Probabilistic Road-Map based Protein Folding

CS365:Artificial Intelligence
Era Jain(Y9209)
Romil Gadia(Y9496)

Advisor:
Dr. Amitabha Mukerjee
Department of Computer Science and Engineering

Abstract

Protein Folding is a biochemical process but it is closely linked to Robotic Motion Planning. At first sight, robots and proteins seem to have little in common. But, they are very similar in their functionality which is based on their motions. Both, proteins and robots, can be modelled by their degrees of freedom(dof). This makes it possible to study motions in these two different domains by applying the same underlying algorithmic framework. Our method to study Protein Folding shall be derived from Probabilistic Road Maps(PRMs), originally developed for Robotic Motion Planning. The traditional methods of Motion Planning in protein are applied on the nodes in a Road Map, where the nodes represent the protein's conformations. Our aim would be to create a smaller set of nodes which is approximate but representative of the protein's energy landscape.

1 Introduction

Proteins are complex biochemical compounds that consist of a series or chain of amino acids folded in a particular way. Protein Folding is thus the process by which a protein attains it's functional shape or conformation (structure with minimum energy/native state) from its constituent polypeptides (chain of amino acids residues) by folding in a particular pattern. A protein can follow infinite number of distinct paths to attain its final conformation of which only a few are energetically feasible. Also, structurally similar proteins can fold in completely different manner.

Motivation

Why are we addressing this problem?
Final Protein structure relates to it's functionality. Diseases like mad cow disease, Alzheimer's disease can be associated with protein misfolding. Thus, finding correct folding pathway can help detect where the misfolding occurred.

Molecular dynamics is a highly complex and intricate field. These small molecular particles have highly dense structures. An average sized protein is composed of around 50-60 amino acids. Each such amino acid is again composed of many atoms which can exist in space in various orientations. These different orientations result in different energies owing to repulsions or attractions. Due to this, a protein can have a huge number of conformations (similar structures varying in their energies). Thus, while folding to it's native state (final state), a protein can move through various conformations resulting in innumerable folding pathways. Thus, finding an approximately correct and energetically feasible pathway is not a trivial task.

Many researches are going on in this area. Researchers are coming up with different models and methodologies. One of the approaches is the Robotic Motion Planning approch which makes use of probabilistic road maps and our project is based on the same.

2 Preliminaries

- Conformation :The different 3-D forms or shapes of varying energies in which a molecule exists.
- Native State: The lowest energy conformation of the protein is its native state or the final state in protein folding.
- Residues: The amino acids present in a protein are the residues of that protein.
- Protein Structure : Protein is a chain of amino acids. Each amino acid can be modelled by majorly 4 dihedral angles phi, psi , chi and omega, out of which chi and omega remain more or less constant across different conformations. Also, the bond lengths can be assumed to be constant for different conformations. Thus, each amino acid can be modelled by (phi, psi) torsional angles pair. (fig. 1(a))

3 Robotic Motion Planning Approach and its analogy with Protein Folding:

1. Protein as an articulated robot:

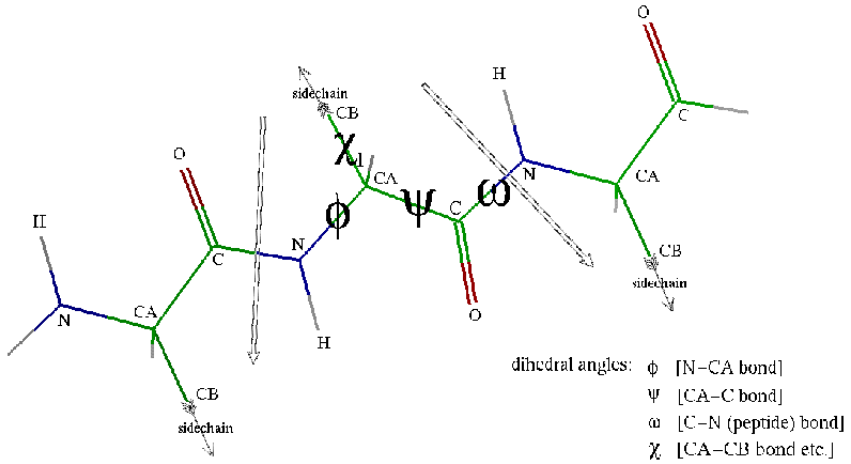


Fig.1(a)

