**Media Engineering and Technology Faculty**
**German University in Cairo**

# Folding Simulations in Side-chain Lattice Protein Models

**Bachelor Thesis**

Author: Mohamed Abou Hamra
Supervisor: Martin Mann
Submission Date: 21 June, 2010

This is to certify that:

(i) the thesis comprimises only my original work toward the Bachelor Degree

(ii) due acknowlegement has been made in the text to all other material used

Mohamed Abou Hamra
21 June, 2010

# Abstract

The thesis presents the K-local move set, a local move set defined generically for lattice protein models. The K-local move set is defined for both backbone and side-chain protein models via constraint satisfaction problems. The use of the constraint-based approach enabled its use for an arbitrary lattice. The K-local move set is then used for a simulation procedure for side-chain protein structures in the face-centered cubic lattice using real protein sequences and structures.

# Contents

# Chapter 1

# Introduction

Protein is a sequence of amino acids, bound by peptide bonds. A protein is a polymer of amino acids. Each peptide bond forms between the amine group of a an amino acid and the carboxyl group of the next amino acid.
Proteins hold various important roles in living organisms. They are one of the basic constituents of cells. They help transferring molecules between cells. Enzymes, which are proteins, participate in many biochemical reactions as catalysts [1].
The function of a protein is an important factor of its function. Thus, the problem of protein structure prediction is considered to be of increasing importance, because it affects our understanding of enzymes, cell structure, and also would help the field of drug design [21]. However, this problem is considered challenging, with no known efficient solution, and even proven to be NP-hard for some of the simple protein structure models [17, 1].
One approach for modeling the protein structure is to assume its monomers are confined to a discrete lattice, with each amino acid being represented by one or more vertices. This allows for a convenient representation of the protein structure, and is known as lattice protein models [1].
A method of protein structure prediction is the simulation of its folding into a stable, kinetically optimal structure [1]. This requires the definition of a move set, a set of possible structural changes that can send a protein structure into other possible protein structures with this structural change applied. In this thesis, the K-local move set, a generic move set is defined using an approach based on constraint satisfaction problems. This move set is generic because it applies to any lattice model, and it is easy to be extended for any protein structure model. It is defined for backbone as well as side-chain protein models and it can apply to any lattice. This approach shows the flexibility given by using constraint programming for defining the possible structural protein changes.
This move set was used for search for protein structures that are locally minimum in their energy. The approach of using a constraint satisfaction problem for the search for local optima is known as a general framework for local search problems, as shown in [15]. Two search methods were used in this thesis, namely the gradient descent and random descending walk approaches. Both of them are methods of reaching structures that are locally optimal in energy. Comparisons are held between different input protein structures and the simulation method and parameters used.

# Chapter 2

# Background

## 2.1 Lattices

A *lattice* in 3D space is a set of vectors, including the origin vector, which is closed under addition and subtraction. The closure of addition and subtraction grants that any integer linear combination of these vectors belongs to the lattice.
A set of vectors $\{\vec{v_1}, \vec{v_2}, \vec{v_3}, ..., \vec{v_n}\}$ is called to *generate* a lattice $L$ if any vector in $L$ can be expressed as an integer linear combination of the generating vectors. In other words, if the lattice $L = \left\{\sum_{i=1}^{i=n} k_i \vec{v_i} : k_i \in \mathbb{Z}\right\}$. The set of generating vectors is called the *basis* of the lattice $L$, denoted as $B_L$, if it is minimal.

### The cubic lattice

The *cubic lattice* is denoted as $\mathbb{Z}^3$, and its basis is

$$B_{\mathbb{Z}^3} = \left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \right\} \tag{2.1}$$

The cubic lattice consists of all points with integer coordinates on a Cartesian three dimensional space. It is shown in figure 2.1.

$$\mathbb{Z}^3 = \left\{ \begin{pmatrix} x \\ y \\ z \end{pmatrix} : x, y, z \in \mathbb{Z} \right\} \tag{2.2}$$

### The face-centered cubic lattice

The *face-centered cubic lattice* is denoted as FCC, and its basis is

$$B_{FCC} = \left\{ \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \right\} \tag{2.3}$$

If the space contains infinite number of equally sized cubes, packed adjacent to each other, the face-centered cubic lattice consists of the center points of the faces of these cubes,

as well as the corner points of the cubes themselves. The face-centered cubic lattice is shown in figure 2.2.

$$FCC = \left\{ \begin{pmatrix} x+z \\ x+y \\ y+z \end{pmatrix} : x, y, z \in \mathbb{Z} \right\} \tag{2.4}$$

### 2.1.1 Neighborhood

To define the notion of neighborhood in a lattice $L$, the *set of neighborhood vectors*, $Neighborhood_L$ is introduced. In the lattice $L$, every point $p_1$ has exactly $|Neighborhood_L|$ neighbors, which are $\{p_1 + \vec{v} : \vec{v} \in Neighborhood_L\}$.
The predicate $Neighbor_L(p_1, p_2)$ is used to denote that $p_1$ is neighboring to $p_2$ in the lattice $L$, where $p_1, p_2 \in L$.

$$Neighbor_L(p_1, p_2) \Leftrightarrow \vec{p_1} - \vec{p_2} \in Neighborhood_L \tag{2.5}$$

Every vertex in the cubic lattice has 6 neighbors, as shown in figure 2.1. The set of neighborhood vectors for the cubic lattice is:

$$Neighborhood_{\mathbb{Z}^3} = \left\{ \begin{pmatrix} \pm 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ \pm 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ \pm 1 \end{pmatrix} \right\} \tag{2.6}$$

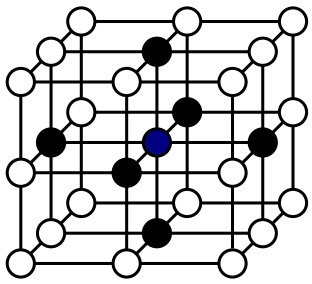

Figure 2.1: The cubic lattice; each vertex has six neighbors

The set of neighborhood vectors for the face-centered cubic lattice is:

$$Neighborhood_{FCC} = \left\{ \begin{pmatrix} \pm 1 \\ \pm 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ \pm 1 \\ \pm 1 \end{pmatrix}, \begin{pmatrix} \pm 1 \\ 0 \\ \pm 1 \end{pmatrix} \right\} \tag{2.7}$$

### 2.1.2 Self-avoiding walk

A *self-avoiding walk* (SAW) in a lattice $L$ is a sequence of lattice points $p = (\vec{p_1}, \vec{p_2}, \vec{p_3}, ..., \vec{p_{|p|}})$, of length $|p|$, where $\forall_{1 \leq i \leq |p|} : p_i \in L$, that satisfies two conditions:

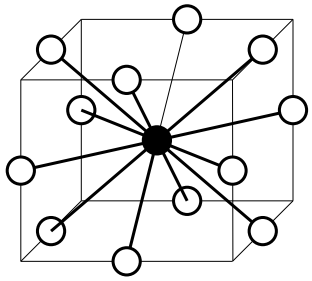

Figure 2.2: The face-centered cubic lattice; each vertex has twelve neighbors

1. *Connectivity*: Every element in the sequence should be a neighbor of the preceding element.

$$\forall_{2 \leq i \leq |p|} : Neighbor_L(p_{i-1}, p_i) \tag{2.8}$$

2. *Self-avoidance*: No two elements should be the same point, i.e., every lattice point is assigned to at most one vertex.

$$\forall_{1 \leq i < j \leq |p|} : p_i \neq p_j \tag{2.9}$$

## 2.2 Lattice proteins

Proteins are polymers of amino acids, bound by peptide bonds. The peptide bond forms between the amine group of an amino acid and the carboxyl group of the next amino acid. Proteins play important roles at various processes in living organisms [1].
One essential feature of a protein is its structure. The structure of a protein is important to determine its functionality. The problem of predicting the structure of the folded protein is of increasing importance lately. Predicting the folded protein structure would greatly affect our understanding of enzymes, cell structure, and also would help the field of drug design [21].
The problem of predicting the protein structure is considered challenging with respect to many aspects. Although the number of theoretically possible protein structures is very large, yet the protein collapses to a certain shape in milliseconds scale, a phenomena known as Levinthal's paradox. So far, there is no efficient solution for this problem [17, 1].
Lattice proteins is a way to model proteins abstractly representing a protein structure as vertices on a lattice. Each vertex is confined in a discretized lattice, and each amino acid is represented by one or more vertices. This allows us to get an approximate, yet convenient representation of protein structures. The backbone model maps every monomer into one lattice point, while the side-chain model maps every monomer to two neighboring lattice points representing the peptide bond and the amino acid chain [1].
Another possible way to model proteins is to model every monomer as two spheres, one representing the peptide bond, and the other represents the amino acid side-chain. This model is known as the tangent sphers model [7], which is an example for off-lattice protein structure model. In this thesis we will only discuss lattice protein structure models.

**Backbone model** A protein structure modelled in the *backbone model* is a chain (SAW) of lattice points, each representing a monomer in the protein. To estimate the energy of a certain configuration, each monomer is assigned to one symbol from a certain alphabet $\Sigma$ (dependent on the model). The energy is estimated based on the interactions between the monomers. A protein structure modelled in backbone model is defined as a pair $(p, R)$, where:

$$\begin{aligned} & p \text{ is a SAW} \\ & R : p \rightarrow \Sigma, \text{ shortened as } r_{p_i} = R(p_i) \in \Sigma \end{aligned} \tag{2.10}$$

As for the alphabet $\Sigma$, one example is $\Sigma = \{H, P\}$, used in the HP model [1]. Hydrophobic monomers are mapped to $H$, whereas polar monomers are mapped to $P$.

**Side-chain model** A protein structure modelled in the *side-chain model* is a chain of lattice points representing the peptide bonds, with each backbone lattice point neighboring a residue that models the amino acid side-chain. The backbone chain as well as the side-chain respect the self-avoidance and connectivity constraints. The structure modelled in side-chain model is defined as a triple $(p, s, R)$, where:

$$\begin{aligned} & p = (\vec{p_1}, \vec{p_2}, \vec{p_3}, ..., \vec{p_{|p|}}) \\ & s = (\vec{s_1}, \vec{s_2}, \vec{s_3}, ..., \vec{s_{|s|}}) \\ & \text{where } |p| = |s| \\ \wedge \quad & \forall_{1 \leq i \leq |p|} : p_i, s_i \in L \\ \wedge \quad & \forall_{1 \leq i \leq |p|} : Neighbor_L(s_i, p_i) \\ \wedge \quad & \forall_{2 \leq i \leq |p|} : Neighbor_L(p_{i-1}, p_i) \\ \wedge \quad & \forall_{1 \leq i,j \leq |p|} : p_i \neq s_j \\ \wedge \quad & \forall_{1 \leq i < j \leq |p|} : p_i \neq p_j \wedge s_i \neq s_j \\ & R : p \cup s \rightarrow \Sigma \cup \{b\}, r_{p_i} = R(p_i) = b, r_{s_i} = R(s_i) \in \Sigma \end{aligned} \tag{2.11}$$

### 2.2.1 Protein structure prediction

To predict the protein structure, it is expected that the protein takes a structure that is of low potential energy, thus, a stable structure. The potential energy of a structure is due to the interactions between the protein atoms. The general approach used to predict the protein structure so far is to simulate its folding [17].
Some of the types of folding simulations include energy local minima search and simulated annealing [5, 6, 17].

### 2.2.2 Energy functions

To determine the potential energy of a structure, some energy functions were developed to allow approximate, but fast calculation of the mutual interactions between the structure monomers.

To calculate the potential energy participated by each pair of vertices, there is a defined function from pairs of vertices to real numbers representing the potential energy participation of a pair in some energy units. As the potential energy introduced out of the interaction between two monomers depends on the distance between them, the energy between two vertices, is thus multiplied by a factor based on the distance of the two vertices. The other main factor in this function is the alphabet symbol each of the vertices represent.
The energy function between two vertices $e$ is more often defined as:

$$e(v_1, v_2) = e_\delta(v_1, v_2) \cdot e_\Sigma(r_{v_1}, r_{v_2}) \tag{2.12}$$

where $e_\delta(v_1, v_2)$ is the contribution to the energy function due to the distance between the vertices, and $e_\Sigma(r_{v_1}, r_{v_2})$ is the contribution due to the internal composition of each of the vertices, as abstracted by the model. $r_{v_i}$ denotes $R(v_i)$ as defined at 2.10 and 2.11, the alphabet symbols representing the type of the vertex.
An interesting characteristic of $e_\delta(v_1, v_2)$ and $e_\Sigma(r_{v_1}, r_{v_2})$ is that they are both symmetric with respect to the order of their inputs, $v_1$ and $v_2$. Which means, $e_\delta(v_1, v_2) = e_\delta(v_2, v_1)$, and $e_\Sigma(r_{v_1}, r_{v_2}) = e_\Sigma(r_{v_2}, r_{v_1})$. This follows from the reality of the mutual interaction between the two vertices. Thus, $e(v_1, v_2)$ is just equal to $e(v_2, v_1)$.

### Distance-based energy functions

In attempt to achieve high precision energy evaluation, the function $e_\delta(v_1, v_2)$ is based on the distance between the monomers. In [5], a distance-based energy function was used to reach for the effects resulting from the concentration of the polypeptide chain, taking the distance between the centers of mass of the chains as a parameter.

