A Multi-Directional Rapidly Exploring Random Graph (mRRG) for Protein Folding*

Shuvra Kanti Nath, Shawna Thomas, Chinwe Ekenna, and Nancy M. Amato†
Parasol Lab, Department of Computer Science and Engineering
Texas A&M University
College Station, TX 77843-3112
{shuvra03,sthomas,cekenna,amato}@cse.tamu.edu

ABSTRACT

Modeling large-scale protein motions, such as those involved in folding and binding interactions, is crucial to better understanding not only how proteins move and interact with other molecules but also how proteins misfold, thus causing many devastating diseases. Robotic motion planning algorithms, such as Rapidly Exploring Random Trees (RRTs), have been successful in simulating protein folding pathways. Here, we propose a new *multi-directional* Rapidly Exploring Random Graph (mRRG) specifically tailored for proteins.

Unlike traditional RRGs which only expand a parent conformation in a single direction, our strategy expands the parent conformation in multiple directions to generate new samples. Resulting samples are connected to the parent conformation and its nearest neighbors. By leveraging multiple directions, mRRG can model the protein motion landscape with reduced computational time compared to several other robotics-based methods for small to moderate-sized proteins. Our results on several proteins agree with experimental hydrogen out-exchange, pulse-labeling, and Φ-value analysis. We also show that mRRG covers the conformation space better as compared to the other computation methods.

Categories and Subject Descriptors

J.3 [**Computer Applications**]: Life and Medical Sciences—*Biology and Genetics*; I.6.3 [**Computing Methodologies**]: Simulation and Modeling—*Applications*

1. INTRODUCTION

Protein folding [27, 20] is the biochemical process by which the protein structure folds into its functional three dimensional native structure. During folding, a protein transitions from the unfolded state to the folded state passing through a sequence of intermediate conformations. Understanding the folding process is of great importance as the protein's functions are related to it. It also helps in understanding why some proteins misfold [17] which results in devastating diseases such as Alzheimer's and Mad Cow.

We assume that the native structure of the protein is known, and our main focus is modeling the folding process to this native structure. There have been both experimental and computational methods developed to study protein motions. Numerous experimental methods such as circular dichroism, fluorescence experiments [28], hydrogen exchange [37], pulse labeling, and NMR spectroscopy [24] have been developed to model the folding process. Experimental methods are not only complex and expensive, it is also difficult to observe the fast moving folding process using these methods. Thus, computational simulation techniques that model this process in a realistic way are needed to study these process. Not only can computational techniques help elucidate the folding process, they can also provide guidance for future experiments. Many traditional simulation methods such as molecular dynamics [19, 7, 10], Monte Carlo methods [6, 16], and simulated annealing [21] provided a single, detailed, high-quality folding pathway but at a large computational expense. As such, they cannot be practically used to study global properties of the folding landscape or produce multiple folding pathways. Statistical mechanical models [25, 1, 23, 8, 11] do provide global information about the folding process, but they cannot be used to produce individual folding pathways. Lattice models [5] are computationally efficient but are theoretical models and not used on real proteins. Robotic motion planning algorithms such as the Probabilistic Roadmap Method (PRM) [15] and Rapidly Exploring Random Tree (RRT) [18] have been applied to model the folding process. These robotics-based methods are quite promising as they can generate multiple folding pathways in a short amount of time (e.g., a few hours on a desktop PC). This enables the study of both individual folding trajectories and global properties of the landscape.

We introduce a new algorithm based on Rapidly Exploring Random Graphs (RRGs) [14] called *multi-directional* RRG (mRRG) for modeling protein folding. RRG is a path planning method which iteratively constructs a graph data structure that encodes representative motions of the object under study. The graph is built by iteratively expanding an existing sample in the graph in a random direction. If the extension is successful, RRG attempts to connect the new

†This work is supported in part by NSF awards CRI-0551685, CCF-0833199, CCF-0830753, IIS-096053, IIS-0917266 by THECB NHARP award 000512-0097-2009, by Chevron, IBM, Intel, Oracle/Sun and by Award KUS-C1-016-04, made by King Abdullah University of Science and Technology (KAUST).

ACM-BCB'12, October 7-10, 2012, Orlando, FL, USA

Approach	Global Landscape Properties	# Paths Produced	Path Quality	Computational Time Required	Native Needed
Trajectory-Based	Poor Coverage	1	Good	Long	No
Statistical Mechanical Model	Good Coverage	0	N/A	Short	Yes
Robotics-Based					
PRM	Good Coverage	Many	Approx	Short	Yes
T-RRT	Poor Coverage	Many	Approx	Very Short	Yes
mRRG	Good Coverage	Many	Approx	Very Short	Yes
Lattice Model	*Not used on real proteins*				

Table 1: Comparison of computation models for modeling protein motion.

sample to its k-nearest neighbors in the graph. mRRG differs from RRG by making m expansion steps from q_{near} instead of a single expansion step. Extension in multiple directions allows a single iteration to explore more of conformation space than RRG [14]. We present a case study for protein G and its variant NuG1, protein L, and protein A. We validate our pathways by comparing the secondary structure formation order with known experimental results. Examining the node distributions of each method, we show that by increasing the number of directions m, we can improve the exploration of the folding landscape. We compare mRRG to T-RRT [13] and rigidity sampling based PRM [35] and show that the exploration with multiple directions gives better coverage than T-RRT and is comparable to PRM for all of the proteins studied. The computational requirements are also comparable to T-RRT. We also examine properties of the resulting folding pathways including path weight (i.e., energetic feasibility) and smoothness.

