzer Zeit gefunden, was auch schon in [29] berichtet wird. Alle anderen Sequenzen sowie Darstellungen eines Grundzustandes jeder Sequenz findet man in Kap. B.2.

Um meine Implementation erstmals zu prüfen, habe ich meine Rechenzeiten mit den Werten aus [29] verglichen. Die Ergebnisse zeigt Tab. 4.2.

Seq.	Rechenzeit hier	Rechenzeit aus [29]	Prozessor hier
1	40,3 s	39,6 s	PII 350MHz
2	141,8 s	287,4 s	PIII 733MHz
3	596,0 s	263,4 s	PII 350MHz
4	5386 s	1170 s	PII 350MHz
5	634,6 s	412,8 s	PII 350MHz
6	758,5 s	568,8 s	PIII 500MHz
7	518,6 s	459,0 s	PIII 500MHz
8	416,9 s	175,2 s	PPro 200MHz
9	9696 s	22718 s	PIII 500MHz
10	371,2 s	53,4 s	PPro 200MHz

Tabelle 4.2: Vergleich der Wiederkehrzeiten des Grundzustandes der Sequenzen aus [29] mit den dort angegebenen Werten. Gezeigt ist hier der Mittelwert der Wiederkehrzeit nach 10^9 Touren. Die Werte aus [29] wurden auf einer 167 MHz Sun ULTRA I gemessen.

Man kann die Werte natürlich nicht direkt vergleichen, da alle Simulationen auf verschiedenen Rechnern liefen, trotzdem habe ich die Zeiten und den Rechnertyp aufgelistet, da man trotzdem qualitative Informationen gewinnen kann. Man sieht einerseits, daß die Rechenzeiten größenordnungsmäßig übereinstimmen und in welcher tatsächlichen zeitlichen Region sie liegen. Man sieht auch, daß bestimmte Ketten (hier vor allem Sequenz 4 und 9) „schwieriger“ sind, unabhängig von der Implementation.

4.4.2 Untersuchung des Algorithmus I

Die Performance meines Algorithmus zeigt Abb. 4.3 anhand der Wiederkehr des Grundzustandes der Sequenz (Seq 48_1) bei 2 verscheidenen Temperaturen, $T = 0.2$ und $T = 0.3$. Die Zeiten sind nicht mit denen aus Tab. 4.2 vergleichbar, da die Simulationen noch einmal auf einem anderen Prozessor liefen.

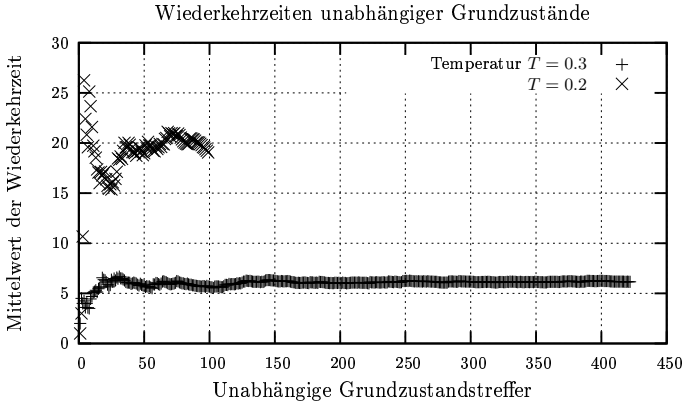


Abbildung 4.3: Perfomance meines nPERM. Der Plot zeigt den Mittelwert der Wiederkehrzeit (in s) unabhängiger Grundzustände bei verscheidenen Temperaturen.

Die Bilder in Abb. 4.4 zeigen die akkumulierten Gewichte erfolgreich beendeter Ketten derselben Sequenz zu bestimmten Energien (Histogramm, **links**) sowie die Anzahl der erfolgreich entstandenen Ketten mit bestimmten Energien („nacktes“ Histogramm, **rechts**) bei Temperaturen zwischen $T = 0.2$ und $T = 1.0$.

Man sieht darin schon, daß bei den Temperaturen $T = 0.2$ und $T = 0.3$ die meisten Ketten mit der Grundzustandsenergie gefunden werden. Eine genauere Untersuchung der Abhängigkeit der gefundenen Ketten mit Grundzustandsenergie zeigt Abb. 4.5. Auch hier sieht man, daß zwischen $T = 0.2$ und $T = 0.3$ die meisten Grundzustände gefunden werden. Bei Temperaturen über $T = 0.4$ wurden innerhalb der 10^8 Touren überhaupt keine Grundzustände gefunden.

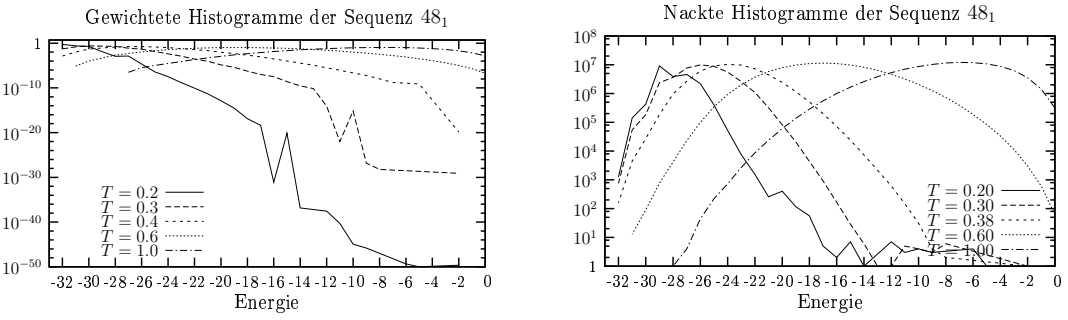


Abbildung 4.4: Histogramme der Sequenz 48_1 bei verschiedenen Temperaturen. **Links** das Histogramm der Gewichte, **rechts** das „nackte“ Histogramm der Treffer der jeweiligen Energie. Die Ergebnisse stammen aus Simulationen mit jeweils 10^8 Touren.

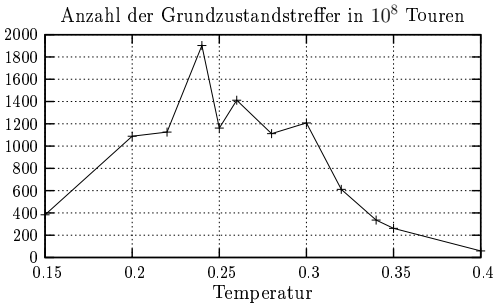


Abbildung 4.5: Anzahl der gefundenen Grundzustände der Sequenz ($\mathrm{Seq}\,48_1$) mit nPERM während 10^8 Touren in Abhängigkeit von der Temperatur.

Bei $T = 0.2$ ist aber die Performance des Algorithmus wesentlich schlechter als bei $T = 0.3$. Es gibt also für die Grundzustandssuche bei HP-Proteinen eine optimale Temperatur, bei der der Grundzustand einerseits sehr oft gefunden wird, andererseits der Algorithmus auch noch befriedigend schnell und flexibel arbeitet. Diese optimale Temperatur ist von der simulierten Sequenz abhängig, wie man auch an den Fakten in Kap. 4.3 sieht. Die Ergebnisse dort sind bei der Temperatur von $T = 0.3$ entstanden. Hätte man die Temperatur höher eingestellt, wäre man wahrscheinlich schneller wieder aus dem lokalen Minimum, das Abb. 4.2 zeigt, herausgekommen.

Für jede zu untersuchende Sequenz jedoch erst die optimale Temperatur zu bestimmen ist allerdings sehr aufwendig und wahrscheinlich wenig sinnvoll. Ich habe deswegen für alle Untersuchungen dieser Art die Temperatur $T = 0.3$ gewählt. In [29] wurde eine ähnliche Temperatur[9] als feste Temperatur gewählt, obwohl auch dort vermerkt ist, daß man die CPU-Zeit verbessern kann, wenn man für jede Kette eine eigene Temperatur einstellt.

Weitere Untersuchungen zur Temperaturabhängigkeit der Grundzustandssuche findet man auch noch einmal in Kap. 5.2.2.

[9] Jedoch angegeben als Boltzmannfaktor: $\exp(1/T) = 18$, was einer Temperatur von $T \approx 0.35$ entspricht.

Kapitel 5

Simulation von HP-Proteinen

5.1 Kurze Proteine auf verallgemeinerten Gittern

5.1.1 Exakte Enumeration

Für erste Untersuchungen von HP-Proteinen auf verschiedenen Gittertypen habe ich sehr kurze Proteine benutzt, da ich dafür noch exakt die Zustandsdichte und somit z.B. die Wärmekapazität bestimmen kann. Tabelle 3.3 zeigt u.a., bis zu welchen Proteingrößen mir das maximal möglich war.

Ich wähle zuerst ein Protein mit 10 Monomeren (Seq 10_1) und vergleiche dessen Zustandsdichte (und somit u.a. auch die Grundzustandsentartung) sowie die Wärmekapazität ganz allgemein auf dem Dreiecksgitter in 2 Dimensionen (2D), dem sc-Gitter in 3 Dimensionen (3D) sowie dem fcc-Gitter in 3D. Besonders interessant sind natürlich die Vergleiche der Eigenschaften von Proteinen auf Gittern mit teilweise gleichen Eigenschaften. So z.B. der Vergleich zwischen Gittern in gleicher Dimension (und somit mit gleichem kritischen Koeffizienten γ) oder etwa zwischen Gittern mit gleicher Koordinationszahl (und unterschiedlicher Dimension). Die Sequenz ist folgende:

$$\mathrm{HPHHHHHPHP}\,. \qquad (\mathrm{Seq}\,10_1)$$

Abbildung 5.1 zeigt drei Grundzustände dieser Sequenz auf den drei erwähnten Gittern, Abb. 5.2 zeigt die Zustandsdichten bzw. die Wärmekapazitäten der Proteine auf den entsprechenden Gittern.

An der Zustandsdichte sieht man direkt, daß der Grundzustand von (Seq 10_1) auf dem 2D Dreiecksgitter sehr schwach entartet ist, was man sich auch vorstellen kann, wenn man sich den Zustand in Abb. 5.1 ansieht. Die hydrophoben Monomere sind kompakt in einem Sechseck angeordnet, so daß sie so die optimale Anzahl an Kontakten bilden.Jede Konformation,

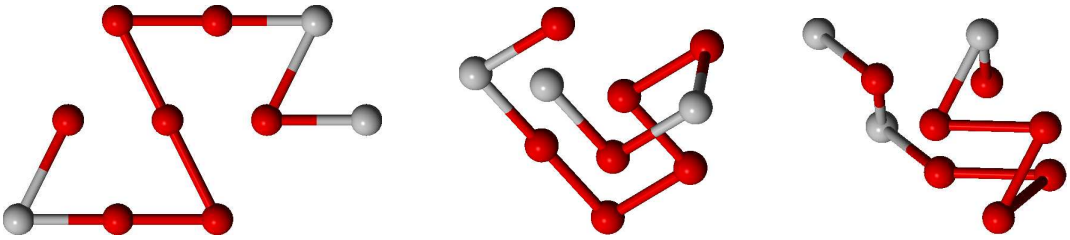


Abbildung 5.1: **Links** Ein Grundzustand der Sequenz (Seq 10_1) auf dem Dreiecksgitter in 2D, **Mitte** auf dem sc-Gitter in 3D und **rechts** ein Grundzustand derselben Sequenz auf dem Tetraedergitter in 3D.

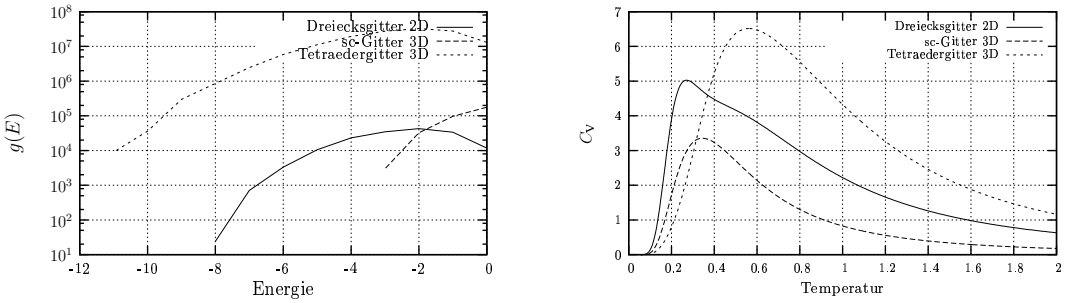


Abbildung 5.2: **Links** Die Zustandsdichten der Konformationen von (Seq 10_1) auf verschiedenen Gittern. Der Grundzustand auf dem 2D Dreiecksgitter ist sehr schwach entartet, im Gegensatz zu den Grundzuständen auf den anderen Gittern. **Rechts** sind die aus den Zustandsdichten berechneten Wärmekapazitäten gezeigt. Die Wärmekapazität auf dem 2D Dreiecksgitter zeigt andeutungsweise einen Doppelpeak.

in der sie nicht in diesem Sechseck angeordnet sind, hat weniger HH-Kontakte. Tabelle 5.1 zeigt die Entartung der Grundzustände auf den verschiedenen Gittern.

An der Wärmekapazität stellt man keinen qualitativen Unterschied zwischen den Gittern in 3 Dimensionen fest. Beide haben einen deutlichen *single peak*, wenn dieser auch in der Lage verschoben ist. Der θ-Übergangspunkt (siehe Kap. 6) hängt offenbar vom Gittertyp ab.

	2D Dreieck	3D Kubisch	3D Tetraeder
Grundzustandsenergie E_0	-8	-3	-11
Entartung des Grundzustandes $g(E_0)$	24	3 000	9 000
Anzahl Konformationen	160 689	308 982	1.411478×10^8
Anteil der Grundzustände	0.000 149	0.009 709	0.000 064

Tabelle 5.1: Grundzustandsenergie und Grundzustandsentartung der Sequenz (Seq 10_1) auf verschiedenen Gittern. Die Grundzustandsentartung und Anzahl der Konformationen wurde wieder durch den globalen Symmetriefaktor k geteilt (siehe Kap. 3.2.1). Ebenso dort hatte man gesehen, daß der verbleibende Symmetriefaktor auf dem 2D Dreiecksgitter $s'^{\,\text{planar}} = 2$ ist. Das heißt, es existieren 12 unabhängige Grundzustände von (Seq 10_1) auf diesem Gitter. Da man diese nicht sofort sieht, aber relativ leicht aufzeichnen kann, sind sie in Kap. B.1 gezeigt.

Als nächstes schauen wir uns zwei Proteine an, welche auf dem kubischen Gitter in 3D *designing sequences* haben [7]:

$$\mathrm{HHHPHPHHPHPH}\,, \qquad (\mathrm{Seq}\,12_1)$$
$$\mathrm{HHPHPHHPHPHH}\,. \qquad (\mathrm{Seq}\,12_2)$$

Was passiert mit dieser Eigenschaft auf einem anderen Gitter in gleicher Dimension? Wir halten also γ fest, ändern aber die Konnektivität μ. Abbildung 5.3 zeigt die Grundzustände dieser Proteine auf dem 3D sc-Gitter und dem 3D fcc-Gitter, Abb. 5.4 zeigt die Zustandsdichten sowie die Wärmekapazitäten.

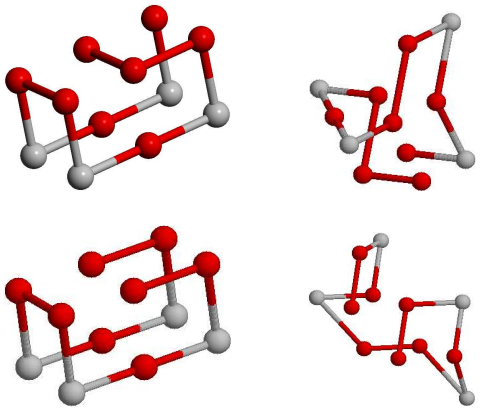


Abbildung 5.3: Grundzustände der Sequenzen ($\mathrm{Seq}\,12_1$) (**oben**) und ($\mathrm{Seq}\,12_2$) (**unten**) auf dem sc-Gitter in 3D (**links**) und dem fcc-Gitter in 3D (**rechts**). Die Sequenzen sind *designed* auf dem sc-Gitter und haben dort die Grundzustandsenergie $E = -7$. Auf dem fcc-Gitter haben sie die Grundzustandsenergie $E = -15$.

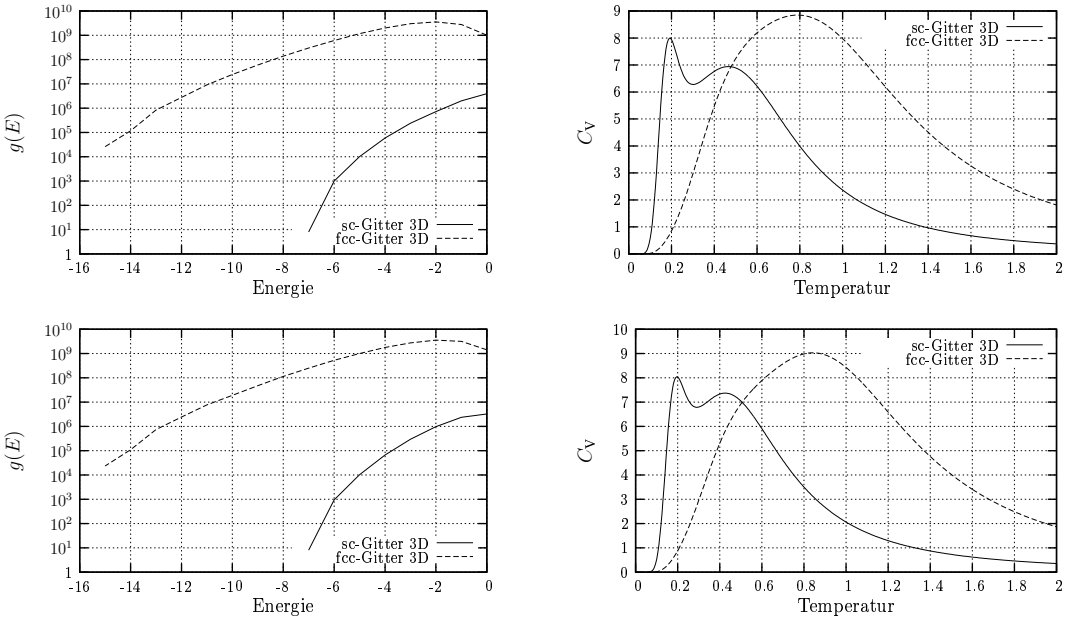


Abbildung 5.4: **Links** Die Zustandsdichten der Sequenzen ($\mathrm{Seq}\,12_1$) (**oben**) und ($\mathrm{Seq}\,12_2$) (**unten**) sowie **rechts** die dazugehörigen Wärmekapazitäten. Diese weisen auf dem 3D sc-Gitter einen Doppelpeak auf, der auf dem 3D fcc-Gitter nicht vorhanden ist. ($\mathrm{Seq}\,12_1$) und ($\mathrm{Seq}\,12_2$) sind auf dem 3D sc-Gitter *designed*. Bis auf Spiegelsymmetrien ist ihr Grundzustand dort nicht entartet.

Die Wärmekapazitäten auf dem 3D sc-Gitter zeigen einen für *designing sequences* typischen, relativ scharfen Peak bei niedrigen Temperaturen[1] [7, 32]. Dieser ist in den Wärmekapazitäten auf dem fcc-Gitter nicht vorhanden, für (Seq 12_2) (Abb. 5.4 rechts unten) kann man ihn höchstens erahnen. Daraus und natürlich auch aus den Zustandsdichten kann man sehen, daß die Eigenschaft *designing* i.a. nur für ein bestimmtes Gitter gilt. Die Konnektivität μ der Gitters ist also entscheident für die Grundzustandsentartung. Die Grundzustandsentartungen der Sequenzen (Seq 12_1) und (Seq 12_2) auf dem fcc-Gitter sind $g(E_0) = 26\,116$ bzw. $g(E_0) = 23\,520$.

Betrachen wir zuletzt noch eine Sequenz, deren Grundzustand weder auf dem 2D Dreicksgitter, noch auf dem 3D sc-Gitter nur schwach oder gar nicht entartet ist. Die beiden Gitter eignen sich für solch einen Vergleich, da auf beiden die Anzahl aller Konformation etwa gleich ist (siehe Tab. 3.3, 9.6×10^8 für das 2D Dreiecksgitter bzw. 3.5×10^9 für das 3D sc-Gitter). Ebenso haben beide Gitter die gleiche Koordinationszahl, die Konnektivität und der kritische Exponent γ sind jedoch verschieden. Die Sequenz ist

$$\text{PHPPPPHPPHHHPPHP}\,. \qquad (\text{Seq}\,16_3)$$

Abbildung 5.5 zeigt jeweils einen Grundzustand dieser Sequenz auf den beiden Gittern, Abb. 5.6 zeigt die Zustandsdichten und die Wärmekapazitäten. Im Verlauf der Wärmekapazität sieht man nun keinerlei qualitativen Unterschied mehr. Auf beiden Gittern ist deutlich ein einfacher Peak zu sehen. Die Grundzustandsentartung auf dem 3D sc-Gitter ist $g(E_0) = 37\,196$ und auf dem 2D Dreicksgitter ist $g(E_0) = 3\,560$.

Der Gittertyp beeinflußt, auch unabhängig von der Dimension, in der Regel nicht das qualitative Verhalten von Gitterproteinen. Nur in Ausnahmefällen hat der Gittertyp Einfluß auf die Übergänge von Proteinen. Jedoch sind diese „Ausnahmefälle“ genau die, nicht nur vor dem biologischen Hintergrund, daß funktionelle Proteine eindeutige Grundzustände haben sollten, welche interessant sind.

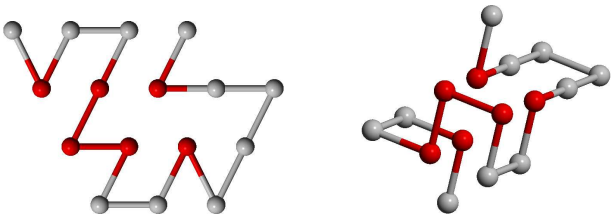


Abbildung 5.5: **Links** Ein Grundzustand der Sequenz (Seq 16_3) auf dem Dreiecksgitter in 2D. **Rechts** Ein Grundzustand dieser Sequenz auf dem 3D sc-Gitter.

[1] In [32] werden die beiden Peaks der Wärmekapazität interpretiert als die Übergänge zwischen den Grundzuständen mit kompakten hydrophoben Kernen und den maximal kompakten Zuständen bzw. den maximal kompakten Zuständen und den *random coils*, also ausgestreckten Zuständen.

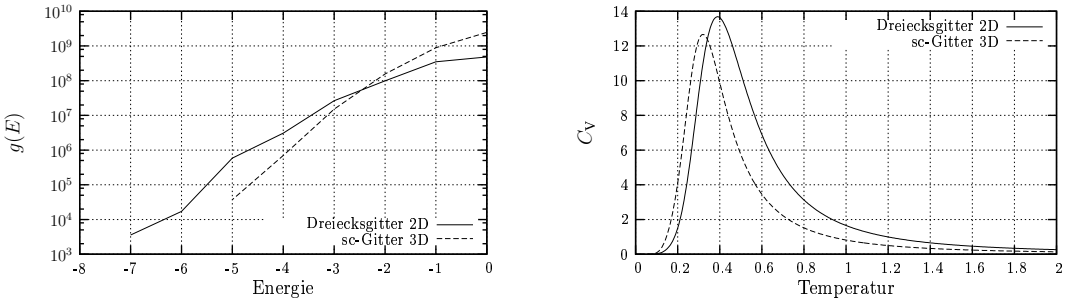


Abbildung 5.6: **Links** Die Zustandsdichte der Sequenz (Seq 16_3) auf dem 2D Dreicksgitter bzw. auf dem 3D sc-Gitter, **rechts** die zugehörigen Wärmekapazitäten. Beide zeigen einen deutlichen *single peak*.

Eine genauere Analyse des Zusammenhangs der Entartung des Grundzustandes mit dem in der Wärmekapazität zu sehenden Peak im Tieftemperaturbereich kann in [7] nachgelesen werden. Dort werden aus einfachen Annahmen zur Entartung des Grundzustandes sowie der Entartung und der Energie des ersten angeregten Zustandes in einem 3-Zustands Modell u.a. folgende Trends abgelesen:

- Das lokale Minimum zwischen den beiden Peaks ist, wenn vorhanden, umso ausgeprägter, je niedriger die Entartung des Grundzustandes ist und umso größer die Entartung des ersten angeregten Zustandes ist.

- Die Peaks entfernen sich umsomehr voneinander, umso größer die Differenz der Entartungen ist und umso kleiner die Energiedifferenz zwischen Grundzustand und erstem angeregten Zustand ist.

5.1.2 *Quasi-Designing Sequences* auf 2D Dreicksgitter

Hier möchte ich die Entwicklung einer kurzen *quasi-designing sequence* auf dem 2D Dreiecksgitter verfolgen, der man gezielt Monomere entfernt. Aus den Ergebnissen der folgenden Betrachtungen war u.a. auch die Idee für Kap. 2.1.2 entstanden, in dem es um das gezielte *design* von HP-Proteinen ging.

Mit *quasi-designing* meine ich, daß der Grundzustand zwar entartet ist, es aber leicht überschaubar ist, welches die Grundzustände sind. Ich beginne mit der Sequenz (Seq 17_1):

$$\text{HHPPHPPHPPHPPHPPH}\,, \qquad (\text{Seq}\,17_1)$$

sie ist *quasi-designing*, wie Abb. 5.7 veranschaulicht, welche einen Grundzustand von (Seq 17_1) zeigt. Alle hydrophoben Monomere bilden einen kompakten Kern, die polaren Monomere sind so in die Sequenz integriert, daß sie sich um diesen Kern mit einigen Freiheitsgraden gruppieren können.

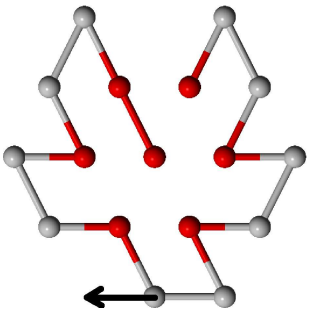


Abbildung 5.7: Ein Grundzustand der Sequenz (Seq 17_1) auf dem Dreiecksgitter in 2D. Der Grundzustand ist entartet, man kann jedoch leicht alle anderen erraten. Zum Beispiel kann man die beiden im Bild untersten Monomere nach links „klappen“ (angedeutet).