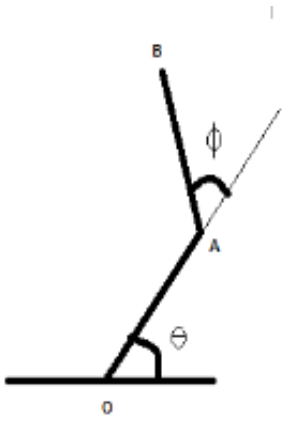
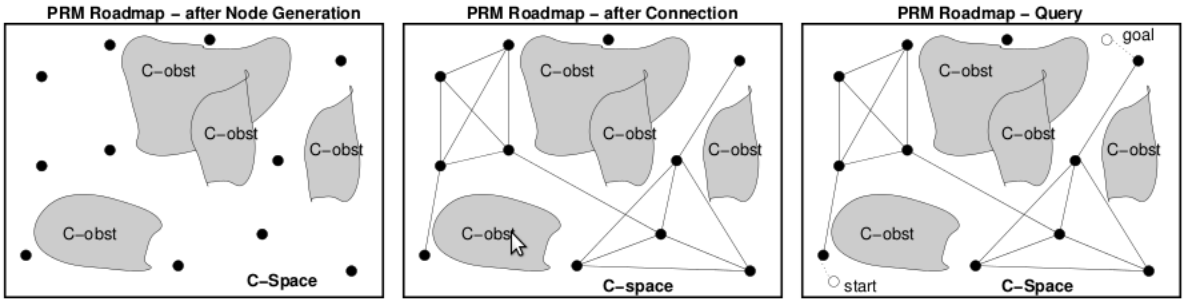


Fig.1(b)

Fig.(a) is how a protein looks like. The structure between the two arrows is an amino acid. Each amino acid is characterized by a set of dihedral angles: ϕ, ψ, χ, ω. Out of which χ and ω remain almost constant and thus their changes can be ignored. The angles of importance are the backbone's phi and psi, any change in them results in a different conformation. Thus we can see that each amino acid has two degree of freedoms. Thus a protein can be modelled as an articulated robot as shown in fig.(b) with phi and psi modelled as two revolute joints having value in the range $[0,2\pi)$

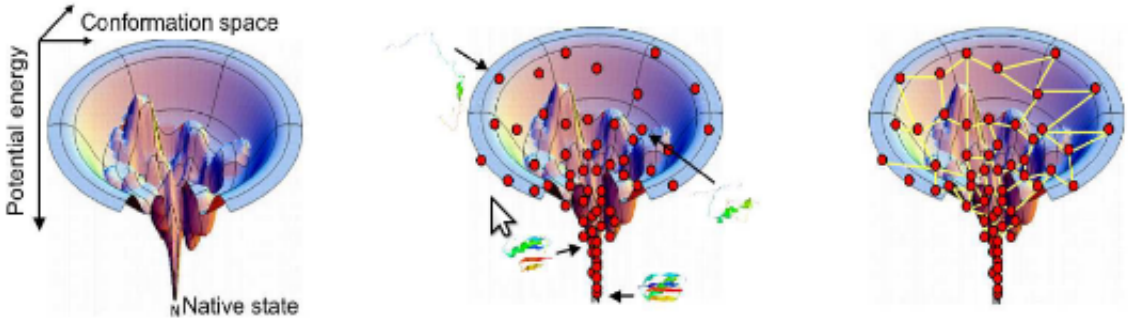
2. Protein energy landscape as C-space map and PRMs



(a) (b) (c)

Figure 3: A PRM roadmap in C-space. A PRM roadmap: (a) after node generation, (b) after the connection phase, and (c) using it to solve a query.

Fig.2



(a) (b) (c)