2. RELATED WORK

Different computational methods have been developed to study protein motion and folding, see Table 1. For each method, the table provides an overview of its ability to study global properties of the folding landscape, how many trajectories it produces and the quality of these trajectories. It also summarizes each method's computational requirements and dependence on knowing the native state.

Molecular dynamics [19, 7, 10] simulates the forces on all the atoms at each timestep to produce a motion trajectory. Monte Carlo methods [6, 16] perform a random walk on the protein's energy landscape that favors lower energy transitions. Replica exchange for both Monte Carlo methods [34, 12] and molecular dynamics [33] simulate many copies of the protein, all at different temperatures, and periodically exchange copies from different temperatures to allow greater access to both high and low temperature states. Simulated annealing [21] is similar except that it periodically increases the simulation temperature to allow the protein to move out of local minima. These trajectory-based methods are computationally intensive as they use complex kinetics and thermodynamics to simulate a single, high-resolution pathway. While distributed computing methods can reduce the computational expense [38], these trajectory-based methods cannot be used to simulate many pathways and thus study global information about the folding landscape. Statistical mechanical models [25, 1, 23, 8, 11] compute statistics about the global energy landscape and use them to infer ensemble properties of the folding process. They are not designed to produce individual trajectories. While computationally efficient, they can only be used to study global averages of the landscape. Lattice models [5] are well studied but cannot be applied to actual proteins.

2.1 Motion Planning Approaches

In the past decade, a number of efforts have focused on adapting robotic motion planning algorithms to model the protein folding process. These robotics-based methods can generate multiple folding pathways in a short amount of time (e.g., a few hours on a desktop PC). Such efficiency enables the study of both individual folding trajectories and global properties of the overall folding landscape. In this section, we discuss how these methods can be applied to model molecular motions and then review the particular approaches most relevant to this work.

The motion planning problem is to find a valid path for a movable object from a start placement to a goal placement. Probabilistic Roadmap Methods (PRMs) [15] and Rapidly Exploring Random Trees (RRTs) [18] are two popular classes of motion planning algorithms that have been highly successful in solving challenging high dimensional motion planning problems. These methods construct a graph or tree data structure that models the motion space of the movable object (C-space) which can be queried to find trajectories in the graph connecting start and goal configurations of the movable object.

By simply changing the model of the movable object from a robot to a protein and the definition of feasibility from collision-free to low energy, these same algorithms may be applied to model protein motion [30, 13]. The resulting graph or tree data structure then models the protein's energy landscape, the set of all protein conformations and their associated energies. In most cases, the energy landscape is thought to be funnel-shaped with the native, folded state corresponding to a conformation of minimal energy at the bottom of the funnel [4, 5, 9]. Every protein has a unique energy landscape that influences how the protein moves and folds. Connections in the graph or tree data structure are weighted based on their energetic feasibility as determined by the energies of the intermediate nodes along the connection. Low edge weights correspond to energetically favorable transitions so that standard graph search algorithms for shortest paths can be used to extract approximate folding pathways from an unfolded state to the native state.

2.1.1 RRG

Rapidly Exploring Random Graphs (RRGs) [14] iteratively explore the C-space by expanding existing samples towards unexplored areas. They attempt k connections from new samples resulting in a graph. RRGs extend Rapidly Exploring Random Trees (RRTs) [18], one of the early sampling-based motion planning methods. In particular, an RRT is an RRG with $k = 1$. Given an initial object placement, RRG expands the graph by first generating a new sample at random, q_{rand}. Then, the nearest sample q_{near} in the graph to q_{rand} is selected for expansion. q_{near} is expanded by walking toward q_{rand} in C-space until an invalid placement is found or a maximum distance, δ_{max}, is reached. The new valid sample q_{new} is added to the graph. Then k connections are attempted between q_{new} and its k-closest neighbors in the graph. Connections (edges) are added if all the intermediate samples along the connection are valid. The process repeats until the graph satisfies a set of constraints such as having a minimum number of nodes or containing a path between a certain start and goal. Since RRGs always select the nearest neighbor to q_{rand} for expansion, growth is biased towards large Voronoi regions, or unexplored regions of C-space. Algorithm 1 outlines the approach where Extend is the walk from q_{near} towards q_{rand} that terminates when an invalid sample is reached or the distance exceeds the maximum.