Abbildung 5.8 zeigt die Zustandsdichte der Sequenz (Seq 17_1) und deren Wärmekapazität. Der Grundzustand ist 12fach entartet (inklusive des Symmetriefaktors $s'^{\text{planar}} = 2$, es gibt also 6 unabhängige Grundzustandskonformationen), die Wärmekapazität zeigt zeigt keinen Doppelpeak, jedoch ist er zu erahnen[2].

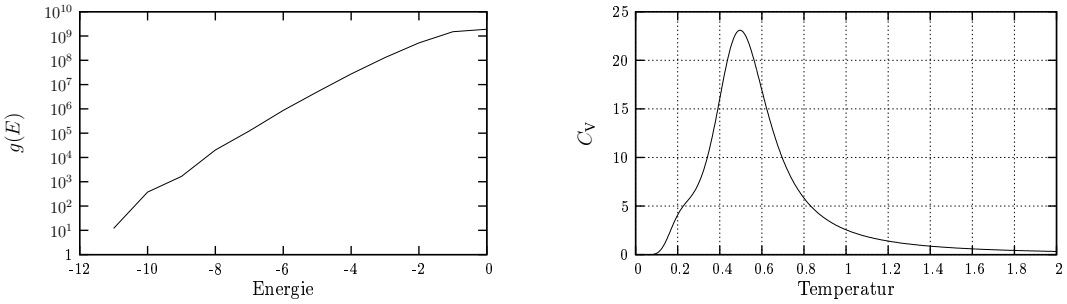


Abbildung 5.8: **Links** Die Zustandsdichte der Sequenz (Seq 17_1) auf dem 2D Dreiecksgitter, **rechts** die daraus erhaltene Wärmekapazität.

Nun kann man z.B. jeweils ein Monomer so entfernen bzw. eines so entfernen und ein anderes in seiner Position verschieben, daß man folgenden Sequenzen erhält:

$$\text{HHPHPPHPPHPPHPPH}\,, \qquad (\text{Seq}\,16_1)$$

$$\text{HHPPHPPHPHPHPPHP}\,. \qquad (\text{Seq}\,16_2)$$

In (Seq 16_1) wurde das Monomer 4 aus (Seq 17_1) entfernt, in (Seq 16_2) wurde Monomer 10 aus (Seq 17_1) entfernt und Monomer 13 an die letzte („ganz rechte“) Position verschoben. Abbildung 5.9 zeigt Grundzustände der so entstandenen Ketten. Die Zustandsdichten und Wärmekapazitäten (Abb. 5.10) unterscheiden sich nicht wesentlich von denen in Abb. 5.8.

[2]Nach dem einfachen Modell in [7] kann dies erklärt werden durch das kleine Verhältnis $g(E_1 = -10)/g(E_0 = -11) = 374/12 \approx 30$. Sieht man sich z.B. nocheinmal Abb. 5.4 an, worin für das 3D sc-Gitter deutliche Doppelpeaks zu sehen sind, sieht man auch, daß dort $g(E_1)/g(E_0) \approx 120$ ist. Es spielen aber sicherlich noch weitere Faktoren eine Rolle.

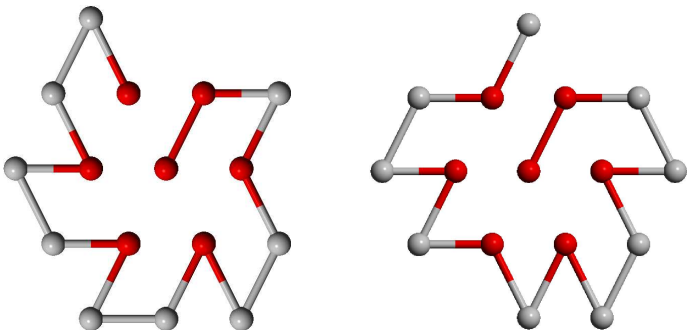


Abbildung 5.9: Grundzustände der Sequenzen (Seq 16_1) und (Seq 16_2) auf dem Dreiecksgitter in 2D. Diese waren entstanden durch das Entfernen bzw. und das Verschieben jeweils eines Monomers aus Sequenz (Seq 17_1).

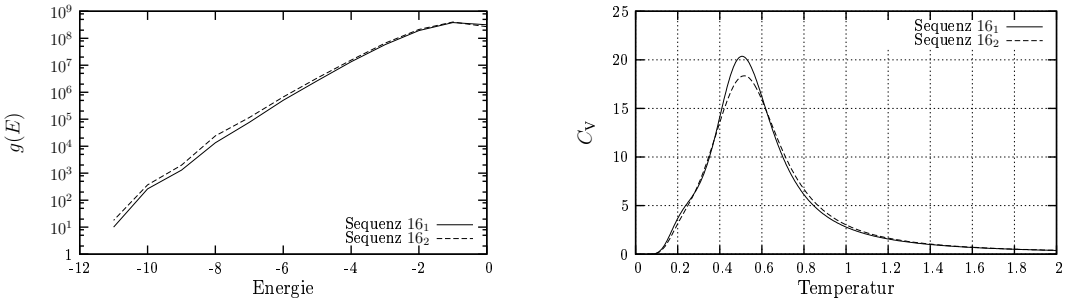


Abbildung 5.10: **Links** Die Zustandsdichten der Sequenzen (Seq 16_1) und (Seq 16_2) auf dem 2D Dreiecksgitter, **rechts** die zugehörigen Wärmekapazitäten.

Durch ähnliche Vorgehensweisen gelangen wir über die Sequenzen (Seq 14_1) und (Seq 14_2)

$$\text{HHPPHPHPHPHPHP}\,, \qquad (\text{Seq}\,14_1)$$

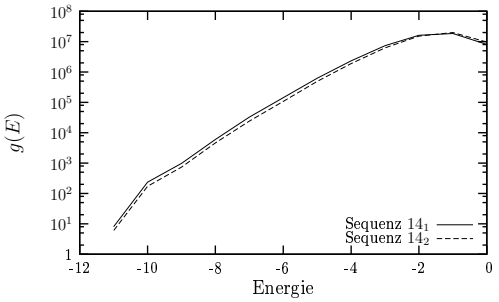
$$\text{HHPPHPPHPHPHPH}\,, \qquad (\text{Seq}\,14_2)$$

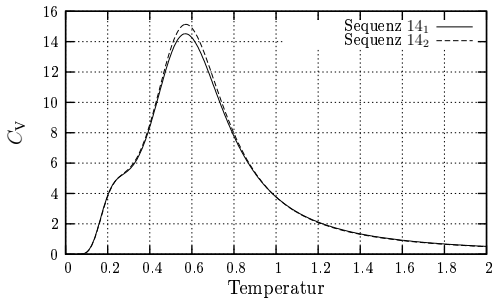
(siehe Abb. 5.11), deren Grundzustände ebenso den kompakten hydrophoben Kern der vorherigen Sequenzen haben und ähnlich stark bzw. schwach entartet sind, zu folgender Sequenz:

$$\text{HHPHPHPHPHPHP}\,. \qquad (\text{Seq}\,13_1)$$

Abbildung 5.12 zeigt einen der beiden Grundzustände der Sequenz (Seq 13_1). Trotzdem der Grundzustand nur 2fach entartet ist, ist der Doppelpeak in der Wärmekapazität nicht sehr stark ausgeprägt. Das kann vielleicht wiederum dadurch erklärt werden, daß der erste angeregte Zustand ebenfalls nur schwach entartet ist, nämlich $g(E_1) = 70$ (siehe Abb. 5.13). Das Verhältnis ist wieder $g(E_1)/g(E_0) \approx 30$ wie für Sequenz (Seq 17_1). Die Entartung des Grundzustandes ist jedoch deutlich geringer, was wahrscheinlich dafür verantwortlich ist, daß der Doppelpeak trotzdem deutlicher sichtbar ist, als in Abb. 5.8.

In diesen beiden Abschnitten habe ich versucht, einige der explizit gestellten offenen Fragen aus [7] ansatzweise zu beleuchten. Keineswegs handelt es sich aber um eine systematische und noch weniger abgeschlossene Untersuchung des Übergangs zwischen Grundzuständen und angeregten Zuständen sowie der Frage nach *designing sequences* auf verallgemeinerten Gittern.





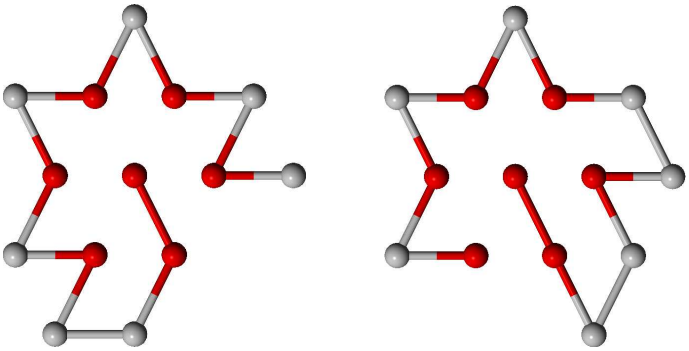


Abbildung 5.11: **Oben** Die Zustandsdichten (**links**) bzw. Wärmekapazitäten (**rechts**) der Sequenzen (Seq 14_1) und (Seq 14_2). **Unten** Zwei Grundzustände dieser Sequenzen.

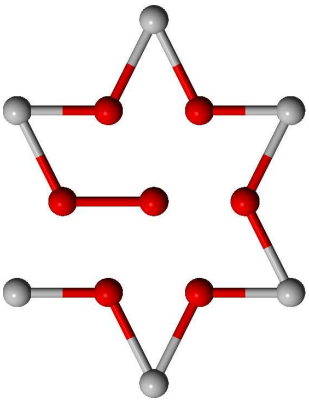
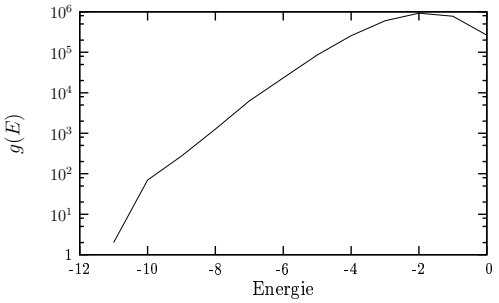


Abbildung 5.12: Ein Grundzustand der Sequenz (Seq 13_1) auf dem Dreiecksgitter in 2D. Der Grundzustand ist 2fach entartet, man sieht leicht den anderen.



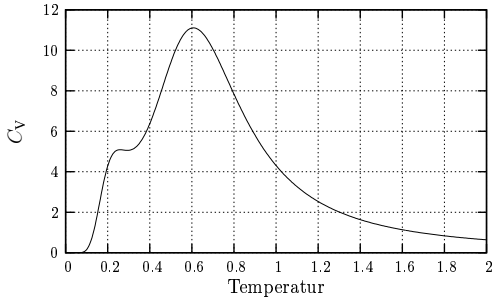


Abbildung 5.13: **Links** Die Zustandsdichte der Sequenz (Seq 13_1) auf dem 2D Dreiecksgitter, **rechts** die zugehörige Wärmekapazität. Der Doppelpeak in der Wärmekapazität deutet sich sehr klar an.

Bevor ich mich jedoch längeren Proteinketten zuwende, bei denen ich keine Resultate aus exakten Enumerationen mehr zur Verfügung habe, möchte ich noch kurz zu einem letzten interessanten Schritt aus obiger Evolutionskette kommen.

Sequenz (Seq 13_1) hat eine sehr prägnante „sternförmige“ Grundzustandskonformation. Lassen sich daraus Grundzustände von längeren Ketten konstruieren? Das erste Ergenis, das ich dazu erhielt zeigt die Konformation in Abb. 5.14, welche folgende Sequenz hat:

$$\text{HHPHPHPHPHPHPHPHH}\,. \qquad (\text{Seq}\,17_2)$$

Der Grundzustand von Sequenz (Seq 17_2) ist 4fach entartet. Zwei der Grundzustände entstehen jedoch allein dadurch, daß die Sequenz symmetrisch ist, sie unterscheiden sich im Aussehen nicht von den anderen beiden. Zwei weitere Grundzustände sind die, welche entstehen, wenn man den unteren Teil der in Abb. 5.14 zu sehenden Konformation nach links schiebt. Der Zustand dann ist aber auch genau der, der durch Spiegelung an einer waagerechten Achse entsteht. Der Doppelpeak in der Wärmekapazität ist ähnlich stark ausgeprägt, wie der in Abb. 5.13 von Sequenz (Seq 13_1) (siehe Abb. 5.15).

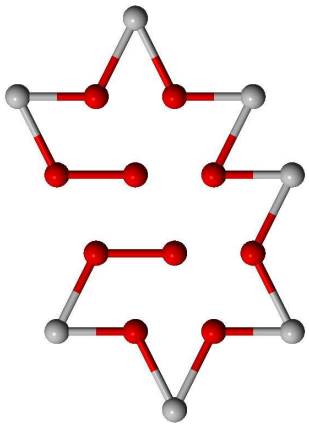


Abbildung 5.14: Ein Grundzustand der Sequenz (Seq 17_2) auf dem Dreiecksgitter in 2D.

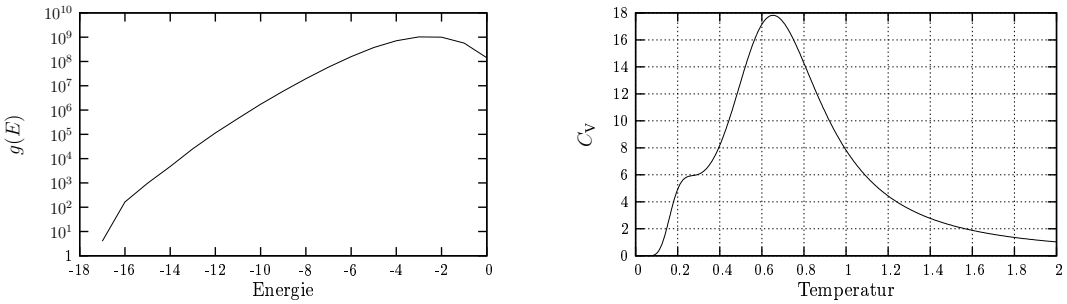


Abbildung 5.15: **Links** Die Zustandsdichte der Sequenz (Seq 17_2) auf dem 2D Dreiecksgitter, **rechts** die zugehörige Wärmekapazität.

5.2 Ergebnisse für längere Proteine

Im folgenden widme ich mich Proteinen mit Längen zwischen $46 \leq n \leq 136$. Wie früher schon gesehen, ist es dafür unmöglich, exakt alle möglichen Konformationen einer gegebenen Sequenz zu erzeugen. Die nächsten Kapitel zeigen daher lediglich Ergebnisse aus Grundzustandssuchen mit dem nPERM Algorithmus. Thermodynamische Untersuchungen habe ich nicht durchgeführt, da sich nPERM dafür im Tieftemperaturbereich zu schlecht verhält. Für thermodynamische Untersuchungen ist sehr viel besser der in [32] vorgestellte multikanonische PERM Algorithmus in der Lage. Erste Ergebnisse, z.B. die Wärmekapazitäten für die Sequenzen ($\mathrm{Seq}\,48_1$) bis ($\mathrm{Seq}\,48_{10}$) findet man ebenso dort.

5.2.1 Ergebnisse auf dem sc-Gitter

Das erste Protein, welches ich hier untersucht habe, ist ein Protein der Länge $n = 46$. Es ist das HP-Modell des realen Proteins Crambin [33]:

$$\mathrm{PPHHHPHHHPPPHPHHPHHPPHPHHHHPHPPHHHHHPHPHHPPHHP}\,. \qquad (\mathrm{Seq}\,46_1)$$

Die Grundzustandsenergie, welche ich hier (und im folgenden immer) mit nPERM gefunden habe, ist $E = -33$. Sie wurde sehr schnell, innerhalb von 6 s, während der Tour $\approx 32\,000$ das erste Mal gefunden. Einen Grundzustand[3] von Sequenz ($\mathrm{Seq}\,46_1$) zeigt Abb. 5.16. Man sieht in Abb. 5.16 gut den kompakten Kern aus hydrophoben Monomeren, welche durch die verfügbaren polaren Monomere so gut wie möglich abgeschirmt werden. Man kann dieses Verhalten als ein erstes optisches Indiz dafür benutzen, wie weit man noch vom wahren, unbekannten Grundzustand „entfernt“ ist: Findet man etwa 2 Cluster aus hydrophoben Monomeren, welche nicht verbunden sind, kann man daraus schließen, daß dies i.a. noch nicht der Zustand minimaler Energie ist.

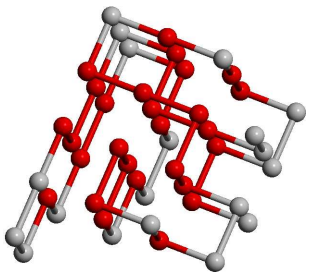


Abbildung 5.16: Ein vermuteter Grundzustand der Sequenz ($\mathrm{Seq}\,46_1$) auf dem sc-Gitter in 3D.

Als nächstes wende ich mich den bereits in Kap. 4.4.1 erwähnten 48meren zu, welche in [29] untersucht wurden und aus [28] stammen. Wie gesagt, habe ich in etwa vergleichbaren

[3] Natürlich kann man jetzt nicht mehr von dem wahren Grundzustand sprechen, weil dieser nicht bekannt ist. Wenn ich hier und im folgenden von „Grundzustand“ spreche, meine ich den vermuteten Grundzustand, bzw. den Zustand, mit der geringsten Energie, der bisher gefunden wurde.

Zeiten (siehe Tab. 4.2) alle dort angegebenen Grundzustände auf dem sc-Gitter in 3D ebenso gefunden. Sie sind in Kap. B.2 zu sehen.

In [29] wurden ebenso 4 weitere, längere Proteine untersucht, welche HP-Modelle von realen Proteinen sind. In Kap. 1.3.2 ist bereits eines davon im Vergleich zu seinem „Original“ gezeigt worden (Es handelte sich dort um Sequenz ($\mathrm{Seq}\,103_1$)). Die Sequenzen und Grundzustände kann man in Kap. B.3 nachlesen. Für zwei dieser Sequenzen habe ich die angegebenen Grundzustände in [29] mit nPERM nicht finden können. Das kann entweder daran liegen, daß meine Implementation[4] prinzipiell sehr viel länger braucht, um die dort angegebenen Zustände zu finden oder ich einfach nicht lange genug gewartet habe[5]. Ein dritter Grund ist möglicherweise eine ungünstige Temperatur, bei der simuliert wurde[6]. Wie früher schon gesehen, kann das drastische Auswirkungen auf die Effizienz haben. Worin ich allerdings mit [29] übereinstimme, ist die Fakt, daß ich mit nPERMss auch deutlich niedrigere Grundzustandsenergien finde, als in [34] angegeben. Tabelle 5.2.2 zeigt die gefundenen Grundzustandsenergien im Vergleich.

	$E_{\min}{}^a$	$E_{\min}{}^b$	$E_{\min}{}^c$
Sequenz ($\mathrm{Seq}\,58_1$)	-42	-44	-44
Sequenz ($\mathrm{Seq}\,103_1$)	-49	-54^d	-53
Sequenz ($\mathrm{Seq}\,124_1$)	-58	-71	-71
Sequenz ($\mathrm{Seq}\,136_1$)	-65	-80	-77

[a] Niedrigste Energie gefunden in [34].
[b] Niedrigste Energie gefunden mit nERMis und individuellen Temperaturen in [29].
[c] Niedrigste Energie gefunden hier mit nPERMss.
[d] Der aktuell niedrigste bekannte Wert ist $E_{\min} = -56$ [32].

Tabelle 5.2: Grundzustandsenergien der Sequenzen ($\mathrm{Seq}\,58_1$), ($\mathrm{Seq}\,103_1$), ($\mathrm{Seq}\,124_1$) und ($\mathrm{Seq}\,136_1$). Die Resultate entstammen aus unterschiedlichen Arbeiten, wobei jeweils ein anderes Verfahren benutzt wurde.

5.2.2 Ergebnisse auf verallgemeinerten Gittern

Auch auf dem 2D Dreiecksgitter und dem 3D fcc-Gitter habe für ich einige der bereits untersuchten Sequenzen Grundzustandssuchen mit nPERM durchgeführt. Die untersuchten Sequenzen und die dazu gefundenen Grundzustandsenergien zeigt die folgende Tabelle 5.3.

Im folgenden möchte ich nocheinmal kurz auf die Effizienz der Grundzustandssuche in Abhängigkeit von der Temperatur eingehen. Ich wähle dazu Sequenz ($\mathrm{Seq}\,124_1$) auf dem zweidimensionalen Dreiecksgitter. Abbildung 5.17 zeigt, in welcher *tour* ein neuer Zustand

[4] H.-P. Hsu et al. benutzten zur Grundzustandssuche einen modifizierten nPERM Algoritmus, den sie nPERMis nennen, wobei „is“ für „importance sampling“ steht, wohingegen meine Implementation dort mit nPERMss bezeichnet wird – für „simple sampling“. nPERMis wurde von mir nicht implementiert, ich kann deswegen nicht sagen, um wieviel schneller es, auf dieses Problem angewandt, wirklich ist.

[5] In [29] werden keine Zeiten angegeben, wann der Grundzustand das erste Mal gefunden wurde.

[6] H.-P. Hsu et al. haben für jede dieser Sequenzen eine andere Temperatur eingestellt, wohingegen ich stets dieselbe benutzte.

	2D Dreiecksgitter		3D sc-Gitter		3D fcc-Gitter	
	$E_{\min}$	Tour[a]	$E_{\min}$	Tour	$E_{\min}$	Tour
Sequenz ($\mathrm{Seq}\,48_1$)	-38	3.7×10^7	-32	8.0×10^5	-69	2.7×10^5
Sequenz ($\mathrm{Seq}\,48_2$)			-34	3.9×10^8	-69	5.1×10^6
Sequenz ($\mathrm{Seq}\,48_3$)			-34	6.6×10^5	-72	4.4×10^6
Sequenz ($\mathrm{Seq}\,48_4$)			-33	1.9×10^7	-71	2.3×10^7
Sequenz ($\mathrm{Seq}\,48_5$)			-32	3.2×10^6	-70	6.2×10^6
Sequenz ($\mathrm{Seq}\,48_6$)			-32	2.0×10^6	-70	4.1×10^6
Sequenz ($\mathrm{Seq}\,48_7$)			-32	6.7×10^6	-70	3.8×10^5
Sequenz ($\mathrm{Seq}\,48_8$)			-31	1.6×10^5	-69	1.8×10^7
Sequenz ($\mathrm{Seq}\,48_9$)			-34	2.9×10^7	-71	1.6×10^6
Sequenz ($\mathrm{Seq}\,48_{10}$)			-33	8.2×10^5	-68	2.6×10^7
Sequenz ($\mathrm{Seq}\,58_1$)	-49	5.9×10^6	-44	4.6×10^8	-94	1.6×10^7
Sequenz ($\mathrm{Seq}\,103_1$)	-56	1.7×10^8	-53	8.8×10^8	-114	9.9×10^8
Sequenz ($\mathrm{Seq}\,124_1$)	-73	4.0×10^8	-71	1.6×10^9	-154	8.3×10^6
Sequenz ($\mathrm{Seq}\,136_1$)	-80	9.9×10^7	-77	2.9×10^8	-167	5.8×10^8

[a]Meint hier immer die Tour, in der $E_{\min}$ das erste Mal gefunden wurde.

Tabelle 5.3: Grundzustandsenergien aller von mir untersuchten Ketten auf verschiedenen Gittern, sowie die Tour, während diese das erste Mal gefunden wurden.

niedrigster Energie das erste Mal gefunden wurde. Man sieht hier wieder, daß die Temperatur entscheidenden Einfluß auf das Auffinden von Zuständen niedriger Energie hat. Bei den Temperaturen $T = 0.25$, $T = 0.28$ und $T = 0.30$ wurden die Zustände mit der Energie $E = -73$ nach etwa 5×10^8 Touren gefunden (etwa 4h[7]), bei der Temperatur $T = 0.32$ nach etwa doppelt so vielen Touren, allerdings schon in etwa dreifacher Zeit und bei der Temperatur $T = 0.35$ innerhalb von 10 Tagen überaupt nicht mehr!

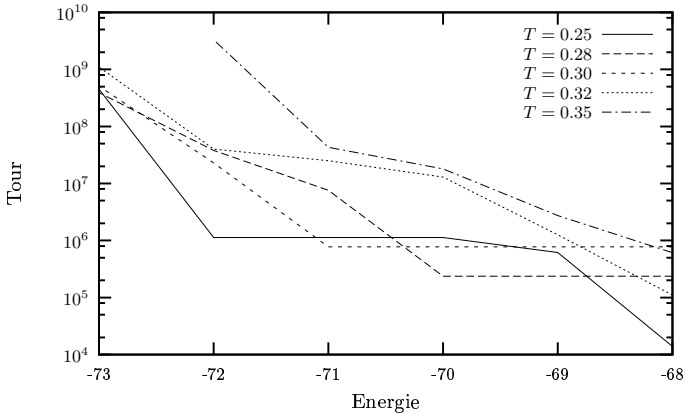


Abbildung 5.17: Die Touren, in denen Zustände mit neuer niedrigster Energie der Sequenz ($\mathrm{Seq}\,124_1$) gefunden wurden (2D Dreiecksgitter).

[7]CPU P4 2Ghz.