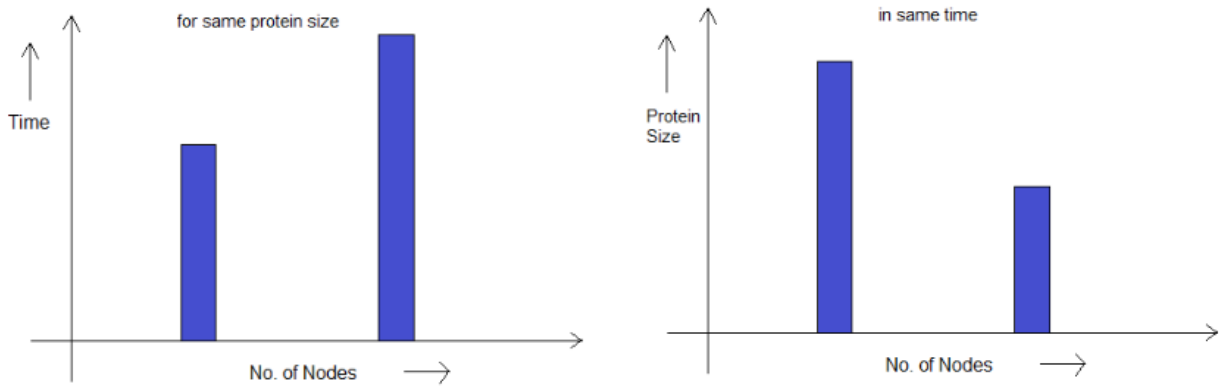
Fig.3

Fig.(2) shows the C-space and Obstacle space of a robot. The possible positions of the robot are shown on the map with dots. These dots are then connected to form a probabilistic road map (probabilistic because the probabilities that a robot moves to a particular location vary with locations, for example probability to move to an obstacle should be very low)

Fig.(3) shows the energy landscape of a protein molecule. The high peaks denote the location of high energy conformations while the low valleys accommodate all the low energy conformations. The bottom most point N represent the native state which is the final state with lowest energy of all the conformations. Just like a robot cannot move to any location randomly, analogously, folding of a protein to any conformation depends on it's energy. The ones with lower energies are favored with higher probablities and thus we can see that on the map there's a high concentration of nodes (representing conformations) on the valleys which gets sparse towards the peaks. Finally a roadmap can be built connecting these nodes, encoding about thousands of pathways, out of which the one which is most energtically feasible is selected.

Importance of Map reduction:

We aim at reduction in map size, that is use of lesser number of conformations/nodes on the graph but still maintaining a representative though approximate map of the protein energy landscape.



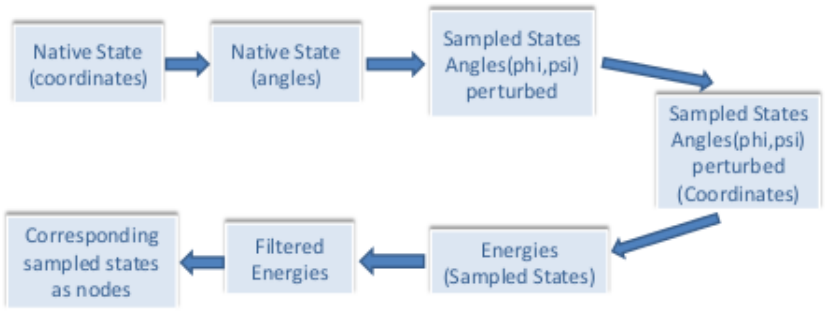
Using the entire protein landscape increases the complexity due to expensive use of memory and high computational time. The reduction in map size is aimed at reducing the number of nodes on which the processing can be done. This reduction results in decreased amount of computational time for a fixed size protein. This is equivalent to saying that given a fixed amount of time for computation, greater sized protein (with more amino acids) can be processed.
The computations on proteins are time consuming and often the computer's computation limit comes into play. This creates a limit on the maximum size protein which could be analysed. This limit being low as compared to most of the proteins of concern makes reducing map not merely a benefit, but a necessity.

4 Methodology

We'll now briefly describe the methodology used to build this reduced sized map ([1], [2])
General paradigm: Firstly, the conformations/nodes are sampled from the protein's energy landscape as can be seen in fig 3(b), and then edges are drawn between the nearby conformations (the ones close in energies) as can be seen in fig 3(c). This strategy favors lower energy conformations and transitions because during sampling, lower energy samples are retained with a higher probability, and during the node connection phase, edges are assigned weight based on their energetic feasibility, that is lower the edge weight for an edge (i,j), greater is the transition probability from i to j.

4.1 Sampling of Nodes

Since the dimensionality of the conformation space is very high therefore uniform sampling would have to be highly dense in order to cover the conformation space effectively. This would be computationally very expensive. However, the fact that the native state of the protein is known apriori can be taken advantage of! This is what our method does precisely. It selects conformations biased around the native state, that is the goal state.



Flowchart 1

Native state in pdb format:

Native states of all the proteins are available in their pdb formats from online protein data banks. We used the pdb file of 1GB1 native state protein. A pdb file contains the x, y, z coordinates of the atoms of all the amino acids in a protein conformation. Following is the snapshot of the pdb file we used.

