EFFICIENT SAMPLING OF PROTEIN FOLDING FUNNELS USING HMMSTR AND PATHWAY GENERATION USING PROBABILISTIC ROADMAPS

By

Yogesh A. Girdhar

A Thesis Submitted to the Graduate

Faculty of Rensselaer Polytechnic Institute

in Partial Fulfillment of the

Requirements for the Degree of

MASTER OF COMPUTER SCIENCE

Approved:

____
Srinivas Akella
Thesis Adviser

Rensselaer Polytechnic Institute
Troy, New York

April 2005
(For Graduation May 2005)

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGMENT

First of all I must thank my advisor Prof. Srinivas Akella for his never ending feedback and motivation, which finally led to the completion of this thesis.

Special thanks to Prof. Chris Bystroff for sharing his bottomless bag of ideas with me. Without his support, I would still be stuck on some trivial part of my thesis.

Thanks to Edward Carlson for spending time with me to get me started on this problem.

Thanks to my parents and my sisters Karuna and Ritu for always standing by my side.

Nilanjan and Jufeng for helping me with my everyday math problems.

Thanks to my other lab mates for tolerating me day and night.

ABSTRACT

Classical techniques for simulating molecular motion such as Molecular Dynamics and Monte Carlo simulations only generate one pathway at a time and have extremely high computational cost. We present a biologically significant and efficient way to predict protein folding pathways using HMMSTR and Probabilistic Roadmaps (PRM), with several order of magnitudes lower computational cost. We show how to perform unbiased sampling of the folding funnel to generate a PRM graph for protein chains of up to 36 residues. This biologically based sampling is achieved by enforcing protein-like local structures using HMMSTR, a hidden Markov model for local sequence-structure prediction, thereby significantly reducing the size of the conformational space. We also show that there exist favored folding pathways (*highways*) that proteins take to reach their native fold or other compact folded states. We evalute our approach with three different proteins: a 36 residue long subdomain from Chicken Villin Headpiece – 1VII(36), a 16-residue long β-hairpin from Protein G – 2GB1(16), and a 28 residue long Fbp28Ww Domain from Mus Musculus – 1E0L(28).

CHAPTER 1
INTRODUCTION

There are two fundamental problems in computational biology when dealing with protein structures:

Protein Structure Prediction: Given the amino acid sequence code of a protein, we would like to predict the final 3D structure of the protein, i.e., the position of each atom in the protein. Anfinsen, in his Nobel Prize winning work [2], showed that a denatured protein can refold spontaneously to its native conformation without any external help. From this observation, it was concluded that the amino acid sequence code of a given protein has enough information to completely predict its 3D shape.

Protein Folding Pathway Prediction: Given the 3D structure of a protein, we would like to predict the folding pathway, i.e., we would like to predict intermediate configurations which the protein takes before it reaches the final configuration. From this we can also infer the order in which the secondary structures of the protein are formed.

Both these problems are equally important, however the latter is much less explored and is the focus of this thesis. The pathway prediction problem is important because it has applications in understanding diseases such as prion diseases, Alzheimer's disease and cystic fibrosis, which are a result of protein misfolding. Also in future this could be used for designing artificial proteins.

1.1 Problem Statement

Protein folding can be visualized to occur on an energy landscape with shape similar to a rugged funnel. This funnel like energy landscape is sometimes referred to as the protein folding funnel. Given the amino-acid sequence and 3D structure of a protein, we wish to generate protein configurations that are characteristic of this

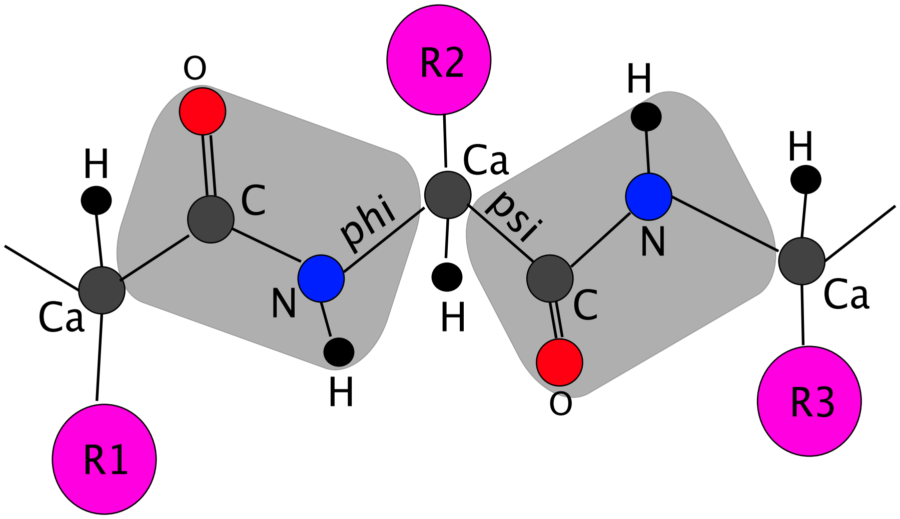


Figure 1.1: Parts of a polypeptide chain. Each peptide unit has two degrees of freedom, corresponding to the psi (ψ) and phi (ϕ) torsion angles. Ca represents the C_α atom, and R_1, R_2, R_3 represent the sidechains.

folding funnel, and then use these configurations to build a protein folding roadmap graph which can be used to generate folding pathways.

1.2 Model of a Protein

Proteins are built up by amino acids that are linked by peptide bonds to form a polypeptide chain [4]. There are only 20 amino acids. What distinguishes one amino acid from other is the side chain attached to its C_α atoms. The peptide units are effectively rigid groups because the electron pair of the $C = O$ bond is delocalized over the peptide group, such that rotation around $C - N$ bond is prevented by an energy barrier. Hence the only degrees of freedom they have are the rotations around two bonds: the $C_\alpha - C$ (the ϕ torsion angle) and the $N - C_\alpha$ (the ψ torsion angle), as shown in Figure 1.1. We assume that conformational changes are solely due to changes in these torsion angles.

We use the atomgroup localframes method [21] to efficiently derive 3D molecular conformations from the values of ϕ and ψ torsion angles. In this method a single local frame is attached to each rotatable bond, and the position of each atom is updated by a single matrix multiplication. This method provides lazy evaluations for atom positions, which means we can accumulate rotation operations and then in one pass calculate the 3D position of all atoms. This greatly reduces the computational cost, when many torsion angles of a conformations are being changed.

1.3 Research Contributions of this Thesis

1.3.1 Sample Generation

- We present a new way to sample the configuration space [14] of a protein using statistical data about known protein like structures.

- We show that this sampling approach reduces the size of conformational space drastically, since we do not try to sample biologically impossible configurations.

- We show that the size of conformational space depends on secondary structure. The C-space size of 36 residue long 1VII is much smaller than 28 residue long 1E0L even when it has more residues. This is because 1VII is mostly made up of helices.

- We present a hydrogen bond energy reducing method to compact the HMM-STR samples and improve the density of near-native samples.

- We show that the resultant graph has smooth transitions from one conformation to the other, and hence this graph can be used with other energy functions.

1.3.2 Pathway Generation

- We use a simple energy function based on van der Waals energy, and approximation of hydrogen-bonds and radius of gyration to generate pathways to the native fold from all the other configurations.

- We show that the resulting pathways have smooth DME (Distance Matrix Error) and energy transitions.

- We generate a connectivity graph of these pathways, and show that there exist *highways* that correspond to popular folding channels.

1.4 Previous Work

Classical techniques for simulating molecular motion such as Molecular Dynamics (MD), and Monte Carlo simulations only generate one pathway at a time and have extremely high computational cost. For a comparison, molecular dynamics simulation of 36 residue long α-helical protein from the villin headpiece (PDB id 1VII), took about 1000 single CPU (500Hz) years of computation [19]. On the other end of the speed spectrum, if we were to just predict the order of formation of secondary structure elements (SSE), then [20] presents an algorithm which runs extremely fast (order of a few seconds). It works by building a graph representation of a protein, where a vertex denotes a SSE and an edge denotes the interactions between two SSEs. The edges are weighted by the strength of the SSE interactions. The basic idea then is to break the weakest interactions to obtain a sequence of unfolding events.

When Probabilistic Roadmaps (PRM) [9] are used to predict protein folding pathways [1] [18], they are capable of giving us actual structural information about intermediate configurations on the folding pathway, while still running in a reasonable amount of time (order of a few hours). Stochastic Roadmaps (a variant of PRM) use transition probabilities, instead of energy difference while building the roadmap graph. These are known to converge to the same results as a Monte Carlo simulation, while being several order of magnitudes faster [3]. [1] [18] use Gaussian sampling about the native state so that density of samples is greater near the known native state. The problem with building the roadmap graph using such native state based sampling is that the generated samples do not have much biological significance, apart from being close to the native fold. This approach, although better than randomly sampling the configuration space, is still inefficient due to the size of the configuration space [7]. What is needed is some way to only sample those parts of the configuration space that are biologically feasible. We do this by using HMMSTR[6] to sample the configuration space.