Abbildung 5.18 zeigt einen Zustand mit der niedrigsten gefundenen Energie $E = -73$. Wie deutlich zu sehen ist, gibt es noch drei separate hydrophobe Zentren, was darauf deutet, daß es Zustände mit noch niedrigerer Energie geben sollte. Jedoch wurden die Zustände mit den Energien $E = -73$ bereits nach etwa 4–13 Stunden (bei $T = 0.25$ bzw. $T = 0.32$) gefunden, wohingegen die komplette Simulation bisher etwa 10 Tage lief und keine weiteren Zustände niedrigerer Energie fand. In Kap. B.3 sind drei weitere Zustände der Energie $E = -73$ gezeigt, alle weisen qualitativ dieselbe Struktur auf.[8]

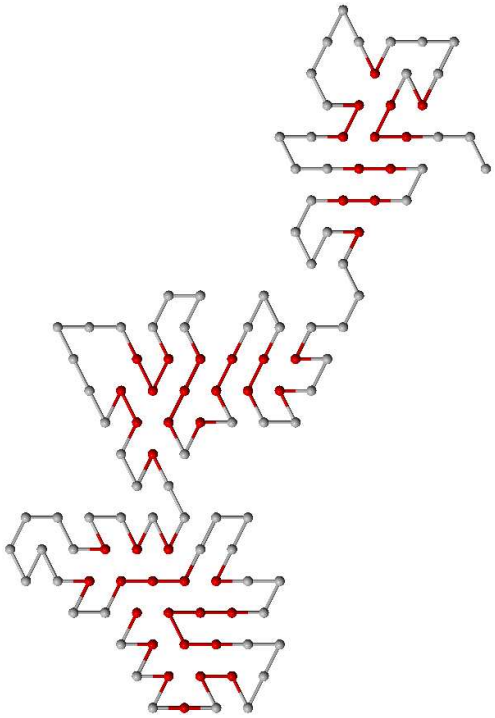


Abbildung 5.18: Ein Zustand mit der Energie $E = -73$ der Sequenz (Seq 124_1) auf dem 2D Dreiecksgitter. Gut zu sehen die drei separaten hydrophoben Regionen.

5.2.3 *Latest News*

Unmittelbar vor dem Fertigstellen dieser Arbeit erhielt ich aus meinen fortlaufenden Simulationen folgende neue Ergebnisse, welche ich hier nur stichpunktartig anfügen möchte, ohne die vorherigen Kapitel umzuschreiben:

- Für Sequenz (Seq 124_1) wurden bei zwei Temperaturen (u.a. bei $T = 0.30$) Zustände mit der Energie $E = -74$ gefunden. Hier findet man jetzt schon nicht mehr nur drei getrennte hydrophobe Zentren. Wie Abb. 5.19 zeigt, „verschmelzen“ hier zwei dieser

[8] Es handelt sich bei den gezeigten Konformationen genau um jene, welche bei den unterschiedlichen Temperaturen zuerst gefunden wurden.

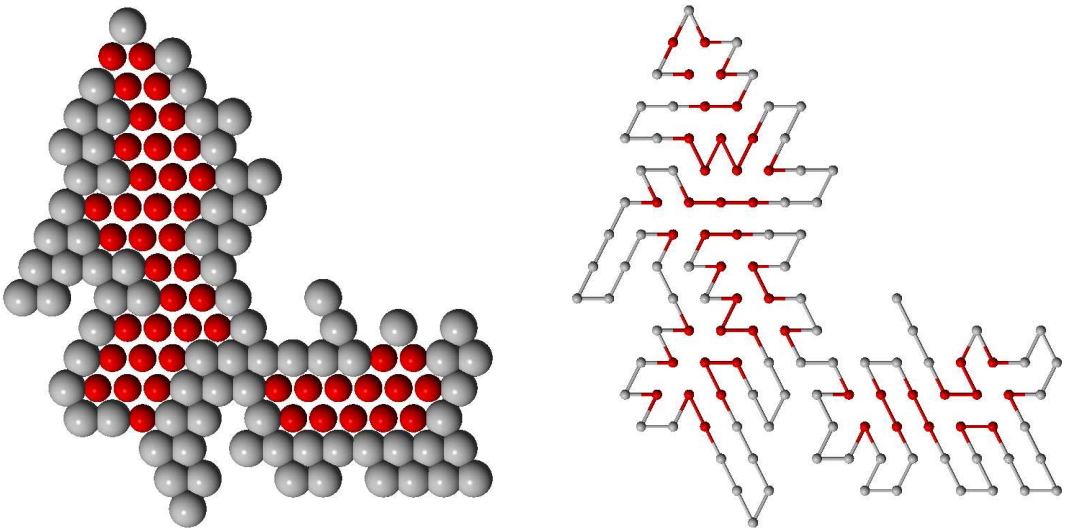


Abbildung 5.19: Ein Zustand mit der Energie $E = -74$ der Sequenz (Seq 124_1) auf dem 2D Dreiecksgitter in zwei verschiedenen Darstellungen. Gut zu sehen die nur noch zwei separaten hydrophoben Regionen.

Zentren zu einem. Dieses ist jedoch mitnichten schon kompakt. Der abgebildete Zustand wurde nach etwa 13 Tagen gefunden.

- Für dieselbe Sequenz wurde bei der Temperatur $T = 0.35$ nach 18 Tagen, während der *tour* 2.5×10^{10}, das erste Mal ein Zustand der Energie $E = -73$ auf dem 2D Dreiecksgitter gefunden.

- Auf dem 3D fcc-Gitter wurden für zwei Sequenzen neue Zustände niedrigster Energie gefunden. Die Energien dieser Zustände sind

  1. $E = -116$ für (Seq 103_1), während der *tour* 7.7×10^9 (vorher $E = -114$),
  2. $E = -168$ für (Seq 136_1), während der *tour* 2.2×10^{10} (vorher $E = -167$).

Kapitel 6

Simulation von Homopolymeren

Homopolymere sind Monomerketten großer Länge aus einer einzigen Sorte von Monomeren mit attraktiver Wechselwirkung zwischen auf dem Gitter benachbarten Monomeren. Sie können beschrieben werden durch *interacting self-avoiding walks*. Die Form der Konformation, die Homopolymere auf Gittern annehmen, hängt von der Güte des Lösungsmittels ab. Polymere in „guten“ Lösungsmitteln (bzw. bei hohen Temperaturen) nehmen eine ausgestreckte Form an, in „schlechten“ Lösungen (bei niedrigen Temperaturen) haben sie eine dichte, kugelförmige Form (siehe Abb. 6.1).

Der Übergangspunkt heißt θ-Punkt, man spricht auch vom θ-Übergang. Dieser ist ein Phasenübergang 2. Ordnung im thermodynamischen Limes. Sehr weit oberhalb der kritischen Temperatur T_θ entspricht die Konformation des Gitterpolymers (fast) der eines Zufallsweges. Verringert man die Temperatur bis unterhalb T_θ, „kollabiert“ das Polymer, da der Entropiebeitrag zur freien Energie nicht länger die gestreckten Konformationen gegenüber den dichtgepackten Konformationen mit sehr niedriger innerer Energie behaupten kann.

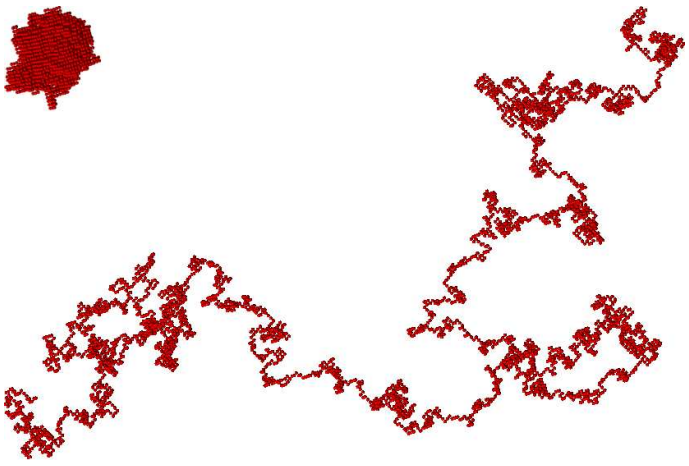


Abbildung 6.1: 2 Konformationen des Homo4096mers auf dem kubischen Gitter bei verschiedenen Temperaturen $T' \ll T_\theta$ (**links oben**) und $T' \gg T_\theta$. Man sieht deutlich die verschiedenen Phasen, dichtgepackt und weit ausgestreckt. Die beiden Konformationen sind in derselben Auflösung gezeigt, so daß man auch die Größenverhältnisse sehen kann. Die Energien sind $E = -5367$ (dicht gepackt) bzw. $E = -1054$ (ausgestreckt).

6.1 Polymerkollaps in 3 Dimensionen

Nimmt man an, daß für den kollabierten Zustand eines Polymers in jedem Volumen viele unkorrelierte Teilstücke des Polymers liegen, kann man diese als einen See unabhängiger kleiner Polymere sehen [30]. Der Gleichgewichtszustand, und damit die Gleichgewichtsgröße und -dichte, ist erreicht, wenn der Druck innerhalb des Polymersees gleich dem äußeren Druck ist, d.h. gleich 0.

$$p_{\mathrm{eq}}(\varrho) = 0\,, \tag{1}$$

$$\varrho \sim n/R^3\,, \tag{2}$$

wenn n die Anzahl der Monomere pro Kette und R der Radius der Polymerkugel ist und somit ϱ die Monomerdichte. Benutzt man für $p(\varrho)$ eine Virialentwicklung erhält man

$$0 = p_{\mathrm{eq}}(\varrho) \simeq TB\varrho^2 + 2TC\varrho^3\,, \tag{3}$$

wobei B der zweite und C der dritte Virialkoeffizient sind, und damit

$$\varrho = -B/2C\,. \tag{4}$$

Der 2. Virialkoeffizient verschwindet bei der Boyle-Temperatur T_n [31] und damit auch die Monomerdichte.

Der θ-Punkt ist von der Dimension und vom Gittertyp abhängig. Er wurde auf mehrere Arten für das kubische Gitter in 3 Dimensionen bestimmt (siehe auch [31]):

- $T_\theta = 3.716 \pm 0.007$ über das Messen des Virialkoeffizienten und Extrapolation $T_\theta = \lim_{n\to\infty}(T_n : B(n, T_n) = 0)$ [31],
- $T_\theta = 3.717 \pm 0.003$ über das Messen der Monomerdichte und Extrapolation nach $\varrho \sim (T_n - T_\theta)^{0.7}$ (entgegen der *mean-field* Vorhersage $\varrho \sim (T_n - T_\theta)^1$) [27].

Wie sieht $T_{\theta,n} = T_\theta(n)$ für endlich lange Ketten aus und wie schnell konvergiert $T_{\theta,n}$ gegen T_θ?

6.1.1 Ergebnisse für kurze Ketten

Mit nPERM versuche ich zunächst für Kettenlängen der Größenordnungen ähnlich denen, für die ich HP-Proteine untersucht habe, auch Polymere zu untersuchen. Ich habe zuerst für verschiedene Kettenlängen ($n = 50 - 2000$) bei verschiedenen Temperaturen Simulationen durchgeführt und die Wärmekapazitäten berechnet,

$$C_V(T) = \frac{\mathrm{d}\langle E(T)\rangle}{\mathrm{d}T}\,. \tag{5}$$

Aus dem Maximum der Wärmekapazität habe ich die Werte für die Übergangstemperaturen abgelesen. Abbildung 6.2 zeigt beispielhaft die Wärmekapazitäten von Polymeren der Längen

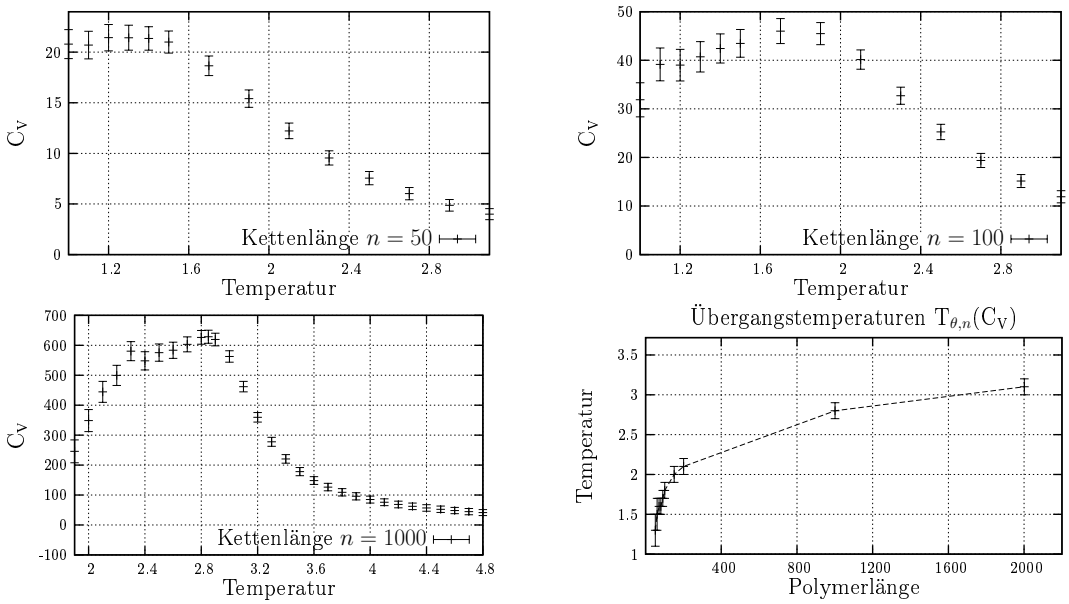


Abbildung 6.2: Wärmekapazitäten von kurzen Polymeren (Beispielhaft: Kettenlängen $n = 50, 100, 1000$), sowie die daraus gewonnenen Übergangstemperaturen als Funktion der Kettenlänge (**rechts unten**).

$n = 50$, $n = 100$, und $n = 1000$. Rechts unten in Abb. 6.2 ist das *finite-size scaling* der Übergangstemperatur zu sehen. Die obere Grenze der Abbildung ist der Wert $T_\theta = 3.717$, von dem die Übergangstemperaturen der kurzen Ketten noch sehr weit entfernt sind.

Natürlich kann man neben der Wärmekapazität noch die Ableitungen anderer Observabler betrachten, wie zum Beispiel der *end-to-end distance* d_{ee} und des *radius of gyration* r_{gyr}. Die *end-to-end distance* ist der geometrische Abstand zwischen dem ersten und dem letzten ($n$ten) Monomer eines Polymers auf dem Gitter, der *radius of gyration* ist der mittlere Abstand aller Monomere vom Polymerschwerpunkt. Er kann als Maß für die Kompaktheit eines Polymers gesehen werden. Die Observablen sind wie folgt definiert:

$$d_{\mathrm{ee}} = |\mathbf{x}_1 - \mathbf{x}_n| \,, \quad r_{\mathrm{gyr}}^2 = \frac{1}{n} \sum_{i=1}^{n} (\mathbf{x}_i - \mathbf{x}_{\mathrm{SP}})^2 \,, \tag{6}$$

wobei $\mathbf{x}_i$ die Position des $i$ten Monomers und $\mathbf{x}_{\mathrm{SP}}$ die Position des Schwerpunkts des Polymers ist. Als Fluktuation der Observablen definiere ich hier, analog zu Gl. (5):

$$\begin{aligned} \frac{\mathrm{d}\langle \mathcal{O} \rangle}{\mathrm{d}T} &= \frac{\mathrm{d}}{\mathrm{d}T} \left(\frac{1}{Z(T)} \sum_i \mathcal{O}_i \, \mathrm{e}^{-\beta(T) E_i} \right) \\ &= \frac{1}{T^2} \left(\frac{\sum_i \mathcal{O}_i \, \mathrm{e}^{-\beta(T) E_i}}{Z(T)} - \frac{\sum_i \mathcal{O}_i \, \mathrm{e}^{-\beta(T) E_i} \sum_i E_i \, \mathrm{e}^{-\beta(T) E_i}}{Z(T)^2} \right) \\ &= \frac{1}{T^2} \left(\langle E \mathcal{O} \rangle - \langle \mathcal{O} \rangle \langle E \rangle \right) \,. \end{aligned} \tag{7}$$

Für die *end-to-end distance* als Observable ergibt das dann

$$\frac{\mathrm{d}\langle d_{\mathrm{ee}}\rangle}{\mathrm{d}T} = \frac{1}{T^2}\left(\langle E d_{\mathrm{ee}}\rangle - \langle d_{\mathrm{ee}}\rangle\langle E\rangle\right) , \tag{8}$$

für den *radius of gyration* ergibt sich ein analoger Ausdruck[1].

Die Fluktuationen beider geometrischer Größen haben wieder, ähnlich der Wärmekapazität und abhängig von der Kettenlänge, Maxima bei bestimmten Temperaturen. Abbildung 6.3 (oben) zeigt exemplarisch die Fluktuationen der *end-to-end distance* und des *radius of gyration* bei zwei verschiedenen Kettenlängen. Abbildung 6.3 (unten) zeigt analog zu Abb. 6.2 (rechts unten) die Übergangstemperaturen dieser Größen im Vergleich zueinander und im Vergleich zu Abb. 6.2. Es zeigt sich, daß die geometrischen Übergänge bei ungefähr gleichen Temperaturen stattfinden und die Übergangstemperaturen von C_{V} deutlich davon abweichen. Alle Übergangstemperaturen sollten für $n \to \infty$ gegen T_θ gehen. T_θ ist in Abb. 6.3 wieder als obere Grenze des Plotbereiches enthalten.

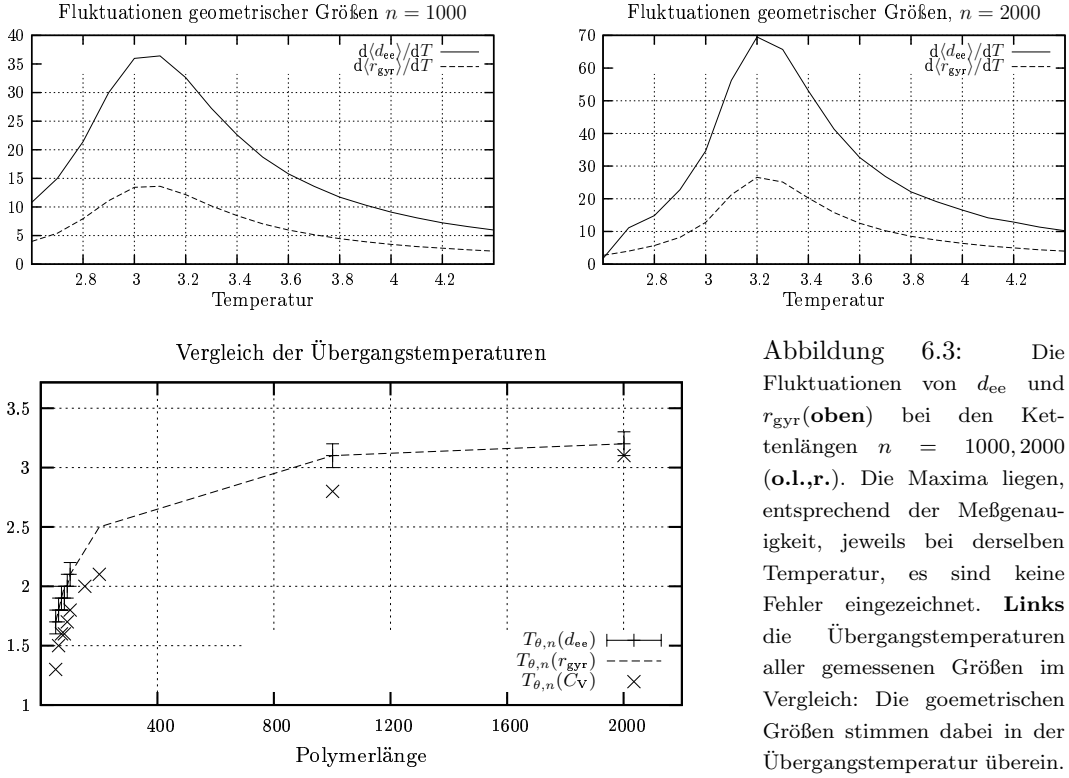


Abbildung 6.3: Die Fluktuationen von d_{ee} und r_{gyr}(**oben**) bei den Kettenlängen $n = 1000, 2000$ (**o.l.,r.**). Die Maxima liegen, entsprechend der Meßgenauigkeit, jeweils bei derselben Temperatur, es sind keine Fehler eingezeichnet. **Links** die Übergangstemperaturen aller gemessenen Größen im Vergleich: Die goemetrischen Größen stimmen dabei in der Übergangstemperatur überein.

[1] Allgemein wird als Fluktuation einer Observablen $\sigma^2 \sim \langle \mathcal{O}^2\rangle - \langle\mathcal{O}\rangle^2$ definiert. Für die $\mathcal{O} = E$ ist das gleich der Definition der Wärmekapazität (5). Für die anderen Observablen sind das natürlich andere Größen. Ich habe Gl. (7) der Analogie zu (5) wegen benutzt und möchte das hier Fluktuation einer Observablen nennen.

6.1.2 Simulation mit Polymeren bis zu Längen von $n \approx 16\,000$

Wie gesehen, kommt man mit kurzen Ketten nur sehr schlecht in die Nähe des tatsächlichen Phasenübergangs. Um z.B. T_θ abzuschätzen, braucht man sehr viel längere Polymere.

Nun ergibt sich erstmals ein Speicherproblem. Die Gewichte $W(t)$ nach Gl. (4.1) werden größer als der Darstellungsbereich im Speicher. Für $n = 4096$ und eine Temperatur von $T \approx 3.5$ kommt $W(t)$ in Größenordnungen von 10^{10^3}. Man kann daher nur noch die Logarithmen der Gewichte benutzen, wodurch sich die Formeln (4.1)-(4.3) abwandeln:

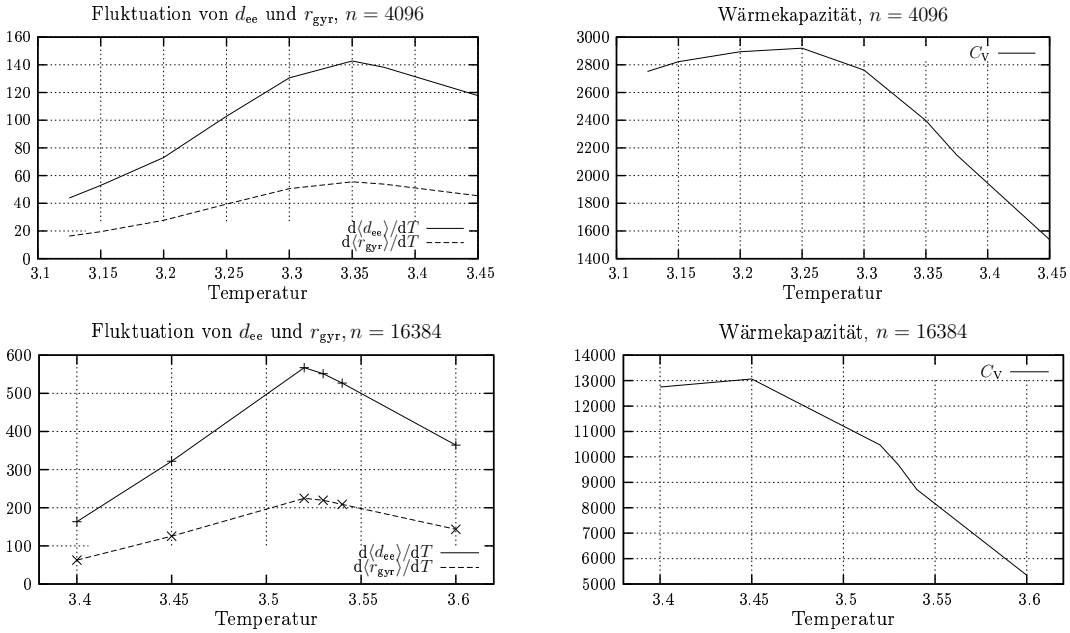
$$\ln W(t) \sim \sum_{i=0}^{t} \ln w_i = \sum_{i=0}^{t} \left(\ln w_{\mathrm{Rose},i} + \ln w_{\mathrm{Boltz},i}\right) , \tag{9}$$

$$\ln w_{\mathrm{Rose},i} + \ln w_{\mathrm{Boltz},i} = \ln m_i + (-E_i/k_\mathrm{B}T) , \tag{10}$$

$$\ln \hat{Z}_{\mathrm{neu}}(l+1) = \ln \hat{Z}_{\mathrm{alt}}(l+1) + \ln\left(1 + \mathrm{e}^{\ln W(l+1) - \ln \hat{Z}_{\mathrm{alt}}(l+1)}\right) . \tag{11}$$

Abbildung 6.4 zeigt analog zu Abb. 6.3 die Fluktuationen sowie die Entwicklung der Übergangstemperaturen für längere Ketten. Man sieht qualitativ kein anderes Verhalten als bei kurzen Ketten, es deutet sich jedoch an, daß es immer „schwieriger“ wird, sich T_θ zu nähern.

Abbildung 6.5 zeigt die normierten Histogramme der Energien, zu denen die Wärmekapazität in Abb. 6.4 (r.o.) berechnet wurde. Es zeigen sich deutliche *single peaks*, die sich mit höheren Temperaturen zu höheren Energien bzw. weniger Kontakten pro Monomer hin verschieben. Beim Übergangspunkt der Wärmekapazität zeigt sich ein leichtes Minimum in der Einhüllenden der Maxima, was eine Folge der größeren Breite des „kritischen“ Histogramms ist.



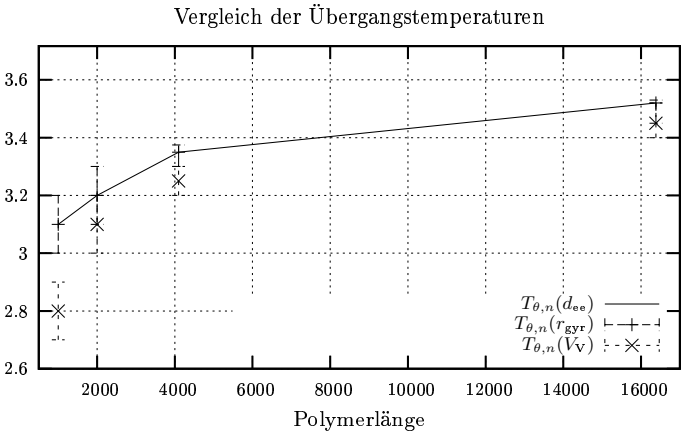


Abbildung 6.4: Die Fluktuationen von d_{ee} und r_{gyr} (**o.l.**, vorige Seite) sowie der Wärmekapazität (**o.r.**, vorige Seite) bei den Kettenlängen $n = 4096$ bzw. $n = 16\,384$. Es sind keine Fehler eingezeichnet. **Links** die Übergangstemperaturen aller gemessenen Größen im Vergleich analog Abb. 6.3, hier für die entsprechend längeren Ketten.