### Contact-based energy functions

As the potential energy introduced by a pair of vertices drops quadratically with the increase of the distance between them, one approximation to follow is to allow participation at the total energy of a protein structure to only the pairs of vertices that are in direct contact (neighbors on the lattice $L$, or touching spheres in the case of tangent sphere models [1]). In other words, $e_\delta(v_1, v_2)$ is defined as follows:

$$e_\delta(v_1, v_2) = \begin{cases} 1 & \text{if } Neighbor_L(v_1, v_2) \\ 0 & \text{otherwise} \end{cases} \tag{2.13}$$

$e_\Sigma(r_{v_1}, r_{v_2})$ is defined usually following a precomputed matrix of constant values, that is shown to be correspondent to the reality. For example, the HP model suggests splitting the vertices of the monomers into two types; hydrophobic, denoted as $H$, and polar, denoted as $P$. The matrix for $e_\Sigma$ using HP models is shown in table 2.1.
The HP model is extremely efficient when used in simulations, due to the very limited domain of it. However, the number of structures optimal according to the HP model is huge [3]. Therefore, the HPNX model, a refinement of the HP model, is sometimes used. The types H, P, N and X resemble hydrophobic, positive, negative and neutral, respectively. The matrix used for HPNX energy function is shown in table 2.2.

The MJ model takes each of the twenty possible amino acids as a representative to a structure unique on its own, and provides a finer matrix than HP and HPNX models, giving the values of the 210 possible pairwise interactions between any two amino acids [16]. The MJ model is used in the simulations designed for this thesis, and its corresponding matrix is given in appendix A.

**Energy function for backbone model**

For backbone model, the total energy of a structure is defined as a sum of all energies computed over all non-consecutive pairs of vertices (representing monomers). The total energy of a protein structure $(p, R)$, given an energy function $e(v_1, v_2)$, is defined as follows:

$$E(p, R) = \sum_{i=1}^{i=|p|} \sum_{j=i+2}^{j=|p|} e(p_i, p_j) \tag{2.14}$$

**Energy function for side-chain model**

For side-chain model, the total energy of a structure is defined as a sum of all energies computed over all pairs of side-chains (representing amino acid chains). The total energy of a protein structure $(p, s, R)$, given an energy function $e(v_1, v_2)$, is defined as follows:

$$E(p, s, R) = \sum_{i=1}^{i=|s|} \sum_{j=i+1}^{j=|s|} e(s_i, s_j) \tag{2.15}$$

## 2.3 Constraint satisfaction problem

A *constraint satisfaction problem*, CSP, is a mathematical problem stated as a set of variables on which some constraints are imposed. Solving a CSP means finding an assignment of values to the variables such that all constraints are satisfied. Formally, a

| | H | P |
|---|---|---|
| H | -1 | 0 |
| P | 0 | 0 |

Table 2.1: Energy function matrix of HP model [9]

| | H | P | N | X |
|---|---|---|---|---|
| H | -4 | 0 | 0 | 0 |
| P | 0 | 1 | -1 | 0 |
| N | 0 | -1 | 1 | 0 |
| X | 0 | 0 | 0 | 0 |

Table 2.2: Energy function matrix of HPNX model [3]

CSP is a triple $< X, D, C >$. $X$ is a set of variables. $D$ is a domain of values. $C$ is a set of constraints. A constraint can be understood as a pair $(t, V)$, where $t$ is a tuple of variables and $V$ is a set of tuples of values, each tuple is of the same size of that of $t$. The meaning of a constraint to be *satisfied* is that there is an evaluation function $u : X \to D$ such that $(u(x_1), u(x_2), ..., u(x_n)) \in V$. A *solution* for the CSP is an evaluation satisfying all constraints in $C$.

## 2.4 CSP for self avoiding walk in 3D

To ease the understanding of the CSP elements, we introduce an easy problem, relevant to the topic of the thesis, and describe its corresponding CSP.
The problem is to compute all SAWs of a certain length $n$ on a lattice $L$. Given that SAWs of length $n$ on an infinite lattice are infinitely many, it is only interesting to investigate SAWs beginning at the origin, $\vec{0}$, as any other SAW of length $n$ would be a translation of one SAW beginning at the origin.

**Problem** For a given $n$, formulate a CSP that any solution of which defines a SAW $p$ of length $n$ on a lattice $L$, such that $p$ starts at the origin.

**Solution** A viable choice of variables of the CSP is to choose $n$ variables: $X = \{p_1, p_2, ..., p_n\}$ representing the vertices of the SAW. The length of the SAW is $n$, a finite value. Thus, a finite domain of points $D$ can be assigned to the variables, depending on the lattice structure. For $\mathbb{Z}^3$ or $FCC$ lattices, the domain is stated explicitly in section 2.4.2, and will be denoted as the finite frame $F \subset L$, so that $\forall_{1 \leq i \leq n} : D_i \subseteq F$. The constraints imposed on the vertices constituting the SAW, $C$, are described as follows:

$$
\begin{aligned}
& |p| = n \\
\wedge \quad & \forall_{1 \leq i \leq |p|} : p_i \in L \\
\wedge \quad & p_1 = \vec{0} \\
\wedge \quad & \forall_{2 \leq i \leq |p|} : Neighbor(p_{i-1}, p_i) \\
\wedge \quad & \forall_{i \neq j} : p_i \neq p_j
\end{aligned}
\tag{2.16}
$$

The neighborhood binary constraint, $Neighbor(p_{i-1}, p_i)$, is not clearly defined, and thus will be discussed in section 2.4.1. One last point to state is that most implemented constraint problem solvers only work with numeric values, preferably integers. Therefore, the domain of 3D points is transformed to an isomorphic domain of integer values as described in section 2.4.3.

### 2.4.1 Neighborhood binary constraint

The neighborhood binary constraint is defined on two variables, each representing a lattice point, so that it enables the constraint solver of branching and searching for a solution respecting the neighborhood constraint. The neighborhood constraint is defined following

its intuitive meaning, that the two vertices represented by variables will be neighbors in the given lattice if the constraint is satisfied.
For the two variables $p_i$ and $p_j$, of domains $D_i$, $D_j$ respectively, the constraints dictating that they are neighboring are:

$$\begin{aligned} & \forall_{p_i \in D_i} \exists_{p_j \in D_j} : Neighbor_L(p_i, p_j) \\ \wedge \quad & \forall_{p_j \in D_j} \exists_{p_i \in D_i} : Neighbor_L(p_i, p_j) \end{aligned} \tag{2.17}$$

The domains of the points $p_i$ and $p_j$ are constrained only to the pairs where $p_i$ is neighboring to $p_j$. The implementation of the neighborhood binary constraint filters any point in each domain which has no neighbor in the opposite domain. Following is a description of the algorithm applied to grant the satisfaction of the constraint:
Two points on a lattice are neighbors if the difference between them is one of the neighborhood vectors. Thus, the neighborhood binary constraint, when applied on two variables $p_i$ and $p_j$, limited to two domains $p_i \in D_i$ and $p_j \in D_j$, where $D_i, D_j \subset L$ works as follows:
Without loss of generality, let's assume that the domain of $p_i$ is smaller than of $p_j$. The binary neighborhood constraint sweeps over all points in $D_i$ and searches for all supporter neighbors in $D_j$. If no supporter was found in $D_j$, the point is removed from $D_i$. After that, $D_j$ is cleaned up from every point that has no supporter neighbor in $D_i$.

### 2.4.2 Frame size problem

To possibly model the lattice $L$ in a computational model, a finite frame $F$ is chosen to have a finite domain for the points. In general, an easily representable frame is defined by a bounding box $[x_1, x_2] \times [y_1, y_2] \times [z_1, z_2]$. Such representation is convenient as it can be transformed to an isomorphic domain of integers using translation and row-major ordering, as described in section 2.4.3. Therefore it is enough to define the boundaries of that box, namely $x_1$, $x_2$, $y_1$, $y_2$, $z_1$, and $z_2$.
In $\mathbb{Z}^3$ or $FCC$ lattices, the difference between two neighbors in any coordinate does not exceed 1. Therefore, every chain $p$ of length $n$ starting from the origin can not reach any point having a coordinate of absolute value higher than $n$. Thus, a sufficient finite frame is $F = [-n-k, n+k] \times [-n-k, n+k] \times [-n-k, n+k]$, where $k \geq 1$, constant, to allow dealing with the set of neighborhood vectors as a set of constant integers even with the problem transformed to the integer domain using raw-major ordering. After transforming the frame into a domain of integers using raw-major ordering, this positive $k$ eliminates both the errors resulting from not handling wrapping around the frame once reaching its boundary, and also eliminates the degeneracy in efficiency resulting from handling the special vertices at the boundaries each time the neighborhood binary constraint is run.
More formally, a frame $F$ is defined as follows:

$$F = \{(x, y, z) : (x, y, z) \in L \wedge x_1 \leq x \leq x_2 \wedge y_1 \leq y \leq y_2 \wedge z_1 \leq z \leq z_2\} \tag{2.18}$$

$Size_F$ denotes the size of a frame $F$, the largest domain of values at one dimension:

$$Size_F = max(x_2 - x_1 + 1, y_2 - y_1 + 1, z_2 - z_1 + 1) \tag{2.19}$$

As stated before, the frame used for this problem is defined as:

$$F = \{(x, y, z) : (x, y, z) \in L \wedge n - k \leq x, y, z \leq n + k\} \tag{2.20}$$

The size of this frame is $Size_F = 2n + 2k + 1 = 2n + 3$, for $k = 1$ as shown sufficient.

### 2.4.3 Indexing

As stated before, most of the implemented constraint solvers handle only numeric values. So, for efficiency and convenience purposes, the points of the lattice are mapped by a one-to-one mapping to non-negative integers using row-major ordering in order to be usable in a constraint solver, namely the Gecode library, an open C++ library that enables solving CSPs natively in C++, efficiently [18].
We will follow a definition of raw-major ordering that is only defined on non-negative integer coordinates. Thus, to get all the points of the frame $F$ in the first octant of the infinite lattice $L$, the starting position of the chain will be translated from the origin to $(n + k, n + k, n + k)$ by a translation vector $\vec{t} = (n + k, n + k, n + k)$. Thus, the new translated frame lies only on non-negative integer coordinates.

$$F_{\vec{t}} = \{(x, y, z) : (x, y, z) \in L \wedge 0 \leq x, y, z \leq 2n + 2k\} \tag{2.21}$$

The frame size is the same, not affected by the translation, $Size_{F_{\vec{t}}} = Size_F = 2n+2k+1$. After that, a point of coordinates $(x, y, z)$ in the frame $F_{\vec{t}}$ of size $Size_{F_{\vec{t}}}$ is mapped, by a one-to-one mapping, to a single integer using row-major ordering as:

$$I(x, y, z) = x + y \cdot Size_{F_{\vec{t}}} + z \cdot Size^2_{F_{\vec{t}}} \tag{2.22}$$

# Chapter 3

# Related Work

Protein structure prediction proposes the question of predicting the spatial structure of a protein knowing its amino acid sequence. Levinthal showed that the number of possible structures of a protein sequence is extremely huge. However, it is known that the folding procedure of a protein happen in the scale of milliseconds, and that it always results into a certain structure, dependent on the amino acid sequence, assumed to be optimal in energy, and kinetically reachable according to the protein formation conditions. This notice, known as Levinthal paradox [17], leads to the realization that reaching the structure of a protein from its amino acid sequence follows a kinetically favorable procedure, known as protein folding. Thus, one approach for answering this question is performing a simulation of the protein folding beginning by the amino acid sequence.
It is computationally intractable to examine every possible structure of the protein in order to predict the most favorable structure. Various models of representing the problem of protein structure prediction was shown to be NP-hard [1]. The approach of predicting the stable structure of a protein by simulating its collapse is named as protein folding simulation [1].

## 3.1 Simulation methods

Ab initio protein structure prediction is based on simulating the physical principles applied on the protein to follow the folding steps. A folding simulation follows one simulation model, such as simulated annealing, Markov chain simulation or energy local minima search. Simulation methods based on the idea of a move set take the protein structure from one state to another, iteratively. The possible next states from a given protein structure are named as its neighboring structures and are defined by one of the allowed move sets applied on that protein structure. Consequently, the simulation method decides for the choice of the next structure from the neighboring structures. Some of the simulation methods include simulated annealing and energy local minima search.

### 3.1.1 Simulated annealing

In [6], a Monte Carlo approach was followed for the choice of the next protein structure. Metropolis criterion was used for accepting or rejecting a move. This approach involves temperature as a parameter in the Metropolis criterion. Controlled heating and cooling

conditions affect the allowed moves. A condition with a higher temperature allows a higher possibility for larger structural changes to occur.

### 3.1.2 Energy local minima search

Other simulation methods used Monte Carlo approaches for finding a ground state for the protein chain, following a path of structures that are descending in energy. This approach is following a folding simulation that searches for structures that are local minima. Considering the structure space of the protein to be fold as a domain of the energy function, and by the use of a move set defining structure neighborhoods, local minima search algorithms can be applied to find protein structures that are locally optimal in energy [17]. Gradient descent is a simple method of finding local minima by iteratively choosing the neighboring structure with lowest energy. This method is used in the thesis experiment, for applying the K-local move set.