Algorithm 1 RRG

Input. An initial placement q_{init}, a minimum and maximum distance δ_{min} and δ_{max}, a number of nearest neighbors k, and an evaluator E.
Output. A graph G rooted at q_{init} that satisfies E.

```
1: G.AddVertex(q_init).
2: while G does not satisfy E do
3:    Let q_rand be a random sample, valid or not.
4:    Let q_near be the nearest sample ∈ G to q_rand.
5:    q_new = Extend(q_near, q_rand, δ_min, δ_max).
6:    G.AddVertex(q_new).
7:    Let K be the k-nearest neighbors ∈ G to q_new.
8:    for each q ∈ K do
9:       if the edge (q_new, q) is valid then
10:         G.AddEdge(q_new, q).
11:      end if
12:   end for
13: end while
```

Notice that graph extension toward similar states already present in the graph does not aid in modeling unexplored regions. To avoid this, q_{new} is not extended if the distance between q_{new} and q_{rand} is less than δ_{min}. This checking is done in the Extend call.

2.1.2 T-RRT

As mentioned in Section 2.1.1, an RRG is an RRT with $k = 1$. In [13], RRT was adapted and applied to model folding pathways for small molecules. The T-RRT algorithm uses a self-tuning strategy that adjusts the simulation temperature T to bias towards unexplored regions and also towards energetically favorable regions. T-RRT differs from standard RRT in that it only adds a new sample with the following probability P_{ij} called the *transition test*:

$$P_{ij} = \begin{cases} e^{\frac{-\Delta E_{ij}}{kT}} & \text{if } \Delta E_{ij} > 0 \\ 1 & \text{if } \Delta E_{ij} \leq 0 \end{cases}$$

where $\Delta E_i = E_j - E_i$ is the energy difference between two nodes, k is the Boltzmann constant, and T is the temperature. Unlike RRG, it only connects a node to its single nearest neighbor. T-RRT was shown to be efficient in finding energy minima and transition paths between them.

2.1.3 PRM

Probabilistic Roadmap Methods (PRMs) [15] also build a graph data structure to model the motion space, or C-space. PRMs first generate a set of random samples, adding valid ones to the graph. Then, for each node in the graph, connections are attempted between it and its nearest neighbors. Like RRG, connections are only added if every intermediate sample along the connection is valid. PRMs were the first robotics-based algorithms applied to model protein folding [30]. The application to proteins was further refined in [36] to use rigidity information to sample conformations in a more physically realistic way.

2.2 Protein Model

All the methods described in Section 2.1 are general in that they support any protein model and energy function. In previous work, PRMs have been applied to backbone models using both coarse-grained and all-atoms energy functions [31]. For the results here, we use the following coarse-grained model that has been shown to work well previously.

A protein is modeled as a sequence of amino acids. For each amino acid, we model the ϕ and ψ backbone torsional angles as flexible and keep all other bond lengths and angles fixed. This is a standard modeling assumption [32]. Thus, the protein is modeled as an articulated linkage, where the flexible atomic bonds are joints ranging between $[0, 2\pi)$. Note that we do not restrict the values of the backbone torsional angles (e.g., to Ramachandran angles) so as to capture both the folded *and* unfolded regions of the landscape. Instead, we allow the energy function to dictate what resulting conformations are feasible or not.

We use a potential energy function to determine the validity of a given protein conformation. The results in this paper use a coarse energy function from [2] which includes standard van der Waals interactions, hydrogen bonding, and electrostatic interactions [19]. If the side chains are too close (less than 2.4Å during sampling and 1.0Å when connecting), the conformation is rejected. Otherwise, the energy is:

$$U_{tot} = \sum_{constraints} K_d\{[(d_i - d_0)^2 + d_c^2]^{1/2} - d_c\} + E_{hp}$$

where K_d is 100 kJ/mol, d_i is the length on the ith constraint, and $d_0 = d_c = 2$Å as shown in [19].

All the robotics-based methods compared use connect conformations with straight-line interpolations. The weight of an edge is a function of the intermediate conformations along the edge. For each pair of consecutive conformations, the probability of transitioning between them is given by the *transition test* (see Section 2.1.2). The weight is then the sum of the negative logarithms of the probabilities of consecutive conformations along the edge. A similar weight function, with different probabilities, was used in [29].

3. mRRG METHOD

T-RRT is efficient at finding a single pathway quickly but only expands a parent node in a single direction, namely towards q_{rand}, causing it to focus on a small number of routes in the energy landscape funnel [3]. PRM is slower but covers the energy landscape better. For example, Figure 1 compares the node distribution for T-RRT and PRM graphs with 250 nodes for protein G, a small 56-residue $\alpha\beta$ protein. Each node's energy is plotted against its Euclidean distance to the native state. The PRM distribution is broader and extends further into the unfolded region of the energy landscape (i.e., larger Euclidean distance and energy values) than T-RRT.