1.5 Probabilistic Roadmaps (PRM)

Probabilistic Roadmaps were originally developed [9] to plan motion for robots with many *degrees of freedom* (dof). The method has two phases: a learning phase and a query phase. In the learning phase, the collision free configuration space of the robot is sampled randomly, and then each of these configurations is connected via a fast *local planner* to a few of its nearest configurations. The roadmap formed is stored as an undirected graph. Once the roadmap graph is built, now path planning queries can be answered, by searching the graph, to find a path through the sampled configurations. If the start and the goal configurations are not present in the graph, then they can be added and then connected to the nearest corresponding configuration. Figure 1.2 shows a simple illustration of Probabilistic Roadmaps. The blue nodes are the sampled collision free configurations. Orange objects represent the obstacles. An edge between two configurations represents a possible local path between the configurations. The green node is the starting configuration, and red node is the goal configuration. The grey line is the path between them.

1.6 HMMSTR

HMMSTR[6] is a hidden Markov model [17] for local sequence-structure. It uses knowledge of preferred orientations of amino acid sequences from data in the Protein Data Bank (PDB) to predict the ϕ and ψ torsion angles of local sequences. Given the amino acid sequence code of a protein and a window size, HMMSTR generates a set of likely ϕ and ψ angles for each overlapping window for the entire sequence. Over 150 motifs are modeled as sequence-structure correlations in HMMSTR. HMMSTR uses PSI-BLAST to generate a list of similar protein sequences.

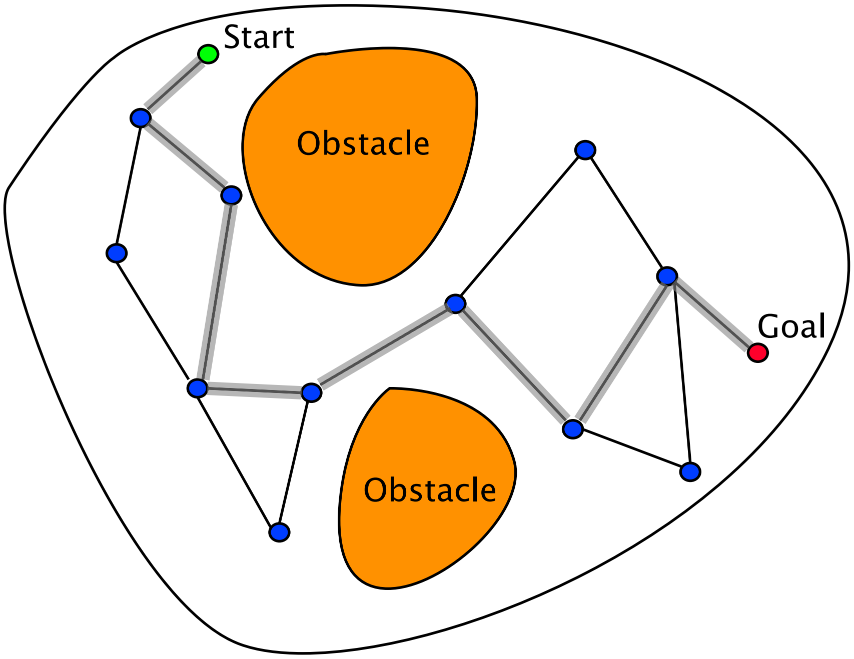


Figure 1.2: Probabilistic Roadmaps. The blue nodes are the sampled collision free configurations. Orange objects represent the obstacles. An edge between two configurations represents a possible path between the configurations. The green node is the starting configuration, and red node is the goal configuration. The grey line is the path between them.

CHAPTER 2
SAMPLING THE FOLDING FUNNEL USING HMMSTR

Our goal is goal is to sample the biologically valid regions of the conformation space. We use HMMSTR to achieve this goal.

2.1 Generating Samples

We use HMMSTR to generate 1000 high scoring local structures (ϕ and ψ angles) for each 5 residue long overlapping window in the amino acid sequence. Here is an example of what we mean by overlapping windows.

$$\underbrace{GATAV}SEWTEYKT... \text{ - Window 1}$$
$$G\underbrace{ATAVS}EWTEYKT... \text{ - Window 2}$$
$$GATAV\underbrace{SEWTE}YKT... \text{ - Window 6}$$

It has been shown that protein fragments of length 5 are sufficient to reconstruct any protein with a reasonable library [12]. Once we have these local structures, we the proceed to building a complete configuration out of these local structures. We start off with a random anchor window in the protein, and then choose a random angle set (ϕ, ψ angles for each residue in the window) with probability proportional to its score. We then walk towards both the ends of the protein from the anchor window, assigning angle set for all remaining windows. At each step of the walk, we shift by two residues and then randomly select an angle set; again with a probability proportional to the score. We only allow angle sets with maximum deviating angle (MDA) less than 100^o to be selected. If there are no such angle sets, then the sample is discarded.

After instantiating torsion angles for each of the residues, we then compute the 3D coordinates of the atoms. Now this sample is checked for van der Waals (VDW) collisions, and if there are no collisions, we accept this sample.

2.2 Van der Waals Collision Checking (VDW)

Van der Waals collision checking is performing by doing pairwise distance calculation of all the atoms and making sure that each of these distances is more than a given cutoff distance. For the case of a oxygen and nitrogen atom pair, we allow a lower cutoff distance, which permits formation of hydrogen bonds. We choose the largest possible cutoff distances for which the native configuration passes the VDW test.

2.3 Problem of Generating Near-native Configurations

Although HMMSTR is good at predicting the local structure of a amino acid sequence, it does not give us any information about long distance interactions between various atoms [6]. By long distance we mean atoms which are more than a window length away. We notice that our HMMSTR generated samples do not have many non-local hydrogen bonds, which are especially important for the formation of β-sheets.

Hence to address these problems, we apply Monte Carlo minimization on each of the samples to minimize its Hydrogen Bond Energy (HBE). As a result of this, we get samples which not only show non-local hydrogen bonding, but are also closer to the native in terms of the DME distance.

2.4 Monte Carlo Minimization of Hydrogen Bond Energy (HBE)

Given a configuration, our goal is to generate a more compact and hydrogen bond rich configuration. We use a very simple, but effective model of the hydrogen bond energy (HBE). We calculate the HBE by summing up the distances between potential hydrogen bonds forming atom pairs. Each N and O atom pair within 4Å of each other is considered a potential hydrogen bond. Moreover for each atom is only allowed to form one hydrogen bond. We use 4Å as the cutoff distance (d_{cutoff}).

$$HBE = \sum_{all\ potential\ H-bonds} (d - 2.8)^2 - (d_{cutoff} - 2.8)^2$$

Here d is the distance between the N and O atoms, and 2.8Å is assumed to be the ideal distance between N and O atoms in a hydrogen bond. Each atom is only allowed to form one hydrogen bond.

To generate a HBE minimized sample we repeatedly modify the torsion angle of the configuration and see if we get a configuration with lower HBE. We always accept the modified (mutated) sample if its energy is lower than the original sample, and accept it with an exponentially decreasing probability if the energy is higher than the original sample. Choosing a low value oh temperature T means that high energy samples are less likely to be accepted. Details of this are given in Algorithm 1 below. We use a Gaussian random number generator with mean 0 and standard deviation of 5, to randomly modify the torsion angles during the mutation step. A smaller standard-deviation would mean smaller steps towards the final compact configuration.

Algorithm 1 MinimizeEnergy(s)

```
$es \leftarrow$ HBE(s) //Energy of the sample
repeat
    for $i = 0$ to $MAX\_ITERATIONS$ do
        $m \leftarrow$ Mutate(s)
        $em \leftarrow$ HBE(m)
        if $em < es$ or UniformRandom(0,1) $< e^{(es-em)/T}$ then
            $accept \leftarrow accept + 1$
            $es \leftarrow em$
            $s \leftarrow m$
        else
            $reject \leftarrow reject + 1$
        end if
    end for
    $acceptrate \leftarrow accept/(accept + reject)$
until $acceptrate$ drops below threshold or $i > MAX\_ITERATIONS$
return $s$;
```

2.5 Radius of Gyration (RG) Minimization

Apart from minimizing HBE, another possible way to improve a HMMSTR sample is by minimizing its radius of gyration (RG). RG is a popular measure of

Algorithm 2 MUTATE(s)

```
repeat
    for each torsion angle $t$ of s do
        $t \leftarrow t + \textsc{NormalRandom}(0, STDDEV)$ //modify by a random amount
    end for
until this sample passes VDW collision check
```

compactness of a conformation. It is defined as the average distance of all atoms from the center. Center is the average position of all the atoms. Minimizing RG however results in biasing each conformation towards a extremely compact ball-like shape. To address this issue we used Dead End Elimination [16] to add sidechains to each HMMSTR sample and then do the RG minimization. But even then we would end up with unrealistically compact configurations.

2.6 Advantages of using HMMSTR sampling

Although Gaussian sampling works well for short and medium length chains, it is expected to fail for longer sequences. This is because a lot of time is wasted in generating samples which might not be biologically possible. Also, any random sampling based technique has a heavy dependence on a good energy function, since using the energy function is the only way to identify a realistic sample. HMMSTR sampling gets around this problem by constructing its samples out of protein-like local structures. Our hypothesis is that since these local structures are parts of stable structures which are at a energy minima, hence a typical HMMSTR sample is more likely to be at a local minima of the energy landscape as compared to a Gaussian samples. As a result, HMMSTR samples serve as good milestone configurations for building the PRM graph. Moreover since no information about the 3D structure of the native is used, HMMSTR sampling is unbiased.