1GB1.pdb

```
ATOM      1  N   MET A   1      -7.173   9.686   8.824
ATOM      2  CA  MET A   1      -7.670   8.280   8.824
ATOM      3  C   MET A   1      -6.504   7.328   8.549
ATOM      4  O   MET A   1      -5.365   7.743   8.470
ATOM      5  CB  MET A   1      -8.749   8.089   7.758
ATOM      6  CG  MET A   1      -9.694   9.292   7.777
ATOM      7  SD  MET A   1      -9.271  10.698   6.720
ATOM      8  CE  MET A   1     -10.391  11.898   7.482
ATOM      9 1H   MET A   1      -6.134   9.686   8.824
ATOM     10 2H   MET A   1      -7.519  10.177   7.974
ATOM     11 3H   MET A   1      -7.520  10.176   9.673
ATOM     12  N   THR A   2      -6.815   6.067   8.419
ATOM     13  CA  THR A   2      -5.744   5.068   8.139
ATOM     14  C   THR A   2      -6.263   4.014   7.155
ATOM     15  O   THR A   2      -7.288   3.404   7.385
ATOM     16  CB  THR A   2      -5.340   4.389   9.449
ATOM     17  OG1 THR A   2      -5.107   5.462  10.358
ATOM     18  CG2 THR A   2      -4.012   3.651   9.314
ATOM     19  H   THR A   2      -7.747   5.778   8.506
ATOM     20  HG1 THR A   2      -4.596   6.134   9.899
ATOM     21  N   TYR A   3      -5.557   3.843   6.066
ATOM     22  CA  TYR A   3      -5.991   2.819   5.061
ATOM     23  C   TYR A   3      -5.147   1.548   5.211
ATOM     24  O   TYR A   3      -4.054   1.589   5.740
ATOM     25  CB  TYR A   3      -5.782   3.375   3.651
ATOM     26  CG  TYR A   3      -6.191   4.851   3.623
ATOM     27  CD1 TYR A   3      -5.349   5.823   4.127
ATOM     28  CD2 TYR A   3      -7.396   5.232   3.072
ATOM     29  CE1 TYR A   3      -5.721   7.152   4.101
ATOM     30  CE2 TYR A   3      -7.765   6.562   3.044
ATOM     31  CZ  TYR A   3      -6.934   7.530   3.565
ATOM     32  OH  TYR A   3      -7.304   8.860   3.539
ATOM     33  H   TYR A   3      -4.734   4.361   5.923
ATOM     34  HH  TYR A   3      -8.181   8.916   3.151
ATOM     35  N   LYS A   4      -5.681   0.441   4.756
ATOM     36  CA  LYS A   4      -4.915  -0.844   4.852
```

Calculation of Dihedral angles:

We next used a python pdb tool [**4a**] to generate the dihedral angles for the native state using it's pdb file. Following is the snapshot of the file with the phi, psi angles.

angles.dat

0	1GB1 MET "A 1"	-141.66	124.80
1	1GB1 THR "A 2"	-100.97	157.15
2	1GB1 TYR "A 3"	-104.10	124.95
3	1GB1 LYS "A 4"	-89.82	150.40
4	1GB1 LEU "A 5"	-127.73	101.86
5	1GB1 ILE "A 6"	-79.76	114.35
6	1GB1 LEU "A 7"	-104.68	38.89
7	1GB1 ASN "A 8"	-58.26	140.48
8	1GB1 GLY "A 9"	-70.01	-27.92
9	1GB1 LYS "A 10"	-104.84	-7.02
10	1GB1 THR "A 11"	-163.02	126.37
11	1GB1 LEU "A 12"	-116.59	136.93
12	1GB1 LYS "A 13"	-159.08	-162.21
13	1GB1 GLY "A 14"	-176.17	131.15
14	1GB1 GLU "A 15"	-146.96	154.21
15	1GB1 THR "A 16"	-104.60	138.[illegible]
16	1GB1 THR "A 17"	-127.25	145.[illegible]
17	1GB1 THR "A 18"	-88.34	122.73
18	1GB1 GLU "A 19"	-139.99	153.13
19	1GB1 ALA "A 20"	-88.23	-27.94
20	1GB1 VAL "A 21"	-142.72	177.56
21	1GB1 ASP "A 22"	-76.33	-40.65
22	1GB1 ALA "A 23"	-53.58	-53.[illegible]
23	1GB1 ALA "A 24"	-65.47	-26.72
24	1GB1 THR "A 25"	-70.22	-38.47
25	1GB1 ALA "A 26"	-61.08	-37.45
26	1GB1 GLU "A 27"	-81.21	-41.33
27	1GB1 LYS "A 28"	-55.95	-55.68
28	1GB1 VAL "A 29"	-63.75	-44.55
29	1GB1 PHE "A 30"	-53.85	-37.26
30	1GB1 LYS "A 31"	-66.55	-27.65
31	1GB1 GLN "A 32"	-79.35	-54.61
32	1GB1 TYR "A 33"	-49.98	-29.67
33	1GB1 ALA "A 34"	-65.77	-33.26
34	1GB1 ASN "A 35"	-70.03	5.40
35	1GB1 ASP "A 36"	-130.79	-26.20