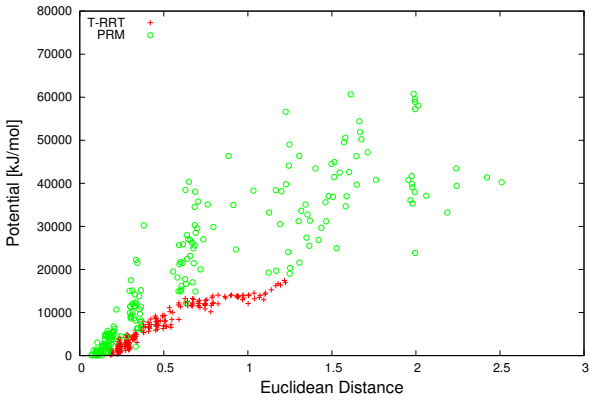


Figure 1: Node distribution comparison of 250 samples from T-RRT and PRM for protein G. PRM yields a broader distribution that extends further into unfolded areas.

In this paper we propose a novel extension of RRG that combines the efficiency of a T-RRT-style search with the breadth of a PRM distribution. *Multi-directional* Rapidly Exploring Random Graph (mRRG) augments the RRG algorithm by expanding the parent node in multiple directions at each expansion step instead of a single bias towards q_{rand}. Expansion in multiple directions yields more expansive graphs that can help broaden the area of the explored folding landscape. Figure 2 sketches the approach and the algorithm is provided in Algorithm 2.

The only additional input for mRRG compared to RRG is the number of expansion directions m. To model protein folding, we set q_{init} to be the known native state. In each iteration, a random configuration q_{rand} is generated (valid or not). The nearest node q_{near} to q_{rand} is selected for expansion and extended towards q_{rand} to create q_{new}. During expansion, each of the intermediate nodes between q_{near} and q_{new} is checked for energy validity. If the transition test is passed, q_{near} is connected to multiple neighbors using $ConnNeighbors$ function. In our case, we chose Brute Force neighborhood finder(Line 1) to choose k-nearest neighbors from q_{new}. After the connection, mRRG selects $m-1$ additional random directions to expand q_{near} towards. Just as with the first direction, q_{near} is extended toward each of these new $m-1$ random directions to create $m-1$ new samples. Then, like T-RRT, each q_{new} is added to the graph if it passes the transition test. Similar to RRG, it is then connected to its k-nearest neighbors. Figure 2 shows a single iteration with $m = 5$ and $k = 2$.

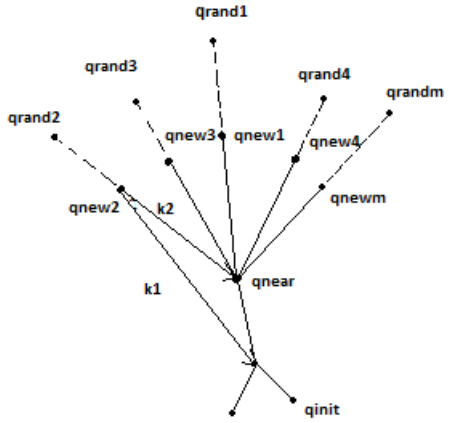


Figure 2: An example of mRRG. Instead of a single q_{rand} for expansion, m random samples are generated to guide expansion from q_{near}. Along each direction, a q_{new} is generated and connected to its k-nearest neighbors.

The method continues expanding the graph until it passes a set of evaluation criteria. Since we are interested in studying protein folding pathways, we stop construction once the secondary structure formation ordering (i.e., the order in which the various α-helices and β-sheets form) along the pathways does not vary between iterations by more than some threshold. This is the same evaluation scheme used previously in applying PRMs to study protein folding [30]. For the results presented here, we evaluate the secondary structure formation ordering after every 250 samples.

4. RESULTS AND DISCUSSION

Here we study the performance of mRRG under different input parameters and compare against T-RRT [13] and PRM [35]. For each method, we construct a graph rooted at the native state. We evaluate the graph after every 250 samples. Construction stops when the secondary structure formation ordering along the folding pathways in the graph stabilizes, i.e., the percentage of pathways following a given ordering does not vary between successive graphs by more than 10%. This is the same evaluation criteria used in [30].

We validate a method's results by comparing its dominant secondary structure formation ordering to the experimentally determined ordering from hydrogen out-exchange, pulse-labeling data, and/or Φ-value analysis. We compare the sample distribution of a graph by looking at its potential vs. native state Euclidean distance plot. We also examine individual folding pathways in the same manner. All methods were implemented using the C++ motion planning library developed by the Parasol Lab of Texas A&M University.

4.1 Proteins Studied

We study the proteins in Table 2. Proteins G, L, and NuG1 are $\alpha\beta$ mixed proteins that while structurally similar, are known to fold differently. The second β-hairpin forms early in protein G but forms late in proteins L and NuG1, a variant of protein G. Protein A is an all α protein of similar size.