CHAPTER 3
SIZE OF CONFORMATIONAL SPACE

Proteins are sequence of amino acids. We model each amino acid with its ψ, ϕ backbone torsion angles. Each of these torsion angle can theoretically take on any value in (0,360], which can be associated with a points on a unit circle, denoted by S^1. This is the conformational space (C-space) of 1 torsion angle. Now a n amino acid long chain has $2n$ of torsion angles, but a change in the first or last torsion angle does not result in any change to the 3D structure of the protein. Hence given a amino acid sequence with n residues, its C-space can be expressed as:

$$C = \{q | q \in (S^1)^{2n-2}\}$$

This equation however does not give as any idea about the size of the C-space which is free (C-free), i.e set of all physically possible conformations of the protein. These impossible conformations correspond to extremely high energies, and a protein is never found in these conformations.

One way to estimate C-free is to discretize the C-space and then count those configurations which are biologically and physically possible (with low energy). The obvious way to do this discretization is by uniformly discretizing the torsion angles. So if we say that each torsion angle can be in 4 different states(for example (0,90], (90,180], (180,270], (270,360]), then the size of C-space can be estimated by the number of unique discrete configurations in C-space. For a n residue long sequence, the size of C-space is given by:

$$|C| = 4^{2n-2}$$

Here we are assuming that the torsion angles are uniformly distributed over $(0, 360]$. The Ramachandran plot however shows that this is not true. The Ramachandran plots show that ψ, ϕ angles are clustered around three main regions (labeled as H, E, L), which correspond to a β-sheet, Right handed α-helix and a Left handed α-helix. This is shown in Figure 3.1. Hence we can discretize C-space

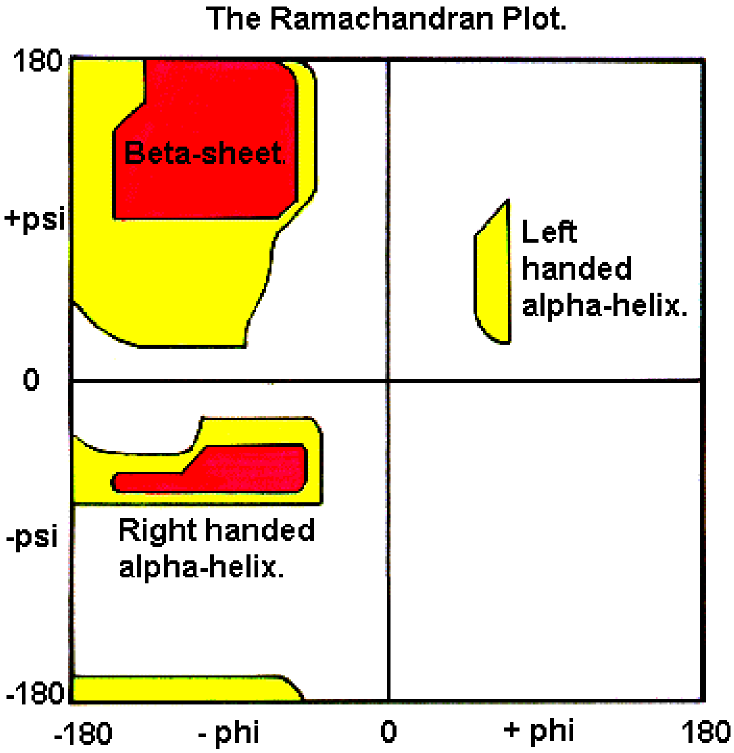


Figure 3.1: The Ramachandran Plot. The ψ, ϕ angles are clustered around three main regions (known as H, E, L), which correspond to a Beta-sheet, Right handed alpha-helix and a Left handed alpha-helix. [From: www.csbsju.edu]

so that each ψ, ϕ pair is labeled as H, E, or L. In other words, each amino acid has 3 degrees of freedom. Now the size of C-space (or maximum possible size of c-free) for a n residue long sequence can be expressed as:

$$|C_{HEL}| = 3^{n-2}$$

Again, $n - 2$ corresponds to ignoring the first and last residues. We now describe a way to compare the size of C-free by measuring measuring the rate at which we get unique samples. Using Algorithm 3 we plot number of unique samples versus total number of samples generated by using HMMSTR. We call this plot the uniqueness plot. Figure 3.2 shows uniqueness plot for different proteins. One main thing to notice is that 1VII(36) is a much larger protein than 1E0L(28), but it still has a much flatter uniqueness curve. Hence it is expected to have a smaller C-free size.

Algorithm 3 UniqueSamples(S)

```
$c \leftarrow 0$ //Number of unique samples
$U \leftarrow \{\}$ //Set of already seen unique samples
for each sample $s \in$ sample set $S$ do
    $different \leftarrow false$
    for each $u \in U$ do
        if IsDifferentHEL(s,u) then
            $different \leftarrow true$
            break
        end if
    end for
    if $different = true$ then
        $U \leftarrow U \cup \{s\}$
        $c \leftarrow c + 1$
    end if
end for
```

Algorithm 4 IsDifferentHEL(s,u)

```
$H \leftarrow (-60, 60)$ //centroid of the H region
$E \leftarrow (-120, 120)$ //centroid of the E region
$L \leftarrow (90, 0)$ //centroid of the L region
for $i = 1$ to NUMRESIDUES(s) do
    $sa \leftarrow (\phi, \psi)$ angles for the $i^{th}$ residue of $s$
    //Now find the region closest to $sa$
    $stype \leftarrow argmin(\text{Distance}(sa, H), \text{Distance}(sa, E), \text{Distance}(sa, L))$
    $ua \leftarrow (\phi, \psi)$ angles for the $i^{th}$ residue of $u$
    //Now find the region closest to $ua$
    $utype \leftarrow argmin(\text{Distance}(ua, H), \text{Distance}(ua, E), \text{Distance}(ua, L))$
    if $stype \neq utype$ then
        return $true$
    end if
end for
return $false$
```

3.1 Predicting Size of C-free

Since we know that there are a finite number of unique samples, the uniqueness plot of a given sample-set should level out eventually. We can define the size of C-free for a given sampling method and a protein as; the number of unique samples found after infinite number of tries. Or in other words, size of c-free is the y-coordinates of the uniqueness plot at $x = \infty$. One way to find the size of c-free is to fit the

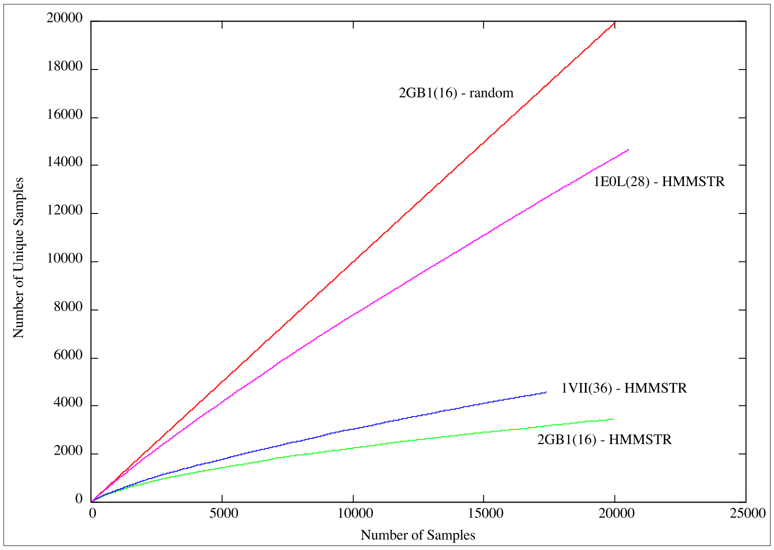


Figure 3.2: Comparison of c-space sizes of 2GB1(16), 1VII(36), 1E0L(28). Even though 1VII has more residues than 1E0L, still its C-space is smaller than 1E0L. This is because 1VII is mostly made up helices, which lowers the complexity of its C-space.

uniqueness plot to a curve and then by looking at the parameters of the curve, we can predict C-free size. For the simple case of random sampling, where each conformation is uniformly likely, we describe the method below.

3.2 C-free Size for Uniform Sampling

Let the expected number of unique samples found after x tries be $f(x)$. Now, if N is the total number of unique samples, then we can write $f(x)$ as:

$$f(x) = f(x-1) + 1 - \frac{f(x-1)}{N}$$

This is like saying, expected number of samples found after x tries is equal to the expected number of samples found after $x-1$ tries, plus the probability of finding a new sample on this try. We here are assuming that all the samples are uniformly distributed in the C-space. Solving this recursion we get:

$$f(x) = N - N\left(1 - \frac{1}{N}\right)^x$$

This is an exponential curve which levels out at $f(x) = N$. This curve however cannot be made to fit to uniqueness curve for samples generated using HMMSTR and then put into the H, E, L bins. This is because the samples are not uniformly distributed over the C-space.