## 3.2 Classification of move sets

Move sets vary according to their ergodicity (completeness). An *ergodic* move set (or set of move sets) enables every structure to reach any possible formation by its application zero or more times. A *non-ergodic* move set can not allow some of the formations to occur starting from a certain structure. For a complete folding simulation, an ergodic move set enables reaching any optimal structure, and thus ergodic move sets are preferable [9].
Move sets are defined typically with respect to a certain model (backbone or side-chain model) in a predefined lattice. However, common move sets, as shown later, are generalized to various lattices and models. The move sets introduced in this thesis are defined using a CSP approach, and thus are defined declaratively, and can apply to any lattice. The introduced move sets were defined for both backbone as well as side-chain models. In [2], the CSP approach to define a move set was used for protein folding simulations in backbone model using the HP energy function. Search for local optima was introduced as a general framework in [15], and is used in the experiments in this thesis.
A *local move set* is a move set affecting at most constant number of consecutive monomers of the protein structure. The number of amino acids involved in a move is inversely related to the probability of the move [20]. Thus, it is suggested that local moves with fewer monomers involved are more likely to occur than moves that change the entire structure of the chain. However local move sets are not ergodic [9]. The ergodicity of a move set is necessary so an the optimal structure would be reachable [9]. Thus, some of the research concentrated on ergodic move sets involving less monomers such as the pull move set [9]. Other methods used Monte Carlo random simulations to the folding giving higher probabilities to moves involving less amino acids [20].

### 3.2.1 Pull move set [9]

A pull move is defined operationally as a move involving one monomer $V_i$ that has a free diagonally-adjacent cell to move into, named as C, with the one other corner of the square defined by the two cells $V_i$ and C is occupied by $V_{i+1}$. The fourth square corner, named as L, must be either empty or filled by $V_{i-1}$, in order to allow the pull move to

occur. The vertex $V_i$ is moved into the free square C. The vertex $V_{i-1}$ is either moved to L (if L is free) or the move step stops only by moving $V_i$. In case $V_{i-1}$ is moved to L, every vertex $V_j$ is moved to the place previously occupied by $V_{j+2}$, starting from $V_{i-2}$ and going by j down until reaching a vertex that is moved and still neighboring its old neighbors, or reaching the end of the chain.

### 3.2.2 Standard local move set for [17]

The standard local move set is a local move set that is originally used for backbone models [17] and was adopted for side-chain models as well [6].
In this move set, either a random single monomer is chosen to be moved, or two consecutive monomers (randomly chosen) are moved. The move is allowed if a single monomer was chosen are either moving a monomer ending the chain, and then it is moved in a way that the protein still respects connectivity and self-avoidance. If the monomer is not at the end of a chain, moving it is allowed if moving it to a diagonally adjacent cell keeps the protein connected and self-avoiding. If a pair of consecutive monomers were chosen to be moved, they are moved in a crankshaft fashion by moving the two consecutive monomers into two other adjacent cells, diagonally adjacent to their original positions [17]. This move set can be defined for more general lattices, and for a larger number of consecutive monomers to be altered. It can be defined also for side-chain models as well [6, 11]. In [4], the local move set was defined for triangular lattices. The standard local move set defined by [17] for backbone models is shown in figure 3.1.

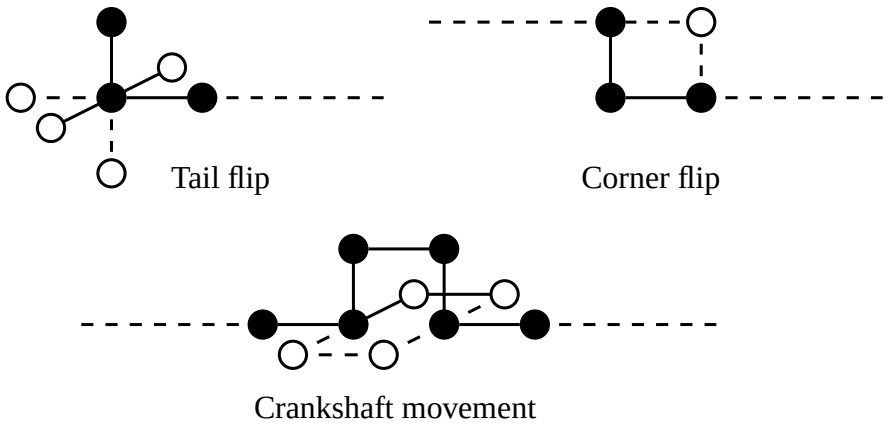


Figure 3.1: The standard local move set for backbone model [17]

Single moves and crankshaft moves were given probabilities of 20% of the tail flip, 20% of the corner flip and 60% of the crankshaft movement. If the positions of the new back bone beads are valid, the side-chain moves are chosen. Otherwise, the side-chain beads are tried to be moved simultaneously [11].
Chapter 4 gives a generic definition for a more general form of this move set, using a CSP definition. The move set used in [17] can be defined as the K-local move set for $K = 2$, applied on backbone model, as defined in sections 4.1. However the CSP approach allowed an easy adoption for this move set into side-chain models, and it allows a more general definition, for $K > 2$, that works using any arbitrary lattice.

### 3.2.3 The pivot move set [10]

The pivot move set chooses a random monomer along the chain (the pivot), then applies reflection, rotation, or both to one side of the chain. The move is allowed if by applying it the structure remains respecting the self avoiding conditions. This move set applies a large structural change to the protein, and thus can be combined with a local move set to simulate more common moves that affect limited number of monomers.

### 3.2.4 MS3 Local move set [11]

The MS3 move set is a local move set that extends the standard local move set. It includes all the moves from the standard local move set, in addition to moves involving two or three consecutive monomers. The monomers are moved into diagonally adjacent positions if the continuity of the chain is maintained as well as self-avoidance. Some examples of the application of the move set are shown in figure 3.2. The probability of a move involving a number of monomers is proportional to a negative exponential of the number of chosen monomers. The MS3 move set was shown to be more efficient than the standard local move set, because it allows more moves that offer a more flexible structural change [11].

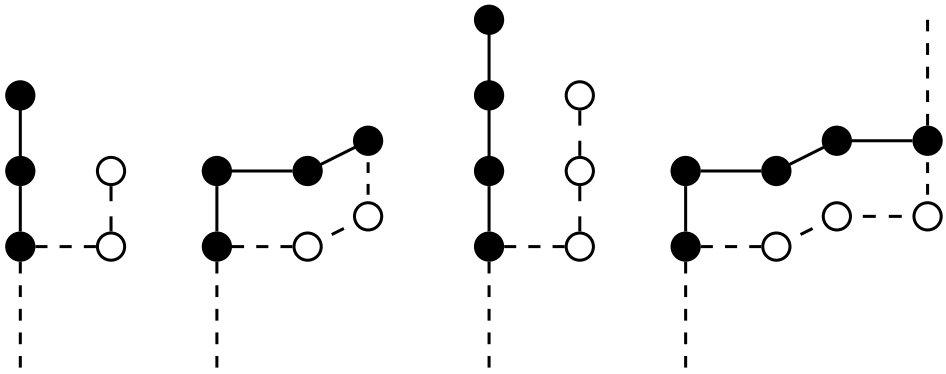

Figure 3.2: Examples of the MS3 moves extending the standard local move set [11]

### 3.2.5 Energy function and spin states [11]

Spin states are introduced to represent the degrees of freedom of the monomers by using $n$ spin degrees (numbered from 0 to $n-1$) and assuming that 0 is the native state. side-chains that are in the native state is the only one contributing to the energy function. Also, a breathing motion is introduced for the structure backbone to allow the side-chains to be located inside the interior of the folded protein [6].
The introduction of side-chains divided the beads into back bone beads and side-chain beads. An energy function was developed as a sum of the interactions among back bone beads together, side-chain beads together, or the interaction between a back bone bead with a side-chain bead [11]. Spin states are commonly used in protein folding simulation with side-chain models, taking into account the energy participation from side-chain beads that are in the native state only [11, 6].

### 3.2.6 Side-chain placement

Some research is directed towards predicting side-chain placements after building a backbone configuration with no initial knowledge of the side-chains [8]. Further research showed that interaction between side-chains does not affect the accuracy of the side-chain placement prediction, although prediction only using the interactions between side-chains was shown significantly accurate. That resulted in the conclusion that both types of interactions are consistent [19].

# Chapter 4

# Interval-based local move set

In this thesis, we introduce CSP definitions to local move sets. The K-local move set used in the experiment of this thesis is presented. The K-local move set follows a generic definition, flexible in application to any model or lattice. The following chapter will introduce CSPs solving the K-local move set for backbone model and side-chain model in $\mathbb{Z}^3$ and FCC lattices. The CSPs are extensible to be defined in other lattices as well, with the only change altered is at the domain of the variables. An optimized variation of the K-local move set is presented later in this chapter and its optimized CSP is presented as well. Next chapter presents a global move set and its CSP definition for both backbone and side-chain models as well.
The advantage of using CSP definitions is providing a generic working description for the move set, extensible to any lattice used, and any length used, as will be shown later in this chapter.

## 4.1 K-local move set in backbone model

For protein structures in backbone model, a generic definition of a local move set is to fix the whole structure, except for a consecutive subinterval of length no more than $K$. This definition is flexible as it can be applied regardless of the lattice containing the structure, and it is easily extensible to other models, such as the side-chain model, as shown in section 4.2.
More formally, given $p$, the protein chain in backbone model, as well as the previously mentioned length $K$, the K-local move set is defined by choosing *start* and *end*, two indices of vertices in the SAW $p$, such that $1 \leq start \leq end \leq min(|p|, start + K - 1)$. The move set applies by fixing the whole protein structure but allowing the subinterval $[p_{start}, p_{end}]$ to take any formation as long as it respects the conditions of connectivity and self-avoidance.

### 4.1.1 CSP for the K-local move set in backbone model

**Problem** Given $p$, a protein structure in backbone model in a lattice $L$, as well as $K < |p|$, formulate a CSP that any solution of which defines another structure $p'$ that results from applying one move out of the K-local move set to the given $p$.

**Solution** The CSP presented to solve this problem is $< X, D, C >$. The CSP contains $|p| + 2$ variables: $X = \left\{start, end, p'_1, p'_2, ..., p'_{|p|}\right\}$ representing *start* and *end*, the two indices as described earlier, as well as the vertices of the altered protein structure, $p'$. Note that $|p'| = |p|$.
The whole solution of the CSP is defined by an assignment for each element of the variables, $X$. However, the move set is only defined in terms of $p'$. Thus, the CSP is preferred not to describe duplicate solutions, in terms of equal $p'$. Two duplicate solutions might occur by finding a solution for the CSP that involves a structure $p'$ different from $p$ only between *start* and *end*, inclusively. However, this same $p'$ is different from $p$ between $start'$ and $end'$, inclusively, as well, such that $start' \neq start \vee end' \neq end$. The two solutions, albeit different for the CSP solver, are considered redundant from the move set point of view. Thus, it is favorable to obtain a CSP that captures every possible $p'$ exactly once.
Consequently, the CSP can be defined in a way that only allows the smallest interval $[p'_{start}, p'_{end}]$ to be chosen for each $p'$. This can be entailed by restricting $p'_{start}$ and $p'_{end}$ to be different from $p_{start}$ and $p_{end}$, respectively. This way, other solutions with other values, $start'$ and $end'$, can not capture the same altered structure, $p'$, that was captured with *start* and *end*. Also, any possible $p'$ can be captured. So, this added condition will grant a complete solution space, with no duplicates of $p'$.
The domains of the variables, $D$, are defined as follows:

$$\begin{aligned} & start, end \in \{1, 2, ..., |p|\} \\ & \forall_{1 \leq i \leq |p|} : p'_i \in D_i \subseteq F, \qquad \text{where } F \text{ is defined in section 4.1.2} \end{aligned} \tag{4.1}$$

The constraints, $C$, of this CSP are described as follows:

$$\begin{aligned} & 1 \leq start \leq end \leq |p| \\ \wedge \quad & end - start < K \\ \wedge \quad & p'_{start} \neq p_{start} \wedge p'_{end} \neq p_{end} \\ \wedge \quad & \forall_{1 \leq u < start} : p'_u = p_u \wedge \forall_{end < u \leq |p|} : p'_u = p_u \\ \wedge \quad & \forall_{max(2,start) \leq u \leq min(end+1,|p|)} : Neighbor_L(p'_{u-1}, p'_u) \qquad \text{connectivity} \\ \wedge \quad & \forall_{1 \leq i < j \leq |p|} : p'_i \neq p'_j \qquad \text{self avoidance} \end{aligned} \tag{4.2}$$

### 4.1.2 Variable Domains for the CSP

Suppose $p = (p_1, p_2, ..., p_{|p|})$ is the given protein structure. Let $p_{min}$ and $p_{max}$ be defined as follows:

$$\begin{aligned} p_{min} &= (min_{1 \leq u \leq |p|}(p_{u_x}), min_{1 \leq u \leq |p|}(p_{u_y}), min_{1 \leq u \leq |p|}(p_{u_z})) \\ p_{max} &= (max_{1 \leq u \leq |p|}(p_{u_x}), max_{1 \leq u \leq |p|}(p_{u_y}), max_{1 \leq u \leq |p|}(p_{u_z})) \end{aligned} \tag{4.3}$$

In other words, $p_{min}$ and $p_{max}$ resemble the corners of a bounding frame containing the whole given protein structure, $p$. That means:

$$\begin{aligned} \forall_{1 \leq u \leq |p|} \quad : \quad & p_{min_x} \leq p_{u_x} \leq p_{max_x} \\ \wedge \quad & p_{min_y} \leq p_{u_y} \leq p_{max_y} \\ \wedge \quad & p_{min_z} \leq p_{u_z} \leq p_{max_z} \end{aligned} \tag{4.4}$$