Protein	pdb	# Residues	Secondary Structure Makeup	Experimental Formation Order
G	1PGA	56	$1\alpha + 4\beta$	[α,β1,β3,β4],β2[1] [α,β4],[β1,β2,β3][2]
G Variant	NuG1	56	$1\alpha + 4\beta$	β1-2,β3-4[3]
A	1BDD	60	3α	[α2,α3],α1[1] [α1,α2,α3][2]
L	2PTL	62	$1\alpha + 4\beta$	[α,β1,β2,β4],β3[1] [α,β1],[β2,β3,β4][2]

Table 2: Proteins studied and their secondary structure formation order from: [1]hydrogen out-exchange experiments [22], [2]pulsed labeling/competition experiments [22], and [3]Φ-value analysis [26]. Brackets indicate no clear order.

Algorithm 2 mRRG for Proteins

Input. An initial placement q_{init}, a minimum and maximum distance δ_{min} and δ_{max}, a number of expansion directions m, a number of nearest neighbors k, and an evaluator E.

Output. A graph G rooted at q_{init} that satisfies E.

$G.AddVertex(q_{init})$.
while G does not satisfy E **do**
  Let q_{rand} be a random sample, valid or not.
  Let q_{near} be the nearest sample $\in G$ to q_{rand}.
  q_{new} = Extend($q_{near}, q_{rand}, \delta_{min}, \delta_{max}$).
  if TransitionTest(q_{new}, q_{near}) **then**
    $G.AddVertex(q_{new})$.
    $G.AddEdge(q_{new}, q_{near})$.
    $ConnNeighbors(q_{new}, nf, lp, G)$
    for $i = 2 \dots m$ **do**
      Let q_{rand} be a random sample, valid or not.
      q_{new} = Extend($q_{near}, q_{rand}, \delta_{min}, \delta_{max}$).
      if TransitionTest(q_{new}, q_{near}) **then**
        $G.AddVertex(q_{new})$.
        $G.AddEdge(q_{new}, q_{near})$.
        $ConnNeighbors(q_{new}, nf, lp, G)$
      end if
    end for
  end if
end while

4.2 Varying m

Here we examine how mRRG performs with differing values of the number of expansion directions m. To isolate out the affect of m, we set $k = 1$.

4.2.1 Case Study on Protein G

Graph Quality. Table 3 compares the running time and resulting graph size for each method for protein G at $m = \{1, 3, 5, 7\}$. Recall that T-RRT is the same as mRRG with $m = 1$. Every method was able to reproduce the correct secondary structure formation order. We see that as m increases, a larger graph is needed before the secondary structure formation order stabilizes. This is due to the fact that larger m values yield bushier graphs which take longer (i.e., more samples) to explore into the unfolded regions. Note that mRRG is able to create samples much faster than PRM: mRRG with $m = 5$ takes much less time than PRM, yet mRRG generates same number of samples.

Algorithm 3 ConnNeighbors

Input. Connecting node q, neighbor finder nf, local planner lp, graph G.

$N = nf.FindNeighbors(q, G)$
for each $n \neq q \in N$ **do**
  if $lp.IsConnectable(q, n)$ **then**
    $G.AddEdge(q, n)$
  end if
end for

Method	m	Time (hr)	# Nodes	# Edges	SSFO Comparison
T-RRT	1	0.157	750	1498	Agreed
mRRG	3	0.094	1250	2500	Agreed
	5	0.063	1000	2000	Agreed
	7	0.417	1500	3000	Agreed
PRM	n/a	0.680	1000	18928	Agreed

Table 3: Method comparison for protein G with different values of m. All methods produced folding pathways that agreed with the experimentally determined secondary structure formation order (SSFO).

Figure 3 and Figure 1 show the node distribution in terms of potential energy vs. Euclidean distance to the native state after 250 samples are created. PRM has the densest distribution near the native state but covers the unfolded regions, albeit sparsely. T-RRT and mRRG have a more even distribution of samples, where increase of m increases the bushiness of the graphs as it searches in more directions.

Figure 4 shows the same distribution plots but for the graph for which secondary structure formation ordering is stabilized. mRRG with $m = 5$ has the most even coverage of the folded, partially folded, and unfolded regions. In addition, mRRG was able to explore the furthest into the unfolded region of the energy landscape. T-RRT was unable to reach a large portion of the unfolded region.

Path Quality. Figure 5 compares the dominant folding pathway as determined by each method for protein G. When m increases, again we see that the pathway contains more unfolded nodes, i.e., is able to reach further up the energy landscape. For example, the Euclidean distance to the native state from the most unfolded conformation for T-RRT is 1.4 while for mRRG, $m = 7$ is 1.8. We also see that mRRG is able to produce smoother pathways than PRM which contains large jumps across the plot. Note that the start of PRM's folding pathway is not the most unfolded conformation in the pathway as one would expect.

4.2.2 Results for Proteins NuG1, A and L

Table 4 shows the performance of T-RRT, mRRG with $m = 5$, and PRM for the remaining proteins. While mRRG needs more time to generate a stable graph, it generates many more nodes with a more even distribution on the landscape.