CHAPTER 4
GENERATING FOLDING PATHWAYS

4.1 Probabilistic Roadmap Graph

After generating the samples, we then build the nearest neighbor graph which represents the protein folding funnel landscape. This is a directed graph where each node of this graph represents a valid low energy (with no VDW collisions) conformation of the protein, and each edge (u, v) represents a possible transition from u to v. We fix the number of outgoing edges to 20 per node.

One simple way to calculate the out edges for a given conformation is to find out the most *similar* conformations. A good way to compare 3D structure of proteins is using RMSD [11]. However RMSD calculations are expensive. We instead use Distance Matrix Error (DME) to compare two conformations. Given a protein conformation, its distance matrix is simply a triangular matrix where each element d_{ij} is the distance between C_α atoms of residues i and j. Now we can define DME distance between two N residue long configurations A and B as:

$$DME(A, B) = \frac{2}{(N-4)(N-5)} \sum_{i=1}^{N-4} \sum_{j=i+4}^{N} |d(A)_{ij} - d(B)_{ij}|$$

This is basically calculating the average difference between the distance matrices elements of the two configurations we are trying to compare. We do not consider nearby residues (less than 4 residues away) while calculating DME. DME is very similar to dRMS distance described in [11].

Although DME is much faster than RMSD, it however cannot differentiate between symmetrical conformations. As a result using DME might allow edges between conformations which are not possible to reach when we try to interpolate between them (linear interpolation of the torsion angles). Even in the case when the two conformations are similar, it still might not be possible to interpolate between them without having a self collision. To solve this problem, given two conformations

A and B, we first calculate the average conformation C (with average torsion angles) and then define the distance between A, B as:

$$distance(A, C, B) = \begin{cases} \infty & C \text{ fails VDW test} \\ DME(A, C) + DME(C, B) & C \text{ passes VDW test} \end{cases}$$

Now for a given conformation, we find $K(= 20)$ conformations with the smallest distances, and add an edge between them. This is described in Algorithm 5

Algorithm 5 FindOutEdges(u,S,K)

Calculates K out-edges of u, given sample-set S.

```
$E \leftarrow \{\}$ //list of out-vertices
for each sample $s$ in $S$ do
    $c \leftarrow$ Average$(u, s)$ //$c$ has averaged torsion angles
    if $c$ passes VDW collision test then
        $d_s \leftarrow$ DME$(u, c)$ + DME$(c, s)$ //distance between $s, u$
        $E \leftarrow E + (s, d_s)$
        if Size$(E) > K$ then
            remove the sample with the greatest distance from $E$
        end if
    end if
end for
return $E$
```

This method can be made more accurate by considering more intermediate conformations between A and B, but we do not do this to save on computational cost.

4.2 Energy

After building the nearest neighbor graph, we now assign a energy value to each node in the graph. We use a very simple energy function which is a linear combination of normalized Radius of Gyration (RG) and normalized Hydrogen Bond Energy (HBE). We give RG a weight of 1.0 and HBE a weight of 5.0 (HBE is favored over RG).

$$Energy(s) = 5 * Normalize(HBE(s)) + 1 * Normalize(RG(s))$$

By normalization, we mean mapping each value to the interval (0,1]. Given a energy value e which lies in the interval $[E_{min}, E_{max}]$, we calculate its normalized value as:

$$Normalize(e) = \frac{e - E_{min}}{E_{max} - E_{min}}$$

4.3 Folding Pathways

We use Dijkstra's shortest path algorithm to find the folding pathways on the PRM graph. Weight of each edge is set to the difference in the energy of the conformations. All the negative edge weights are set to zero. Also Dijkstra's algorithm is used for finding paths to each node from a single source, but we need to find a path from all nodes to a single target (the native conformation). Hence we invert all the edges and then run the algorithm.

4.4 Problems with PRM Graph Building

The algorithm described above works well for small proteins like the 16 residue long segment of 2GB1 protein. However, this method failed for 28 residue long 1E0L, and 36 residue long 1VII. For the case of 1E0L, no in edges for the native configuration were found. We were able to fix this problem by converting this directed graph to a undirected graph as shown in Figure 4.1. However, for the case of 1VII, the algorithm failed to find any incoming or outgoing edges for the native configuration. As a result we were not able to generate a valid PRM graph, and as a result were not able to generate pathways.

4.5 Interpolation

Once the pathways have been generated, we now use simple linear interpolation of the torsion angles to get interpolated configurations. We use this linear interpolation to generate 10 intermediate configurations between each pair of adjacent configurations in the path.

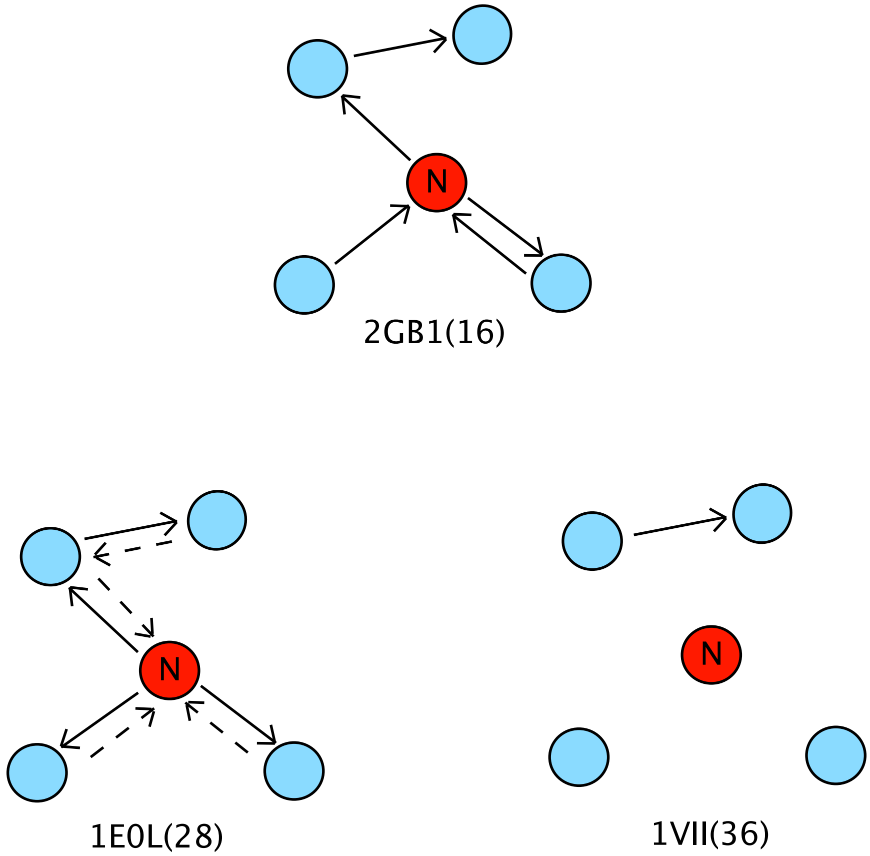


Figure 4.1: Sampling Problem: For 1VII(36), the native (N) was not connected to any other sample. For 1E0L, the native was connected to other samples, only through outgoing edges (solid arrows). As a result we had to add edges in the other direction (dashed arrows) for all the edges in the graph, making the graph undirected. The native configuration of 2GB1(16) however was well connected to other configurations through both incoming and outgoing edges.

CHAPTER 5
RESULTS

5.1 Generating Samples

Our primary goal was to generate samples which as unbiased by the knowledge of native structure. We generated about 20,000 HMMSTR samples each for 2GB1(16), 1VII(36) and 1E0L(28) protein fragments.

The only input to the sample generation process is the amino acid sequence code of the protein. HMMSTR takes this sequence code and gives us torsion angles of local structures. We use these local structures to build complete configurations. Hence sampling is not biased by the 3D shape of the protein's native configuration. Figure 3.2 shows the uniqueness plot (number of unique samples found vs. total number of samples) for these three sample sets. For comparison the figure also shows the uniqueness plot of randomly generated 2GB1(16) samples wich pass the van der Waals (Section 2.2) collision check. We notice that HMMSTR samples have a much smaller C-free size when compared to randomly generated samples. Another interesting observation is that although 1VII has more residues than 1E0L, its C-free is much smaller than 1E0L. This suggests that the size of C-free does not depend entirely on the number of residues for a given sequence of amino acids, but also on its secondary structures. In the case of 1VII, the smaller C-free size might be due to helices in its structure, which restrict the motion of torsion angles, and thus reduce the overall number of degrees of freedom when compared to a β-sheet protein.

We improved our sampling by minimizing Hydrogen Bond Energy (HBE) of each HMMSTR sample as described in Section 2.4. HMMSTR samples only model

PDB ID	Size	Sequence
2GB1	16	GEWTYDDATKTFTVTE
1E0L	28	SEWTEYKTADGKTYYYNNRTLESTWEKP
1VII	36	MLSDEDFKAVFGMTRSAFANLPLWKQQNLKKEKGLF

Table 5.1: Amino acid sequences of proteins used for testing.