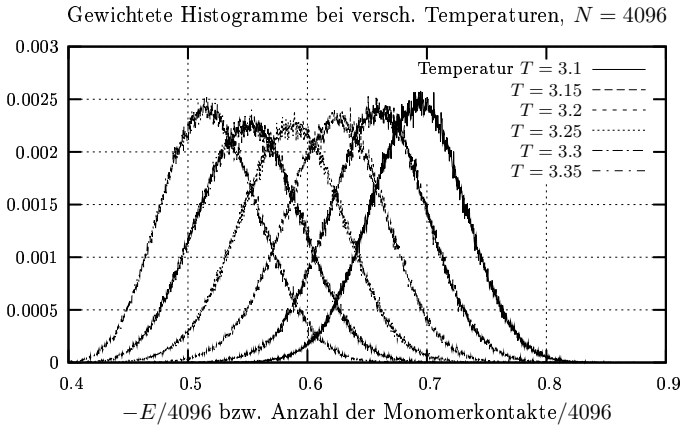


Abbildung 6.5: Die normierten gewichteten Histogramme der Energie bei Temperaturen zwischen $T = 3.1$ (*rechts*) und $T = 3.35$ (*links*) für Polymere der Länge $n = 4096$. Bei der Übergangstemperatur der Wärmekapazität (siehe Abb. 6.4) ist das Maximum der Verteilung am niedrigsten. Alle Histogramme haben deutlich einen einfachen Peak.

6.2 *First-order like*: Phasenübergang in 4D bei endlichen Kettenlängen

In vier Dimensionen gibt es nun eine Überraschung. In [35] findet man Abb. 6.6. Dort ist völlig analog zu Abb. 6.5 bei verschiedenen Temperaturen unterhalb und oberhalb der Übergangstemperatur das Energiehistogramm abgebildet. Bei der Temperatur[2] $T = 4.538$ sieht man einen Doppelpeak wie man ihn von Phasenübergängen 1. Ordnung her kennt (siehe z.B. [36]).

Abbildung 6.7 zeigt die Ergebnisse meiner Simulation in 4 Dimensionen. Als Temperatur ist $T = 4.538$ eingestellt, die Kettenlänge ist $n = 4096$. Ich beobachte genau wie in Abb. 6.6

[2] In Abb. 6.6 angegeben ist ω, der Boltzmannfaktor pro Kontakt zwischen Monomeren. Die Umrechnung zu Temperaturen ist $\omega = \mathrm{e}^{-\beta\epsilon} = \mathrm{e}^{1/T}$. Die Werte sind entsprechend:

ω	1.0	1.182	1.2465	1.26
T	∞	5.981	4.538	4.33

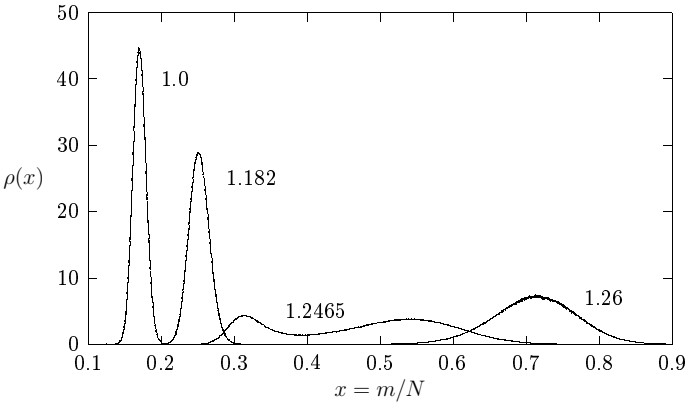


Abbildung 6.6: *Figure 12* aus http //arxiv.org/e-print/cond-mat/9907434 [35]: „*Internal energy density distributions at* $\omega = 1.0$, 1.182, 1.2465 *and* 1.26 *for* $N = 4096$.“ $x = m/N$ bezeichnet hier die Kontakte pro Monomer, die zur Energie beitragen. ω ist das Boltzmanngewicht pro Kontakt: $\omega = \mathrm{e}^{-\beta\epsilon}$, $\epsilon = -1$ (siehe Fußnote 2).

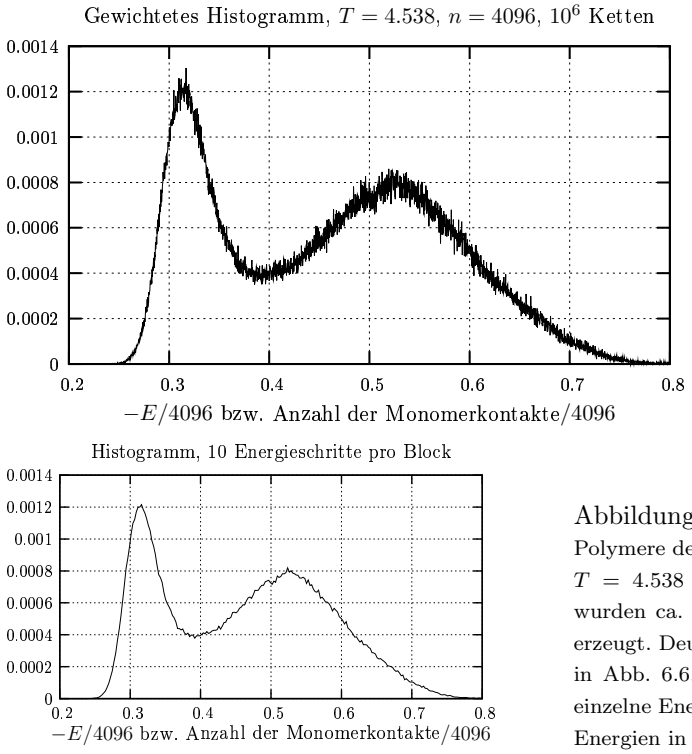


Abbildung 6.7: Das gewichtete Histogramm für Polymere der Länge $n = 4096$ bei der Temperatur $T = 4.538$ ($\omega = 1.2465$) in 4 Dimensionen. Es wurden ca. 1 000 000 Polymere (siehe Fußnote 6) erzeugt. Deutlich zu sehen ist ein Doppelpeak wie in Abb. 6.6. **Oben** ist das Histogramm für jede einzelne Energie gezeigt, **unten** wurden jeweils 10 Energien in einem *bin* zusammengefaßt.

einen Doppelpeak, wobei auffällt, daß der Niedrigenergiepeak (der „rechte“ Peak) viel größere Fluktuationen aufweist, als der Hochenergiepeak („links“), was in Abb. 6.6 so nicht zu sehen ist. Teilabbildung 6.7 (unten) zeigt denselben Doppelpeak in einer etwas niedrigeren Auflösung. Es sind die gezeigten Energieintervalle des Histogramms verbreitert worden.

Wieviele Polymerketten tatsächlich zu jedem Peak beitragen, zeigt Abb. 6.8. Dort zu sehen ist das „nackte“ Histogramm, d.h. die Anzahlen der Ketten mit einer bestimmten Energie (wohingegen Abb. 6.7 die Summen der Gewichte von Ketten mit einer bestimmten Energie zeigte). Man sieht deutlich, daß zum Hochenergiepeak mehr Ketten beitragen, als

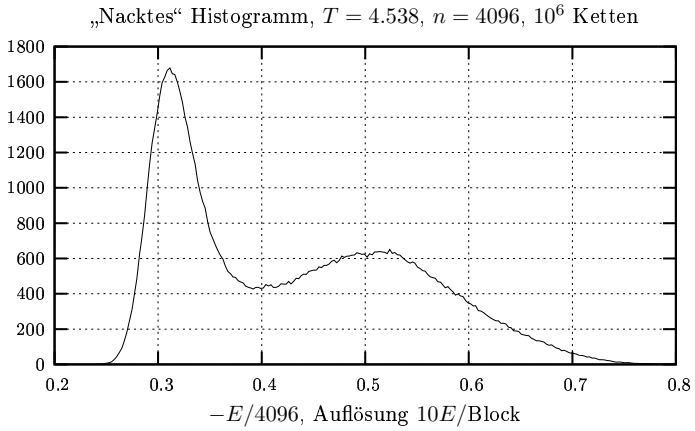


Abbildung 6.8: Das „nackte“ Histogramm zu Abb. 6.7. D.h. hier sind die Anzahlen der Ketten aus derselben Simulation mit bestimmten Energien aufgezeichnet. Es sind wieder Energieblocks der Breite $\triangle E = 10$ verwendet worden.

zum Niedrigenergiepeak, etwa 3 mal so viel. Im gewichteten Histogramm war das Verhältnis der Peakhöhen nicht so groß, da Polymere mit niedrigen Energien ein deutlich höheres Gewicht haben.

Wenn man hier von einem Phasenübergang spricht, ist die Frage, wieviele Polymere sich in der Hochenergiephase befinden und wieviele in der Niedrigenergiephase. Ein Zustand soll in der Niedrigenergiephase sein, wenn seine Energie niedriger ist als diejenige, bei der das Histogramm sein Minimum hat[3] bzw. analog für die Hochenergiephase. Ich berechne nun die Summe aller Ketten bis zu der Energie, bei der das Histogramm sein Minimum hat und die Summe aller Ketten mit größerer Energie sowie die Energie, bei der genausoviele Polymere mit niedrigerer Energie zum Histogramm beitragen, wie Polymere mit höherer Energie, also die „Mitte“ des Histogramms.

Das Minimum des Histogramms in Abb. 6.8 liegt bei $x = 0.392$ bzw. $E = -1605$. Die Anzahl der Polymere mit höherer Energie („links“) ist 47028, die Anzahl der Polymere mit niedrigerer Energie („rechts“) ist 53272. Die Differenz beträgt hier 6244 Ketten, also etwa 0.6% der Gesamtanzahl an Ketten. In Anbetracht dieses kleinen Fehlers kann man sagen, daß „rechts“ und „links“ des Minimums gleich viele Ketten zum Histogramm beitragen[4]. Die gleiche Analyse ergibt für das gewichtete Histogramm in Abb. 6.7 analog:

- $x = 0.392$ bzw. $E = -1605$ für das Minimum des Histogramms, genau wie für das gewichtete Histogramm in Abb. 6.8,

- $x = 0.469$ bzw. $E = -1920$ als der Punkt, an dem „rechts“ und „links“ gleich viel Gewicht verteilt ist.

[3]Es gibt noch viele weitere Entscheidungskriterien, wann ein Zustand in einer bestimmten Phase ist. Das hier verwendete ist sicherlich das Einfachste.

[4]Die Grenze, bei der die Differenz der Anzahlen der Ketten unter den Peaks minimal ist, liegt bei $x = 0.4101$ bzw. $E = -1680$. Sie ist dort 75.

6.2.1 Untersuchung des Algorithmus II

Die Laufzeit des Algorithmus ist für große Polymere sehr hoch, deswegen ist es sinnvoll, nach Implementationen zu suchen, die die Laufzeit deutlich verringern bzw. es erlauben, in gleicher Zeit eine sehr viel bessere Statistik zu erhalten. Eine erste Idee dazu ist die Parallelisierung des Problems. Bei PERM bzw. nPERM bietet sich folgende Möglichkeit an:

- Starte eine Simulation und erzeuge damit eine bestimmte Anzahl von Ketten, z.B. 100 000,
- kopiere den Zustand der Simulation und lasse n unabhängige Simulationen weiterlaufen, z.B. $n = 10$.

Der erste Punkt soll zur Einregulierung der oberen und unteren Grenzen $W^{<,>}$ für alle Kettenlängen dienen. Sind diese relativ stabil[5], werden neue Simulationen mit diesen Startparametern initialisiert. Die Grenzen $W^{<,>}$ sind auch innerhalb der verteilten Prozesse variabel und entwickeln sich i.a. auch unterschiedlich.

Die Erwartung ist nun, daß z.B. 10 Prozesse, die jeweils 100 000 Polymere erzeugen, dasselbe Histogramm liefern, wie ein *single run*, der 1 000 000 Polymere erzeugt. Ich habe also genau wie beschrieben 10 parallele Prozesse jeweils 100 000 Ketten erzeugen lassen[6] und das Histogramm mit dem in Abb. 6.7, welches genau das eines *single runs* mit 10^6 Ketten ist, verglichen. Das Ergebnis zeigt Abb. 6.9. Es weicht sehr stark von dem vorher erhaltenen Histogramm ab. Besonders auffällig ist der Ausläufer im Niedrigenergiebereich. Um nun diese Abweichung weiter verfolgen zu können, müssen wir uns ansehen, welche Daten genau in die Verteilung eingehen. Dazu sehen wir uns zuerst an, wieviele Polymere in jeder *tour* entstehen.

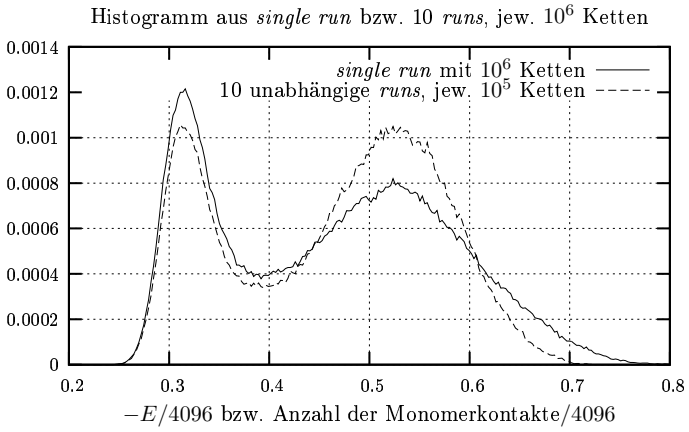


Abbildung 6.9: Vergleich des Histogramms aus dem *single run* (siehe Abb. 6.7) mit dem durch Summation der Einzelhistogramme der 10 parallelen, kurzen *runs* entstandenen Histogramm. Deutlich zu sehen ist der Unterschied vor allem im Niedrigenergiesektor.

[5] Was das heißt, werden wir später sehen.

[6] Natürlich darf man nicht sofort abbrechen, wenn eine bestimmte Anzahl von Polymeren entstanden ist, da dann die Gewichte aller womöglich noch im Speicher vorhandenen Kopien verlorengingen. Tatsächlich lasse ich die *tour*, in der das $n$te Polymer entsteht noch zu Ende laufen und breche dann ab.

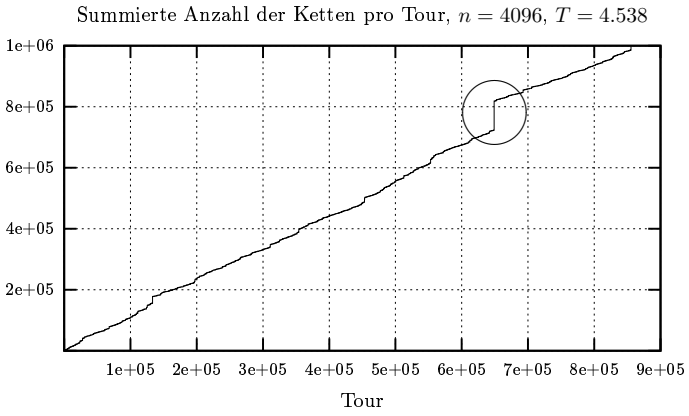


Abbildung 6.10: Der Knackpunkt: Gezeigt ist die summierten Anzahl der Ketten die pro *tour* entstehen. *Eingekreist* ist der deutlichste Sprung, d.h. eine *tour*, bei der sehr viele, i.a. stark korrelierte Ketten entstehen. Es handelt sich wieder um dieselbe Simulation wie in Abb. 6.7.

Abbildung 6.10 zeigt die Summe der bisher entstandenen Ketten pro *tour* zur in Abb. 6.7 betrachteten Simulation. Es fällt zuerst auf, daß pro *tour* im Mittel etwa 1 Polymer entsteht, der „Anstieg“ der Funktion ist etwa 1. Bei genauerem Hinsehen sieht man aber auch, daß es Touren gibt, in denen sehr viele Polymere entstehen, von denen man annehmen muß, daß sie stark korreliert sind. Der markanteste Sprung (im Bild eingekreist) passiert bei der *tour* in der Nähe von 650 000 (genau bei *tour* 648 951). Dort entstehen in einer *tour* etwa 100 000 Polymere (genau 95 127). Wenn diese stark korreliert sind, d.h. im Besonderen, wenn diese sich alle in einer Phase befinden, verfälscht das stark das Histogramm. Untersuchen wir also genau die in der markierten *tour* entstandenen Ketten. In Abb. 6.11 sieht man den Anteil des Histogramms, der allein aus den Ketten dieser einen *tour* verursacht wird im Vergleich zum gesamten Histogramm. Wie man sieht, ist genau diese *tour* „schuld“ an dem Ausläufer des Histogramms des *single runs* gegenüber dem Histogramm aus den einzelnen parallelen *runs* (siehe Abb. 6.9). Genau die Ketten aus dieser *tour* erzeugen den Ausläufer. Da sie aber alle stark korreliert sind und selbst nicht der „richtigen“ Verteilung genügen, verfälschen sie in dem Moment sehr stark das Histogramm. Erst nach sehr langer Simulationszeit würde dieser Einfluß wieder kompensiert werden[7].

Da Touren, die sehr viele, stark korrelierte Ketten erzeugen[8] die Statistik wie gesehen stark verfälschen, kann man versuchen, diese zu unterdrücken. Dazu gibt es einige Ansätze, von denen ich einen genauer untersucht habe:

a) Verwirf alle Touren mit mehr als x Ketten.

b) Erhöhe die obere Grenze $W^{>}$ mit der Anzahl der bereits entstandenen Ketten in dieser *tour*, z.B. $W'^{>} = W^{>}\left(1 + 10^3\, n\right)^2$, wobei n die Zahl der bereits entstandenen Ketten in dieser *tour* ist [37].

[7] Auch bei den kurzen parallelen Simulationen habe ich die analogen Plots zu Abb. 6.10 für alle einzelnen *runs* untersucht, aber keine derart großen Sprünge gefunden.

[8] Nicht in allen Touren, die sehr viele Ketten erzeugen, sind diese auch stark korreliert. Ich habe Beispiele gefunden, in denen sich die Ketten aus einer *tour* etwa der „wahren“ Verteilung gemäß verteilen.

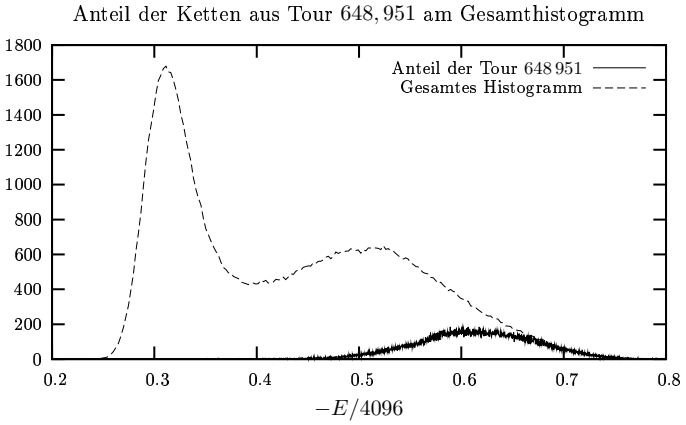


Abbildung 6.11: Der Anteil der Ketten, die in der *tour* 648 951 (in Abb. 6.10 eingekreist) entstehen im Vergleich mit der Gesamtanzahl der Ketten mit bestimmten Energien (siehe Abb. 6.8). Alle Ketten aus dieser *tour* fallen in den Ausläufer des Niedrigenergiesektors.

c) Führe parallele Simulationen durch und verwirf alle Simulationen, die große „Sprünge“ wie in Abb. 6.10 aufweisen.

Abbildung 6.12 zeigt analog zu Abb. 6.10 die summierte Anzahl der Ketten, die pro *tour* entstehen, wenn ich Methode a) verwende. Ich habe dabei jede *tour* verworfen, die mehr als 1 000 Ketten erzeugt. Sehr deutlich sieht man, daß die Funktion jetzt sehr viel „glatter“ ist, es also keine Touren mehr gibt, die die Statistik allein sehr stark beeinflußen. Wie man auch sieht, brauche ich dadurch aber auch etwa 200 000 Touren mehr, ich verwerfen also etwa jede fünfte *tour*. Im *inset* in Abb. 6.12 sieht man, daß zu Anfang beide Simulationen tatsächlich identisch sind.

Abbildung 6.13 zeigt das Histogramm, welches aus dieser modifizierten Version entsteht. Links sieht man das Histogramm im Vergleich mit dem Histogramm aus dem „normalen“ *single run*, rechts das Histogramm nach 1 000 000 Ketten im Vergleich mit dem nach 500 000 Ketten. Man sieht, daß das Histogramm stark von dem erwarteten Ergebnis abweicht und daß es sich offenbar auch nicht stark ändert, zumindest nicht während der von mir simulierten Zeiten. Dieses Resultat ist einigermaßen ernüchternd und zeigt sofort die Nachteile von a) auf. Bei der Interpretation sollte man vorsichtig sein, es gibt mehrere Möglichkeiten:

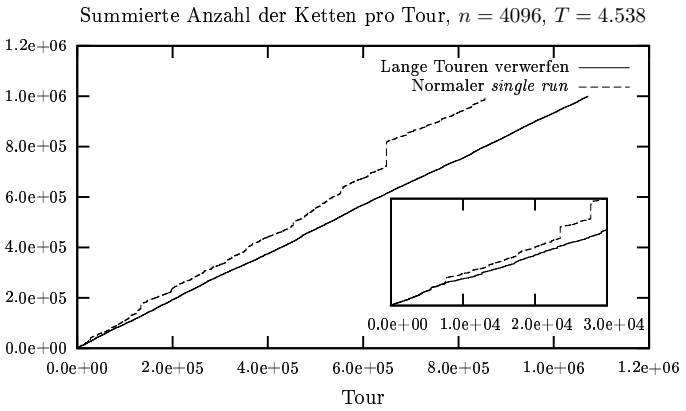


Abbildung 6.12: Die summierte Anzahl der Ketten, die pro *tour* entstehen, wenn ich Methode a) zur Vermeidung langer Touren benutze. Im Vergleich dazu nocheinmal der Plot aus Abb. 6.10 als gestrichelte Linie. Das *inset* zeigt den Anfang der Simulation. Während der ersten $\sim$ 8 000 Touren sind beide Simulationen identisch.

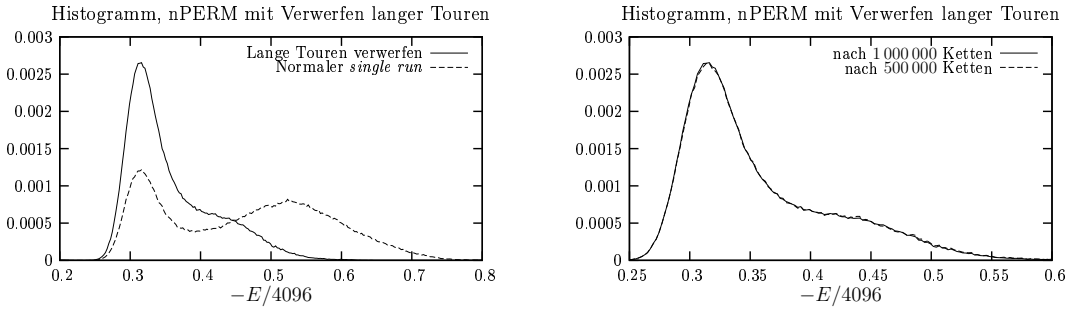


Abbildung 6.13: Das Histogramm aus 10^6 Ketten, nachdem jede *tour* mit mehr als 10^3 Ketten verworfen wurde im Vergleich mit dem Histogramm aus Abb. 6.7 (**links**). Der Niedrigenergiepik fehlt fast vollständig. **Rechts** das Histogramm nach 10^6 Ketten im Vergleich mit dem nach 5×10^5 Ketten. Beide sind nahezu identisch.

- Methode a) erzeugt prinzipiell falsche Ergebnisse. Es ist offenbar so, daß Polymere mit niedrigen Energien bevorzugt in Touren entstehen, die sehr viele Polymere hervorbringen. Oder besser gesagt: In der das wirkliche Gewicht eines Polymers auf sehr viele Kopien verteilt ist. Wenn man diese nun systematisch verwirft, verwirft man damit systematisch auch den Niedrigenergiepeak, bzw. verwirft ihn bevorzugt gegenüber dem Hochenergiepeak.

- Die Methode ist nicht prinzipiell falsch, nur muß man wesentlich länger warten, ehe genug niedrigenergetische Polymere in kurzen Touren entstehen.

Abbildung 6.14 untersucht den letzten Punkt näher. Sie zeigt die Entwicklung des Mittelwertes der erfolgreichen Touren, d.h. der Touren, in denen überhaupt Ketten bis zu voller Länge entstehen, sowie des Mittelwertes der Ketten, die pro (erfolgreicher) *tour* entstehen. Man sieht, daß etwa jede 10. Tour überhaupt nur Ketten liefert und daß in jeder dieser Touren im Mittel 10 Polymere entstehen. Wenn man bedenkt, daß pro *tour* effektiv nur das Gewicht genau eines Polymers entsteht, heißt das, ich habe eine effektive Statistik von etwa 100 000 Ketten. Das ist natürlich sehr wenig. Man könnte nun weitere Simulationen durchführen, in denen diese Rate erhöht wird.

nPERM bietet die Möglichkeit, durch einen Parameter in der Berechnung der Grenzen $W^{<,>}$ die Anzahl der erfolgreichen Touren zu beeinflußen. Diese Untersuchungen und deren Ergebnisse sind jedoch noch nicht ausreichend durchgeführt bzw. analysiert worden, so daß dies eine Aufgabe für die weitere Forschung bleibt.

Möglichkeit c) ist im Endeffekt nur eine andere Realisierung von a), die Untersuchung von b) steht ebenso noch aus. Ich favorisiere deswegen zu diesem Zeitpunkt die parallele Implementation ohne Selektion. Abbildung 6.15 zeigt das Resultat einer solchen Implemetierung im Vergleich zu den bereits in Abb. 6.9 gezeigten. Hier sind nun 10 Simulationen mit jeweils 300 000 erzeugten Ketten durchgeführt worden, insgesamt tragen also 3×10^6 Ketten bei.