Method	m	Time (hr)	# Nodes	# Edges	SSFO Comp.
Protein G Variant					
T-RRT	1	0.025	500	1000	Disagreed
mRRG	5	0.043	1000	2000	Agreed
PRM	n/a	0.297	499	7742	Agreed
Protein A					
T-RRT	1	0.037	750	1500	Agreed
mRRG	5	0.096	2000	4000	Agreed
PRM	n/a	0.550	748	12830	Agreed
Protein L					
T-RRT	1	0.027	500	1000	Agreed
mRRG	5	0.039	1000	2000	Agreed
PRM	n/a	2.330	500	10282	Agreed

Table 4: Method comparison for proteins G variant, A, and L. All methods produced folding pathways that agreed with the experimentally determined secondary structure formation ordering (SSFO) except T-RRT for G variant.

4.3 Varying k

Here we study the affects of k, the number of neighbors each new sample is connected to. For mRRG, we fix m to 5.

Table 5 shows the statistics of the dominant folding pathway with varying values of k. Increasing k increases the running time without a measurable improvement in path quality. Figure 6 plots the potential vs. Euclidean distance to the native state for each pathway. Again, increasing k does not yield a significant improvement. We conclude that for these applications, $k = 1$ could be comfortably used. Table 6 shows a similar trend for the remaining proteins.

k	Time (hr)	Path Length	Path Weight
1	0.063	47	2361
3	0.104	53	4029
5	0.120	55	4577

Table 5: Protein G path statistics for mRRG for varying k.

Protein	k	Time (hr)	Path Length	Path Weight
G Variant	1	0.043	56	3271
	3	0.086	48	2888
A	1	0.096	74	164
	3	0.250	78	170
L	1	0.039	68	3382
	3	0.088	65	4511

Table 6: Path statistics for mRRG on proteins G variant and L under varying k.

5. CONCLUSIONS

We present a new *multi-directional* Rapidly Exploring Random Graph (mRRG) approach for studying protein folding based on traditional RRGs. Unlike traditional RRGs which only expand in a single direction, mRRG expands in multiple directions in each iteration step. We compare our method to two popular approaches: T-RRT and PRM. We show that our method is effective in achieving better energy landscape coverage and more unfolded pathways quickly as compared to T-RRT and PRM. Future work includes application of mRRG approach for more complex proteins of larger size and to other types of protein movement such as transitions between two given conformations in a binding interaction.

References

[1] E. Alm and D. Baker. Prediction of protein-folding mechanisms from free-energy landscapes derived from native structures. *Proc. Natl. Acad. Sci. USA*, 96(20):11305–11310, 1999.

[2] N. M. Amato and G. Song. Using motion planning to study protein folding pathways. *J. Comput. Biol.*, 9(2):149–168, 2002. Special issue of Int. Conf. Comput. Molecular Biology (RECOMB) 2001.

[3] M. Arnold, Y. Baryshnikov, and S. M. LaValle. Convex hull asymptotic shape evolution. In *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, 2012.

[4] R. Baldwin. Protein folding: matching speed and stability. *Nature*, 369:183–184, 1994.

[5] J. Bryngelson, J. Onuchic, N. Socci, and P. Wolynes. Funnels, pathways, and the energy landscape of protein folding: A synthesis. *Protein Struct. Funct. Genet*, 21:167–195, 1995.

[6] D. Covell. Folding protein α-carbon chains into compact forms by Monte Carlo methods. *Proteins: Struct. Funct. Genet.*, 14(4):409–420, 1992.

[7] V. Daggett and M. Levitt. Realistic simulation of naive-protein dynamics in solution and beyond. *Annu. Rev. Biophys. Biomol. Struct.*, 22:353–380, 1993.

[8] P. Das, S. Matysiak, and C. Clementi. Balancing energy and entropy: A minimalist model for the characterization of protein folding landscapes. *Proc. Natl. Acad. Sci. USA*, 102(29):10141–10146, 2005.

[9] K. A. Dill and H. S. Chan. From Levinthal to pathways to funnels: The new view of protein folding kinetics. *Nat. Struct. Biol.*, 4:10–19, 1997.

[10] Y. Duan and P. Kollman. Pathways to a protein folding intermediate observed in a 1-microsecond simulation in aqueous solution. *Science*, 282:740–744, 1998.

[11] P. Faccioli, M. Sega, F. Pederiva, and H. Orland. Dominant pathways in protein folding. *Phys. Rev. Lett.*, 97:108101–108104, 2006.

[12] C. J. Geyer. *Computing Science and Statistics: the 23rd Sympón the Interface*, pages 156–163, 1991.

[13] L. Jaillet, F. J. Corcho, J.-J. Pérez, and J. Cortés. Randomized tree construction algorithm to explore energy landscapes. *J. Comput. Chem.*, 32:3464–3474, 2011.

[14] S. Karaman and E. Frazzoli. Incremental sampling-based algorithms for optimal motion planning. In *Proceedings of Robotics: Science and Systems*, Zaragoza, Spain, June 2010.

[15] L. E. Kavraki, P. Švestka, J. C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. Robot. Automat.*, 12(4):566–580, August 1996.