PDB ID	Residues	HMMSTR Samples				HBE Compact Samples			
		Min	Max	Avg	Std. Dev.	Min	Max	Avg	Std. Dev.
2GB1	16	1.40	13.25	8.34	2.00	0.93	11.43	5.40	1.75
1E0L	28	3.04	21.72	10.19	2.98	2.71	19.70	8.04	2.53
1VII	36	2.69	18.75	7.96	2.01	2.40	17.64	7.29	1.89

Table 5.2: DME Distance of samples from the native (in Å).

the local interactions between the atoms, and HMMSTR cannot predict long distance interactions (hydrogen bonds). To address this we add HBE minimized samples, which basically brings atoms, which might form hydrogen bonds, closer to each other. After HBE minimization, the HMMSTR samples were on an average closer to native, as seen in Table 5.2. Figure 5.1 shows the distribution of these samples as a function of their DME distance from the native configuration. Please note that the HBE minimization process does not use any information about the 3D structure of the native, hence it maintains our goal of having unbiased sampling.

One thing to notice in Figure 5.1 is that 1VII(36) sampling does not gain significantly from HBE minimization. It shows almost the same distribution of HBE minimized samples as HMMSTR samples. This is because most of the 1VII(36) HMMSTR samples we generate are entirely made up of helices, which already have hydrogen bonds. Hence HBE minimization does not do much for this protein.

5.2 Smoothness of Pathways

To analyze the smoothness of pathways, we first define the following terms:

Path Smoothness (p)**:** Given a path, we define its smoothness as the average length of an edge in the path. The distance metric used here is DME.

Average Path Smoothness (p_{avg})**:** Given all paths to the native from every node in the graph, we calculate the average of each path's smoothness. This is a good measure of overall smoothness of paths, for a given energy function. This value depends on the energy function used, since the paths are calculated by minimizing the overall energy of the path.

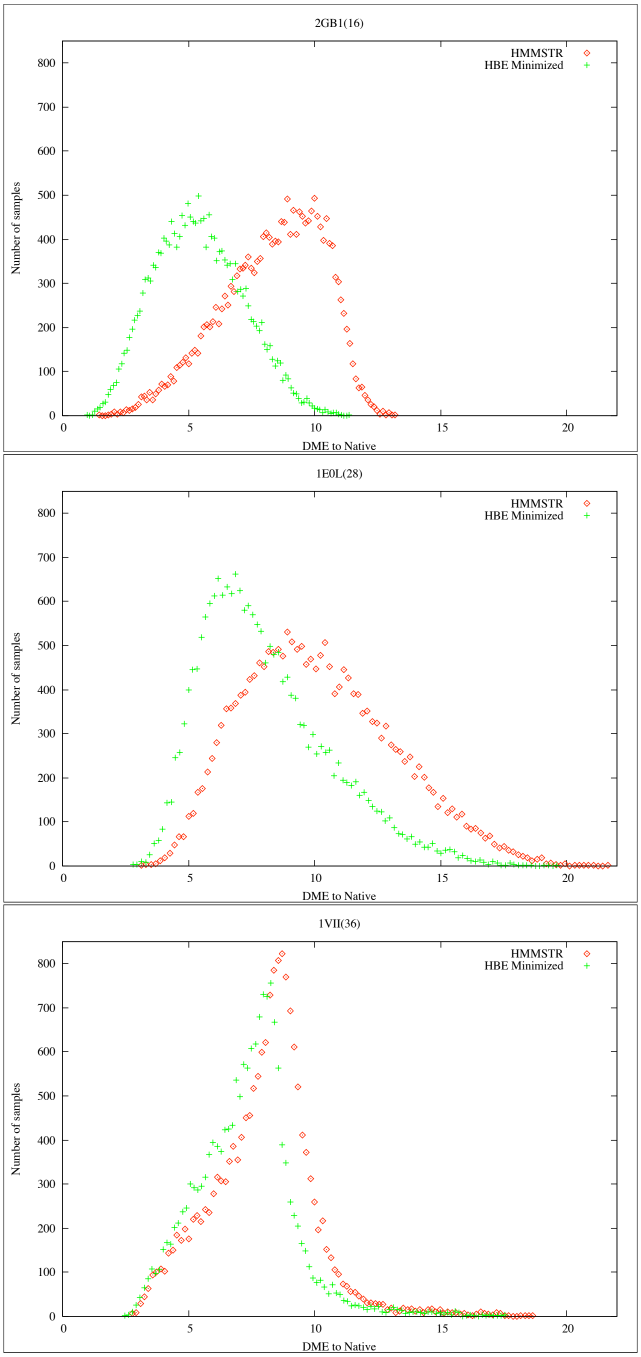


Figure 5.1: DME-to-native distribution of samples. The X axis represents the DME distance of a sample from the native configuration, and Y axis represents frequency. In each case we can see that HBE minimized samples are on an average closer to native. However for 1VII(36), this difference is very small. This is because most 1VII(36) samples are primarily made of helices, which already have hydrogen bonds (HB) in place. Hence HBE minimization does not change them significantly.

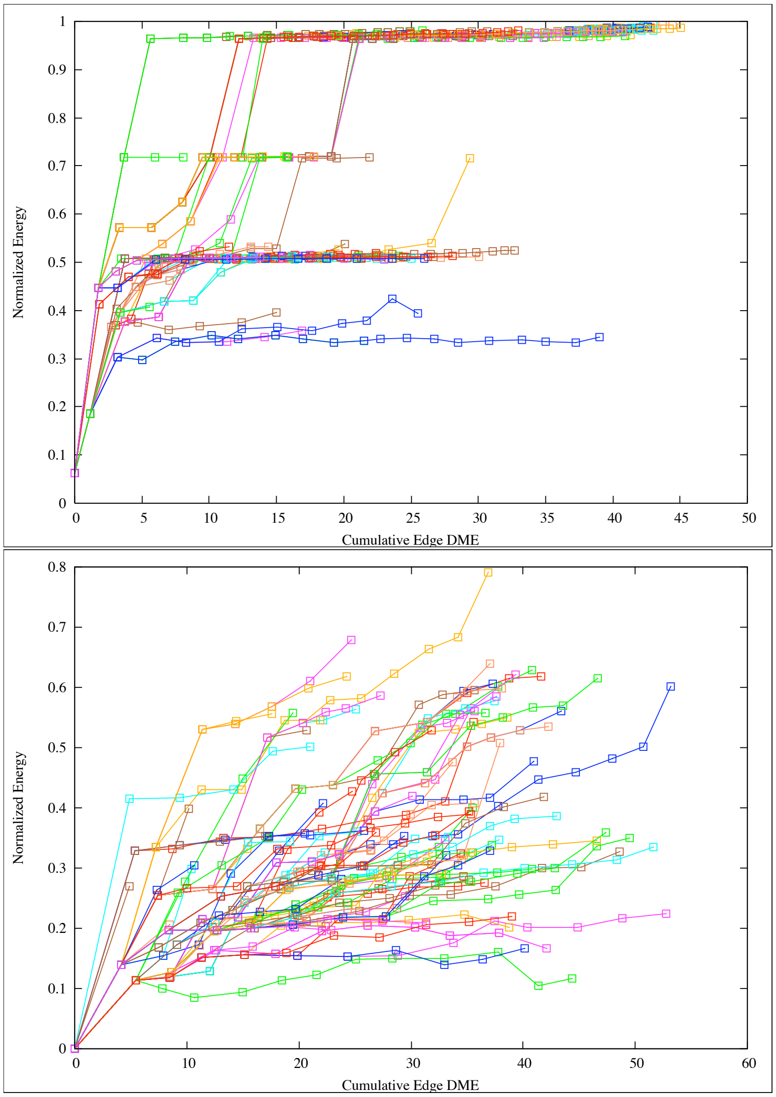


Figure 5.2: Smoothness of paths: 16 residue long 2GB1 (top), 28 residue long 1E0L bottom. Each figure shows 100 random pathways. These paths are without any interpolated configurations. The X axis represents the cumulative edge length, which is the sum of all edge lengths till the given configuration (where edge length is the DME distance between the two configurations forming the edge). The Y axis represents normalized energy (Section 4.2) of the configuration. Going from left to right represents unfolding of the protein.

PDB ID	Residues	n_H	n_M	p_{avg} (σ)	g (σ)
2GB1	16	20000	19974	$1.882\AA$ (0.299)	$4.406\AA$ (0.986)
1E0L	28	20545	20539	$3.750\AA$ (0.304)	$2.207\AA$ (0.933)
1VII	36	17400	16600	–	–

Table 5.3: Smoothness values for sample proteins. n_H is number of HMM-STR samples, n_M is the number of HBE minimized samples, p_{avg} is the average path smoothness, g is the graph smoothness, σ is the standard deviation

Graph Smoothness (g): Given a graph, we calculate its smoothness by averaging the length of every edge. We use DME as the distance metric. This value is independent of the energy function used, and represents the average smoothness of pathways, given any energy function.

Table 5.3 summarizes these smoothness values.

Another visualization of smoothness of pathways is shown in Figure 5.2. The figure shows 100 random pathways. Each point on the curve represents a configuration on the path. The X axis represents the cumulative length of the edge (in DME), and the Y axis represents normalized energy. The cumulative edge length is the sum of all edge lengths (where the edge length is the DME distance between the two configurations forming the edge) till the given configuration. This figure can be viewed as a cross-section of the folding funnel.