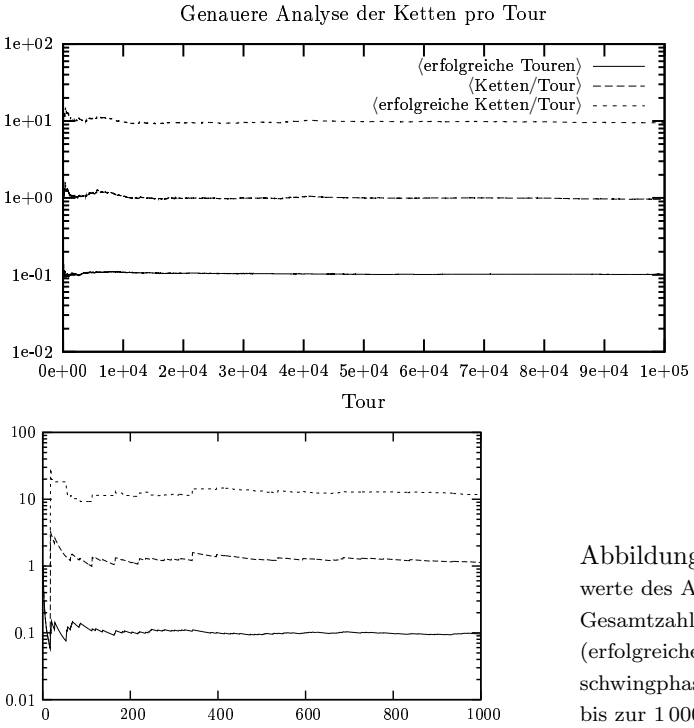


Abbildung 6.14: Die Entwicklung der Mittelwerte des Anteils der erfolgreichen Touren an der Gesamtzahl der Touren sowie der Ketten, die pro (erfolgreicher) Tour entstehen. **Links** die „Einschwingphase“, die Entwicklung der Mittelwerte bis zur 1 000. Tour (**oben** bis zur 100 000. Tour).

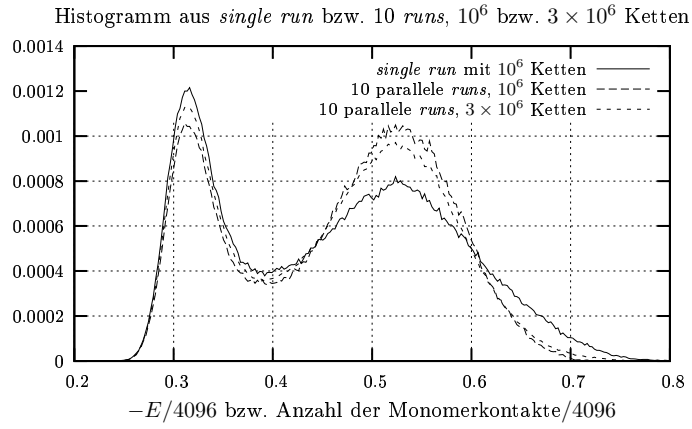


Abbildung 6.15: Vergleich des Histogramms aus dem *single run* (siehe Abb. 6.7) mit denen durch Summation der Einzelhistogramme der 10 parallelen *runs* entstandenen Histogramme mit ebenfalls 10^6 bzw. 3×10^6 Ketten.

6.2.2 Untersuchung der Phasen

Zur näheren Untersuchung des Phasenübergangs bzw. der zwei Phasen benutze ich Polymere der Kettenlänge 16 384. Eine Abschätzung für die Übergangstemperatur findet sich wieder in [35], dort findet man analog zu Abb. 6.6 eine Abbildung (hier Abb. 6.16), die die Energiehistogramme in der Nähe der Übergangstemperatur zeigt.

Für die dort angegebenen Temperaturen $T_1 = 5.008$ bzw. $T_2 = 5.039$ habe ich ebenfalls diese Histogramme aufgenommen, zu sehen in Abb. 6.17. Qualitativ erhalte ich die gleichen Ergebnisse, die Ausläufer der Peaks verhalten sich jedoch wieder unterschiedlich.

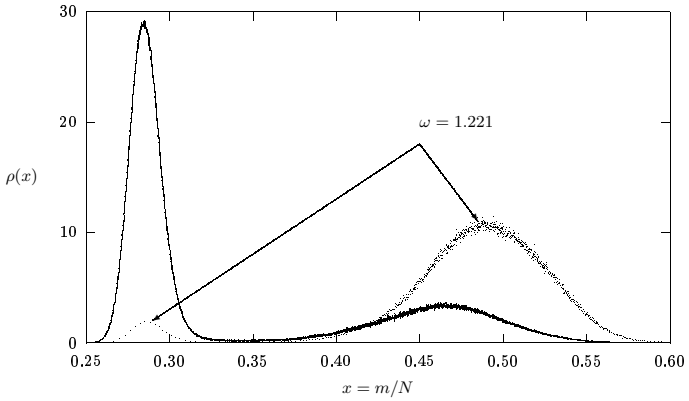


Abbildung 6.16: *Figure 13* aus http://arxiv.org/e-print/cond-mat/9907434 [35]. „*Internal energy density distributions at* $\omega = 1.2195$ *and* 1.2210 *for* $N = 16384$.“ $x = m/N$ bezeichnet wieder die Kontakte pro Monomer, die zur Energie beitragen. $\omega = 1.221$ entspricht $T = 5.008$, $\omega = 1.2195$ entspricht $T = 5.039$.

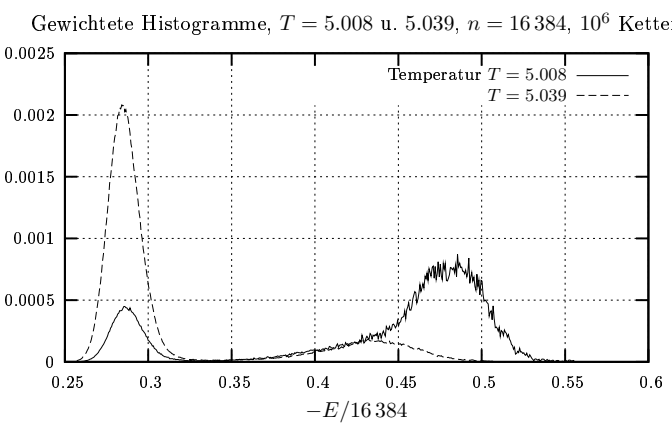


Abbildung 6.17: Histogramme von jeweils 10^6 Polymeren der Länge $n = 16\,384$ bei den Temperaturen $T_1 = 5.008$ und $T_2 = 5.039$. Beide weisen schon bzw. noch einen Doppelpeak auf. Die Übergangstemperatur liegt wahrscheinlich zwischen T_1 und T_2.

Offensichtlich liegt die tatsächliche Übergangstemperatur zwischen T_1 und T_2, wobei ich darauf hinweisen möchte, daß dies ein für die bisherigen Verhältnisse relativ schmales Temperaturintervall ist[9]: $T_2 - T_1 = 0.031$. Für die folgenden Ergebnisse, die alle einer Simulation entstammen, habe ich eine Temperatur $T_1 < T = 5.023 < T_2$ gewählt.

Abbildung 6.18 zeigt die „Zeitreihe“ dieser Simulation, d.h. hier die Energie der einzelnen entstandenen Polymere in fortlaufender Reihenfolge. Das Signal erinnert sehr stark an „typische“ Energiezeitreihen von Modellen am 1. Ordnung Phasenübergang (Vgl. z.B. [36] *Figure 2*). Am Phasenübergang „springt“ die Zeitreihe zwischen zwei (oder i.a. mehr) Phasen spontan hin und her. Genau dieses Verhalten zeigt auch Abb. 6.18. Das Springen zwischen den Phasen spiegelt sich auch stark in der Entwicklung der Schranken $W^{<,>}$ wieder (siehe Abb. 6.19). Die ersten etwa 150 000 Ketten befinden sich alle in der Hochenergiephase, die Grenzen stabilisieren sich sehr schnell, schon nach etwa 100 Touren fluktuieren sie nur noch schwach im Vergleich zur Differenz der beiden Grenzen. Sobald aber der Übergang in die Niedrigenergiephase erfolgt, werden die Grenzen sehr schnell sehr viel größer, da die Gewich-

[9] Z.B. ist aus Abb. 6.3 ersichtlich, daß sich die Fluktuationen von d_{ee} und r_{gyr} um ihren Maximalwert über einen wesentlich größeren Temperaturbereich kaum ändern. Das macht eine genaue Bestimmung der Übergangstemperatur sehr schwierig.

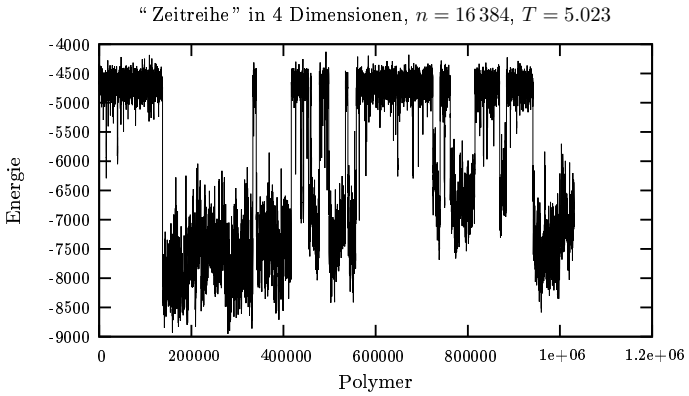


Abbildung 6.18: Die „Zeitreihe“ der Simulation mit Polymeren der Länge $n = 16\,384$ bei der Temperatur $T = 5.023$. Gezeigt sind die Energien der entstandenen Polymere während der Simulation in der Reihenfolge ihrer Entstehung. Deutlich zu sehen ist das „Springen“ zwischen zwei Phasen hoher bzw. niedriger Energie.

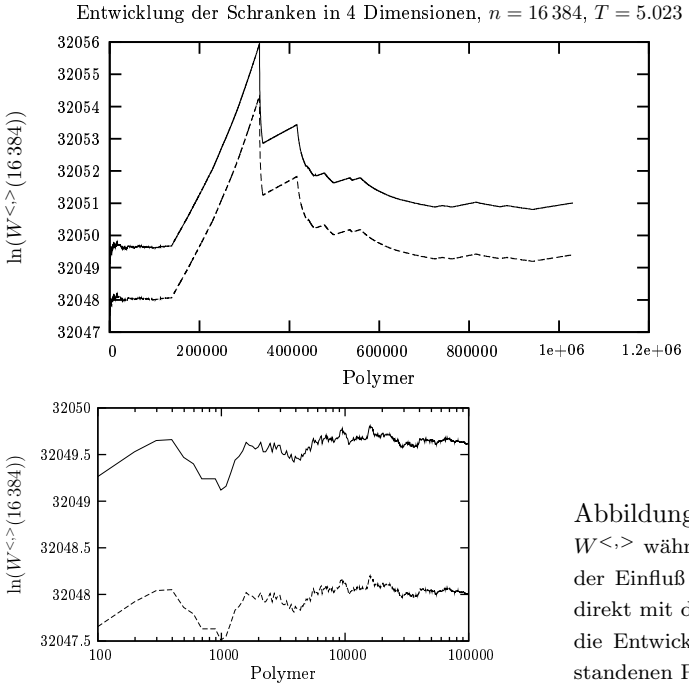


Abbildung 6.19: Die Entwicklung der Grenzen $W^{<,>}$ während einer Simulation. Gut sichtbar ist der Einfluß der zwei Phasen. Die „Zeitachse“ ist direkt mit der aus Abb. 6.18 vergleichbar. **Unten** die Entwicklung während der ersten 100 000 enstandenen Polymere.

te der entstehenden Ketten sehr viel größer sind als die vorhergehenden. Das hat zur Folge, daß in der Folgezeit sehr viele Ketten aussterben, da ihr Gewicht unter die untere Schranke fällt, die durch den gerade erwähnten Effekt jetzt sehr hoch liegt.

Vielleicht kann ich an dieser Stelle einige reale Zeiten angeben, die die Simulation in den beschriebenen Phasen verbracht hat:

- In der ersten Hochenergiephase (bis etwa 150 000 Polymere) verweilte die Simulation etwa 3 Tage[10]. In dieser Zeit war für einen Phasenübergang noch kein Anzeichen zu sehen. Das Histogramm beschränkte sich (natürlich) auf den Hochenergiepeak und die Entwicklung der Schranken verlief wenig spektakulär (wie in Abb. 6.21 unten gezeigt).

[10] Simuliert auf einem P4 2400Mhz PC.

- In der ersten Hochenergiephase (etwa Polymer 150 000 bis 350 000) verweilte die Simulation etwa 5 Tage, der Niedrigenergiepeak war nun „schlagartig“ vorhanden und die Schranken wuchsen sehr stark.

- In der zweiten Niedrigenergiephase (ganz kurz um das Polymer 350 000 herum) verweilte die Simulation ebenso ganze 5 Tage, in denen aber nur etwa 5 000 Polymere entstanden! Zur Erinnerung: Zu diesem Zeitpunkt waren die optimalen Grenzen stark überschätzt, wahrscheinlich sind sehr viele Ketten dadurch ausgestorben.

Die gesamte Simulation dauerte etwa 1 Monat. Es dauert also in der Nähe der Übergangstemperatur im Gegensatz zu ersten Vermutungen sehr lange, ehe sich die Grenzen $W^{<,>}$ stabilisieren. Sind diese noch nicht stabil bzw. die Breite zwischen der oberen und der unteren Grenze sehr klein im Gegensatz zu den Fluktuationen der Grenzen, wirkt sich das negativ auf die Effizienz des Algorithmus, d.h. die Anzahl der produzierten Ketten, aus.

Im folgenden möchte ich die Phasen, in denen sich die Polymere befanden, anhand der gemessenen Observablen näher beschreiben. Abbildung 6.20 zeigt Phasenplots der entstandenen Polymere über verschiedenen Variablen, Abb. 6.21 die dazugehörigen Histogramme.

Energie–End-to-End Distance Deutlich erkennt man hier die zwei Phasen, die man auch im Energiehistogramm (analog Abb. 6.7) sehen würde. Die Niedrigenergiephase ist in

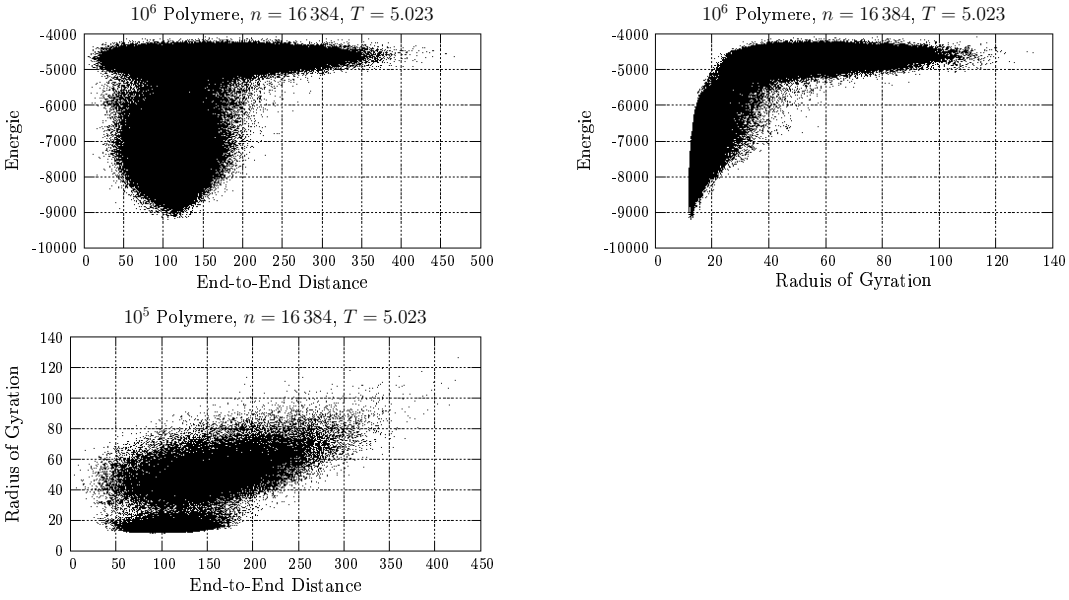


Abbildung 6.20: Die gemessenen Observablen von jeweils 10^6 bzw. 10^5 Polymeren der Kettenlänge $n = 16\,384$ in 4 Dimensionen in Abhängigkeit voneinander. Deutlich zu sehen in den Plots **oben**, welche die Energie beinhalten, die beiden Phasen der niedrigen bzw. hohen Energien, sowie die möglichen geometrischen Verhältnisse in diesen Phasen.

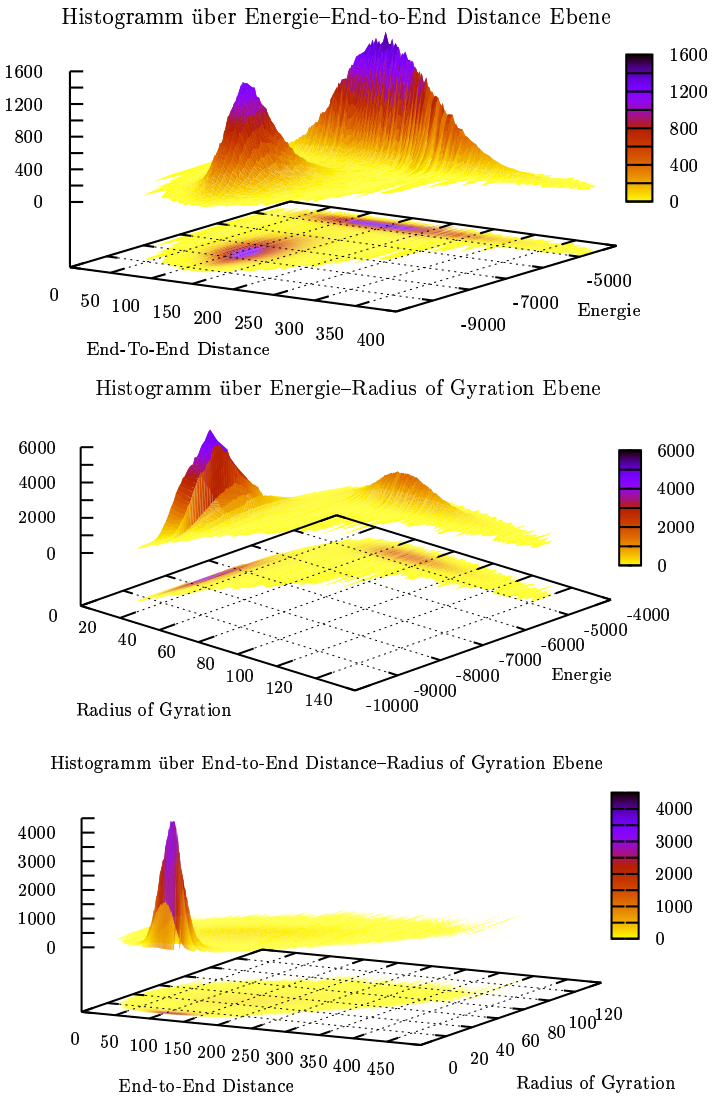


Abbildung 6.21: Die Histogramme aus jeweils 10^6 Polymeren der Kettenlänge $n = 16\,384$ in 4 Dimensionen über verschiedenen Observablen. **Oben** über der Energie–End-to-End Distance Ebene, **Mitte** über der Energie–Radius of Gyration Ebene und schließlich **unten** über der End-to-End Distance und dem radius of Gyration.

der *end-to-end distance* nach oben begrenzt ($d_{\mathrm{ee,max}} \approx 200$), was in diesem Fall ein direktes Maß für die maximale Ausdehnung der in dieser Phase entstandenen Polymere ist. Sie ist deutlich kleiner als die, die in der Hochenergiephase tatsächlich maximal erreicht wird. In der Hochenergiephase entstehen also auch ausgestreckte Polymere. Daß in der Hochenergiephase auch sehr kleine (sogar kleiner als $d_{\mathrm{ee,min}}$ in der Niedrigenergiephase) *end-to-end distances* erreicht werden können ist klar, wenn man sich etwa ein Polymer vorstellt, bei welchem das letzte Monomer sehr nahe (bzw. direkt neben) dem ersten gelegen ist und sonst einen „Kreis“ bildet. In der Niedrigenergiephase ist es sehr viel unwahrscheinlicher, daß die Endmonomere so nahe zusammenkommen, da die Kompaktheit des entstehenden Polymers ein „Zurücklaufen“ der letzten Monomere zum Anfang sehr erschwert. Besser erkennt man die Geometrie der Phasen aber am *radius of gyration*.

Energie–Radius of Gyration Im Prinzip treffen die gerade gemachten Aussagen auch hier zu, nur ist die Interpretation eine etwas andere. Der *radius of gyration* ist ein Maß für die mittlere Entfernung der Monomere vom Polymermittelpunkt. In der Niedrigenergiephase sind die einzelnen Monomere deutlich dichter gepackt als in der Hochenergiephase. Der *radius of gyration*, der tatsächlich angenommen wird hat eine sehr scharfe untere Grenze, welche bei den niedrigsten Energien angenommen wird. In der Hochenergiephase gibt es bei diesen r_{gyr} keine Polymere mehr (im Gegensatz zur Observablen d_{ee}).

Radius of Gyration–End-to-End Distance Dieser Plot ist die notwendige Kombination der beiden oben erwähnten und der Vollständigkeit halber gezeigt. Man sieht an dem Histogramm in Abb. 6.21 einen alles andere sehr stark überragenden einzelnen, relativ „schlanken“ Peak. D.h., daß in der Niedrigenergiephase alle Polymere in einen sehr stark begrenzten geometrischen Raum fallen (nämlich die Region der in Abb. 6.1 gezeigten „Kugeln“), wohingegen sich die Polymere in der Hochenergiephase über (fast) die gesamten Ebene verteilen.

Allgemein fällt noch auf, daß die Phasen, wie auch schon vorher an den Histogrammen zu sehen, nicht strikt voneinander getrennt sind. Auch in dem Raum zwischen den Maxima der Phasen liegen noch viele Polymere.

6.2.3 Zurück in 3 Dimensionen

Nach den Beobachtungen in 4 Dimensionen stellt sich die Frage, ob man auch in 3 Dimensionen das Verhalten eines *pseudo-first-order* Übergangs feststellen kann. Da dieser Effekt auch in 5 Dimensionen gefunden wurde, dort bei deutlich niedrigeren Kettenlängen [38], ist die naive Vermutung, daß man in 3 Dimensionen nun zu längeren Ketten gehen muß, um den Effekt zu finden, wenn er vorhanden ist. Diese Vermutung stützt auch die Tatsache, daß bei bisher betrachteten Kettenlängen ($n = 4096$, siehe Abb. 6.5) noch keinerlei Doppelpeak zu erkennen oder zu vermuten war. Analog zu Abb. 6.5 zeigt Abb. 6.22 die Histogramme für Polymere in 3 Dimensionen der Kettenlänge $n = 16\,384$ in der Nähe des Übergangspunktes.

Anhand der Energiehistogramme ist somit kein außergewöhnliches Verhalten von Polymeren mit endlichen Längen in 3 Dimensionen zu erkennen. Allerdings erfahren geometrische Größen wie die *end-to-end distance* und der *radius of gyration* als Funktion der Kettenlänge und der Temperatur eine deutliche Veränderung ihres Monotonieverhaltens beim Durchgang durch den θ-Punkt. In [31] sieht man Daten, die den Anschein erwecken, daß sich im Limes $n \to \infty$ die inverse Dichte

$$\langle \varrho^{-1} \rangle = r_{\mathrm{gyr}}^3 / n \tag{12}$$

abrupt ändert, was auf einen Phasenübergang 1. Ordnung hindeuten würde. Da die obere kritische Dimension 3 ist, sollte jedoch die Vorhersage der *Mean-Field* Theorie richtig sein, daß der Phasenübergang 2. Ordnung ist. Zum jetzigen Zeitpunkt ist diese Frage wahrscheinlich noch nicht endgültig zu entscheiden.

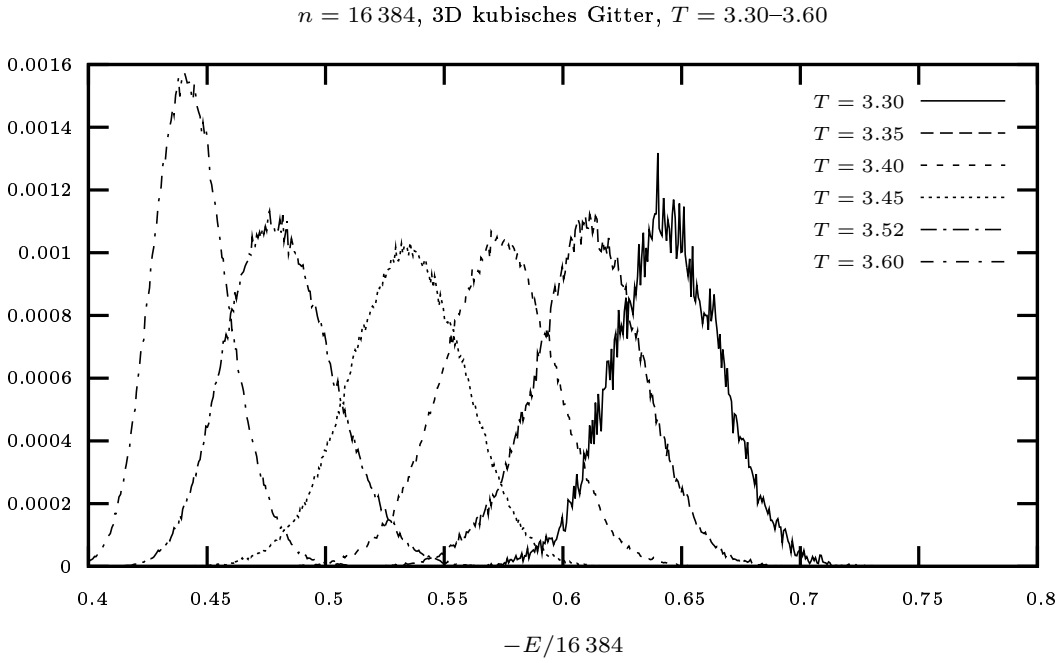


Abbildung 6.22: Die normierten gewichteten Histogramme der Energie bei Temperaturen zwischen $T = 3.30$ (*rechts*) und $T = 3.60$ (*links*) für Polymere der Länge $n = 16\,384$ in 3 Dimensionen. Bei der Übergangstemperatur der Wärmekapazität (siehe Abb. 6.4) ist das Maximum der Verteilung am niedrigsten. Alle Histogramme haben aber immer noch deutlich einen einfachen Pik. (Das Histogramm bei $T = 3.30$ besteht nur etwa aus 1/10 der Ketten im Vergleich zu den anderen, daher die größere Unschärfe.)