To limit the domain of $p'$, consider either point of $p_{start-1}$ or $p_{end+1}$. In some cases, *start* might be equal to 1 or *end* might be equal to $|p|$, however, both can not happen simultaneously, because $end - start < K < |p|$. So, let $p_{pivot}$ be $p_{start-1}$, or $p_{end+1}$, in case $p_{start-1}$ is not defined. In other words, $p_{pivot}$ is a vertex neighboring to one end of the interval $[p'_{start}, p'_{end}]$. It follows that the subinterval $[p'_{max(1,pivot-K)}, p'_{min(|p'|,pivot+K)}]$ contains the subinterval $[p'_{start}, p'_{end}]$, because $end - start < K$.
$p_{pivot}$ is outside the subinterval that was chosen to be altered for the local move set, so $p'_{pivot} = p_{pivot}$. In $\mathbb{Z}^3$ or $FCC$ lattices, the difference between two neighbors in any coordinate does not exceed 1. Thus, any vertex in the subinterval $[p'_{max(1,pivot-K)}, p'_{min(|p'|,pivot+K)}]$ is at most $K$ units distant away from $p_{pivot}$, in any coordinate. From equation 4.4, $p_{pivot}$ is contained between $p_{min}$ and $p_{max}$. Thus,

$$\begin{aligned} \forall_{max(1,pivot-K) \leq u \leq min(|p'|,pivot+K)} \quad : \quad & p_{min_x} - K \leq p'_{u_x} \leq p_{max_x} + K \\ \wedge \quad & p_{min_y} - K \leq p'_{u_y} \leq p_{max_y} + K \\ \wedge \quad & p_{min_z} - K \leq p'_{u_z} \leq p_{max_z} + K \end{aligned} \tag{4.5}$$

In addition, any other vertex outside $[p'_{start}, p'_{end}]$ is kept unchanged, so it is contained between $p_{min}$ and $p_{max}$, as well. Therefore, for any point $p'$,

$$\begin{aligned} \forall_{1 \leq u \leq |p'|} \quad : \quad & p_{min_x} - K \leq p'_{u_x} \leq p_{max_x} + K \\ \wedge \quad & p_{min_y} - K \leq p'_{u_y} \leq p_{max_y} + K \\ \wedge \quad & p_{min_z} - K \leq p'_{u_z} \leq p_{max_z} + K \end{aligned} \tag{4.6}$$

As mentioned in section 2.4.2, the implementation of the neighborhood binary constraint dictates an additional margin $k = 1$ for the frame. So, the bounding box for the frame can be defined as:

$$F = [p'_{min_x}, p'_{max_x}] \times [p'_{min_y}, p'_{max_y}] \times [p'_{min_z}, p'_{max_z}] \tag{4.7}$$

The definition of $p'_{min}$ and $p'_{max}$ is:

$$\begin{aligned} p'_{min} &= p_{min} - (K + k, K + k, K + k) \\ p'_{max} &= p_{max} + (K + k, K + k, K + k) \end{aligned} \tag{4.8}$$

In order to get all the points of the frame $F$ in the first octant of the infinite lattice $L$, $p'_{min}$ is translated to the origin by a translation vector $\vec{t} = -p'_{min}$. Then, the translated frame is:

$$F_{\vec{t}} = \{\vec{v} : v + p'_{min} \in F\} \tag{4.9}$$

The frame size is given as described in equation 2.19. So any point in $F_{\vec{t}}$ can be mapped, by a one-to-one mapping, to a domain of integers using row-major ordering, as in equation 2.22.

## 4.2 K-local move set in side-chain model

The K-local move set defined for backbone model in section 4.1, is extensible to other protein structure models. It can be defined to a protein structure in side-chain model, similarly, by fixing the whole structure, except for a consecutive subinterval of length no more than $K$. The backbone vertices, as well as their side chains, which belong to that chosen subinterval can take any formation, as long as the structure is maintained as a valid side-chain protein structure.

### 4.2.1 CSP for the K-local move set in side-chain model

**Problem** Given $(p, s)$, a protein structure modelled by side-chain model in a lattice $L$, as well as $K < |p|$, formulate a CSP that any solution of which defines another structure $(p', s')$ that results from applying one move out of the K-local move set to the given $(p, s)$.

**Solution** The CSP presented to solve this problem is $< X, D, C >$. The CSP contains $2|p| + 2$ variables: $X = \left\{start, end, p'_1, p'_2, ..., p'_{|p|}, s'_1, s'_2, ..., s'_{|p|}\right\}$. *start* and *end* define the interval, where only vertices that are allowed to take other positions; $[p_{start}, p_{end}]$, and their corresponding side-chains $[s_{start}, s_{end}]$. The other $2|p|$ variables describe the vertices of the altered protein structure, $(p', s')$. Note that $|p'| = |s'| = |p| = |s|$.
Similar to the case handled at backbone model in section 4.1, the two variables *start* and *end* are preferred not to describe duplicate $(p', s')$ structures. Thus, the smallest intervals $[p_{start}, p_{end}]$ and $[s_{start}, s_{end}]$ are chosen, by restricting one of $p'_{start}$ and $s'_{start}$ to be different than their corresponding original vertices, $p_{start}$ and $s_{start}$, also one of $p'_{end}$ and $s'_{end}$ must be different than $p_{end}$ and $s_{end}$.
The domains of the variables, $D$, are defined as follows:

$$
\begin{aligned}
& start, end \in \{1, 2, ..., |p|\} \\
& \forall_{1 \le i \le |p|} : p'_i \in D_{p_i} \subseteq F \\
& \forall_{1 \le i \le |p|} : s'_i \in D_{s_i} \subseteq F, \qquad \text{where } F \text{ is defined in section 4.2.2}
\end{aligned} \tag{4.10}
$$

The constraints of this CSP, $C$ are described as follows:

$$
\begin{aligned}
& 1 \le start \le end \le |p| \\
\wedge \quad & end - start < K \\
\wedge \quad & (p'_{start} \neq p_{start} \vee s'_{start} \neq s_{start}) \\
\wedge \quad & (p'_{end} \neq p_{end} \vee s'_{end} \neq s_{end}) \\
\wedge \quad & \forall_{1 \le u < start} : p'_u = p_u \wedge s'_u = s_u \\
\wedge \quad & \forall_{end < u \le |p|} : p'_u = p_u \wedge s'_u = s_u \\
\wedge \quad & \forall_{max(2,start) \le u \le min(end+1,|p|)} : Neighbor_L(p'_{u-1}, p'_u) \quad \text{connectivity of backbone} \\
\wedge \quad & \forall_{start \le u \le end} : Neighbor_L(s'_u, p'_u) \quad \text{connectivity of side-chains} \\
\wedge \quad & \forall_{1 \le i < j \le |p|} : p'_i \neq p'_j \wedge s'_i \neq s'_j \wedge \forall_{1 \le i,j \le |p|} : p'_i \neq s'_j \quad \text{self avoidance}
\end{aligned} \tag{4.11}
$$

### 4.2.2 Variable Domains for the CSP

The argument followed in section 4.1.2 applies for side-chain modelled protein structures, but with $k = 2$ instead of $k = 1$, to accommodate that side-chain vertices are at most one unit away from their corresponding backbone vertices.
More formally, given $(p, s)$ the initial protein structure. $p_{min}$ and $p_{max}$ are defined to be the corners of the bounding frame containing the backbone vertices.

$$\begin{aligned} p_{min} &= (min_{1\leq u\leq |p|}(p_{u_x}), min_{1\leq u\leq |p|}(p_{u_y}), min_{1\leq u\leq |p|}(p_{u_z})) \\ p_{max} &= (max_{1\leq u\leq |p|}(p_{u_x}), max_{1\leq u\leq |p|}(p_{u_y}), max_{1\leq u\leq |p|}(p_{u_z})) \end{aligned} \tag{4.12}$$

To limit the domain of $(p', s')$, consider the point $p_{pivot}$, similar to the one defined in section 4.1.2. $p_{pivot}$ is a vertex neighboring to one end of the interval $[p'_{start}, p'_{end}]$. Similarly, follows that the subinterval $[p'_{max(1,pivot-K)}, p'_{min(|p'|,pivot+K)}]$ contains the subinterval $[p'_{start}, p'_{end}]$, because $end - start < K$. The resulting domains for $p'_u = (p'_{u_x}, p'_{u_y}, p'_{u_z})$ are the same as solved in equations 4.5 and 4.6.
Side-chain vertices are at most 1 unit away from their corresponding backbones, thus,

$$\begin{aligned} \forall_{1\leq u\leq |p'|} \quad : \quad & p_{min_x} - K - 1 \leq s'_{u_x} \leq p_{max_x} + K + 1 \\ \wedge \quad & p_{min_y} - K - 1 \leq s'_{u_y} \leq p_{max_y} + K + 1 \\ \wedge \quad & p_{min_z} - K - 1 \leq s'_{u_z} \leq p_{max_z} + K + 1 \end{aligned} \tag{4.13}$$

As mentioned in section 2.4.2, the implementation of the neighborhood binary constraint dictates an additional margin $k = 1$ for the frame. So, to accommodate side-chains additionally, $k$ is chosen to be 2, and the bounding box for the frame can be defined in the same way at equations 4.7 and 4.9, for the new value of $k = 2$.
Reducing the domain to that the finite frame, of size that is given in equation 2.19, any point in $F_{\vec{t}}$ can be mapped, by a one-to-one mapping, to a domain of integers using row-major ordering, as in equation 2.22.

## 4.3 An optimized variation of the K-local move set

Similar to the move set defined in section 4.1, another move set can be defined by choosing a subinterval of the protein chain, $(p_{start}, s_{end})$, of length at most $K$. The optimized variation is choosing not to alter the boundaries of the chosen subinterval, and only allowing the open subinterval $(p_{start}, s_{end})$ to take an alternative formation.
The advantage from such formulation to the move set is its efficiency. The domains of the variables chosen for the CSP will be much less as shown later in 4.3.2. However, applying this local move set alone in a folding simulation will not allow the ends of the protein chain to take different vertices other than their initial ones (because they are always excluded from being moved). Thus, for any folding simulation to use this local move set, it must include other possibilities of move sets, preferably global ones, such as the pivot move set or the pull move set, which would also grant the ergodicity of the possible move sets.
In this section, the CSP for this move set, defined on backbone model, will be presented.

### 4.3.1 CSP for the open K-local move set in backbone model

The open K-local move set for a protein structure in the backbone model $p$ can be defined by choosing two vertices $p_{start}$ and $p_{end}$, of distance no more than $K \geq 3$, and allowing the chain in between (excluding $p_{start}$ and $p_{end}$) to take any formation as long as it maintains the conditions of connectivity and self-avoidance. The rest of the chain outside the interval $(p_{start}, p_{end})$ should be left unchanged.

**Problem** Given $p$, a protein structure modelled by backbone model in a lattice $L$, as well as $K < |p|$, formulate a CSP that any solution of which defines another structure $p'$ that results from applying one move out of the open K-local move set to the given $p$.

**Solution** The CSP solving this problem, $< X, D, C >$, contains $K$ variables: $X = \{start, end, q_1, q_2, ..., q_{K-2}\}$. The variables $start$ and $end$ denote the previously described indices. The variables $q_1, q_2, ...q_{end-start-1}$ resemble the vertices allowed to be altered from the given protein chain $p$, which are $p'_{start+1}$, $p'_{start+2}$, ..., $p'_{end-1}$. Finally, $p'$ will be described as $(p_1, p_2, ..., p_{start}, q_1, q_2, ..., q_{end-start-1}, p_{end}, p_{end+1}, ..., p_{|p|})$.
To eliminate duplicate possibilities of $p'$, $p'_{start+1}$ and $p'_{end-1}$, resembled by $q_1$ and $q_{end-start-1}$, they are restricted to be different than $p_{start+1}$ and $p_{end-1}$, respectively. Also, $q_{end-start}$, $q_{end-start+1}$, ..., $q_{K-2}$ has to be set to an arbitrary value, in order to have a unique solution per $p'$.
The constraints of this CSP, $C$ are described as follows:

$$
\begin{aligned}
& 1 \leq start \leq end \leq |p| \\
\wedge \quad & 1 < end - start < K \\
\wedge \quad & q_1 \neq p_{start+1} \wedge q_{end-start-1} \neq p_{end-1} \\
\wedge \quad & \forall_{end-start \leq u \leq K-2} : q_u = 0 \\
\wedge \quad & Neighbor_L(p_{start}, q_1) \wedge Neighbor_L(q_{end-start-1}, p_{end}) \\
\wedge \quad & \forall_{2 \leq u \leq end-start-1} : Neighbor_L(q_{u-1}, q_u) && \text{connectivity} \\
\wedge \quad & \forall_{1 \leq i < j \leq end-start-1} : q_i \neq q_j \\
\wedge \quad & \forall_{1 \leq i \leq start} \forall_{1 \leq j \leq end-start-1} : p_i \neq q_j \\
\wedge \quad & \forall_{end \leq i \leq |p|} \forall_{1 \leq j \leq end-start-1} : p_i \neq q_j && \text{self avoidance} \quad (4.14)
\end{aligned}
$$