We were not able to generate a PRM graph for 1VII, and as a result not able to generate pathways. This is because while trying to add an edge, the vdW collision check on the averaged configuration (with average torsion angles), failed for each and every try (Section 4.4).

5.3 Folding Highways

After generating pathways, we want to identify the characteristic folding pathways, which we call *folding highways*. Hence we built a *connectivity graph* of the pathways which is a subgraph of the PRM graph in which each node represents a configuration and an edge represents a path segment towards the native. We define the *popularity* of a given node as the number of times the node appears in a pathway. Figures 5.5, 5.6 show example connectivity graphs. A darker circle represents

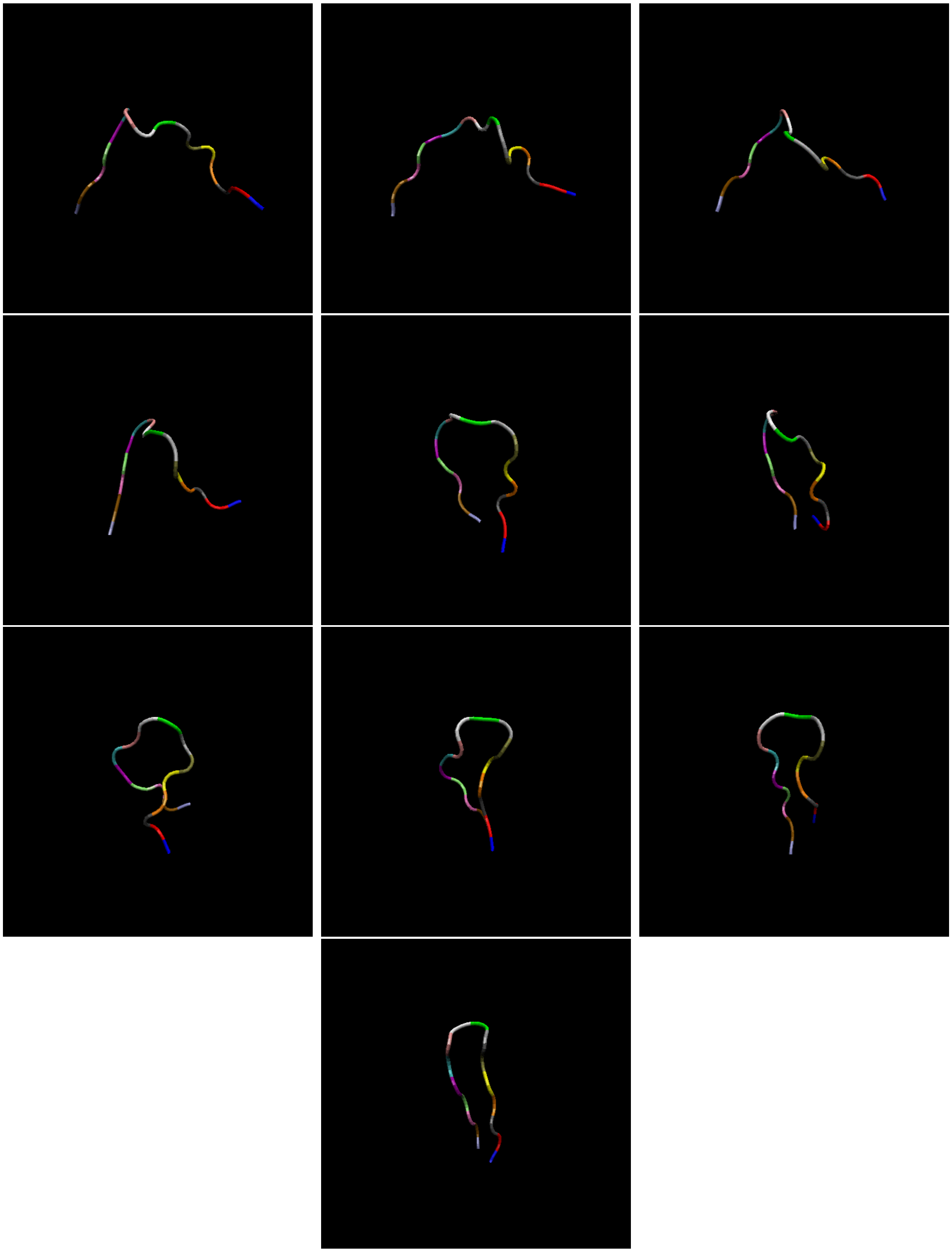


Figure 5.3: 2GB1(16) folding pathway snapshots (going from left to right, top to bottom)

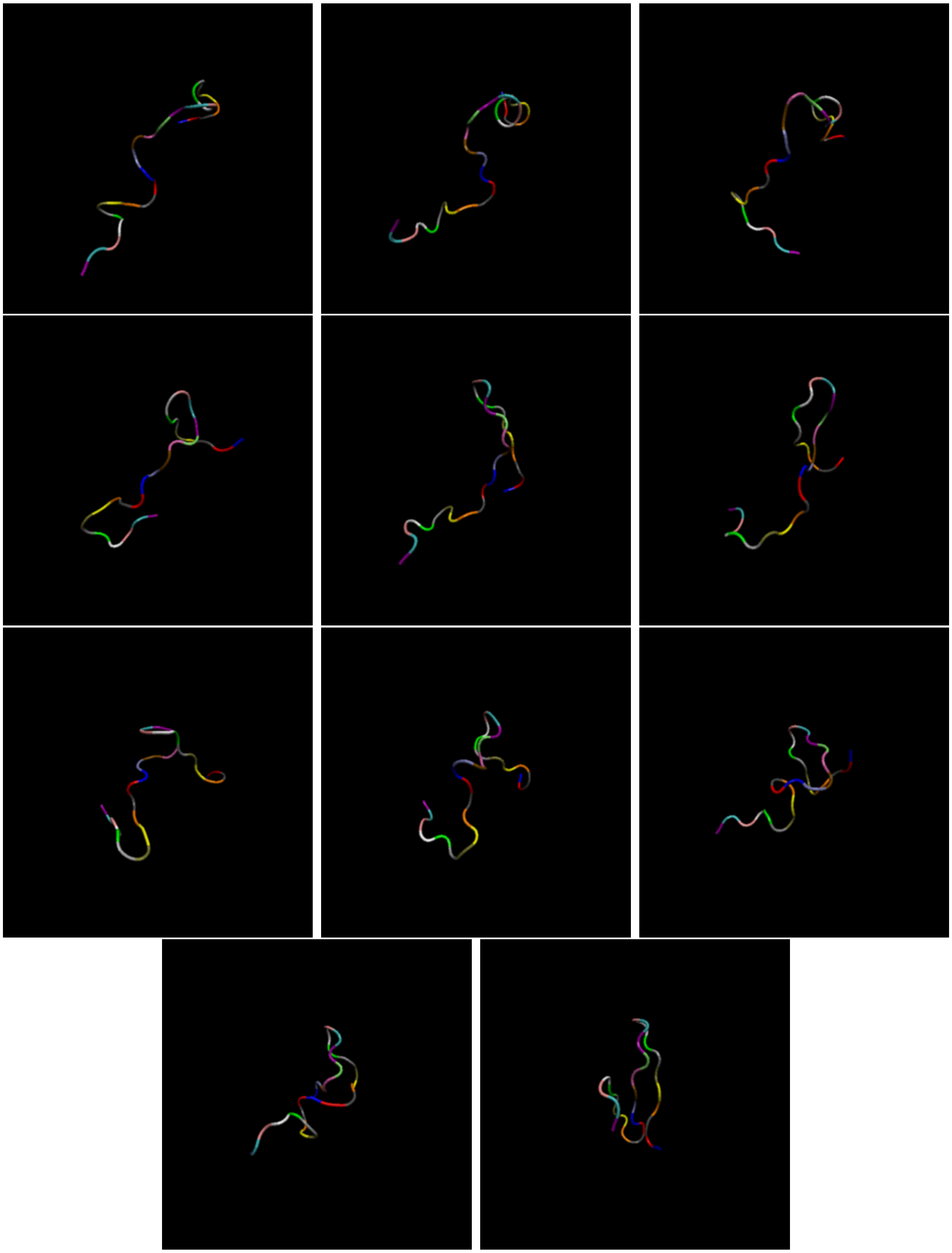


Figure 5.4: 1E0L(28) folding pathway snapshots (going from left to right, top to bottom). The first β-hairpin (N-terminus) is the one on right hand side in the last (native) configuration.

a more popular node, and a smaller circle represents a lower energy node.

We can see that there are a few preferred highways (popuular pathways) which a protein takes to reach the final native state. These highways are also shown separately in Figures 5.5 and 5.6. Our observation is that for 1E0L and 2GB1, the highways are mainly made up of low energy conformations represented by small circles.