Biased Gaussian Sampling:

We generated a set of normal distributions around the native state and sampled from these distributions. The set of standard deviations (STDs) we used was 5 , 10 , 20 , 40 , 80 , 160 . The selection of such a set of STDs is so as to capture the details around the goal (done by the smaller STDs), and also ensure proper roadmap coverage of the conformation space of the protein (done by the large STDs). The sampling is biased because greater number of selections will be around the native state while higher energy states away from the goal will be sparser in number (owing to the set of STDs). In order to simplify sampling a bit, instead of taking a n-dimensional gaussian around the native state, n being the number of residues (amino acids in the protein chain), we built 1-dimensional gaussian for each amino acid of the native state, for all the STDs in the set. This way we were able to generate around 6000 nodes on the map.

Energy calculation for each node:

Next we generated the pdb files for each of these nodes using their dihedral angles. We did this with the help of a molecular package, Crankite[5b]. It has a pre-built binary (lipa) which generated pdb files from the given dihedral angles. Then we calculated the energies for all the nodes using their pdb files. Following is the energy function we used:

$$\mathrm{E(c)} = \begin{cases} 2 * E_{max}^{con} & \text{if } r_{min} < 1.0\dot{A} \\ 2 * E_{max}^{gen} & \text{if } 1.0\dot{A} < r_{min} < 2.4\dot{A} \\ \sum_{rest} K_d\{[(d_i - d_0)^2 + {d_c}^2]^{\frac{1}{2}} - d_c\} + E_{hydro} & \text{o/w} \end{cases}$$

d_i = separation b/w atoms forming hydrogen or disulphide bond
$d_0 = d_c = 2.0\dot{A}$

This function uses a step function approximation of the van der Waals component of energy. It only considers the contributions due to the repulsions from the side chains (the carbon chains located at the C_{beta} atom of every amino acid). Each side chain is treated as a single large R atom and r_{min} is the minimum distance between any two R atoms in the protein conformation. Whenever r_{min} is less than $1\dot{A}$ during node connection or less than $2.4\dot{A}$ during node generation, this function returns a very high potential. K_d is 100KJ/Mol, $d_o = d_c = 2\dot{A}$, and d_i is the separation between those H and O atoms that form hydrogen bonds. We took the approximation that the O and H atoms at a distance less than $4\dot{A}$ form strong Hydrogen bonds and have thus considered the reduction in energy only due to them. E_{hydro} is the increase in energy by 20KJ/mol whenever the R atoms of any two hydrophobic amino acids (any two non polar amino acids are hydrophobic) come within a distance of $6\dot{A}$.

Filtering of nodes:

Finally, nodes were filtered based on their energies. The ones with lower energies had a greater retention probability. Following is the probability function we used:

$$\text{P(accept q)} = \begin{cases} 1 & \text{if } E(q) < E_{min} \\ \frac{E_{max} - E(q)}{E_{max} - E_{min}} & \text{if } E_{min} \leq E(q) \leq E_{max} \\ 0 & \text{if } E_{max} < E(q) \end{cases}$$

P(accept q) is the probability to retain a node q. E(q) is it's energy. E_{min} is the energy of open chain protein conformation while E_{max} is twice of E_{min}. After filtering we were left with about 4500 nodes.