### 4.3.2 Variable Domains for the CSP

The domains of the variables, $D$ are:

$$
\begin{aligned}
& start, end \in \{1, 2, ..., |p|\} \\
& \forall_{1 \leq i \leq K-2} : q_i \in D_i \subseteq F \qquad (4.15)
\end{aligned}
$$

The chosen frame, $F$, will be smaller than that shown in section 4.1.2. To realize that, the subinterval of the structure from $p_{start}$ to $p_{end}$, is of length $l = end - start + 1 \leq K$. This subinterval is translated into a finite frame $F$. This finite lattice should allow any SAW of the same length $l$ to lie completely within it. In $\mathbb{Z}^3$ or FCC, the size of this lattice $Size_F$ was found to be bounded from above by a constant added to the length of

the subchain $Size_F = l + k$, where $k \geq 1$. To prove this, let the point $\vec{F_c} = \frac{\vec{p_{start}} + \vec{p_{end}}}{2}$ to be in the center of the frame $F$, defined as:

$$F = \left\{ (x, y, z) : (x, y, z) \in L \wedge max(|x - F_{c_x}|, |y - F_{c_y}|, |z - F_{c_z}|) \leq \frac{l + k}{2} \right\} \quad (4.16)$$

The notion of a distance between two points on a lattice is used in the proof; $Dis(p_1, p_2)$ will denote the length of the shortest possible SAW from $p_1$ to $p_2$.
Using the neighborhood vectors of the lattices $\mathbb{Z}^3$ or FCC, $Dis(p_1, p_2) \geq |p_{1_x} - p_{2_x}|$. The reason is, that every neighborhood vector either adds to the x-dimension or not. Thus, it needs at least a chain of $|p_{1_x} - p_{2_x}|$ neighbors to reach from $p_1$ to $p_2$.
Similarly,

$$\begin{aligned} &Dis(p_1, p_2) \geq |p_{1_x} - p_{2_x}| \\ &Dis(p_1, p_2) \geq |p_{1_y} - p_{2_y}| \\ &Dis(p_1, p_2) \geq |p_{1_z} - p_{2_z}| \end{aligned} \quad (4.17)$$

The symbol $SAW_{start,c,end}$ will denote a SAW that goes from $p_{start}$ to $p_{end}$ through $p_c$. The length of this SAW is at least as long as sum of lengths of the shortest SAW from $p_{start}$ to $p_c$ and the shortest SAW from $p_c$ to $p_{end}$:

$$|SAW_{i,c,j}| \geq Dis(p_{start}, p_c) + Dis(p_c, p_{end}) \quad (4.18)$$

**Statement** In the lattices $\mathbb{Z}^3$ or FCC, any lattice point of a SAW of length $l$ from $p_{start}$ to $p_{end}$ will not lie outside the frame $F$ previously defined.

**Proof** Assume a lattice point $p_c \notin F$ is a part of a SAW $SAW_{start,c,end}$ of length $l$ beginning by $p_{start}$ and ending in $p_{end}$, the proof will show that the length of the SAW is necessarily larger than $l$ and thus arrive at a contradiction.
As assumed, $p_c$ lies outside $F$. This implies that at least one of the following three statements is true:

$$\begin{aligned} &|p_{c_x} - \frac{p_{start_x} + p_{end_x}}{2}| > \frac{l + k}{2} \\ &|p_{c_y} - \frac{p_{start_y} + p_{end_y}}{2}| > \frac{l + k}{2} \\ &|p_{c_z} - \frac{p_{start_z} + p_{end_z}}{2}| > \frac{l + k}{2} \end{aligned} \quad (4.19)$$

By the symmetry between the three coordinates, and without loss of generality, we assume that the first one of them is true:

$$|p_{c_x} - \frac{p_{start_x} + p_{end_x}}{2}| > \frac{l + k}{2} \quad (4.20)$$

The length of the self-avoiding walk is derived as follows:

$$\begin{aligned}
|SAW_{start,c,end}| &= l \text{ as assumed} \\
&\geq Dis(p_{start}, p_c) + Dis(p_c, p_{end}) \\
&\geq |p_{c_x} - p_{start_x}| + |p_{c_x} - p_{end_x}| \\
&\geq |2p_{c_x} - p_{start_x} - p_{end_x}| && \text{subadditivity} \\
&\geq 2|p_{c_x} - \frac{p_{start_x} + p_{end_x}}{2}| \\
&> 2(\frac{l+k}{2}) && \text{as } p_c \text{ is assumed to be outside } F \\
&> l + k && \text{contradiction} \qquad (4.21)
\end{aligned}$$

That proves the sufficiency of a frame of size $Size_F = l + k$, where $k \geq 3$, a constant to avoid off-by-one errors resulting from integer division as well as the offset discussed in section 2.4.2. In side-chain modelled protein structures, $k \geq 5$ is used to accommodate the side-chains attached to the backbone vertices. As $l = end - start + 1 \leq K$, the frame of size $Size_F = K + k$ will be sufficient for any value of *start* and *end*.

### 4.3.3 Indexed model

To get all the points of the frame $F$ in the first octant of the infinite lattice $L$, the point $\lfloor \frac{\vec{p_{start}} + \vec{p_{end}}}{2} \rfloor$ is to be translated to $(\lfloor \frac{l+k}{2} \rfloor, \lfloor \frac{l+k}{2} \rfloor, \lfloor \frac{l+k}{2} \rfloor)$ by a translation vector $\vec{t} = (\lfloor \frac{l+k}{2} \rfloor, \lfloor \frac{l+k}{2} \rfloor, \lfloor \frac{l+k}{2} \rfloor)\rfloor) - \lfloor \frac{\vec{p_{start}} + \vec{p_{end}}}{2} \rfloor$.
The translated frame is:

$$F_{\vec{t}} = \{\vec{v} : v + p'_{min} \in F\} \qquad (4.22)$$

The translated frame size is:

$$Size_{F_{\vec{t}}} = Size_F = K + k \qquad (4.23)$$

Finally, any point in $F_{\vec{t}}$ can be mapped, by a one-to-one mapping, to a single integer using row-major ordering, as in equation 2.22, making sure that any point used at the constraints from the original $p$ is translated and mapped to integers as well.

# Chapter 5

# Non-restricted global move set

## 5.1 Non-restricted global move set in backbone model

For a protein structure modelled in backbone model $p$, a non-restricted move set is defined by allowing at most $K < |p|$ residues to be altered, unlike the K-local move set, independent of their relative positions.

**Problem** Given a protein structure $p$ modelled by backbone model in a lattice $L$, and an integer $K$, where $1 \leq K < |p|$, formulate a CSP, that any solution of which describes another protein structure $p'$ that is different from $p$, with at most $K$ residues different from $p$.

**Solution** The CSP solving this problem, $< X, D, C >$, contains $|p|$ variables: $X = \left\{p'_1, p'_2, ..., p'_{|p|}\right\}$ representing the vertices of the altered protein structures, $p'$, where $|p'| = |p|$.
The constraints, $C$, of the CSP that defines the protein structure after applying the move set, $p'$ are described as follows:

$$\begin{aligned} & \forall_{2 \leq i \leq |p|} : Neighbor_L(p'_i, p'_{i-1}) \\ \wedge \quad & \forall_{1 \leq i < j \leq |p|} : p'_i \neq p'_j \\ \wedge \quad & 1 \leq |\{i : 1 \leq i \leq |p|, p_i \neq p'_i\}| \leq K \end{aligned} \tag{5.1}$$

The domains of the variables, $D$, are defined as follows:

$$\forall_{1 \leq i \leq |p|} : p'_i \in D_i \subseteq F \tag{5.2}$$

The frame chosen, $F$, is identical to that chosen in equation 4.7. It can be shown that for any residues $p'_a$ that was altered by applying the move set, there is a SAW beginning from it to a residue that was not altered, of length at most $K$, because the number of altered residues is at most $K$. Therefore, any vertex will be limited in domain by the same frame $F$, defined in equation 4.7, where $p'_{min}$ and $p'_{max}$ are defined exactly the same way in equation 4.8, for $k = 1$. The frame is to be translated, as described in equation 4.9, to enable representing the lattice points of the protein structure in an integer domain.

## 5.2 Non-restricted global move set in side-chain model

For a protein structure modelled in side-chain model $(p, s)$, a non-restricted move set is defined by allowing at most $K < |p|$ residues to be altered, independent of their positions.

**Problem** Given a protein structure $(p, s)$ modelled by side-chain model in a lattice $L$, and an integer $K$, where $1 \leq K < |p|$, formulate a CSP, that any solution of which describes another protein structure $(p', s')$ that is different from $(p, s)$, with at most $K$ residues different from $(p, s)$.

**Solution** The CSP solving this problem, $< X, D, C >$, contains $2|p|$ variables: $X = \left\{p'_1, p'_2, ..., p'_{|p|}, s'_1, s'_2, ..., s'_{|p|}\right\}$ representing the vertices of the altered protein structures, $(p', s')$, where $|p'| = |s'| = |p| = |s|$.
The constraints of the CSP, $C$, that defines the protein structure after applying the move set, $p'$ are described as follows:

$$\begin{aligned} & \forall_{2 \leq i \leq |p|} : Neighbor_L(p'_i, p'_{i-1}) \\ \wedge \quad & \forall_{1 \leq i \leq |p|} : Neighbor_L(p'_i, s'_i) \\ \wedge \quad & \forall_{1 \leq i < j \leq |p|} : p'_i \neq p'_j \wedge s'_i \neq s'_j \\ \wedge \quad & \forall_{1 \leq i,j \leq |p|} : p'_i \neq s'_j \\ \wedge \quad & 1 \leq |\{i : 1 \leq i \leq |p|, p_i \neq p'_i \vee s_i \neq s'_i\}| \leq K \end{aligned} \tag{5.3}$$

The domains of the variables, $D$, are defined as follows:

$$\forall_{1 \leq i \leq |p|} : p'_i \in D_i \subseteq F \tag{5.4}$$

The frame chosen, $F$, is identical to that chosen in equation 4.7. It can be shown that for any backbone vertex, $p'_a$, of a residue that was altered by applying the move set, there is a SAW beginning from it to a residue that was not altered, of length at most $K$, because the number of altered residues is at most $K$. Therefore, any vertex will be limited in domain by the same frame $F$, defined in equation 4.7, where $p'_{min}$ and $p'_{max}$ are defined exactly the same way as in equation 4.8, for $k = 2$, in order to allow for side-chains, that are at most one unit away from a backbone vertex. The frame is to be translated, as described in equation 4.9, to enable representing the lattice points of the protein structure in an integer domain.

# Chapter 6

# Experiment

The experiment of this thesis is performing two types of protein folding simulations for lattice protein models including side chains. The first simulation approach is a gradient descent applied on the energies of the protein structures. The second simulation approach is following a random sequence of protein structures, strictly descending in energy.
Known protein structures were used as inputs for both simulations. The monomer sequences for these proteins are described in table 6.1. The lattice used in the simulations is the FCC lattice, in order to provide a higher resolution of the modelled protein [14]. The initial structure of each protein sequence is a result from one of two sources:

1. the fitting of real protein structures into the FCC lattice following [12].
2. optimal structures, according to the HP energy model, generated by [13].

| PDB ID | Monomer Sequence (original and HP modelled) |
|---|---|
| 1BAZ-A | SKMPQVNLRWPREVLDLVRKVAEENGRSVNSEIYQRVMESFKKEGRIGA<br>PPHPPHPHPHPPPHHPHHPPHPPPPPPPHPPPHHPPHHPPHPPPPPHPP |
| 1J8E-A | GSHSCSSTQFKCNSGRCIPEHWTCDGDNDCGDYSDETHANCTNQ<br>PPPPHPPPPHPHPPPPHHPPPHPHPPPPPHPPHPPPPPPPHPPP |
| 1RH6-A | MYLTLQEWNARQRRPRSLETVRRWVRESRIFPPPVKDGREYLFHESAVKVDLNRP<br>HHHPHPPHPPPPPPPPPHPPHPPHHPPPPHHPPPHPPPPPHHHPPPPHPHPHPPP |
| 1Z0J-B | IEEELLLQQIDNIKAYIFDAKQCGRLDEVEVLTENLRELKHTLAKQKGGTD<br>HPPPHHHPPHPPHPPHHHPPPPHPPHPPHPHHPPPHPPHPPPHPPPPPPPP |
| 2DS5-A | GKLLYCSFCGKSQHEVRKLIAGPSVYICDECVDLCNDIIREEI<br>PPHHHHPHHPPPPPPHPPHHPPPPHHHHPPHHPHHPPHHPPPH |
| 2EQ7-C | LAMPAAERLMQEKGVSPAEVQGTGLGGRILKEDVMRH<br>HPHPPPPPHHPPPPHPPPPHPPPPHPPPHHPPPHHPP |
| 2HBA-A | MKVIFLKDVKGMGKKGEIKNVADGYANNFLFKQGLAIEATPANLKALEAQKQ<br>HPHHHHPPHPPHPPPPPHPPHPPPHPPPHHHPPPHPHPPPPPPHPPHPPPPP |

Table 6.1: Original and HP monomer sequences for the input proteins

The MJ energy function[16] was used in all simulations, as a contact-based energy function, described by equation 2.13. The matrix defining $e_{\Sigma}(r_{v_1}, r_{v_2})$ is given in appendix A.

The total energy for a protein structure in side chain model was calculated as previously shown in equation 2.15.