5.4 Comparison of Results with Experimental Data

5.4.1 2GB1(16)

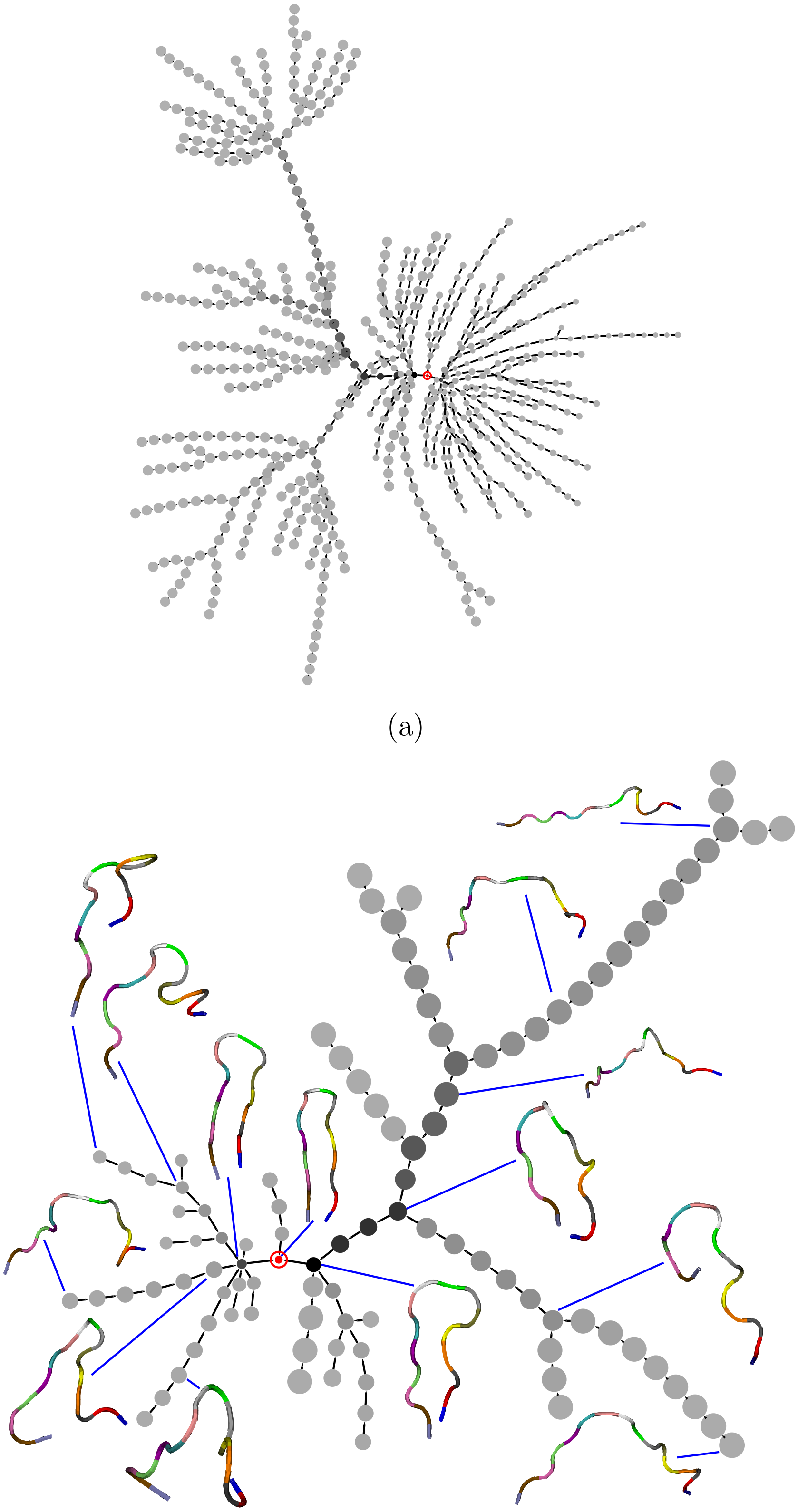
The 2GB1 β-hairpin folds in 6μs at room temperature, which is about 30 times slower than the rate of α-helix formation [15]. Experimental and theoretical analysis [15] shows that this β-hairpin is stabilized by hydrogen bonding and hydrophobic interactions. Its folding shows a two-state behavior, and a funnel-like partially rugged energy landscape. It is predicted that the folding is either initiated from the β turn (middle of the chain), or from the hydrophobic cluster which forms the middle portion of the final hairpin structure.

Our Observations

Looking at the pathways in Figure 5.3 and 5.5, we see that the folding is initiated from the middle of the chain, but at the same time, the hydrophobic cluster is the first one to take its final shape. For other pathways we see a more zipper-like behavior, in which hydrogen bonds are formed in order of increasing distance from the loop.

5.4.2 1E0L(28)

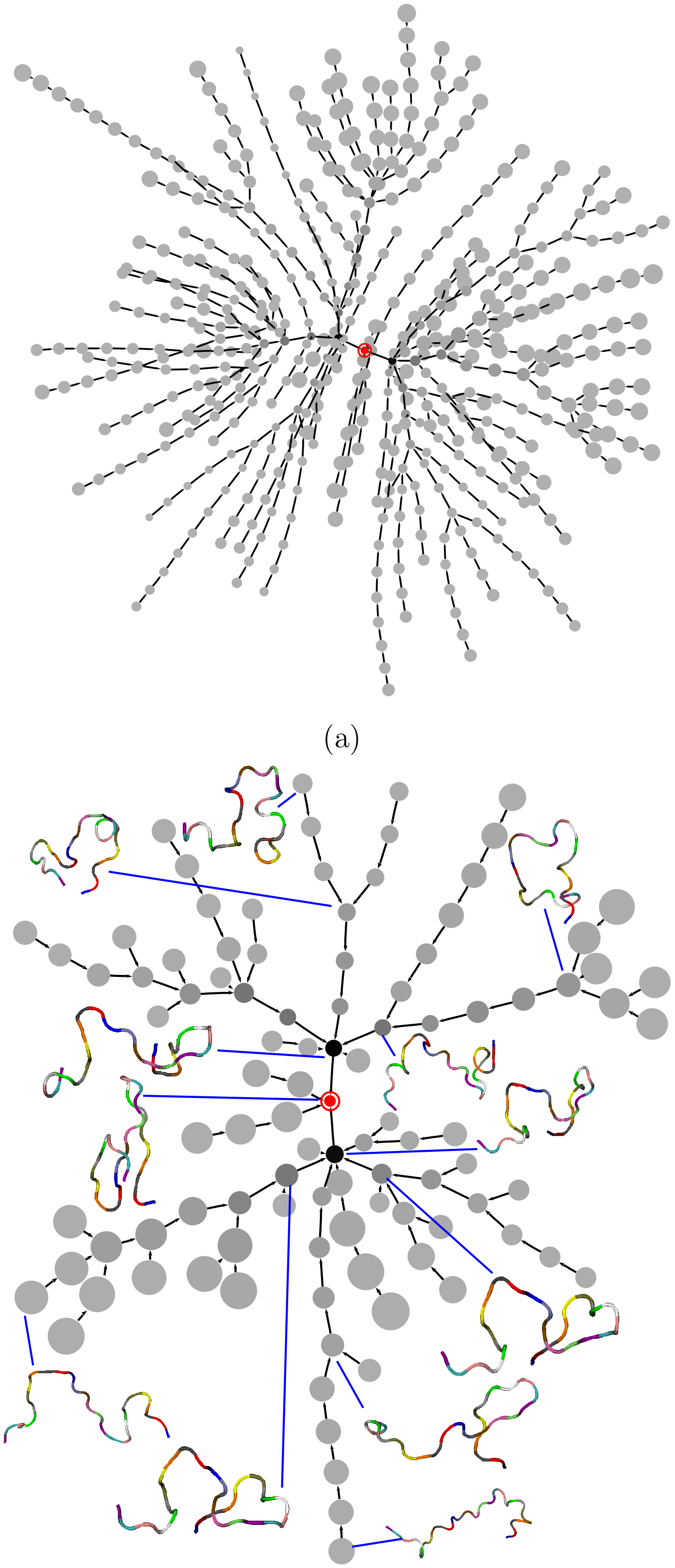
Molecular dynamics simulations of the thermal unfolding of 1E0L [8] suggest that at all temperatures, the second and third strands (C-terminus β-sheets) are the first to separate during unfolding. This means that the first β-hairpin (N-terminus) is formed first during the folding process.



(a)

(b)

Figure 5.5: Connectivity graphs representing 100 random folding pathways of 2GB1 (a) and 100 of the most popular paths (b). A smaller node represents lower energy, and a darker node represents more popular node.



(a)

(b)

Figure 5.6: Connectivity graphs representing 100 random folding pathways of 1E0L (a) and 100 of the most popular paths (b). A smaller node represents lower energy, and a darker node represents more popular node.

Our Observation

We do see the first β-hairpin forming first along a few highways as shown in Figure 5.4 and 5.6, but not all the highways. We calculated the order of formation of loops on 100 random pathways, and found that the first β-hairpin forms first, 47% of times. We calculate the order of secondary structure formation by calculating the age of each of the secondary structures. A secondary structure is said to be formed when 90% of its contacts have been established permanently.

5.4.3 1VII(36)

1VII is a small α-helical protein. Molecular dynamics simulation of 1VII's folding [19] suggests the following:

- When started from completely elongated structure, most of the simulations collapse immediately to an intermediate state, which has a radius of gyration and the Solvent Accessible Surface Area (SASA) of the native state, within 20 ns. The experimental folding time is approximately 10μs.