4.2 Connection of Nodes

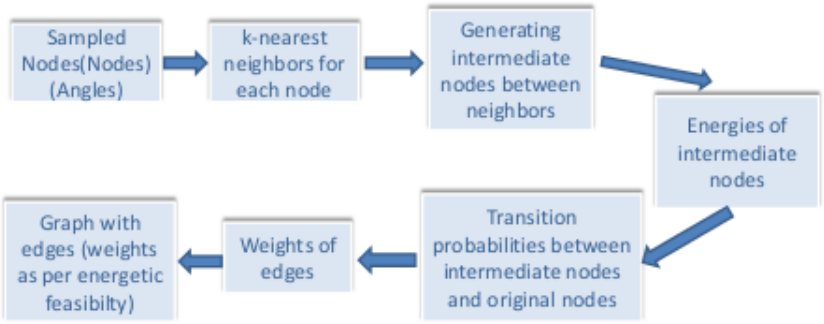
Once, we obtain the sampled nodes, next step is to connect the nodes with appropriate weights.
The steps in node connection are:

a) Finding k-nearest neighbours for each sampled node:

For each node, that is selected, k-nearest neighbours are found based on distances dependent on phi and psi as follows:
$\text{dist}(n_1,n_2) = \sqrt{[\sum(\phi_{n1_i} - \phi_{n2_i})^2 + \sum(\psi_{n1_i} - \psi_{n2_i})^2]}$
The basic idea is that the k nearest neighbours contain the neighbour from the most feasible path.

b) Generating intermediate nodes between neighbour nodes:

The intermediate nodes are generated to get the representation of the path from one node to the other. They are equally spaced on the straight line connecting the two nodes and are linear in terms of phi and psi parameters.



flowchart 2

c) Energies of the intermediate nodes:

The energies of the intermediate nodes are found out from their calculated dihedral angles.

d) Weights of the intermediate nodes

The weights are assigned to the edges as follows:
If their is fall in energy from one node (sampled as well as intermediate) to other, then no contribution is made to the weight.
If there is increase in the energy then contribution to weight is proportional to the change in energy as per the following formula:-

$P_i = e^{-\triangle \frac{E_i}{kT}}$ if $\triangle_i > 0$
1 if $\triangle_i \leq 0$

where

$\triangle E_i = E(c_{i+1})$ - $E(c_i)$

wt(q1,q2) = $\sum_0^{n-1} -log(P_i)$

4.3 Querying the Roadmap:

Once the map (V, E) is generated, we next query the roadmap to find the most suitable pathway among the thousands encoded.

Dijkstra's Algorithm Approach:

Selecting the shortest path, that is, the path with least weight edges will give us the most energetically feasible path since least weight edge (i,j) from i, represent the most energetically feasible transition from i. However, it's a deterministic approach, that is, it will generate the same least weighted pathway. But protein folding is a stochastic process. A protein might fold from a high energy conformation to it's native state in a non monotonic fashion, that is, it might not go to a lower energy conformation always, their might be high energy peaks in between.
Also, choosing the least weight edge always, might get the folding process stuck in some local minima and it may never reach the native state. Therefore, we need some kind of a stochastic algorithm to query the graph.

Monte-Carlo Simulation:

It extracts the path randomly based on the transition probabilities. Each edge(i,j) is assigned a transition probability which varies inversely as it's weight. Thus probability to transition from i to j will be higher if it's a lower weight edge, however this doesn't totally discard the possibility to transition to a higher energy node. The process starts from any random node and keep selecting nodes iteratively based on the transition probabilities unless the native state is reached.

5 Results

Sampling:

1. Generated the dihedral angles from the pdb file of 1GB1 protein (native state)
2. Sampled about 6000 nodes (conformations) from Gaussian distributions around the dihedral angles of the native state
3. Generated pdb files for each of these conformations from their dihedral angles
4. Calculated energies from these pdb files for all the nodes

5. Filtered the nodes based on their energies

For 1: we used a python pdb tool[**5a**]
For 3: we used a molecular package, Crankite[**5b**]
For 2, 4 and 5: we wrote the code

Connection:

1. We connected every node with it's 5 nearest neighbors using 5 intermediate nodes per connection. Our local planner uses straight line interpolation.
2. Calculated the weights for every edge connecting a node with it's 5 nearest neighbors.
3. Generated adjacency list containing all the nodes with their connections.