Schluß

In dieser Arbeit habe ich ein einfachstes Modell für Proteine, das HP-Modell, untersucht. Das HP-Modell kann z.B. Aussagen treffen über die Grundzustandskonformation einer Monomersequenz, nicht jedoch über die Dynamik der Faltung an sich. Die Aussagen sind sehr stark abhängig vom Gittertyp, der für das Modell gewählt wird. So ist z.B. der Grundzustand von bestimmten Sequenzen auf dem kubischen Gitter in 3 Dimensionen eindeutig, wohingegen er für einen anderen Gittertyp mit gleicher Koordinationszahl stark entartet ist.

Das HP-Modell ist zu stark vereinfacht, um auch nur ansatzweise die Struktur von realen Proteinen wiederspiegeln zu können. Wie wir in Kapitel 1 gesehen haben, ist z.B. die Sekundärstruktur realer Proteine abhängig von Wechselwirkungen, die das Modell gar nicht beinhaltet. Dennoch stößt man bereits mit diesem einfachen Modell, für realistische Proteingrößen, an die Leistungsgrenze heutiger Computer.

Das Modell kann jedoch schon gut strukturelle Übergänge in Polymeren beschreiben, die auftreten, wenn man etwa die Temperatur bzw. die Güte des Lösungsmittels ändert. In Kapitel 5 haben wir an einigen Proteinen ein Verhalten festgestellt, welches nebem dem bekannten Übergang zwischen ausgestreckten und kollabierten Konformationen auf einen weiteren Übergang, den zwischen Zuständen mit maximal kompaktem hydrophoben Kern und geometrisch maximal kompakten Zuständen, schließen läßt [32]. In Kapitel 6 wurde der θ-Übergang zwischen ausgestreckten und kollabierten Polymeren in drei und vier Dimensionen näher untersucht.

Natürlich ist das Modell auch an sich interessant, so habe ich z.B. Eigenschaften des PERM Algorithmus' mit diesem Modell gut untersuchen können. Dies ist sehr wichtig, da die Effizienz des Algorithmus' sehr sensibel von diversen Parametern abhängt. Den Algorithmus an einfachsten Modellen zu untersuchen ist also unabdingbar für dessen möglichen Einsatz für kompliziertere, realistischere Modelle. Der erste kleine Schritt hin zu realistischeren Modellen war die Verallgemeinerung des Gittertyps, womit ich direkt die Erhöhung der Koordinationszahl k der Gitter meine. Ich habe einige Proteine auf dem Dreiecksgitter in 2 Dimensionen bzw. auf dem fcc-Gitter in 3 Dimensionen untersucht, welche vorher auf dem quadratischen bzw. kubischen Gitter untersucht worden. Ich habe für diese Proteine obere Schranken für die Grundzustandsenergie auf diesen verallgemeinerten Gittern angegeben.

Diese Arbeit war für mich ein Einstieg in die Arbeit mit Proteinmodellen und deren Simulationsmethoden. Daraus ergeben sich natürlich lange Listen möglicher nächster Schritte bzw. der offen gebliebenen Fragen. Im folgenden werde ich jeweils einige davon auflisten.

Offene Fragen

- Mit PERM habe ich Grundzustandssuchen für Proteine durchgeführt, deren Längen fernab derer liegen, für die Resultate aus exakten Enumerationen vorhanden bzw. zu erwarten sind. Kann man trotzdem ein Kriterium finden, um zu entscheiden ob man die minimale Energie schon erreicht hat bzw. wie weit man davon noch entfernt ist, wenn man einen Zustand geringer Energie findet?
- In Kap. 6 wird für Homopolymere in 4 Dimensionen ein Verhalten sichtbar, daß typisch ist für 1. Ordnung Phasenübergänge. Wie verhält sich das *finite size scaling* für Homopolymere in 4 Dimensionen?
- In [35] gibt es Anzeichen, daß der beschriebene Effekt nur in einem bestimmten Kettenlängenfenster auftritt. Verschwindet er tatsächlich wieder für längere Polymere?
- Gibt es dieses Fenster auch in 3 Dimensionen? Ich habe bis zu Kettenlängen von $n \approx 16\,000$ kein ähnliches Verhalten zu dem in 4 Dimensionen finden können.

Die nächsten Schritte

- Als erstes bietet sich hier die Lösung vom Gitter an. Das Gitter gab uns bis jetzt automatisch die Garantie, daß sich 2 Monomere nicht beliebig nahe kommen können. Löst man sich vom Gitter muß man somit Potentiale benutzen, die diesen Fakt wieder berücksichtigen. Ein solches Modell ist z.B. das AB-Modell [39, 40]. Es kennt jedoch, wie das HP-Modell nur zwei Monomertypen. Ebenso repräsentiert jedes Monomer im Modell noch eine komplette Aminosäure. Abbildung 5 oben in [39] sieht übrigens meinen Abbildungen in Kap. 5.1.2 sehr ähnlich! Welche Schlüsse sind daraus erlaubt?
- Von den letzten beiden Beschränkungen kann man sich ebenso lösen. Man kommt dann zu *all atom* Modellen, in denen alle existierenden Aminosäuren in ihrer kompletten Struktur aus einzelnen Atomen berücksichtigt werden. Für die *all atom* Repräsentation existieren verschiedene Modellkraftfelder z.B. ECEPP [41]. Ist PERM für diese Modelle noch effektiv?
- Abgesehen von Modelländerungen, lohnt es sich auch, die Methoden, die zum Einsatz kamen, auszuweiten. Zum Beispiel kann man PERM mit der Idee der multikanonischen Simulation verbinden. Die Resultate die damit bereits erzielt worden sind vielversprechend. Relativ mühelos fand der so verfeinerte Algorithmus z.B. für Sequenz ($\mathrm{Seq}\,103_1$) eine neue obere Schranke für die Grundzustandsenergie auf dem kubischen Gitter ($E_{\min} = -56$) [32].

- Wie gesehen hängt die Effizienz von PERM bei der Grundzustandssuche von der Temperatur ab, bei der simuliert wird. Man kann nun z.B. versuchen, die Temperatur auf nützliche Art und Weise während der Grundzustandssuche zu variieren (*simulated annealing*).

♢

Mein
erster Dank
gilt meiner Frau,
die während dieser
Arbeit oft zu wenig mei-
ner Zeit bekam. Trotzdem gab
sie mir viel Kraft für die besonders
anstrengenden Zeiten in diesem Jahr. Be-
sonderer Dank geht an meine Betreuer Prof.
Dr. W. Janke und Dr. M. Bachmann die jederzeit
auf meine entstehenden Probleme eingingen und viel Zeit
investierten, um über diese zu reden. Ebenso geht mein Dank
an R. Schiemann für Fehlersuche und ständige Bereitschaft, über die
„kleinen“ Probleme zu diskutieren, sowie an S. Wenzel für die
Zusammenarbeit an vielen kleinen Programmen, die das
Leben erleichtern. Die Simulationen liefen auf Rech-
nern des Instituts für Theoretische Physik der
Universität Leipzig. Mein Dank gilt den
Menschen, die dieses System während
meiner Diplomarbeitszeit auf-
rechterhielten: M. Wei-
gel, M. Hellmund,
E. Bittner und
A. Nussbau-
mer.

♢

Anhang A

Techniken

A.1 Analyse der *self-avoiding walks* I

Idee war, durch numerische Verfahren die Zahlen μ und γ in Gl. (3.1) zu bestimmen. Ich hatte die Größe r_n definiert als

$$r_n = \mu \left[1 + (\gamma - 1)\, n^{-1} + \frac{1}{2}\gamma(\gamma - 1) n^{-2} + \mathcal{O}\left(n^{-3}\right) \right] . \tag{1}$$

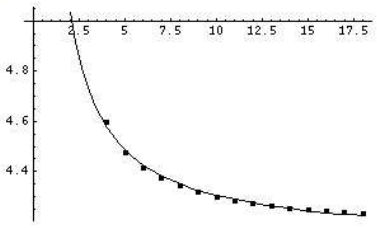
Durch Auszählen haben wir die Anzahl c_n aller Wege erhalten. Ein linearer Fit (Fit bis zur 1. Ordnung) liefert μ ohne bias als Schnittpunkt mit der Ordinate (r_n aufgetragen gegen n^{-1}) und γ mit bias aus dem Anstieg der Funktion. Mit bias, da der Anstieg auch μ mit dessen Fehler enthält. Ich habe auch einen Fit bis zur 2. Ordnung durchgeführt. Die Ergebnisse werden genauso abgelesen, allerdings erwarte ich den Fehler kleiner. Die folgenden Abbildungn A.1 und A.2 zeigen die Analyse mit *Mathematica*.

```
In[37]:= liste1 = Table[x, {x, 4, 18}]

Out[37]= {4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18}

In[27]:= liste2 = {5, 23, 103, 455, 1991, 8647, 37355, 160689, 6.888610*10^5, 2.944823*10^6,
           1.255920*10^7, 5.345578*10^7, 2.271319*10^8, 9.636276*10^8, 4.082888*10^9, 1.727899*10^10}

Out[27]= {5, 23, 103, 455, 1991, 8647, 37355, 160689, 688861., 2.94482×10^6,
          1.25592×10^7, 5.34558×10^7, 2.27132×10^8, 9.63628×10^8, 4.08289×10^9, 1.7279×10^10}

In[39]:= liste3 = Table[N[liste2[[x]] / liste2[[x - 1]]], {x, 2, 16}]

Out[39]= {4.6, 4.47826, 4.41748, 4.37582, 4.34304, 4.32, 4.30167,
          4.28692, 4.27492, 4.26484, 4.2563, 4.24897, 4.24259, 4.237, 4.23205}

In[40]:= data = Transpose[{liste1, liste3}]

Out[40]= {{4, 4.6}, {5, 4.47826}, {6, 4.41748}, {7, 4.37582}, {8, 4.34304},
          {9, 4.32}, {10, 4.30167}, {11, 4.28692}, {12, 4.27492}, {13, 4.26484},
          {14, 4.2563}, {15, 4.24897}, {16, 4.24259}, {17, 4.237}, {18, 4.23205}}

In[41]:= f1[x_] = Fit[data, {1, x^(-1)}, x]

Out[41]= 4.12362 + 1.8207/x
```

```
In[48]:= f2[x_] = Fit[data, {1, x^(-1), x^(-2)}, x]
```

$$\text{Out[48]= } 4.16393 + \frac{2.44941}{x^2} + \frac{1.1181}{x}$$

```
In[46]:= gr2 = Plot[f1[x], {x, 2, 18}, DisplayFunction → Identity]
In[50]:= gr3 = Plot[f2[x], {x, 2, 18}, DisplayFunction → Identity]
In[47]:= Show[gr1, gr2]
```




```
In[51]:= Show[gr1, gr3]
```

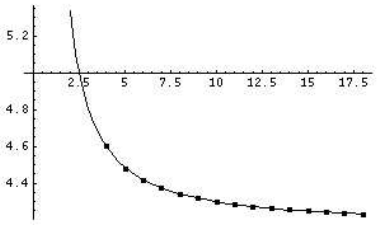


Abbildung A.1: Fit an die Meßwerte nach Gl. (3.3) für das 2D Dreiecksgitter. Der obere Plot zeigt den Fit bis zur 1., der untere den Fit bis zur 2. Ordnung.

```
In[53]:= Regress[data, {1, x^(-1)}, x, OutputList → {BestFit}]
```

$$\text{Out[53]= } \{\text{BestFit} \to 4.12362 + \frac{1.8207}{x},$$

		Estimate	SE	TStat	PValue
ParameterTable →	1	4.12362	0.00497762	828.431	0.
	$\frac{1}{x}$	1.8207	0.040217	45.2718	1.11022×10^{-15}

RSquared → 0.993697, AdjustedRSquared → 0.993212, EstimatedVariance → 0.000073887,

		DF	SumOfSq	MeanSq	FRatio	PValue
ANOVATable →	Model	1	0.151434	0.151434	2049.54	1.11022×10^{-15}
	Error	13	0.000960531	0.000073887		
	Total	14	0.152395			

```
In[54]:= Regress[data, {1, x^(-1), x^(-2)}, x, OutputList → {BestFit}]
```

$$\text{Out[54]= } \{\text{BestFit} \to 4.16393 + \frac{2.44941}{x^2} + \frac{1.1181}{x},$$

		Estimate	SE	TStat	PValue
	1	4.16393	0.00387877	1073.52	0.
ParameterTable →	$\frac{1}{x}$	1.1181	0.0633981	17.6362	6.02447×10^{-10}
	$\frac{1}{x^2}$	2.44941	0.21684	11.2959	9.45736×10^{-8}

RSquared → 0.999458, AdjustedRSquared → 0.999368, EstimatedVariance → 6.88071×10^{-6},

		DF	SumOfSq	MeanSq	FRatio	PValue
ANOVATable →	Model	2	0.152312	0.0761561	11068.1	0.
	Error	12	0.0000825685	6.88071×10^{-6}		
	Total	14	0.152395			

Abbildung A.2: Regressionsanalyse zu den Fits in Abb. A.1. Auch hier wieder bis zur 1. bzw. zur 2. Ordnung. Dokumentation der Begriffe und Größen z.B. in [42].

Daraus kann man für μ

$$\mu_{1.\mathrm{Ord}} = 4.124 \pm 0.005 \quad \text{und} \quad \mu_{2.\mathrm{Ord}} = 4.164 \pm 0.004 \tag{2}$$

und für $\mu(\gamma - 1)$

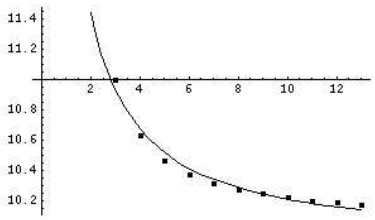
$$\mu(\gamma-1)_{1.\mathrm{Ord}} = 1.82 \pm 0.04 \quad \text{und} \quad \mu(\gamma-1)_{2.\mathrm{Ord}} = 1.12 \pm 0.06 \tag{3}$$

ablesen. Für γ erhält man so

$$\gamma_{1.\mathrm{Ord}} = 1.44 \quad \text{und} \quad \gamma_{2.\mathrm{Ord}} = 1.27\,. \tag{4}$$

Abbildungen A.3 und A.4 zeigen in Auszügen völlig analog die Fits und Regressionsanalysen für das Tetraedergitter.

```
In[11]:= data = Transpose[{liste1, liste3}]
Out[11]= {{3, 11.}, {4, 10.6364}, {5, 10.4701}, {6, 10.3763}, {7, 10.3173},
          {8, 10.2764}, {9, 10.2458}, {10, 10.2221}, {11, 10.2032}, {12, 10.1878}, {13, 10.175}}
In[12]:= f1[x_] = Fit[data, {1, x^(-1)}, x]
Out[12]= 9.90234 + 3.08583/x
In[13]:= f2[x_] = Fit[data, {1, x^(-1), x^(-2)}, x]
Out[13]= 10.0906 + 6.08112/x^2 + 0.691644/x
In[21]:= Show[gr1, gr2]
```




```
In[22]:= Show[gr1, gr3]
```

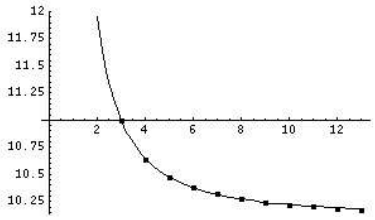


Abbildung A.3: Fit an die Meßwerte nach Gl. (3.3) für das Tetraeder- bzw. fcc-Gitter (Auszüge). Der obere Plot zeigt den Fit bis zur 1., der untere den Fit bis zur 2. Ordnung.

Hieraus erhält man für μ

$$\mu_{1.\mathrm{Ord}} = 9.90 \pm 0.03 \quad \text{und} \quad \mu_{2.\mathrm{Ord}} = 10.091 \pm 0.007 \tag{5}$$

```
In[24]:= Regress[data, {1, x^(-1)}, x, OutputList → {BestFit}]

                                                            Estimate   SE          TStat     PValue
Out[24]= {BestFit → 9.90234 + 3.08583/x, ParameterTable → 1      9.90234    0.0262156   377.727   0.
                                                       1/x    3.08583    0.153488    20.1046   8.67113 × 10^-9 ,

   RSquared → 0.978219, AdjustedRSquared → 0.975798, EstimatedVariance → 0.00151414,

                       DF   SumOfSq     MeanSq       FRatio    PValue
   ANOVATable → Model  1    0.612011    0.612011     404.196   8.67113 × 10^-9 }
                Error  9    0.0136273   0.00151414
                Total  10   0.625638

In[25]:= Regress[data, {1, x^(-1), x^(-2)}, x, OutputList → {BestFit}]

Out[25]= {BestFit → 10.0906 + 6.08112/x^2 + 0.691644/x ,

                          Estimate   SE           TStat     PValue
                    1     10.0906    0.00749395   1346.5    0.
   ParameterTable → 1/x   0.691644   0.0895936    7.71979   0.0000563932 ,
                    1/x^2 6.08112    0.223511     27.2073   3.58865 × 10^-9

   RSquared → 0.999767, AdjustedRSquared → 0.999709, EstimatedVariance → 0.0000182126,

                       DF   SumOfSq       MeanSq         FRatio    PValue
   ANOVATable → Model  2    0.625492      0.312746       17172.    2.88658 × 10^-15 }
                Error  8    0.000145701   0.0000182126
                Total  10   0.625638
```

Abbildung A.4: Regressionsanalyse zu den Fits in Abb. A.3.

und für $\mu(\gamma - 1)$

$$\mu(\gamma - 1)_{1.\mathrm{Ord}} = 3.09 \pm 0.15 \quad \text{und} \quad \mu(\gamma - 1)_{2.\mathrm{Ord}} = 0.69 \pm 0.09\,. \tag{6}$$

Für γ erhält man dann

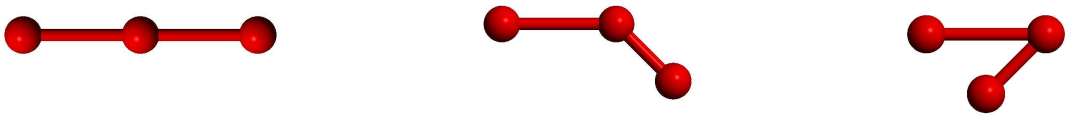
$$\gamma_{1.\mathrm{Ord}} = 1.31 \quad \text{und} \quad \gamma_{2.\mathrm{Ord}} = 1.07\,. \tag{7}$$

A.2 Analyse der *self-avoiding walks* II

Hier werde ich genauer Betrachtung der Symmetriefaktoren aus Kap. 3.2.1 führen. Die Symmetriefaktoren für das 3D sc-Gitter sind bereits in [7] ausführlich dargelegt worden, sie sind: $k = 6$, $s'^{\,\mathrm{planar}} = 4$ und $s'^{\,\mathrm{räumlich}} = 8$.

Für das Dreiecksgitter in 2 Dimensionen ist $s'^{\,\mathrm{planar}}$ auch recht schnell gefunden. Es gibt nur eine Ebene und man braucht genau eine Spiegelachse (die man frei wählen kann), um alle möglichen Konfigurationen aus dieser Spiegelung und den globalen Rotationen (und natürlich der Menge aller nicht durch Symmetrietransformationen ineinander überführbarer Konformationen) erzeugen zu können. Die entsprechende Rotation ist zweizählig, demzufolge ist $s'^{\,\mathrm{planar}} = 2$. Abbildung A.5 veranschaulicht das am Beispiel eines 3mers.

Etwas komplizierter zu bestimmen sind schon die $s_i'^{\,\mathrm{planar}}$ für das fcc-Gitter in 3 Dimensionen. Sieht man sich z.B. Abb. 2.4 an, stellt man fest, daß es hier schon 2 unabhängige Gitterebenen gibt, von denen eine 4zählig ist und eine 2zählig. In der 4zähligen Ebene liegt



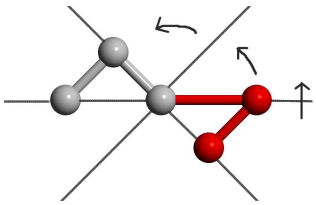


Abbildung A.5: **Oben** die Menge aller nicht durch Symmetrietransformation ineinander überführbaren Konformationen des 3mers auf dem 2D Dreiecksgitter, **links** ein Beispiel, wie die dritte davon durch eine Spiegelung an der waagerechten Achse und zwei globale Drehungen in eine andere Überführt werden kann. Für die Spiegelung hätte man auch jede andere Achse benuzten können und wäre durch mehr oder weniger globale Drehungen zum selben Ergebnis gelangt.

ein Dreiecksgitter, in der anderen ein Quadratgitter. Es folgt $s_1'^{\text{planar}} = 4$, $s_2'^{\text{planar}} = 2$. Analog zu Abb. A.5 zeigt Abb. A.6 die Menge der unabhängigen Konformationen eines 3mers auf dem fcc-Gitter. Naturgemäß liegen diese noch in Ebenen, so daß die $c_{n,i}'^{\text{räumlich}}$ alle Null sind. Zwei dieser Konformationen liegen in Ebenen die 4zählig ineinander überführt werden und in denen ein Dreiecksgitter liegt, die dritte befindet sich auf einem Quadratgitter mit 2zähliger Rotation. Bestimmen wir nach Gl. (3.5) also für diesen Fall c_3/k:

$$c_3/k = (1+4\,c_{n,1}^{\text{planar}} + 2\,c_{n,2}^{\text{planar}} + \sum_j s_j'^{\text{räumlich}} c_{n,j}^{\text{räumlich}}) = 1+4\cdot 2+2\cdot 1+\sum_j s_j'^{\text{räumlich}}\cdot 0 = 11\,. \tag{8}$$

Genau diesen Wert liest man auch in Tab. 3.3 ab. Sehr viel schwieriger wird die Bestimmung der $s_j'^{\text{räumlich}}$. Das soll jedoch nicht mehr Inhalt dieser Arbeit sein.

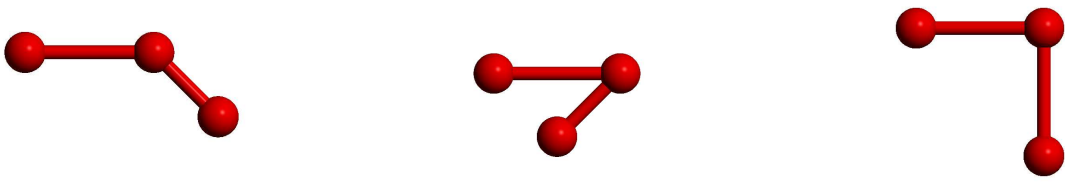


Abbildung A.6: Die Menge aller planaren, nicht durch Symmetrietransformation ineinander überführbaren Konformationen des 3mers auf dem 3D fcc-Gitter. Die Konformationen **links** liegen in einem 2D Dreiecksgitter, welches 4zählig durch das 3d Gitter gedreht werden kann, die Konformation **rechts** liegt in einem 2D Quadratgitter, welches 2zählig ist (siehe auch Abb. 2.4).