## 6.1 Gradient descent simulation

The gradient descent method is a general approach used to find a local minimum (or a local maximum) of a function. In a general sense, it begins by a random point in the domain space of the function, computes the gradient of the function at that point, and takes a step towards the direction of maximum decreasing rate of change. It continues iteratively until it reaches a point of zero gradient.
In this thesis, the gradient descent simulation method was used to reach protein structures that represent a local minimum with respect to their neighboring protein structures, according to some energy function. At every iteration, each neighboring structure of the current protein structure is computed, by applying a single valid move from the K-local move set, defined in section 4.2. Then, the neighboring structure of minimum energy is selected for the next iteration. In the case that different structures have the same energy, which is minimum, one random structure of minimum energy is selected.

**Problem** Given a protein structure modelled using the side-chain model in the FCC lattice, simulate the folding of a protein structure by applying an iterative gradient descent approach on this structure, till reaching a structure with local minimum energy, according to the MJ energy function. The neighboring structures of a structure are defined by the K-local move set defined in 4.2. Given a protein structure $S$ in side-chain model, $N_K(S)$ gives the set of protein structures that result from applying the K-local move set to $S$, using $K$ as the maximum length of a subinterval of residues to be altered from the structure $S$.

**Input protein structures** The input protein is one sequence of monomers from the presented sequences in table 6.1. For each experiment, the initial protein structure, $S_{in} = (p_{in}, s_{in}, R)$, is a structure of this known protein, that was either extracted by fitting the structure to the FCC lattice as described in [12], or one optimal structure, with respect to the HP energy function, according to [13]. Because of the binary nature of the HP energy function, it is usually the case that there is a class of protein structures that are optimal with respect to the HP energy function, as the minimal energy of a structure is achieved through a certain number of H-H contact-based interactions. Thus, in the simulations that use an HP optimal structure as an initial structure, a single random structure out of the optimal class is chosen per simulation. The simulations for each experiment are run multiple times (10 to 1000 times) in order to explore the different paths affected by the random choices resulting from energy equalities.

**Simulation parameters** Aside from the input protein structure as well as its monomer sequence, the other parameters of the simulation are $K$, the value used in applying the K-local move set throughout the simulation. Also, a random seed, $Seed$, is used as a parameter in the simulation, in order to resolve ties between different protein structures with equal minimum energy, using a standard pseudorandom number generator. This

pseudorandom number generator is also used in choosing the initial HP optimal structure to be used in the simulation out of the class of HP optimal structures.

**Algorithm** Gradient-Descent($S_{in}$, $K$, $Seed$)
  $Randomizer \Leftarrow$ Pseudorandom-Number-Generator($Seed$)
  $X \Leftarrow S_{in}$
  **while** $\exists_{n \in N_K(X)} : E(n) < E(X)$ **do**
    $\nabla \Leftarrow \{n : n \in N_K(X) \wedge \forall_{m \in N_K(X)} : E(n) \leq E(m)\}$
    $X \Leftarrow Randomizer$.Get-Random-Element($\nabla$)
  **end while**
  **return** $X$

## 6.2 Random descending walk simulation

The second type of simulation performed in this thesis is the random descending walk simulation. It begins by a protein structure, and at every iteration, it moves to a randomly chosen neighbor structure, such that it is strictly lower in energy. The neighboring structures are generated by applying a single move from the K-local move set. The simulation continues until it reaches a structure that represents a local minimum with respect to the energy function when compared to its neighboring structures.
The random descending walk approach, when applied repeatedly, often allows the simulation to reach protein structures of lower energy than the gradient descent simulation. The gradient descent simulation only reaches the local minima that are reachable by following the structures of least energy at every iteration. The random descending walk allows a wider range of paths, and it reaches to structures of lower energy that might not be reachable by the gradient descent simulation, because the formation of such structure might require passing by an intermediate structure, that is not necessarily the least energy as a neighbor to its predecessor structure.

**Problem** Given a protein structure modelled using the side-chain model in the FCC lattice, simulate the folding of a protein structure by applying an iterative random descending walk approach on this structure, till reaching a structure with local minimum energy, according to the MJ energy function. The neighboring structures of a structure are defined by the K-local move set defined in 4.2. Given a protein structure $S$ in side-chain model, $N_K(S)$ gives the set of protein structures that result from applying the K-local move set to $S$, using $K$ as the maximum length of a subinterval of residues to be altered from the structure $S$.

**Input protein structures** The random descending walk simulation was only applied on optimal protein structures, with respect to the HP energy function, generated according to [13]. The input protein for each experiment is one of the monomer sequences presented in table 6.1. For each experiment, multiple simulations were performed (10 to 1000 times) in order to explore the different paths taken by each random descending walk. As there are multiple protein structures that are optimal in energy according to

the HP model, a random structure out of them was chosen to be a starting structure for each simulation.

**Simulation parameters** The parameters in the simulation are $K$, the value used in applying the K-local move set throughout the simulation. Also, a random seed, $Seed$, is used as a parameter for the simulation, in order to select a random structure of lower energy out of the set of neighboring structures, using a standard pseudorandom number generator. This pseudorandom number generator is also used in choosing the initial HP optimal structure out of the class of HP optimal structures.

**Algorithm** Random-Descending-Walk($S_{in}$, $K$, $Seed$)
  $Randomizer \Leftarrow$ Pseudorandom-Number-Generator($Seed$)
  $X \Leftarrow S_{in}$
  **loop**
    $Next \Leftarrow N_K(X)$
    $Randomizer$.Random-Shuffle($Next$)
    $X' \Leftarrow$ First-Lower-Energy($Next, X$)
    **if** $X' = X$ **then**
      **return** $X$
    **end if**
  **end loop**
**proc** First-Lower-Energy($Next, X$)
  **for** $i = 1$ to $|Next|$ **do**
    **if** $E(Next_i) < E(X)$ **then**
      **return** $Next_i$
    **end if**
  **end for**
  **return** $X$

This implementation of the random descending walk simulation is more efficient than the gradient descent simulation. Every evaluation for the energy of a structure takes $O(|S|^2)$ time, and this implementation does not necessarily evaluate the energy of every neighboring structure, except at the very last iteration, when no neighbor of higher energy is found.

## 6.3 Symmetry elimination

The function $N_K(X)$ used in sections 6.1 and 6.2 removes duplicate protein structures up to rotation, translation and reflection. If applying the move set on a protein structure leads to two structures that are equivalent to each other via rotation, translation and reflection, and thus having the same energy, only one of them is yielded in the set in order not to influence the uniformly random choice among structures by having symmetric structures.
The set of neighboring protein structures are generated using the CSP definition mentioned earlier in section 4.2. Each resulting structure is translated then to a representa-

tion invariant up to rotation, translation and reflection, named as the normalized absolute move string.
For a backbone modelled protein structure, an *absolute move string* describes the protein structure in terms of the direction each monomer is located with respect to the previous monomer in the protein chain. For a protein chain of length $n$, an absolute move string of $n-1$ symbols is used to describe locations of the monomers, invariantly up to translation. The first monomer of the chain does not participate in the move string as it can be translated to any point in the lattice, and any following monomer is described using a single symbol representing its location relative to the last monomer in the chain. In side-chain modelled protein structures, $n$ other symbols are added to the string in order to indicate the positions of the side-chains relative to their backbone vertices. Thus, the absolute move string is invariant for any structure, up to translation.
The *normalized absolute move string* is, however, a unique representation for a protein structure up to rotation, translation and reflection. The normalization step applied to an absolute move string is forcing the first move symbols in the absolute move string to rotate and mirror the structure in a unique representation, by favoring the first transition to take a certain direction out of all degrees of freedom, thus rotating the whole structure after fixing one degree of freedom. The normalization continues until it exhausts all degrees of freedom, or finishes the absolute move string. Thus, introducing the order between the degrees of freedom gives a unique absolute move string, namely the normalized absolute move string, that is invariant up to rotation and reflection as well.

## 6.4 Results for lattice-fitted protein structures

The structures for the protein sequences given in table 6.1 were fitted into the FCC lattice, following [12]. These structures were used as initial structures in a gradient descent simulation, using the K-local move set , with $K = 1$, 2 and 3. The number of simulations per protein sequence is given in table 6.2.

| Simulation parameters | $K = 1$ | $K = 2$ | $K = 3$ |
|---|---|---|---|
| Number of simulations | 1000 | 300 | 20 |

Table 6.2: Number of gradient descent simulations per protein sequence

The results of these simulations show that the lattice-fitted structures of the real protein sequences are not optimal in energy according to the MJ energy function model. The potential energy for most structures were close to zero (sometimes positive). One reason might be the use of a contact energy function, which only takes into account neighboring side chain vertices for calculating the total energy of a structure.
For every input protein sequences, each of the gradient descent simulations reached some locally optimal structure. The structure of minimum potential energy out of them is inspected. Tables 6.3, 6.4 and 6.5 present the potential energy of the lowest energy structure, $E_{min}$, the number of iterations to reach that optimal structure, $\#iterations$, as well as the number of simulations producing that minimum energy structure, $\#sim$, for $K = 1$, 2 and 3, respectively. $E(S_{in})$ is the potential energy for the initial input structure. Table 6.6 shows the variation of the results with respect to $K$.

| PDB ID | $E(S_{in})$ | $E_{min}$ | $\#it$ | $\#sim$ |
|---|---|---|---|---|
| 1BAZ-A | $-3.73$ | $-23.08$ | 39 | 3/1000 |
| 1J8E-A | $-3.54$ | $-18.73$ | 33 | 68/1000 |
| 1RH6-A | $+1.33$ | $-24.69$ | 52 | 1/1000 |
| 1Z0J-B | $+2.05$ | $-26.10$ | 47 | 1/1000 |
| 2DS5-A | $-4.35$ | $-23.76$ | 30 | 242/1000 |
| 2EQ7-C | $-3.07$ | $-17.07$ | 26 | 84/1000 |
| 2HBA-A | $-3.04$ | $-21.46$ | 31 | 358/1000 |

Table 6.3: Simulation results for lattice-fitted structures, $K = 1$

| PDB ID | $E(S_{in})$ | $E_{min}$ | $\#it$ | $\#sim$ |
|---|---|---|---|---|
| 1BAZ-A | $-3.73$ | $-29.57$ | 31 | 4/300 |
| 1J8E-A | $-3.54$ | $-25.92$ | 39 | 1/300 |
| 1RH6-A | $+1.33$ | $-33.61$ | 47 | 1/300 |
| 1Z0J-B | $+2.05$ | $-33.97$ | 47 | 1/300 |
| 2DS5-A | $-4.35$ | $-29.97$ | 33 | 1/300 |
| 2EQ7-C | $-3.07$ | $-19.54$ | 21 | 38/300 |
| 2HBA-A | $-3.04$ | $-29.85$ | 38 | 1/300 |

Table 6.4: Simulation results for lattice-fitted structures, $K = 2$

| PDB ID | $E(S_{in})$ | $E_{min}$ | $\#it$ | $\#sim$ |
|---|---|---|---|---|
| 1BAZ-A | $-3.73$ | $-31.51$ | 28 | 1/20 |
| 1J8E-A | $-3.54$ | $-30.76$ | 37 | 2/20 |
| 1RH6-A | $+1.33$ | $-38.17$ | 34 | 1/20 |
| 1Z0J-B | $+2.05$ | $-35.95$ | 36 | 1/20 |
| 2DS5-A | $-4.35$ | $-34.36$ | 32 | 1/20 |
| 2EQ7-C | $-3.07$ | $-20.58$ | 14 | 2/20 |
| 2HBA-A | $-3.04$ | $-30.62$ | 34 | 1/20 |

Table 6.5: Simulation results for lattice-fitted structures, $K = 3$

| PDB ID | $E(S_{in})$ | $K = 1$ $E_{min}$ $(\#it)$ | $K = 2$ $E_{min}$ $(\#it)$ | $K = 3$ $E_{min}$ $(\#it)$ |
|---|---|---|---|---|
| 1BAZ-A | $-3.73$ | $-23.08$ (39) | $-29.57$ (31) | $-31.51$ (28) |
| 1J8E-A | $-3.54$ | $-18.73$ (33) | $-25.92$ (39) | $-30.76$ (37) |
| 1RH6-A | $+1.33$ | $-24.69$ (52) | $-33.61$ (47) | $-38.17$ (34) |
| 1Z0J-B | $+2.05$ | $-26.10$ (47) | $-33.97$ (47) | $-35.95$ (36) |
| 2DS5-A | $-4.35$ | $-23.76$ (30) | $-29.97$ (33) | $-34.36$ (32) |
| 2EQ7-C | $-3.07$ | $-17.07$ (26) | $-19.54$ (21) | $-20.58$ (14) |
| 2HBA-A | $-3.04$ | $-21.46$ (31) | $-29.85$ (38) | $-30.62$ (34) |

Table 6.6: Variation of $E_{min}$ and $\#it$ with respect to $K$

The results show that the lattice-fitted protein structures are far from optimal according to the MJ contact energy function model. The structure of minimum energy decreases in energy for increasing $K$. The reason is that larger $K$ allows the move set to contain all move sets of smaller $K$, and it allows more changes in the protein structure. Thus, the higher $K$ is, the simulation is able to reach structures of lower potential energy. There is no clear relation between the number of iterations needed to reach that lowest energy structure and $K$. The number of iterations needed to reach a structure is mostly dependent on number of consecutive structural changes needed to reach that specific structure. However, the trend between the number of iterations and $K$ appears when investigating the average number of steps of all simulations as shown in table 6.7. The higher values of $K$ allow the protein structural changes to reach a stable model faster, because of the larger freedom given for structural changes with respect to higher $K$. The average energy of the final structure also decreases with increasing $K$.