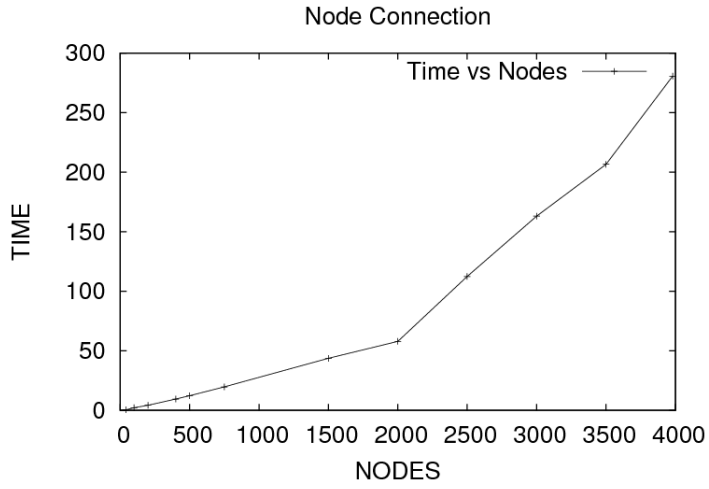
Code for connection was completely written by us.

Statistics:

1. Avg time for node generation: 5 min approx. for about 4500 nodes
2. Time for node connection(for different number of sampled nodes):
   (for k = 5 neighbour nodes and l = 5 intermediate nodes)

NODES	TIME
41	32sec
100	2min 16sec
200	4min 18sec
400	9min 32sec
500	12min 20sec
750	19min 46sec
1500	43min 45sec
2000	57min 55sec
2500	112min 25sec
3000	163min 03sec
3500	206min 33sec
3980	280min 32sec

The above computations have been done on Intel(R) Core(TM) i7 CPU , 930@2.80GHz hardware.
Thus, 3980 nodes could be connected in 280 minutes.



6 Scope of Improvement:

Use of rigidity analysis in node sampling:

In our currently employed strategy we make perturbations in the phi, psi angles of all the amino acids of the protein. However a protein can be classified to have the following three parts: flexible, slightly flexible and rigid. Thus based on the rigidity factor, not all the amino acids of a protein might undergo deviations. We could use method of rigidity analysis to find out the non-rigid amino acids and exclude them from perturbations. This could fine tune the sampling process.

Better strategies for node connection:

We have used many approximations while connecting nodes. For example our distance function is the sum of squares of differences between the phi and psi vectors of two nodes. We could come up with a better distance function which could also take into account the energy differences. Likewise, we can also think of a better way of generating intermediate nodes.

Map analysis:

We could use a prebuilt monte carlo tool to analyse our map and generate the appropriate pathways. We could also generate the energy v/s conformation plots for the protein folding process through various pathways and project our results more vividly.

Analysis of mutations:

We could analyse more proteins and compare their results. We could analyse the pathways of Proteins NuG1 and NuG2, which are mutants of 1GB1 and compare the differences in their folding behaviours.

7 Conclusion

We succeeded in generating the probabilistic road map for 1GB1 protein. The reduction in map size employed by our strategy drastically reduced the computational time of roadmap generation, for a protein with 56 amino acids.

References

[**1**] A Motion Planning Approach to Studying Molecular Motions, Lydia Tapia, Shawna Thomas, Nancy M. Amato, Communications in Information and Systems, 10(1):53-68, 2010. Also, Technical Report, TR08-006, Parasol Laboratory, Department of Computer Science, Texas A&M University, Nov 2008.
[**2**] Intelligent Motion Planning and Analysis with Probabilistic Roadmap Methods for the Study of Complex and High-Dimensional Motions, Lydia Tapia, Ph.D. Thesis, Parasol Laboratory, Department of Computer Science, Texas A&M University, College Station, Texas, Dec 2009.
[**3**] Using Motion Planning to Study Protein Folding Pathways, Guang Song, Nancy M. Amato, Journal of Computational Biology, 9(2):149-168, Nov 2002. Also, In Proc. Int. Conf. Comput. Molecular Biology (RECOMB), pp. 287-296, Apr 2001. Also, Technical Report, TR00-026, Parasol Laboratory, Department of Computer Science, Texas A&M University, Oct 2000.
[**4**] Image Source:
https://parasol-www.cse.tamu.edu/groups/amatogroup/foldingserver/
[**5**] Code Sources:
a) http://code.google.com/p/pdb-tools/
b) Crankite program suite by Alexei Podtelezhnikov, Ph.D., Computational Biochemist, Biophysicist, and Bioinformatician
https://sites.google.com/site/crankite/