A.3 Die Funktion `step(l)`

Hier wird zunächst, auszugsweise, der von mir verwendete Quellcode zu dem in Kap. 4.2 erläuterten Algorithmus gezeigt, implementiert für das 3D sc-Gitter mit 6 nächsten Nachbarn. Der Algorithmus arbeitet hier noch auf einem festen Gitter, so dass z.B. die Energie sehr einfach berechnet werden kann. Man sieht sich dazu einfach die Belegung der benachbarten Gitterplätze an:

```
void berechneenergie(int l)
{
  energie=0;
  if (gitter[xpos+1][ypos][zpos]==1) energie--; // rechter Nachbar
  if (gitter[xpos][ypos+1][zpos]==1) energie--; // oberer Nachbar
  if (gitter[xpos-1][ypos][zpos]==1) energie--; // linker Nachbar
  if (gitter[xpos][ypos-1][zpos]==1) energie--; // unterer Nachbar
  if (gitter[xpos][ypos][zpos-1]==1) energie--; // hinterer Nachbar
  if (gitter[xpos][ypos][zpos+1]==1) energie--; // vorderer Nachbar
  if (sequenz[l-1]==1) energie++; // falsche Linkenergie wieder abziehen
}
```

Das Feld `gitter[x][y][z]` ist folgendermassen belegt:

$$\texttt{gitter[x][y][z]} = \begin{cases} 0 & \text{wenn mit polarem Monomer belegt} \\ 1 & \text{wenn mit hydrophobem Monomer belegt} \\ 2 & \text{wenn leer}\,. \end{cases}$$

`sequenz[l]` ist 1, wenn sich an der $l$ten Stelle der Sequenz ein hydrophobes Monomer befindet und sonst 0. `berechneenergie(int l)` wird natürlich nur für hydrophobe Monomere aufgerufen. Hier nun die Implementation der fundametalen Funktion `step(l)`:

```
void step(int l)
{
  if (l<maxl)
    {
      // freie Nachbarn bestimmen
      int inliste[6]; // 0 wenn Nachbar [i] leer, sonst 1
      int m=6; // m zaehlt Anzahl freier Nachbarplaetze
      for (int i=0;i<6;i++) inliste[i]=0;
      if (gitter[xpos+1][ypos][zpos]!=2){ m--;inliste[0]=1; };
      if (gitter[xpos][ypos+1][zpos]!=2){ m--;inliste[1]=1; };
      ...
      if (gitter[xpos][ypos][zpos-1]!=2){ m--;inliste[5]=1; };
```

```
int neuricht=0;
if (m!=0) // wenn noch Nachbarplaetze frei sind, ...
  {
    // Waehle neue Richtung fuer n+1. Pkt.
    do
        neuricht=(int)((double)6*rand()/(RAND_MAX+1.0));
    while (inliste[neuricht]==1);

    //zu neuem Punkt gehen
    if (neuricht==0) xpos++; // Nach rechts
    if (neuricht==1) ypos++; // Nach oben
    ...
    if (neuricht==5) zpos--; // Nach vorne

    energie=0;
    // Energie des naechsten Pkt. wird berechnet
    if (sequenz[l+1]==1) berechneenergie(l+1);

    // Gewicht aktualisieren
    gewicht[l+1]=m*exp((-energie)/(0.3));
    Grossgewicht[l+1]=Grossgewicht[l]*gewicht[l+1];

    // gucken ob Grenzen ueber- o. unterschritten werden
    if (Grossgewicht[l+1]>WoG[l+1])
      {
        // Punkt auf Level l+1 setzen
        xdatenbank[l+1]=xpos;
        ydatenbank[l+1]=ypos;
        zdatenbank[l+1]=zpos;
        gitter[xpos][ypos][zpos]=sequenz[l+1];

        // Statistik
        c[l+1]++; // zaehlt, wie oft ein level schon besucht wurde
        Znakt[l+1]=Znakt[l+1]+Grossgewicht[l+1];
        WoG[l+1]=(Znakt[l+1]/Znakt[0])*(c[l+1]/c[0])*(c[l+1]/c[0]);
        WuG[l+1]=(0.2)*WoG[l+1];

        int copies=1;
        // bzw. int copies=int(1+sqrt(Grossgewicht[l+1]/WoG[l+1]));
```

```
    //Lege Kopien an
    Grossgewicht[l+1]/=(1.0+copies);
    for (int x=1;x==copies;x++)
      {
        step(l+1); // Aufruf der Kopien

        //Fuers zurueckkehren
        newreset(l+1); // stellt alten Zust. des Gitt. wieder her
        xpos=xdatenbank[l+1];
        ypos=ydatenbank[l+1];
        zpos=zdatenbank[l+1];
      }
    step(l+1); // normaler Aufruf
  }
else if (Grossgewicht[l+1]<WuG[l+1])
  {
    // Kette wird sterben...
    Grossgewicht[l+1]*=2;
    int zufall=(int)((double)2*rand()/(RAND_MAX+1.0));
    if (zufall==0)
      {
        // Stirbt doch nicht!
        // Punkt auf Level l+1 setzen
        ...
        // Statistik
        ...
        step(l+1); // normaler Aufruf
      };
  }
else
  {
    // Alles innerhalb normaler Parameter
    // Punkt auf Level l+1 setzen
    ...
    // Statistik
    ...
    step(l+1); // normaler Aufruf
  }
```

```
        } // Ende if (m!=0)
    } // Ende if (l<lmax)
  else if (l>=maxl)
    {
      // max. Kettenlaenge erreicht
      Hier ist das Protein fertig gewachsen. Man kann nun:
      - Observable wie end-to-end distance oder radius of gyration messen,
      - das Protein abspeichern, um es spaeter zu visualisieren,
      - usw.
    } // Ende if (l=>maxl)
} // Ende step(l)
```

Als letztes möchte ich noch den Quellcode für den in Kap. 4.4 besprochenen Algorithmus nPERM zeigen. Und zwar in der Implementation, in der ich mich schon vom festen Gitter gelöst habe. Dies wurde notwendig, da der vorhandene Speicherplatz für die Implementation auf einem festen Gitter für längere Polymere nicht mehr ausreicht. Dadurch kann allerdings z.B. auch die Energie nicht mehr so leicht wie oben berechnet werden. Man muß nun jedes Mal „durch das ganze Polymer gehen“ und nachsehen, ob das jeweilige Monomer neben dem aktuell betrachteten liegt. Die folgende Implementation ist außerdem speziell für Homopolymere geschrieben, es wird daher nicht mehr zwischen hydrophoben und polaren Monomeren unterschieden. Die in Kap. 6.1.2 angesprochene Modifiaktion ist noch nicht vorgenommen:

```
void step(int l)
{
  int nnfrei[7]; // freie naechste Nachbarn
  int nnfreineu[7]; // freie naechste Nachbarn fuer naechste Kopie
  int m; // Anzahl freier naechster Nachbarn
  int mneu; // ... fuer naechste Kopie
  for (int i=1;i<=6;i++) nnfrei[i]=globalnnfrei[i];
  m=globalm;

  if (l<=maxl)
  {
    if (m!=0)
    {
      //W_pred(l+1) berechnen
      Predgewicht[l+1]=Grossgewicht[l]*m;

      if (Predgewicht[l+1]>WoG[l+1])
      {
        int copies=(int)(Predgewicht[l+1]/WoG[l+1]);
```

```
if (copies>m) copies=m; // copies=min[m,W_pred/WoG]

// Kopien anlegen (inkl. dem ‘‘normalen’’ Weitergehen)
for (int i=0; i<copies; i++)
{
  int wuerfel=1+(int)(double(m-i)*RAN01());
    //Wuerfelt zw. 1 und Anz. der NOCH freien Nachbarn
  int neuricht=nnfrei[wuerfel];
  for(int j=wuerfel; j<=m; j++) nnfrei[j]=nnfrei[j+1];
    // Liste der JETZT NOCH freien Nachbarn wird aktualisiert
       (alle oberhalb der getroffenen ruecken nach)

  //zu neuem Punkt gehen
  if (neuricht==1) {xpos++;} // Nach rechts
  if (neuricht==2) {ypos++;} // Nach oben
  ...
  if (neuricht==6) {zpos--;} // Nach vorne

  // Energie des naechsten Pkt. wird berechnet,
     sowie dessen freie Nachbarn bestimmt
  energie=0;
  mneu=6;
  for (int i=0;i<6;i++) inliste[i]=0;
  for (int i=0;i<l+1;i++)
  {
    if ((xdatenbank[i]==(xpos+1))&&(ydatenbank[i]==(ypos))
                                  &&(zdatenbank[i]==(zpos)))
      {//if ((sequenz[l+1]==1)&&(sequenz[i]==1))
         // kaeme fuer Proteine hinzu
        energie--; mneu--; inliste[0]=1;}
    if ((xdatenbank[i]==(xpos))&&(ydatenbank[i]==(ypos+1))
                               &&(zdatenbank[i]==(zpos)))
        energie--; mneu--; inliste[1]=1;
    ...
    if ((xdatenbank[i]==(xpos))&&(ydatenbank[i]==(ypos))
                               &&(zdatenbank[i]==(zpos-1)))
        energie--; mneu--; inliste[5]=1;
  }
```

```
// Buchhaltung
xx=1;
for (int i=1;i<=6;i++)
{
  if (inliste[i-1]==0)
  {
    nnfreineu[xx]=i;
    xx++;
  }
}

//if ((sequenz[l]==1)&&(sequenz[l+1]==1))
  // kaeme fuer Proteine hinzu
energie++; // Linkenergie abziehen

// Gewicht aktualisieren
gewicht[l+1]=((double)m/(double)copies)*Boltzfak[-energie];
Grossgewicht[l+1]=Grossgewicht[l]*gewicht[l+1];

//Punkt auf Level l+1 setzen
xdatenbank[l+1]=xpos;
ydatenbank[l+1]=ypos;
zdatenbank[l+1]=zpos;
Eges[l+1]=Eges[l]+energie;

// Statistik
c[l+1]++; // zaehlt, wie oft ein level schon besucht wurde
Znaktarray[l+1]=Znaktarray[l+1]+Grossgewicht[l+1];
WoG[l+1]=(Znaktarray[l+1]/Znaktarray[0])*(c[l+1]/c[0])
                                        *(c[l+1]/c[0]);
WuG[l+1]=(0.2)*WoG[l+1];

// Buchhaltung
for (int i=1;i<=6;i++) globalnnfrei[i]=nnfreineu[i];
globalm=mneu;

step(l+1);
```

```
    // Fuers zurueckkehren
    for (int i=1;i<=6;i++) globalnnfrei[i]=nnfrei[i];
    globalm=m;
    xpos=xdatenbank[l];
    ypos=ydatenbank[l];
    zpos=zdatenbank[l];

  } // Ende for (i=0;i<copies)
} // Ende if (Predgewicht>WoG)

else if (Predgewicht[l+1]<WuG[l+1])
{
  // Kette wird sterben...
  int zufall=(int)((double)2*RAN01());
  if (zufall==0)
  {
    // Stirbt doch nicht!
    int wuerfel=1+(int)((double)m*RAN01());
       //Wuerfelt zw. 1 und Anz. der NOCH freien Nachbarn
    int neuricht=nnfrei[wuerfel];

    //zu neuem Punkt gehen
    ...
    // Energie des naechsten Pkt. wird berechnet,
       sowie dessen freie Nachbarn bestimmt
    ...
    // Buchhaltung
    ...
    // Gewicht aktualisieren
    ...
    //Punkt auf Level l+1 setzen
    ...
    // Statistik
    ...
    // Buchhaltung
    ...

    step(l+1);
```

```
        }; // Ende if (zufall==0)
      } // Ende if (Predgewicht<WuG)
      else
      {
        // Alles innerhalb normaler Parameter
        int wuerfel=1+(int)((double)m*RAN01());
           //Wuerfelt zw. 1 und Anz. der NOCH freien Nachbarn
        int neuricht=nnfrei[wuerfel];

        //zu neuem Punkt gehen
        ...
        // Energie des naechsten Pkt. wird berechnet,
           sowie dessen freie Nachbarn bestimmt
        ...
        // Buchhaltung
        ...
        // Gewicht aktualisieren
        ...
        //Punkt auf Level l+1 setzen
        ...
        // Statistik
        ...
        // Buchhaltung
        ...

        step(l+1);

      } // Ende Alles innerhalb normaler Parameter
    } // Ende if (m!=0)

    if (l==maxl)
    {
      Polymer fertig...
    }
  } // Ende if (l<=lmax)
} // Ende step(l)
```

Die Funktion, aus der `step(l)` zum ersten Mal aufgerufen wird, ist folgende:

```
int main()
{
  ...
  energie=0;
  Eges[0]=0;

  // Schranken fuer 1. Durchlauf
  for (int i=0;i<=maxl;i++)
  {
    Znaktarray[i]=0;
    WoG[i]=1E+1000;
    WuG[i]=0;
    c[i]=0;
  }

  // Boltzmannfaktoren ausrechnen
  for (int i=0;i<=12;i++)
    Boltzfak[i]=expl((i)/(temp));

  for (tour=0;hitsever<100000;tour++)
  {
    ...
    gewicht[0]=1;
    Grossgewicht[0]=gewicht[0];

    Znaktarray[0]=Znaktarray[0]+Grossgewicht[0];
    c[0]++;

    for (int i=1;i<=6;i++) globalnnfrei[i]=i;
    globalm=6;

    step(0);
  }
  ...
}
```

`hitsever` zählt dabei, wie oft ein Polymer vollständig erzeugt wurde.

Anhang B

Sequenzen und Zustände niedriger Energie

B.1 Alle Grundzustände des 10mers auf dem 2D Dreiecksgitter

Hier werden alle 12 unabhängigen (siehe Tab. 5.1) Grundzustände von Sequenz (Seq 10_1) auf dem 2D Dreiecksgitter gezeigt.

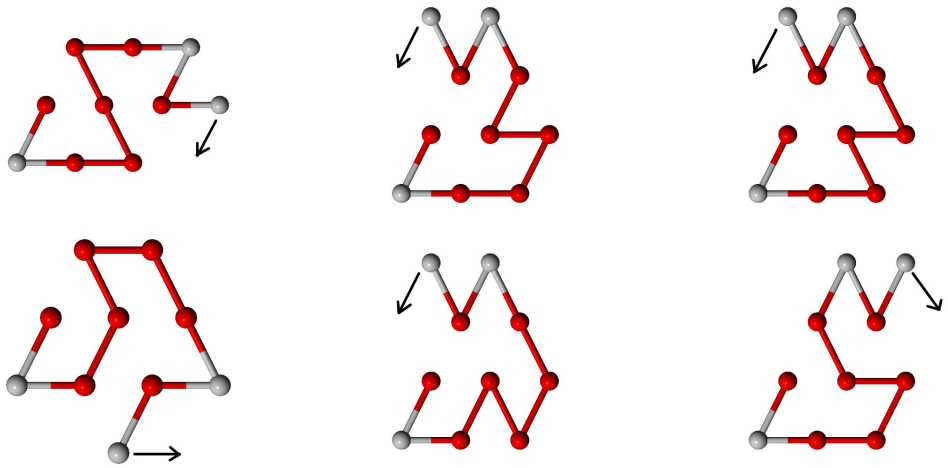


Abbildung B.1: Alle unabhängigen Grundzustände von Sequenz (Seq 10_1) auf dem 2D Dreiecksgitter. Jeweils 2 sind in einer Teilabbildung zu sehen, der Übergang zwischen diesen ist gekennzeichnet durch einen Pfeil.

B.2 48mere

Gezeigt werden die 10 Sequenzen aus [28] der Länge $n = 48$ sowie Zustände mit der in Tab. 5.3 angegebenen minimalen Energie auf verschiedenen Gittern.

HPHHPPHHHHPHHHPPHHPPHPHHHPHPHHPPHHPPPHPPPPPPPPHH (Seq 48_1)
HHHHPHHPHHHHHPPHPPHHPPHPPPPPPHPPHPPPHPPHHPPHHHPH (Seq 48_2)
PHPHHPHHHHHHPPHPHPPHPHHPHPHPPPHPPHHPPHHPPHPHPPHP (Seq 48_3)
PHPHHPPHPHHHPPHHPHHPPPHHHHHPPHPHHPHPHPPPPHPPHPHP (Seq 48_4)
PPHPPPHPHHHHPPHHHHPHHPHHHPPHPHPHPPHPPPPPPHHPHHPH (Seq 48_5)
HHHPPPHHPHPHHPHHPHHPHPPPPPPPHPHPPHPPPHPPHHHHHHPH (Seq 48_6)
PHPPPPHPHHHPHPHHHHPHHPHHPPPHPHPPPHHHPPHHPPHHPPPH (Seq 48_7)
PHHPHHHPHHHHPPHHHPPPPPPHPHHPPHHPHPPPHHPHPHPHHPPP (Seq 48_8)
PHPHPPPPHPHPHPPHPHHHHHHPPHHHPHPPHPHHPPHPHHHPPPPH (Seq 48_9)
PHHPPPPPPHHPPPHHHPHPPHPHHPPHPPHPPHHPPHHHHHHPPHH (Seq 48_{10})

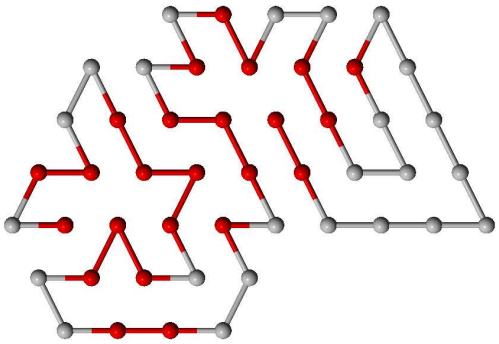
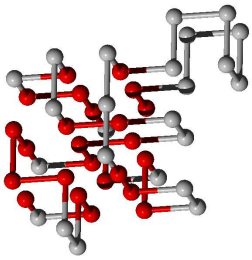
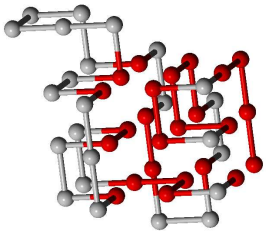


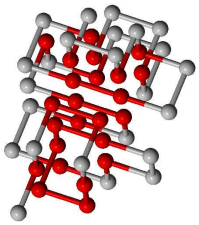
Abbildung B.2: Zustand mit der Energie $E_{\min} = -38$ der Sequenz (Seq 48_1) auf dem 2D Dreiecksgitter.



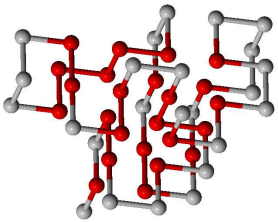
(Seq 48_1) $E = -32$



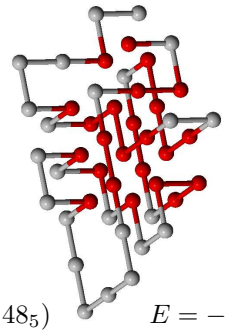
(Seq 48_2) $E = -34$



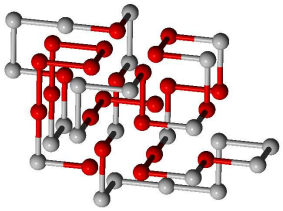
(Seq 48_3) $E = -34$



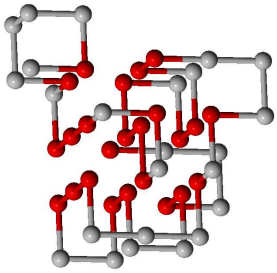
(Seq 48_4) $E = -33$



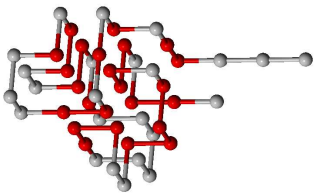
(Seq 48_5) $E = -32$



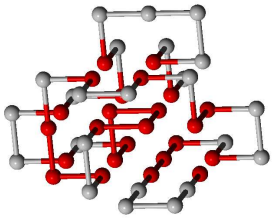
(Seq 48_6) $E = -32$



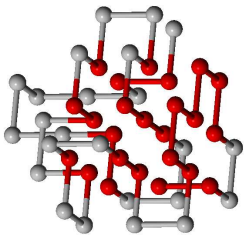
(Seq 48_7) $E = -32$



(Seq 48_8) $E = -31$

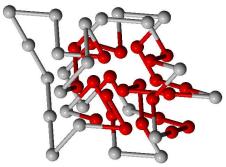


(Seq 48_9) $E = -34$

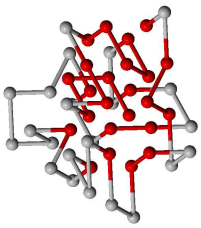


(Seq 48_{10}) $E = -33$

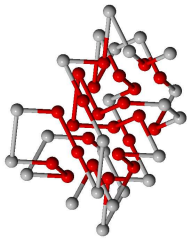
Abbildung B.3: Zustände der Sequenzen (Seq 48_1)-(Seq 48_{10}) (zu lesen von links nach rechts und von oben nach unten) mit den in Tab. 5.3 angegebenen Grundzustandsenergien auf dem 3D sc-Gitter.



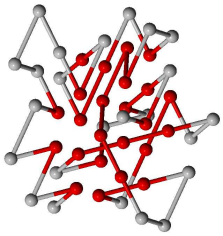
(Seq 48_1) $E = -69$



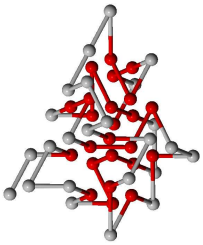
(Seq 48_2) $E = -69$



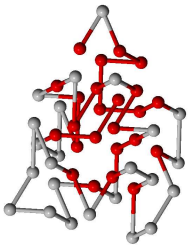
(Seq 48_3) $E = -72$



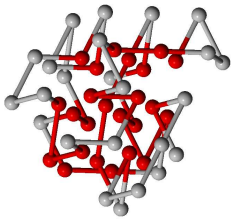
(Seq 48_4) $E = -71$



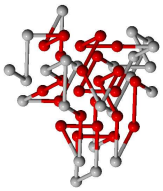
(Seq 48_5) $E = -70$



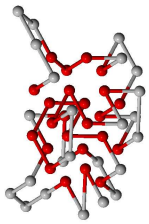
(Seq 48_6) $E = -70$



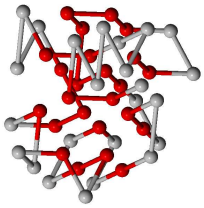
(Seq 48_7) $E = -70$



(Seq 48_8) $E = -69$



(Seq 48_9) $E = -71$



(Seq 48_{10}) $E = -68$

Abbildung B.4: Zustände der Sequenzen (Seq 48_1)-(Seq 48_{10}) (zu lesen von links nach rechts und von oben nach unten) mit den in Tab. 5.3 angegebenen Grundzustandsenergien auf dem 3D fcc-Gitter.

B.3 HP-Modelle realer Proteine

Gezeigt sind hier vier Sequenzen, welche HP-Modelle realer Proteine sind, mit ihren von mir gefundenen Zuständen niedrigster Energie auf dem 3D sc-Gitter (siehe Abb. B.5), dem 2D Dreiecksgitter (siehe Abb. B.6) sowie dem 3D fcc-Gitter (siehe Abb. B.7). Die Sequenzen sind die folgenden:

- (Seq 58_1) für das Protein BPTI, aus [33].
- (Seq 103_1) für Cytochrome c,
- (Seq 124_1) für Ribonuclease A,
- (Seq 136_1) für ein Staphylococcal-Nuclease Fragment, alle aus [15].

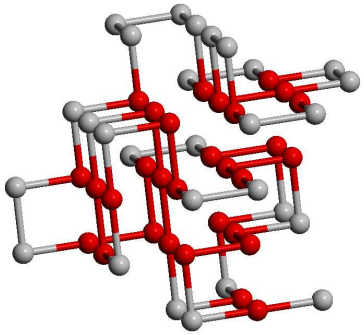
$$PHPH_3PH_3P_2H_2PHPH_2PH_3PHPHPH_2P_2H_3P_2HPHP_4HP_2HP_2H_2P_2HP_2H \qquad (\text{Seq } 58_1)$$

$$P_2H_2P_5H_2P_2H_2PHP_2HP_7HP_3H_2PH_2P_6HP_2HPHP_2HP_5H_3P_4H_2PH_2P_5H_2$$
$$P_4H_4PHP_8H_5P_2HP_2 \qquad (\text{Seq } 103_1)$$

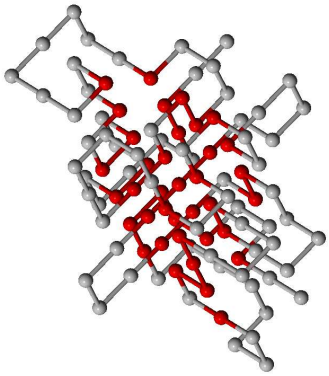
$$P_3H_3PHP_4HP_5H_2P_4H_2P_2H_2P_4HP_4HP_2HP_2H_2P_3H_2PHPH_3P_4H_3P_6H_2P_2$$
$$HP_2HPHP_2HP_7HP_2H_3P_4HP_3H_5P_4H_2PHPHPHPH \qquad (\text{Seq } 124_1)$$

$$HP_5HP_4HPH_2PH_2P_4HPH_3P_4HPHPH_4P_{11}HP_2HP_3HPH_2P_3H_2P_2HP_2HP$$
$$HPHP_8HP_3H_6P_3H_2P_2H_3P_3H_2PH_5P_9HP_4HPHP_4 \qquad (\text{Seq } 136_1)$$

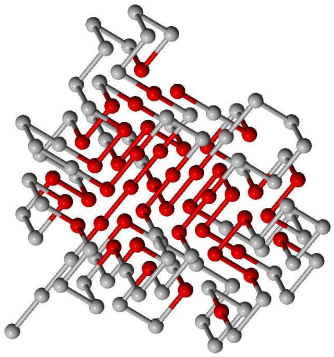
Weiterhin zeigt Abb. C.7 Zustände der Sequenz (Seq 124_1) auf dem 2 dimensionalen Dreiecksgitter mit der Energie $E = -73$, der bisher niedrigsten, die gefunden wurde. Man kann allerdings vermuten, daß es noch Zustände mit deutlich niedrigerer Energie geben wird, da alle gezeigten Zustände noch 3 getrennte hydrophobe Zentren aufweisen.



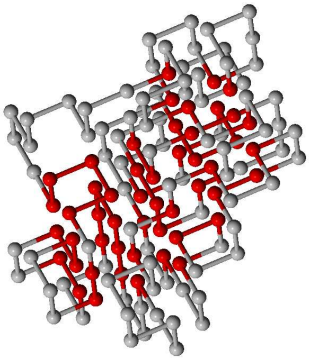
(Seq 58_1) $E = -44$



(Seq 103_1) $E = -53$

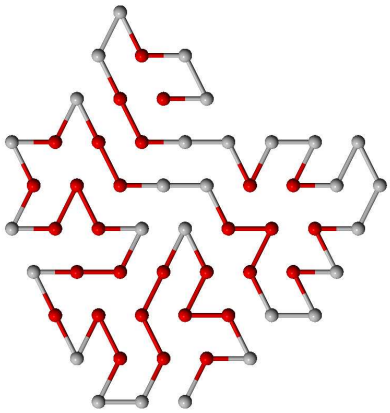


(Seq 124_1) $E = -71$

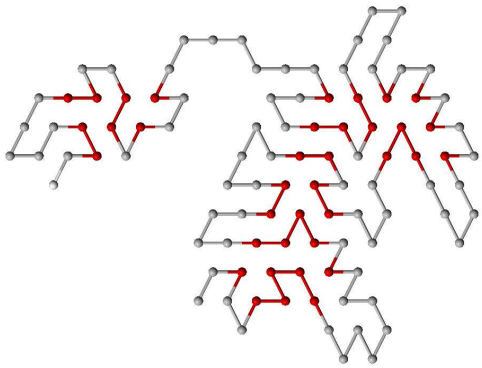


(Seq 136_1) $E = -77$

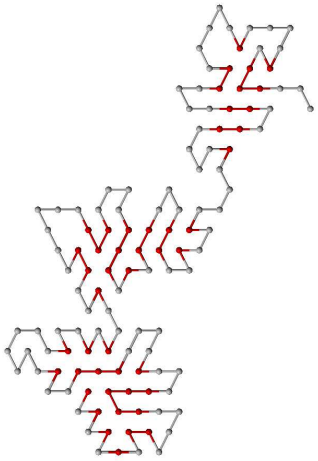
Abbildung B.5: Konformationen niedrigster Energie der Sequenzen (Seq 58_1), (Seq 103_1), (Seq 124_1) und (Seq 136_1) auf dem 3D sc-Gitter.