- This early intermediate contains many conformations with some small amount of native secondary structure, but with incorrect backbone topology and side-chain packing.

- The intermediate is a compact hydrophobically collapsed globular structure.

- Only a small number of trajectories fold concurrently with the hydrophobic collapse.

Our Observations

We were not able to generate a PRM graph for 1VII using HMMSTR and HBE minimized samples. We believe the reason for this is the use of linear torsion angle interpolation, which generated invalid intermediate configurations (with vdW collisions). Since 1VII is mainly made up of α-helices that are very tightly packed, even small changes in its torsion angles are likely to cause vdW collisions. As a result the average intermediate configuration, which has the average torsion angles of the two given configurations, is likely to have collisions.

PDB ID	Residues	Num. HMMSTR samples	Compacting	Graph Building
2GB1	16	20000	4h50m	192h
1E0L	28	20545	15h17m	468h
1VII	36	17400	26h35m	728h(failed)

Table 5.4: Running time of various time consuming steps. Time shown is for a single Itanium2 CPU time. These times are approximate because the system used was a time-shared computer.

5.4.4 Folding Kinetics

We know that entropy is one of the main barriers to folding. Hence the time of folding should be proportional to the size of conformational space (C-free). Now since 1E0L has a much bigger C-free size as compared to 2GB1 and 1VII as predicted in Chapter 3, we should expect much higher folding times for 1E0L. This is in fact what has been observed experimentally. WW domain 1E0L is known to fold in about $30\mu S$, which is very large time when compared to $4.3\mu S$ for 1VII and $6\mu S$ for 2GB1 [13].

5.5 Timing Data

The steps of HBE minimization and graph generation are the bottlenecks of the whole process in terms of the computational run time. Table 5.4 shows approximate runtimes for each of these steps for each protein tested.

5.6 Discussion

We have presented a new technique to sample the folding funnel of a given protein. We do this without using any structural information about the native conformation; hence this method is unbiased by the native state. The samples are built using HMMSTR, which results in having samples that are much more likely to be biologically feasible as compared to random sampling. This also greatly reduces the size of the sampled C-space. We were able to generate pathways for 2GB1(16) and 1E0L(28), but not for 1VII(36). This was because we were not able to connect the native configuration of 1VII(36) to any of the other samples in the PRM graph. We feel this problem could be eliminated if a more realistic local planner is used (for example something which interpolates in 3D space and models the bonds as springs). Currently we just linearly interpolate the torsion angles while generating intermediate configurations between two given configurations.

For 1E0L(28), our results do not match completely with the experimental predictions. It is predicted that the first β-hairpin forms first, whereas we observe this phenomenon only in 47% of the cases. Moreover we see that all the secondary structures are formed between the last two configurations (native and one before that) in any given path. We feel these results could be improved if we do denser sampling and if we use a more realistic energy function. We currently consider all N and O atom pairs within a cutoff distance as hydrogen bonds which is not very accurate. Also while doing vdW collision check and calculating hydrogen bond energy, we do not consider the sidechains. We believe that addressing all these problems should give us much better results.

CHAPTER 6
FUTURE WORK

We have presented a biologically significant and efficient way to sample the protein folding funnel. We use these samples to build a PRM graph which can be used to predict the protein folding pathways. Our work shows promising results and opens up several different avenues which we plan on exploring in the future.

- While generating HMMSTR samples, we chose a window size of 5 residues. Since this might not be the best choice for longer proteins, we hope to explore bigger window lengths while sampling these proteins.
- We present a way to compare the C-free sizes of proteins, by plotting their uniqueness curves. We show how to find the equation of this curve in the case when the samples are uniformly distributed across all the bins (Section 3.2). However we were not able to come up with the equation of a curve, in the case when we use HMMSTR samples and try to put them in H,E,L bins.
- We currently use a very simple energy function which does not entirely capture the complexity of folding funnel. In the future we plan on using a more realistic energy function (for example CHARMM [5]), as that might give us better results.
- We do not model sidechains of the protein, because that increases the degrees of freedom (d.o.f) of the protein, and as a result increases the computation costs. In the future we hope to include sidechains.
- We linearly interpolate the torsion angles to find intermediate conformations between two given conformations. This looks visually pleasing but is not very biologically correct. In the future we hope to incorporate a more accurate interpolation technique. One such interpolation technique is by modeling bonds as springs [10]. This should also hopefully allow us to generate realistic and

valid intermediates which pass van der Waals check, and as a result produce a PRM graph for 1VII and other longer proteins.

- We plan to use a more accurate model of hydrogen bond and van der Waals energy functions. Ideally a hydrogen bond consists of a linear arrangement of N, O, and H atoms such that the distance between O and H atoms is $1.8\AA$ and distance between O and N atoms is $2.8\AA$. We currently only model the latter constraint.

- Although the HMMSTR samples lie in the valid regions of the Ramachandran plot since they are built from fragments of actual proteins, we do not consider the plot while generating the HBE minimized samples. As a result some of the HBE minimized samples have invalid ϕ and ψ angles. We hope to fix this in the future by constraining the angles to be within the allowed regions.

LITERATURE CITED

[1] Nancy M. Amato, Ken A. Dill, and Guang Song. Using motion planning to map protein folding landscapes and analyze folding kinetics of known native structures. *Journal of Computational Biology*, 10(3):239–255, 2003.

[2] Christian B. Anfinsen. Principles that govern the folding of protein chains. *Science*, 181(4096):223–230, July 1973.

[3] Mehmet Serkan Apaydin, Douglas L. Brutlag, Carlos Guestrin, David Hsu, and Jean-Claude Latombe. Stochastic roadmap simulation: An efficient representation and algorithm for analyzing molecular motion. *International Conference on Research in Computational Molecular Biology*, 2002.

[4] Carl Branden and John Tooze. *Introduction to Protein Structure*. Garland Publishing, Inc., 1999.

[5] B. R. Brooks, R. E. Bruccoleri, B. D. Olafson, D. J. States, S. Swaminathan, and M. Karplus. Charmm: A program for macromolecular energy, minimization, and dynamics calculations. *Journal of Computational Chemistry*, 4:187–217, 1983.

[6] Christopher Bystroff, Vesteinn Thorsson, and David Baker. HMMSTR: a hidden Markov model for local sequence-structure correlations in proteins. *Journal of Molecular Biology*, 301(1):173–190, August 2000.

[7] Ken A. Dill. Polymer principles and protein folding. *Protein Science*, 8:1166–1180, 1999.

[8] Neil Ferguson, Jose Ricardo Pires, Florian Toepert, Christopher M. Johnson, Yong Ping Pan, Rudolf Volkmer-Engert, Jens Schneider-Mergener, Valerie Daggett, Hartmut Oschkinat, and Alan Fersht. Using flexible loop mimetics to extend phi-value analysis to secondary structure interactions. *Proceedings of the National Academy of Science of the United States of America*, 98(23):13008–13013, November 2001.

[9] Lydia E. Kavraki, Petr Svestka, Jean-Claude Latombe, and Mark Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, 1996.

[10] Moon N. Kim, Robert L. Jernigan, and Gregory S. Chirikjian. Efficient generation of feasible pathways for protein conformation transitions. *Biophysical Journal*, 83:1620–1630, 2002.

[11] Patrice Koehl. Protein structure similarities. *Current Opinion in Structural Biology*, 11:348–353, 2001.

[12] Rachel Kolodny, Patrice Koehl, Leonidas Guibas, and Michael Levitt. Small libraries of protein fragments model native protein structures accurately. *Journal of Molecular Biology*, 323(2):297–307, 2002.

[13] Jan Kubelka, James Hofrichter, and William A. Eaton. The protein folding 'speed limit'. *Current Opinion in Structural Biology*, 14:76–88, 2004.

[14] Jean-Claude Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, 1991.

[15] Victor Munoz, Peggy A. Thompson, James Hofrichter, and William A. Eaton. Folding dynamics and mechanism of beta-hairpin formation. *Nature*, 390:196–199, November 1997.

[16] N. A. Pierce, J. A. Spriet, J. Desmet, and S. L. Mayo. Conformational splitting: A more powerful criterion for dead-end elimination. *Journal of Computational Chemistry*, 21(11):999–1009, 2000.

[17] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, February 1989.

[18] Guang Song, Shawna Thomas, Ken A. Dill, J. Martin Scholtz, and Nancy M. Amato. A path planning-based study of protein folding pathways with a case study of hairpin formation in protein G and L. *Proceedings of 7th Pacific Symposium on Biocomputing (PSB)*, pages 240–251, January 2003.

[19] Bojan Zagrovic, Christopher D. Snow, Michael R. Shirts, and Vijay S. Pande. Simulation of folding of a small alpha-helical protein in atomistic detail using worldwide-distributed computing. *Journal of Molecular Biology*, 323:927–937, 2002.

[20] Mohammed J. Zaki, Vinay Nadimpally, Deb Bardhan, and Chris Bystroff. Predicting protein folding pathways. *Bioinformatics*, 20:i386–i393, August 2004.

[21] Ming Zhang and Lydia E. Kavraki. A new method for fast and accurate derivation of molecular conformations. *Journal of Chemical Information and Computing Sciences*, 42:64–70, 2002.