[16] A. Kolinski and J. Skolnick. Monte Carlo simulations of protein folding. *Proteins Struct. Funct. Genet.*, 18(3):338–352, 1994.

[17] P. Lansbury. Evolution of amyloid: What normal protein folding may tell us about fibrillogenesis and disease. *Proc. Natl. Acad. Sci. USA*, 96(7):3342–3344, 1999.

[18] S. M. LaValle and J. J. Kuffner. Rapidly-exploring random trees: Progress and prospects. In *New Directions in Algorithmic and Computational Robotics*, pages 293–308. A. K. Peters, 2001. book contains the proceedings of the International Workshop on the Algorithmic Foundations of Robotics (WAFR), Hanover, NH, 2000.

[19] M. Levitt. Protein folding by restrained energy minimization and molecular dynamics. *J. Mol. Biol.*, 170:723–764, 1983.

[20] M. Levitt, M. Gerstein, E. Huang, S. Subbiah, and J. Tsai. Protein folding: the endgame. *Annu. Rev. Biochem.*, 66:549–579, 1997.

[21] M. Levitt and A. Warshel. Computer simulation of protein folding. *Nature*, 253:694–698, 1975.

[22] R. Li and C. Woodward. The hydrogen exchange core and protein folding. *Protein Sci.*, 8(8):1571–1591, 1999.

[23] S. Matysiak and C. Clementi. Optimal combination of theory and experiment for the characterization of the protein folding landscape of s6: How far can a minimalist model go? *J. Mol. Biol.*, 343(1):235–248, 2004.

[24] A. Mittermaier and L. E. Kay. New tools provide new insights in NMR studies of protein dynamics. *Science*, 312(5771):224–228, 2006.

[25] V. Muñoz, E. R. Henry, J. Hoferichter, and W. A. Eaton. A statistical mechanical model for β-hairpin kinetics. *Proc. Natl. Acad. Sci. USA*, 95:5872–5879, 1998.

[26] S. Nauli, B. Kuhlman, and D. Baker. Computer-based redesign of a protein folding pathway. *Nature Struct. Biol.*, 8(7):602–605, 2001.

[27] G. N. Reeke, Jr. Protein folding: Computational approaches to an exponential-time problem. *Ann. Rev. Comput. Sci.*, 3:59–84, 1988.

[28] H. Roder, K. Maki, and H. Cheng. Early events in protein folding explored by rapid mixing methods. *Chem. Rev.*, 106:1836–1861, 2006.

[29] A. P. Singh, J.-C. Latombe, and D. L. Brutlag. A motion planning approach to flexible ligand binding. In *Int. Conf. on Intelligent Systems for Molecular Biology (ISMB)*, pages 252–261, 1999.

[30] G. Song. *A Motion Planning Approach to Protein Folding.* Ph.D. dissertation, Dept. of Computer Science, Texas A&M University, December 2004.

[31] G. Song, S. Thomas, K. Dill, J. Scholtz, and N. Amato. A path planning-based study of protein folding with a case study of hairpin formation in protein G and L. In *Proc. Pacific Symposium of Biocomputing (PSB)*, pages 240–251, 2003.

[32] M. J. Sternberg. *Protein Structure Prediction.* OIRL Press at Oxford University Press, 1996.

[33] Y. Sugita and Y. Okamoto. Replica-exchange molecular dynamics methods for protein folding. *Chemical Physics Letters*, 314:141–151, 1999.

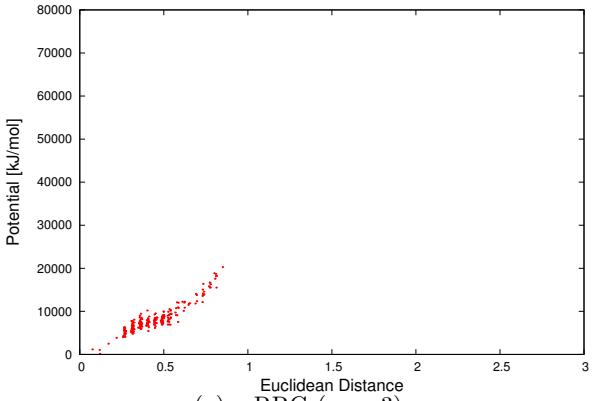
[34] R. H. Swendsen and J. S. Wang. Replica monte carlo simulation of spin glasses. *Phys. Rev. Lett.*, 57:2607–2609, 1986.

[35] S. Thomas, X. Tang, L. Tapia, and N. M. Amato. Simulating protein motions with rigidity analysis. In *Proceedings of the 10th annual international conference on Research in Computational Molecular Biology*, RECOMB'06, pages 394–409, Berlin, Heidelberg, 2006. Springer-Verlag.