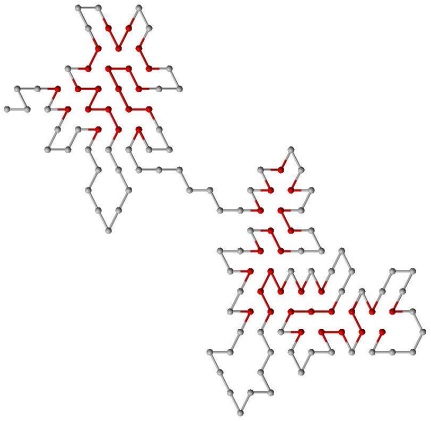
(Seq 58_1) $E = -49$



(Seq 103_1) $E = -56$

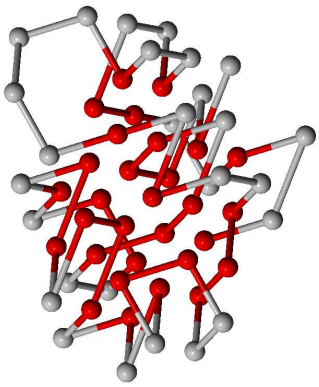


(Seq 124_1) $E = -73$

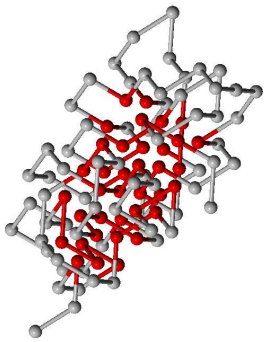


(Seq 136_1) $E = -80$

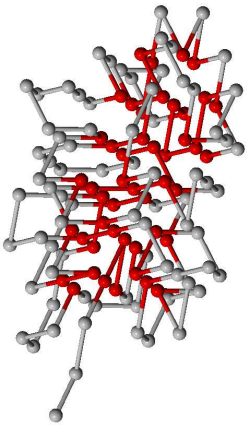
Abbildung B.6: Konformationen niedrigster Energie der Sequenzen (Seq 58_1), (Seq 103_1), (Seq 124_1) und (Seq 136_1) auf dem 2D Dreiecksgitter.



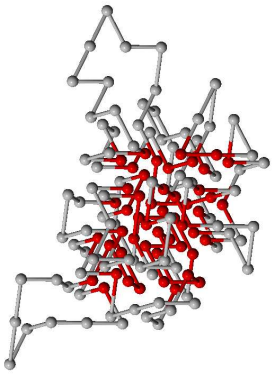
(Seq 58_1) $E = -94$



(Seq 103_1) $E = -114$



(Seq 124_1) $E = -154$



(Seq 136_1) $E = -167$

Abbildung B.7: Konformationen niedrigster Energie der Sequenzen (Seq 58_1), (Seq 103_1), (Seq 124_1) und (Seq 136_1) auf dem 3D fcc-Gitter.

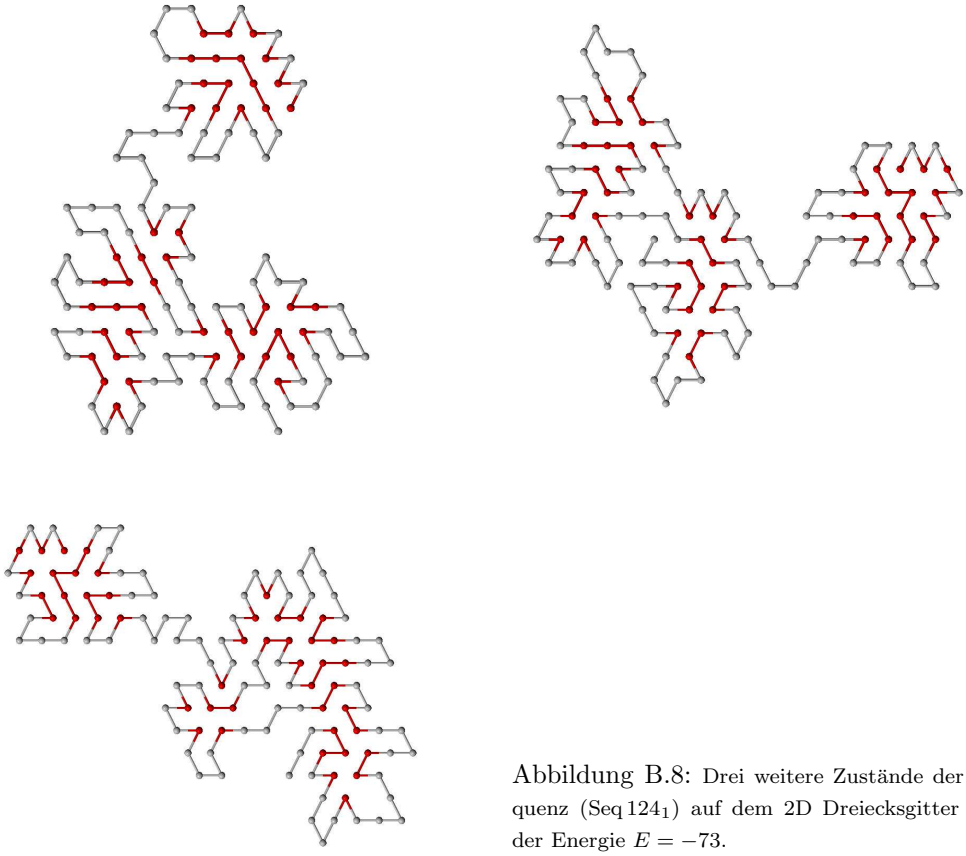


Abbildung B.8: Drei weitere Zustände der Sequenz (Seq 124_1) auf dem 2D Dreiecksgitter mit der Energie $E = -73$.

Anhang C

Galerie

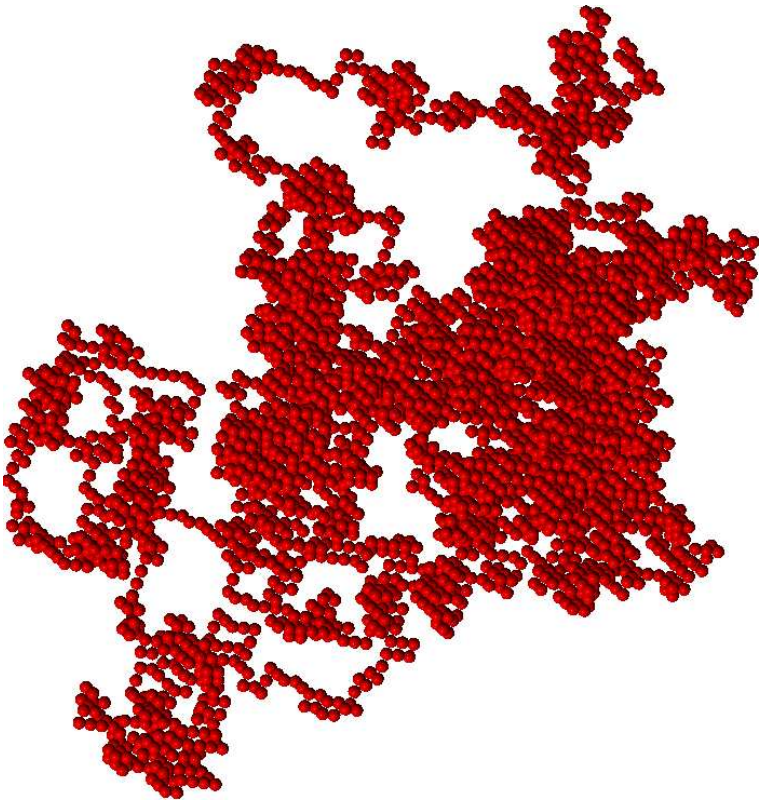


Abbildung C.1: Ein Homo4096mer mit der Energie $E = -2107$ auf dem kubischen Gitter in 3 Dimensionen. Es ist das erste Polymer, das volle Kettenlänge durch den Kettenwachstumsalgorithmus bei der Temperatur $T = 3.1$ erreicht hat. Das muß nicht zwangsläufig innerhalb der ersten Tour passieren und ist deshalb i.a. kein Zufallsweg mehr. Tatsächlich entstand das abgebildete Polymer erst in der 7.Tour.

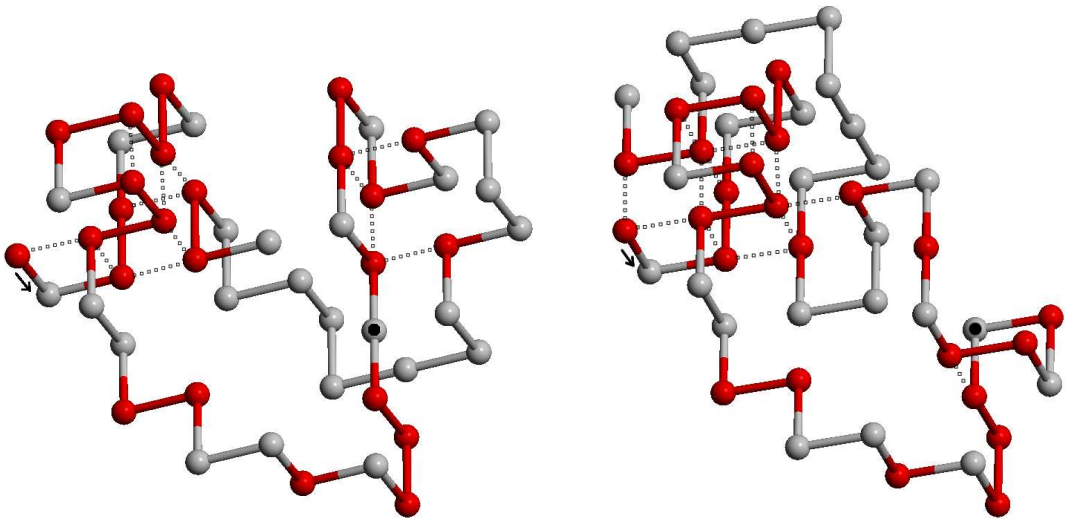


Abbildung C.2: Schnappschuß aus der Faltung von Sequenz (Seq 48_1) auf dem 3D sc-Gitter. **Links** ein Zustand mit der Energie $E = -12$. **Rechts** der nachfolgende Zustand mit der Energie $E = -13$. Die Konformation ist bis zu dem mit • gekennzeichtem Monomer identisch zur linken, der „Rest“ der Kette schmiegt sich jetzt jedoch besser an den *cluster* am Beginn der Konformation an. (Der Pfeil kennzeichnet den Anfang der Konformation.)

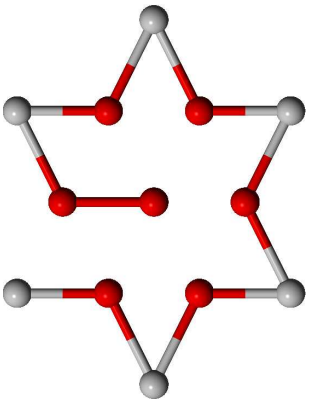


Abbildung C.3: Die Sequenz HHPHPHPHPHPHP auf dem 2D Dreiecksgitter.

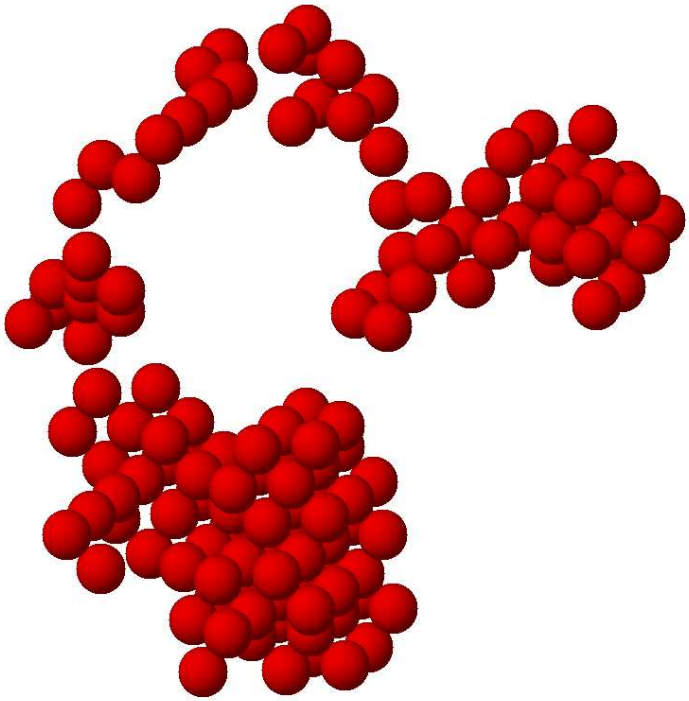


Abbildung C.4: Schnappschuß aus der Faltung des Homo136mers auf dem 3D Tetraedergitter. Die Umgebung ist sehr kalt, es wird kurz darauf seinen Grundzustand einnehmen.

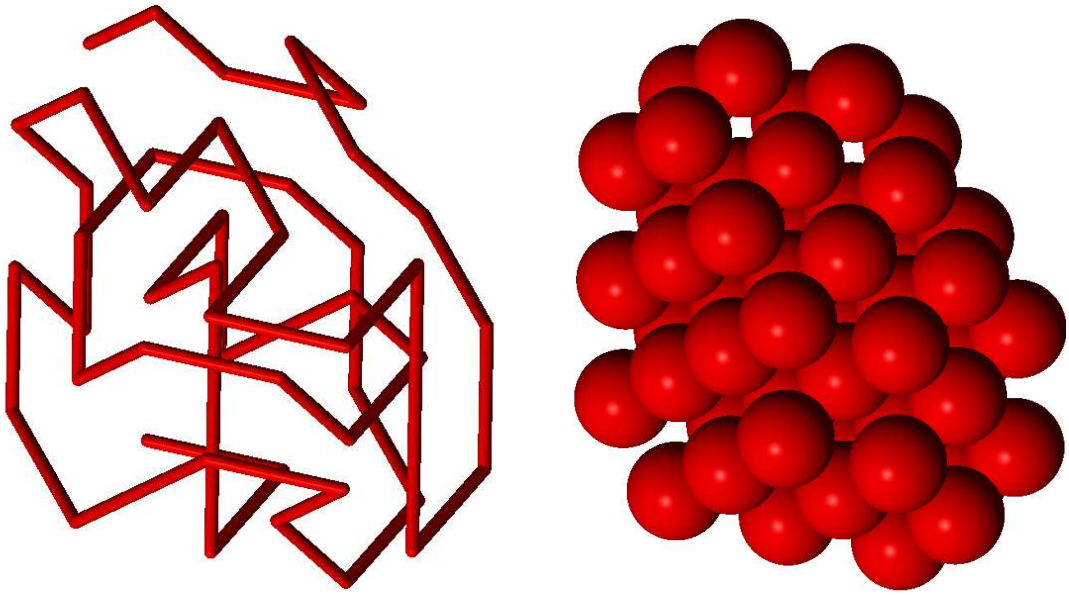


Abbildung C.5: Kompakter Zustand des Homo50mers auf dem 3D Tetraedergitter, zerlegt in *backbone* (**links**) und Sphären am Ort der Monomere (**rechts**).

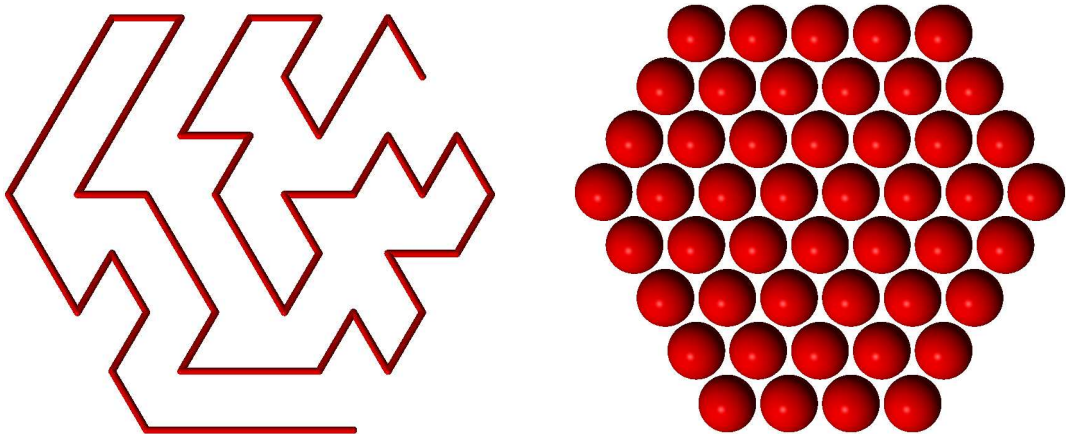


Abbildung C.6: Kompakter Zustand des Homo48mers auf dem 2D Dreiecksgitter, zerlegt in *backbone* (**links**) und Sphären am Ort der Monomere (**rechts**).

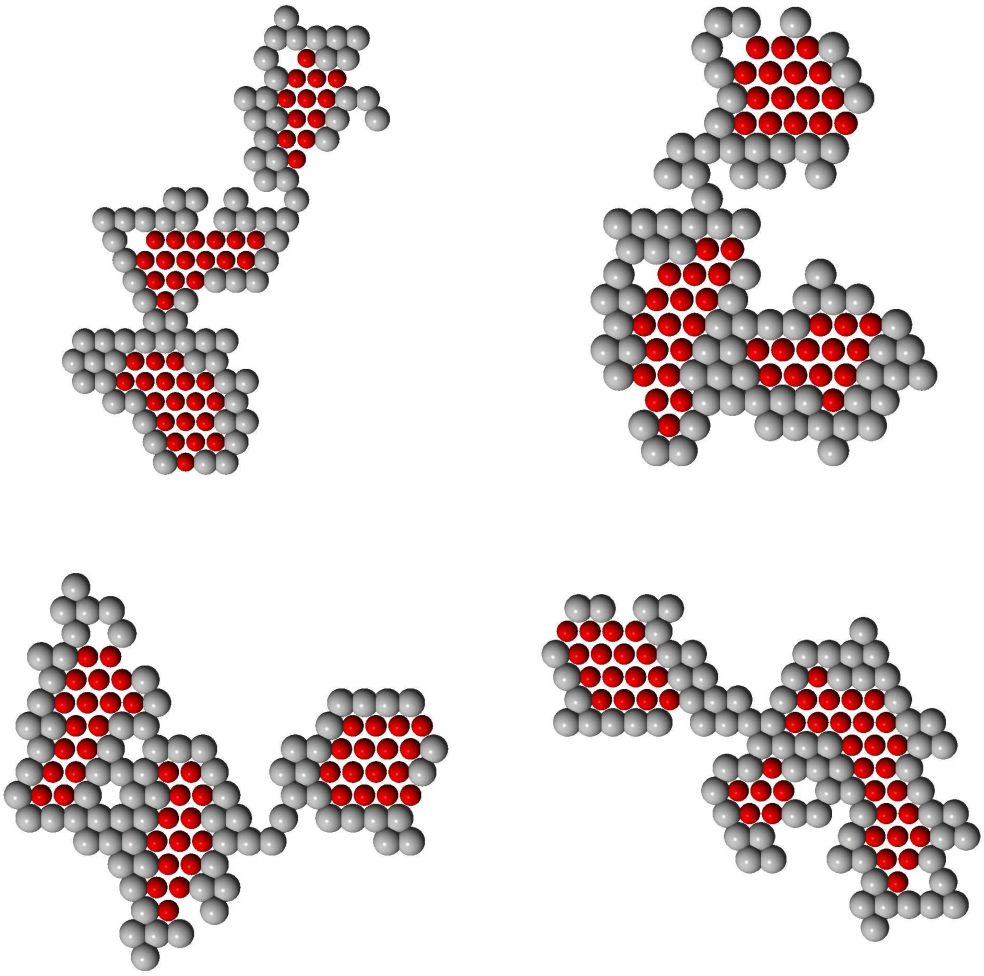


Abbildung C.7: Die Zustände der Sequenz (Seq 124_1) auf dem 2D Dreiecksgitter mit der Energie $E = -73$ in einer anderen Darstellung (siehe Abb. 5.18 und C.7). Besser zu sehen hier die „Kristallisation“ in separaten hydrophoben Kernen.

Literaturverzeichnis

[1] http://www.psc.edu/science/Kollman98/kollman98.html

[2] C.B. Anfinsen, *Studies on the principles that govern the folding of protein chains*, Nobel Lecture (1972); C.B. Anfinsen, Science **181** (1973) 223.

[3] A.L. Lehninger, D.L. Nielson, M.M. Fox, *Principios de Bioquímica*, 2da Ed. (Ediciones Omega, S.A., Barcelona, 2001) (Originalausgabe: *Principles of Biochemistry*, 2nd Ed. (Worth Publishers, Inc., New York, 2001)); T.E. Creighton, *Proteins: Structures and Molecular Properties* (W.H. Freeman and Co., New York, 1993).

[4] J. De Kyte, R.F. Doolittle, J. Mol. Biol. **157** (1982) 105.

[5] M.H. De Klapper, Biochem. Biophys. Res. Commun. **78** (1977) 1018.

[6] R. Sayle, RasMol Molecular Renderer (1995), © R. Sayle 1992–1999, H.J. Bernstein 1998–2001.

[7] R. Schiemann, *Exact enumeration of 3D lattice proteins*, Diploma Thesis, Universtät Leipzig (2003).

[8] K.A. Dill, Prot. Sci. **8** (1999) 1166.

[9] Y. Okamoto, *Predicting protein tertiary structures from the first principles*, Assoc. Asia Pacif. Phys. Soc. Bull. **5** (1995) 4.

[10] Y. Okamoto, Recent Research Developments in Pure & Appl. Chem. **2** (1998) 1.

[11] F.A. Momany, R.F. McGuire, A.W. Burgess, H.A. Scheraga, J. Phys. Chem. **79** (1975) 2361.

[12] M. Vásquez, G. Némenthy, H.A. Scheraga, Chem. Rev. **94** (1994) 2183.

[13] R.A. Pierotti, Chem. Rev. **76** (1965) 717.

[14] K.A. Dill, Biochem. **24** (1985) 1501; K.F. Lau, K.A. Dill, Macromol. **22** (1989) 3986.

[15] E.E. Lattmann, K.M. Fiebig, K.A. Dill, Biochem. **33** (1994) 6158.

[16] C. Kittel, *Introduction to Solid State Physics*, 4th Ed. (John Wiley and Sons, Inc., New York, 1978).

[17] R. Agarwala, S. Batzoglou, V. Dančík, S.E. Dacatur, M. Farach, S. Hannenhalli, S. Skiena, Proceedings of the 8$^{\text{th}}$ Annual ACM-SIAM Symposium on Discrete Algorithms (SODA ′97) 390.

[18] M.N. Rosenbluth, A.W. Rosenbluth, J. Chem. Phys. **23** (1955) 356.

[19] A. Irbäck, C. Troein, J. Biol. Phys. **28** (2002) 1.

[20] D. MacDonald, S. Joseph, D.L. Hunter, L.L. Moseley, N. Jan, A.J. Guttmann, J. Phys. A: Math. Gen. **33** (2000) 5973.

[21] A.D. Sokal, *Monte Carlo methods for the self-avoiding walk* in *Monte Carlo and Molecular Dynamics Simulations in Polymer Science*, ed. by K. Binder (Oxford University Press, New York, 1995) 51.

[22] D. MacDonald, D.L. Hunter, K. Kelly, N. Jan, J. Phys. A: Math. Gen. **25** (1992) 1429.

[23] M. Bachmann, R. Schiemann, personal communication.

[24] M. Chen, K.Y. Lin, J. Phys. A: Math. Gen. **35** (2002) 1501.

[25] B. Nienhuis, Phys. Rev. Lett. **49** (1982) 1062.

[26] P. Grassberger, W. Nadler, *„Go with the winners"-simulations*, Proceedings der Heraeus-Ferienschule, *Vom Billiardtisch bis Monte Carlo: Spielfelder der statistischen Physik* (Chemnitz, Oktober 2000), cond-mat/0010265; P. Grassberger, Comp. Phys. Comm. (Proceedings der CCP2001, Aachen, 2001), cond-mat/0201313.

[27] P. Grassberger, Phys. Rev. E **56** (1997) 3682.

[28] K. Yue, K.M. Fiebig, P.D. Thomas, H.S. Chan, E.I. Shakhnovich, K.A. Dill, Proc. Natl. Acad. Sci. USA **92** (1995) 325.

[29] H.-P. Hsu, V. Mehra, W. Nadler, P. Grassberger, J. Chem. Phys. **118** (2002) 444.

[30] A.Yu. Grosberg, D.V. Kuznetsov, Macromol. **25** (1992) 1970.

[31] P. Grassberger, R. Hegger, J. Chem. Phys. **102** (1995) 6881.

[32] M. Bachmann, W. Janke, *Multicanonical chain-growth algorithm*, Phys. Rev. Lett. **91** (2003) 208 105; *Thermodynamics of lattice heteropolymers*, cond-mat/0310707; *Density of states for HP lattice proteins*, Acta Physica Polonica B **34** (2003) 4689 (Proceedings of the Workshop on Random Geometry, Krakow, 2003).

[33] K.A. Dill, K.M. Fiebig, H.S. Chan, Proc. Natl. Acad. Sci. USA **90** (1993) 1942.

[34] L. Toma, S. Toma, Protein Sci. **5** (1996) 147.

[35] T. Prellberg, A.L. Owczarek, *Four-dimensional polymer collapse: Pseudo-first-order transition in interacting self-avoiding walks*, Phys. Rev. Lett. **62** (2000) 3780.

[36] W. Janke, *First-order phase transitions*, to appear in *Computer Simulations of Surfaces and Interfaces*, NATO Advanced Study Institute, Albena, Bulgaria, September 9–20, 2002, edited by D.P. Landau, A. Milchev, and B. Dünweg (Kluwer, Dordrecht, 2003) (in print)

[37] P. Grassberger, personal communication.

[38] A.L. Owczarek, T. Prellberg, *Monte Carlo investigation of lattice models of polymer collapse in five dimensions*, Int. J. Mod. Phys. C **14** (2003) 621.

[39] F.H. Stillinger, T. Head-Gordon, Phys. Rev. E **52** (1995) 2872.

[40] H.-P. Hsu, V. Mehra, P. Grassberger, Phys. Rev. E **68** (2003) 037703.

[41] F.A. Momany, R.F. McGuire, A.W. Burgess, H.A. Scheraga, J. Chem. Phys. **79** (1975) 2361.

[42] M. Kofler, *Mathematica* (Addison-Wesley, Bonn, München, Paris u.a., 1992), Kap. 12.

Erklärung nach Paragraph 20(5) Prüfungsordnung Hiermit erkläre ich, daß ich die Diplomarbeit selbständig verfaßt habe und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Alle Stellen der Arbeit, die wörtlich oder sinngemäß aus Veröffentlichungen oder aus anderweitigen fremden Äußerungen entnommen wurden, sind als solche kenntlich gemacht.

Ferner erkläre ich, daß die Arbeit noch nicht in einem anderen Studiengang als Prüfungsleistung verwendet wurde.

Leipzig, den 05.01.04

Thomas Vogel