| PDB ID | $E(S_{in})$ | $K=1$ $\overline{E(X)}$ ($\overline{\#it}$) | $K=2$ $\overline{E(X)}$ ($\overline{\#it}$) | $K=3$ $\overline{E(X)}$ ($\overline{\#it}$) |
|---|---|---|---|---|
| 1BAZ-A | $-3.73$ | $-21.06$ (32.61) | $-27.42$ (29.05) | $-30.18$ (26.20) |
| 1J8E-A | $-3.54$ | $-17.48$ (27.87) | $-22.74$ (25.25) | $-27.28$ (28.80) |
| 1RH6-A | $+1.33$ | $-21.60$ (39.40) | $-29.27$ (33.44) | $-34.80$ (27.60) |
| 1Z0J-B | $+2.05$ | $-24.56$ (39.45) | $-29.38$ (34.66) | $-33.18$ (31.05) |
| 2DS5-A | $-4.35$ | $-23.30$ (28.30) | $-28.66$ (26.94) | $-31.00$ (23.75) |
| 2EQ7-C | $-3.07$ | $-16.50$ (25.74) | $-19.08$ (19.40) | $-20.34$ (13.40) |
| 2HBA-A | $-3.04$ | $-21.34$ (29.95) | $-25.68$ (29.56) | $-28.00$ (23.45) |

Table 6.7: Variation of $\overline{E(X)}$ and $\overline{\#it}$ with respect to $K$

## 6.5 Results for HP optimal protein structures

The structures for the protein sequences given in table 6.1 were used to generate classes of equivalent structures that are optimal according to the HP energy model, using [13]. For each simulation, one of the equivalent structures was chosen to perform that simulation. The input protein sequences were used to perform both gradient descent and random descending walk simulations, using the K-local move set , with $K = 1$, 2 and 3. The number of simulations per protein sequence for each algorithm is given in table 6.8.

| Simulation parameters | $K=1$ | $K=2$ | $K=3$ |
|---|---|---|---|
| Number of simulations | 1000 | 100 | 10 |

Table 6.8: Number of simulations per protein sequence

The results of these simulations show that the lattice-fitted structures of the real protein sequences are not optimal in energy according to the MJ energy function model. The potential energy for most structures were close to zero (sometimes positive). One reason

might be the use of a contact energy function, which only takes into account neighboring side chain vertices for calculating the total energy of a structure.
For every input protein sequences, each of the gradient descent simulations reached some locally optimal structure. The structure of minimum potential energy out of them is inspected. The same was performed for random descending walk simulations.

| | | Gradient Descent | | Random Descending Walk | |
|---|---|---|---|---|---|
| PDB ID | $\overline{E(S_{in})}$ | $E_{min}$ $[E(S_{in})]$ | $\#it$ | $E_{min}$ $[E(S_{in})]$ | $\#it$ |
| 1BAZ-A | $-10.02$ | $-28.19$ $[-7.32]$ | 29 | $-29.10$ $[-11.93]$ | 57 |
| 1J8E-A | $-12.31$ | $-28.32$ $[-14.68]$ | 36 | $-28.38$ $[-9.45]$ | 59 |
| 1RH6-A | $-13.75$ | $-29.92$ $[-14.54]$ | 40 | $-32.09$ $[-13.59]$ | 67 |
| 1Z0J-B | $-13.45$ | $-32.18$ $[-14.58]$ | 35 | $-32.65$ $[-10.73]$ | 66 |
| 2DS5-A | $-7.29$ | $-26.34$ $[-10.56]$ | 33 | $-25.98$ $[-6.73]$ | 52 |
| 2EQ7-C | $-5.91$ | $-20.28$ $[-7.09]$ | 26 | $-19.51$ $[-5.67]$ | 51 |
| 2HBA-A | $-11.11$ | $-27.54$ $[-12.04]$ | 33 | $-27.92$ $[-14.90]$ | 55 |

Table 6.9: Simulation results for HP optimal structures, $K = 1$

| | | Gradient Descent | | Random Descending Walk | |
|---|---|---|---|---|---|
| PDB ID | $\overline{E(S_{in})}$ | $E_{min}$ $[E(S_{in})]$ | $\#it$ | $E_{min}$ $[E(S_{in})]$ | $\#it$ |
| 1BAZ-A | $-10.02$ | $-32.08$ $[-10.27]$ | 33 | $-32.62$ $[-11.88]$ | 79 |
| 1J8E-A | $-12.31$ | $-30.43$ $[-12.73]$ | 25 | $-30.59$ $[-10.47]$ | 92 |
| 1RH6-A | $-13.75$ | $-28.82$ $[-14.46]$ | 22 | $-35.64$ $[-14.23]$ | 94 |
| 1Z0J-B | $-13.45$ | $-34.81$ $[-14.72]$ | 28 | $-34.32$ $[-14.35]$ | 65 |
| 2DS5-A | $-7.29$ | $-29.89$ $[-5.23]$ | 32 | $-30.89$ $[-12.54]$ | 70 |
| 2EQ7-C | $-5.91$ | $-21.61$ $[-5.98]$ | 23 | $-21.99$ $[-3.30]$ | 93 |
| 2HBA-A | $-11.11$ | $-30.46$ $[-15.76]$ | 29 | $-31.82$ $[-12.57]$ | 89 |

Table 6.10: Simulation results for HP optimal structures, $K = 2$

| | | Gradient Descent | | Random Descending Walk | |
|---|---|---|---|---|---|
| PDB ID | $\overline{E(S_{in})}$ | $E_{min}$ $[E(S_{in})]$ | $\#it$ | $E_{min}$ $[E(S_{in})]$ | $\#it$ |
| 1BAZ-A | $-10.02$ | $-33.07$ $[-10.64]$ | 38 | $-34.60$ $[-9.37]$ | 124 |
| 1J8E-A | $-12.31$ | $-29.33$ $[-14.70]$ | 15 | $-32.35$ $[-11.62]$ | 113 |
| 1RH6-A | $-13.75$ | $-35.12$ $[-11.51]$ | 24 | $-37.59$ $[-14.23]$ | 103 |
| 1Z0J-B | $-13.45$ | $-34.71$ $[-11.95]$ | 23 | $-37.69$ $[-11.8]$ | 121 |
| 2DS5-A | $-7.29$ | $-31.00$ $[-7.49]$ | 24 | $-32.53$ $[-5.41]$ | 116 |
| 2EQ7-C | $-5.91$ | $-21.64$ $[-2.89]$ | 23 | $-25.10$ $[-5.71]$ | 96 |
| 2HBA-A | $-11.11$ | $-30.91$ $[-10.66]$ | 28 | $-35.56$ $[-12.06]$ | 137 |

Table 6.11: Simulation results for HP optimal structures, $K = 3$

Tables 6.9, 6.10 and 6.11 present the potential energy of the lowest energy structure, $E_{min}$, the energy, according to MJ energy function, of the HP optimal structure, that resulted in

this structure of lowest energy, $E(S_{in})$ and the number of iterations to reach that optimal structure, for both the gradient descent simulations as well as random descending walk simulation.
There were no simulations that tied for the lowest energy structure except for two pairs of simulations. Thus, the number of simulations producing that minimum energy structure was discarded. The tables present the results for $K = 1$, 2 and 3, respectively. $\overline{E(S_{in})}$ is the average potential energy for the initial input structures. Table 6.12 shows the variation of the $E_{min}$ with respect to $K$.

| | | Gradient Descent | | | Random Descending Walk | | |
|---|---|---|---|---|---|---|---|
| | | $K = 1$ | $K = 2$ | $K = 3$ | $K = 1$ | $K = 2$ | $K = 3$ |
| PDB ID | $\overline{E(S_{in})}$ | $E_{min}$ | $E_{min}$ | $E_{min}$ | $E_{min}$ | $E_{min}$ | $E_{min}$ |
| 1BAZ-A | −10.02 | −28.19 | −32.08 | −33.07 | −29.10 | −32.62 | −34.60 |
| 1J8E-A | −12.31 | −28.32 | −30.43 | −29.33 | −28.38 | −30.59 | −32.35 |
| 1RH6-A | −13.75 | −29.92 | −28.82 | −35.12 | −32.09 | −35.64 | −37.59 |
| 1Z0J-B | −13.45 | −32.18 | −34.81 | −34.71 | −32.65 | −34.32 | −37.69 |
| 2DS5-A | −7.29 | −26.34 | −29.89 | −31.00 | −25.98 | −30.89 | −32.53 |
| 2EQ7-C | −5.91 | −20.28 | −21.61 | −21.64 | −19.51 | −21.99 | −25.10 |
| 2HBA-A | −11.11 | −27.54 | −30.46 | −30.91 | −27.92 | −31.82 | −35.56 |

Table 6.12: Variation of $E_{min}$ with respect to $K$

The results show that the HP optimal protein structures are more energy optimal when compared to real lattice-fitted protein structures, according to the MJ contact energy function model. One reason is that the computation of HP optimal models depend on the HP energy function as a contact-based energy function, which leads to a higher number of touching side-chains. The structure of minimum energy generally decreases in energy for increasing $K$, for both gradient descent and random walk simulations, because larger $K$ allows the move set to contain all move sets of smaller $K$. For the same value of $K$, the simulation reaches lower energy structures using the random descending walk approach than the gradient walk approach. The reason is that a random descending walk allows the simulation to choose higher energy structures, which are not compactly folded, as intermediate structures. Thus, adding the possibility of reaching other local minima that are not reachable by any possible gradient descent. The same correlation appears between $\overline{E(X)}$ and $K$ as show in table 6.13.
The correlation between the average number of iterations and $K$ appears to be the same for gradient descent simulations, as appears in table 6.14, because higher values of $K$ allow the protein structural changes to reach a stable model faster.
However, for random descending walks, the number of iterations are much bigger. Moreover, for larger values of $K$, the average number of iterations is larger and is significantly larger than the number of iterations in gradient descent simulations. The reason is that random descending simulations do not follow a steep decrease in potential energy, which is the case in gradient descent. This allows the random descending walk simulation to reach a larger number of local minima. However, the progress of the simulation, although always guaranteed to halt, it follows a gradual descent to the final structure.

Random descending walks are provably finite. The reason is that potential energy of a protein structure is finite, and that the simulation follows structures iteratively, in strictly descending order. It is worth mentioning that the random descending walk algorithm is significantly faster than the gradient descent algorithm, because the random descending walk algorithm does not evaluate the energy of every neighboring structure in order to choose the structure of the following iteration, unlike the gradient descent algorithm, which evaluates the potential energy for every neighboring structure, consuming $O(|S|^2)$ time for each energy computation.

| | | Gradient Descent | | | Random Descending Walk | | |
|---|---|---|---|---|---|---|---|
| | | $K=1$ | $K=2$ | $K=3$ | $K=1$ | $K=2$ | $K=3$ |
| PDB ID | $\overline{E(S_{in})}$ | $\overline{E(X)}$ | $\overline{E(X)}$ | $\overline{E(X)}$ | $\overline{E(X)}$ | $\overline{E(X)}$ | $\overline{E(X)}$ |
| 1BAZ-A | −10.02 | −23.50 | −27.07 | −30.03 | −23.41 | −27.58 | −32.22 |
| 1J8E-A | −12.31 | −23.73 | −26.73 | −28.40 | −23.29 | −27.40 | −30.35 |
| 1RH6-A | −13.75 | −26.34 | −30.54 | −33.35 | −27.20 | −31.52 | −34.86 |
| 1Z0J-B | −13.45 | −28.43 | −30.78 | −33.48 | −28.21 | −31.55 | −34.03 |
| 2DS5-A | −7.29 | −20.70 | −25.22 | −27.39 | −20.72 | −25.79 | −29.41 |
| 2EQ7-C | −5.91 | −16.13 | −18.54 | −20.71 | −16.21 | −19.12 | −20.94 |
| 2HBA-A | −11.11 | −27.54 | −26.58 | −28.72 | −24.33 | −27.74 | −31.28 |

Table 6.13: Variation of $\overline{E(X)}$ with respect to $K$

| | Gradient Descent | | | Random Descending Walk | | |
|---|---|---|---|---|---|---|
| | $K=1$ | $K=2$ | $K=3$ | $K=1$ | $K=2$ | $K=3$ |
| PDB ID | $\overline{\#it}$ | $\overline{\#it}$ | $\overline{\#it}$ | $\overline{\#it}$ | $\overline{\#it}$ | $\overline{\#it}$ |
| 1BAZ-A | 24.26 | 22.50 | 23.80 | 43.74 | 69.52 | 100.20 |
| 1J8E-A | 25.11 | 22.99 | 20.40 | 45.32 | 69.37 | 91.70 |
| 1RH6-A | 24.50 | 24.42 | 24.30 | 45.72 | 69.25 | 86.60 |
| 1Z0J-B | 26.66 | 22.74 | 21.30 | 47.19 | 68.30 | 85.00 |
| 2DS5-A | 27.14 | 23.18 | 21.40 | 45.96 | 70.61 | 101.60 |
| 2EQ7-C | 18.04 | 16.45 | 17.10 | 33.67 | 53.60 | 67.30 |
| 2HBA-A | 24.86 | 20.83 | 19.30 | 46.28 | 72.62 | 98.40 |

Table 6.14: Variation of $\overline{\#it}$ with respect to $K$

The results show that the type of the initial structure (HP optimal or lattice-fitted) does not affect the final energy of the structure. For some of the input sequences, using HP optimal structures as initial structures yielded lowest energy structures. In other input sequences, starting with lattice-fitted structures as initial structures yielded the lowest energy structures. In some cases there were no significant difference between them.
Although random descending walk simulations do not limit the choice of next protein structure, but only requiring any structure of strictly lower energy, random descending walk simulations exhibit a very stable behavior in reaching the lowest energy structures as indicated by the relation between $\overline{E(X)}$ or $E_{min}$ and $K$, as shown in tables 6.12 and

6.13. Especially for larger values of $K$, a gradient descent simulation might allow an initial structure to reach a local minimum structure, for which there is no way out to reach other local minimal structures, using few steps of larger structural changes leading towards the lowest energy structure for every iteration. However, for random descending walks, given the same number of simulations, and the same value of $K$, all possible random descending walks are allowed and are eventually traced, because the random descending walk approach treats all neighbor structures of lower energy indistinguishably.