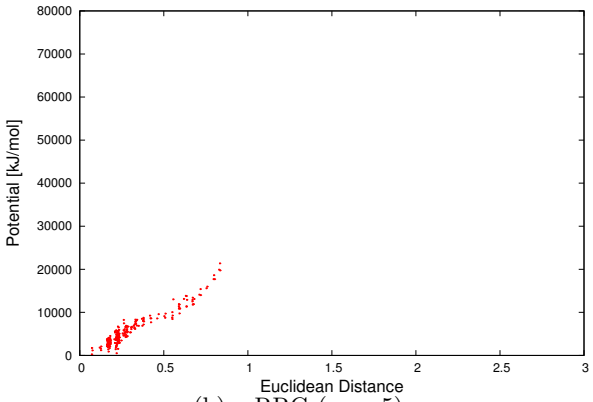
[36] S. Thomas, X. Tang, L. Tapia, and N. M. Amato. Simulating protein motions with rigidity analysis. *J. Comput. Biol.*, 14(6):839–855, 2007. Special issue of Int. Conf. Comput. Molecular Biology (RECOMB) 2006.

[37] T. E. Wales and J. R. Engen. Hydrogen exchange mass spectrometry for the analysis of protein dynamics. *Mass Spec. Rev.*, 25(1):158–170, 2006.

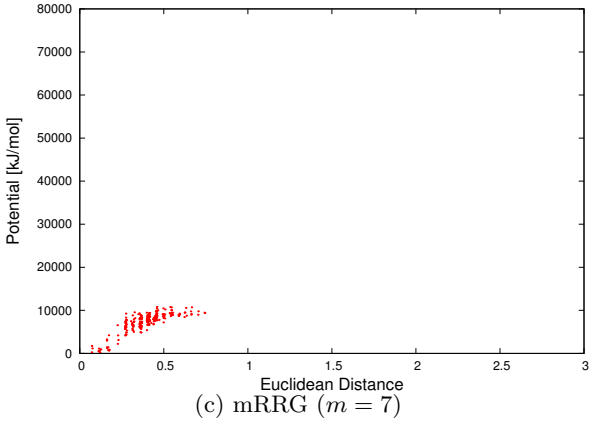
[38] B. Zagrovic, C. D. Snow, M. R. Shirts, and V. S. Pande. Simulation of folding of a small alpha-helical protein in atomistic detail using worldwide-distributed computing. *J. Mol. Biol.*, 323:927–937, 2002.



(a) mRRG ($m = 3$)



(b) mRRG ($m = 5$)



(c) mRRG ($m = 7$)

Figure 3: Node distribution comparison of the first 250 samples for protein G using mRRG. Figure 1 gives PRM and T-RRT distributions. PRM has the densest distribution near the native state while T-RRT and mRRG distributions are more even. mRRG with $m > 1$ reaches further into the unfolded regions than T-RRT.

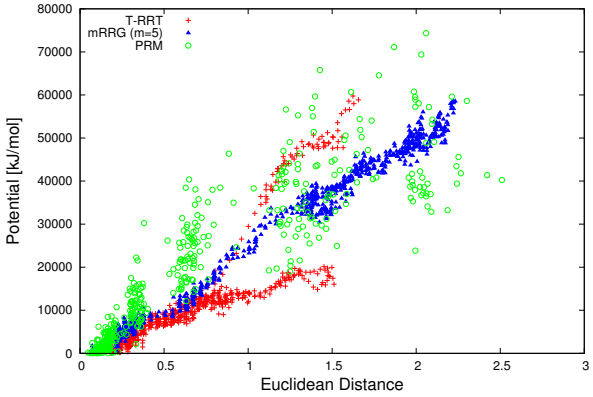


Figure 4: Node distribution comparison for protein G after the secondary structure formation ordering in the graph stabilized. T-RRT is unable to reach a large portion of the unfolded region.

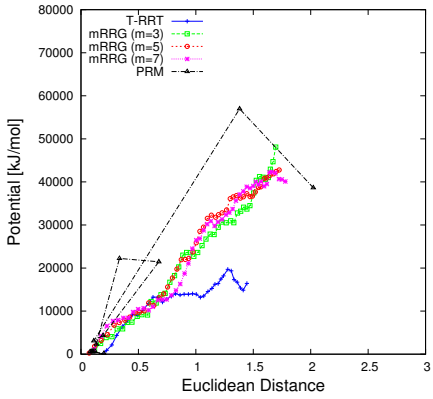


Figure 5: Comparison of the dominant folding pathway for protein G as determined by T-RRT, mRRG, and PRM.

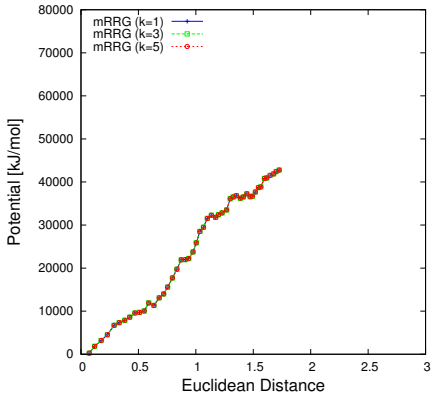


Figure 6: Path profile comparison for protein G under varying k. All paths are nearly identical indicating that k does not significantly change path quality.