# Chapter 7

# Conclusion

The K-local move set provides a generic approach for defining a local move set. Using a generic neighborhood function allows the CSP definition to apply on any lattice. The CSP definition itself is extensible to any applicable protein structure model, as shown for backbone and side-chain models. The CSP definition provided is a general definition covering the local move sets defined in [17, 6, 4]. A more efficient local move set was defined and can be used for folding simulations when augmented with an ergodic move set.
The combination between CSP approach for defining the move set and the local optima search techniques was applied in the thesis, applying the framework defined in [15]. The results extracted lead to the conclusions that lattice-fitted protein structures are far from optimal according to contact-based energy functions and that HP optimal structures are good candidates for starting optima search for structures according to a more precise energy function, such as the MJ energy function. The relations between the energies of the optimal structures found, the number of simulation iterations and $K$ was shown and explained.
The results for the simulations suggest the use of distance-based energy function for lattice-fitted protein structures. Also, the absence of tied simulations for HP optimal structures suggests performing higher number of simulations, especially for larger values of $K$. It appeared that the use of gradient descent simulation might cause some of the optimal structures to be unreachable for more permissible move sets as shown by the trend for simulation results of some HP optimal structures, even though the average result of the simulations asserted that permissible move sets enable reaching structures of lower energy. Thus, the use of an ergodic move set, combined with a realistic search method is suggested for the search for globally optimal protein structures.

# Appendix A

# Energy function matrix of MJ model

| | C | M | F | I | L | V | W | Y | A | G | T | S | Q | N | E | D | H | R | K | P |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| C | -1.06 | 0.19 | -0.23 | 0.16 | -0.08 | 0.06 | 0.08 | 0.04 | 0.00 | -0.08 | 0.19 | -0.02 | 0.05 | 0.13 | 0.69 | 0.03 | -0.19 | 0.24 | 0.71 | 0.00 |
| M | 0.19 | 0.04 | -0.42 | -0.28 | -0.20 | -0.14 | -0.67 | -0.13 | 0.25 | 0.19 | 0.19 | 0.14 | 0.46 | 0.08 | 0.44 | 0.65 | 0.99 | 0.31 | 0.00 | -0.34 |
| F | -0.23 | -0.42 | -0.44 | -0.19 | -0.30 | -0.22 | -0.16 | 0.00 | 0.03 | 0.38 | 0.31 | 0.29 | 0.49 | 0.18 | 0.27 | 0.39 | -0.16 | 0.41 | 0.44 | 0.20 |
| I | 0.16 | -0.28 | -0.19 | -0.22 | -0.41 | -0.25 | 0.02 | 0.11 | -0.22 | 0.25 | 0.14 | 0.21 | 0.36 | 0.53 | 0.35 | 0.59 | 0.49 | 0.42 | 0.36 | 0.25 |
| L | -0.08 | -0.20 | -0.30 | -0.41 | -0.27 | -0.29 | -0.09 | 0.24 | -0.01 | 0.23 | 0.20 | 0.25 | 0.26 | 0.30 | 0.43 | 0.67 | 0.16 | 0.35 | 0.19 | 0.42 |
| V | 0.06 | -0.14 | -0.22 | -0.25 | -0.29 | -0.29 | -0.07 | 0.02 | -0.10 | 0.16 | 0.25 | 0.18 | 0.24 | 0.50 | 0.34 | 0.58 | 0.19 | 0.30 | 0.44 | 0.09 |
| W | 0.08 | -0.67 | -0.16 | 0.02 | -0.09 | -0.07 | -0.12 | -0.04 | -0.09 | 0.18 | 0.22 | 0.34 | 0.08 | 0.06 | 0.29 | 0.24 | -0.12 | -0.16 | 0.22 | -0.28 |
| Y | 0.04 | -0.13 | 0.00 | 0.11 | 0.24 | 0.02 | -0.04 | -0.06 | 0.09 | 0.14 | 0.13 | 0.09 | -0.20 | -0.20 | -0.10 | 0.00 | -0.34 | -0.25 | -0.21 | -0.33 |
| A | 0.00 | 0.25 | 0.03 | -0.22 | -0.01 | -0.10 | -0.09 | 0.09 | -0.13 | -0.07 | -0.09 | -0.06 | 0.08 | 0.28 | 0.26 | 0.12 | 0.34 | 0.43 | 0.14 | 0.10 |
| G | -0.08 | 0.19 | 0.38 | 0.25 | 0.23 | 0.16 | 0.18 | 0.14 | -0.07 | -0.38 | -0.26 | -0.16 | -0.06 | -0.14 | 0.25 | -0.22 | 0.20 | -0.04 | 0.11 | -0.11 |
| T | 0.19 | 0.19 | 0.31 | 0.14 | 0.20 | 0.25 | 0.22 | 0.13 | -0.09 | -0.26 | 0.03 | -0.08 | -0.14 | -0.11 | 0.00 | -0.29 | -0.19 | -0.35 | -0.09 | -0.07 |
| S | -0.02 | 0.14 | 0.29 | 0.21 | 0.25 | 0.18 | 0.34 | 0.09 | -0.06 | -0.16 | -0.08 | -0.20 | -0.14 | -0.14 | -0.26 | -0.31 | -0.05 | 0.17 | -0.13 | 0.01 |
| Q | 0.05 | 0.46 | 0.49 | 0.36 | 0.26 | 0.24 | 0.08 | -0.20 | 0.08 | -0.06 | -0.14 | -0.14 | 0.29 | -0.25 | -0.17 | -0.17 | -0.02 | -0.52 | -0.38 | -0.42 |
| N | 0.13 | 0.08 | 0.18 | 0.53 | 0.30 | 0.50 | 0.06 | -0.20 | 0.28 | -0.14 | -0.11 | -0.14 | -0.25 | -0.53 | -0.32 | -0.30 | -0.24 | -0.14 | -0.33 | -0.18 |
| E | 0.69 | 0.44 | 0.27 | 0.35 | 0.43 | 0.34 | 0.29 | -0.10 | 0.26 | 0.25 | 0.00 | -0.26 | -0.17 | -0.32 | -0.03 | -0.15 | -0.45 | -0.74 | -0.97 | -0.10 |
| D | 0.03 | 0.65 | 0.39 | 0.59 | 0.67 | 0.58 | 0.24 | 0.00 | 0.12 | -0.22 | -0.29 | -0.31 | -0.17 | -0.30 | -0.15 | 0.04 | -0.39 | -0.72 | -0.76 | 0.04 |
| H | -0.19 | 0.99 | -0.16 | 0.49 | 0.16 | 0.19 | -0.12 | -0.34 | 0.34 | 0.20 | -0.19 | -0.05 | -0.02 | -0.24 | -0.45 | -0.39 | -0.29 | -0.12 | 0.22 | -0.21 |
| R | 0.24 | 0.31 | 0.41 | 0.42 | 0.35 | 0.30 | -0.16 | -0.25 | 0.43 | -0.04 | -0.35 | 0.17 | -0.52 | -0.14 | -0.74 | -0.72 | -0.12 | 0.11 | 0.75 | -0.38 |
| K | 0.71 | 0.00 | 0.44 | 0.36 | 0.19 | 0.44 | 0.22 | -0.21 | 0.14 | 0.11 | -0.09 | -0.13 | -0.38 | -0.33 | -0.97 | -0.76 | 0.22 | 0.75 | 0.25 | 0.11 |
| P | 0.00 | -0.34 | 0.20 | 0.25 | 0.42 | 0.09 | -0.28 | -0.33 | 0.10 | -0.11 | -0.07 | 0.01 | -0.42 | -0.18 | -0.10 | 0.04 | -0.21 | -0.38 | 0.11 | 0.26 |

Table A.1: Energy function matrix of MJ model [16]

# Bibliography


[1] Srinivas Aluru, editor. *Handbook of Computational Molecular Biology*, chapter Protein Structure Prediction with Lattice Models. Chapman & Hall CRC Computer and Information Science Series, 2006.

[2] Rolf Backofen. Using constraint programming for lattice protein folding. In Russ B. Altman, A. Keith Dunker, Lawrence Hunter, and Teri E. Klein, editors, *Proceedings of the Pacific Symposium on Biocomputing (PSB'98)*, volume 3, pages 387–398, 1998.

[3] Rolf Backofen, Sebastian Will, and Erich Bornberg-Bauer. Application of constraint programming techniques for structure prediction of lattice proteins with extended alphabets. *Bioinformatics*, 15(3):234–242, 1999.

[4] Hans-Joachim Böckenhauer, Abu Zafer Dayem Ullah, Leonidas Kapsokalivas, and Kathleen Steinhöfel. A local move set for protein folding in triangular lattice models. In *WABI '08: Proceedings of the 8th international workshop on Algorithms in Bioinformatics*, pages 369–381, Berlin, Heidelberg, 2008. Springer-Verlag.

[5] R.I. Dima and D. Thirumalai. Exploring protein aggregation and self-propagation using lattice models: Phase diagram and kinetics. *Protein Science*, 11(5):1036–1049, 2002.

[6] Jun Shimada Edo Kussell and Eugene I. Shakhnovich. Sidechain dynamics and protein folding. In *Bridging Time Scales: Molecular Simulations for the Next Decade, Lecture Notes in Physics*, volume 605, pages 303–321, 2003.

[7] William E. Hart and Sorin Istrail. Lattice and off-lattice side chain models of protein folding: Linear time structure prediction better than 86% of optimal. *J. Comput. Biol*, 4:241–259, 1997.

[8] E. S. Huang, P. Koehl, M. Levitt, R. V. Pappu, and J. W. Ponder. Accuracy of sidechain prediction upon near-native protein backbones generated by ab initio folding methods. *Proteins*, 33(2):204–217, November 1998.

[9] N. Lesh, M. Mitzenmacher, and S. Whitesides. A complete and effective move set for simplified protein folding. In *7th Annual International Conference on Research in Computational Molecular Biology (RECOMB) 2003*, pages 188–195. ACM Press, 2003.

[10] Neal Madras and Alan D. Sokal. The pivot algorithm: A highly efficient monte carlo method for the self-avoiding walk. *Journal of Statistical Physics*, 50(1):109–186, January 1988.

[11] D. K. Klimov Mai Suan Li and D. Thirumalai. Folding in lattice models with side chains. *Computer physics communications*, 147(1-2 (761 p.)):625–628, 2002.

[12] Martin Mann, Daniel Maticzka, Rhodri Saunders, and Rolf Backofen. Classifying protein-like sequences in arbitrary lattice protein models using LatPack. *HFSP Journal*, 2(6):396, 2008. Special issue on protein folding: experimental and theoretical approaches.

[13] Rolf Backofen Martin Mann and Sebastian Will. Equivalence classes of optimal structures in hp protein models including side chains. In *Proceedings of Workshop on Constraint Based Methods for Bioinformatics - WCB'09*, 2009. submitted.

[14] Britt H. Park and Michael Levitt. The complexity and accuracy of discrete state models of protein structure. *Journal of Molecular Biology*, 249(2):493 – 507, 1995.

[15] Gilles Pesant, Michel Gendreau, and Wim Nuijten. A constraint programming framework for local search methods. *Journal of Heuristics*, 5:255–279, 1999.

[16] Miyazawa S. and Jernigan RL. Estimation of effective inter-residue contact energies from protein crystal structures: Quasi-chemical approximation. *Macromolecules*, 18:534–552, 1985.

[17] Eugene Sali, Andrej; Shakhnovich and Martin Karplus. Kinetics of protein folding. a lattice model study of the requirements for folding to the native state. *Journal of Molecular Biology*, 235(5):1614–1636, 1994.

[18] Guido Tack. Gecode: an open constraint solving library. Presentation at Workshop on Open-Source Software for Integer and Constraint Programming (OSSICP), Paris, France, May 2008.

[19] R. Tanimura, A. Kidera, and H. Nakamura. Determinants of protein side-chain packing. *Protein science : a publication of the Protein Society*, 3(12):2358–2365, December 1994.

[20] Jose L. Parra Yanxin Liu, Prem P. Chapagain and Bernard S. Gerstman. Lattice model simulation of interchain protein interactions and the folding dynamics and dimerization of the gcn4 leucine zipper. *Journal of Chemical Physics*, 128:045106–045106–10, 2008.

[21] Y ZHANG. Progress and challenges in protein structure prediction. *Current Opinion in Structural Biology*, 18(3):342–348, June 2008.