Studying Protein Folding Using Motion Planning Techniques

Susan S. Lin
sslin@eecs.berkeley.edu

Dr. Nancy M. Amato, Faculty Advisor
Guang Song, Grad Student Advisor
{amato,gsong}@cs.tamu.edu

Department of Computer Science
Texas A&M University
College Station, TX 77843-3112

August 6, 2001

Abstract

The goal of this project is to use PRM (probabilistic roadmap) methods to study protein folding. Given a goal (native fold) configuration, we are able to construct a roadmap and derive a set of possible paths for the protein to follow. To do so, we model proteins as multi-link tree-like robots with many degrees of freedom. Our work concentrates on improving our techniques for studying the potential landscape of folding pathways. We propose to focus our energy on developing methods that find natural groupings of paths.

[1]This research supported in part by NSF CAREER Award CCR-9624315, NSF Grants IIS-9619850, ACI-9872126, EIA-9975018, EIA-9805823, and EIA-9810937, DOE ASCI ASAP Level 2 Grant B347886, and a Hewlett-Packard Equipment Grant. Susan Lin supported by the NSF through the CRA-W DMP program.

1 Introduction

Folding is a very common process in our lives, ranging from the macroscopic level – paper folding or gift wrapping – to the microscopic level – protein folding. In most instances, while one desires a particular final state to be reached (e.g., the package is wrapped, or the protein's structure is obtained), the knowledge of the dynamic folding process used to reach a particular state is of interest as well. For this reason, we believe motion planning has great potential to help us understand folding. In particular, while motion planning does have the ability to answer questions about the reachability of certain goal states from other states, its primary objective is to in fact determine the motions required to reach the goal.

The problem of folding (and unfolding) is an interesting research topic and has been studied in several application domains. Lu and Akella [8] consider a carton folding problem and its appli-

cations in packaging and assembly. In computational geometry, there are various paper folding problems, such as, given gluing instructions for a polygon, construct the unique convex polyhedron to which it folds [9]. In computational biology, one of the most important outstanding problems is protein folding, i.e., folding a one-dimensional amino acid chain into a three-dimensional protein structure.

There are large and ongoing research efforts whose goal is to determine the native folds of proteins (see, e.g., [10, 7]). In this paper, we assume we already know the native fold, and our focus is on the folding process, i.e., how the protein folds to that state from some initial state. Many researchers have remarked that knowledge of the folding pathways might provide insights into and a deeper understanding of the nature of protein folding [5, 11]. Although there have been some recent experimental advances [4], computational techniques for simulating this process are important because it is difficult to capture the folding process experimentally.

Our approach is based on the successful *probabilistic roadmap (*PRM*)* motion planning method [6]. We have selected the PRM paradigm due to its proven success in exploring high-dimensional configuration spaces (the configuration space, or C-space, of a movable object is the space consisting of all possible positions and orientations of the object). A major strength of PRMs is that they are quite simple to apply requiring only the ability to randomly generate points in C-space, and then test them for feasibility. The protein folding problem has a complication in that the way in which the protein folds depends on factors other than the purely geometrical constraints. Nevertheless, we show that these additional factors can be dealt with in a reasonable fashion within the PRM framework.

Since this work builds upon our previous work [13, 12] we will describe this process in Section 2. In Section 3.1, we present a new method for path selection. Section 3.2 deals with a method for determining path similarity, and we present some preliminary results in Section 4.

2 Previous Work

To apply the PRM framework to folding processes, we must define the configuration spaces of the objects we are interested in folding. In particular, we model the amino acid sequence as a multi-link tree-like articulated 'robot', where fold positions (atomic bonds) correspond to joints and areas that cannot fold (atoms) correspond to links. For the amino acid sequence of the protein, we consider all atomic bond lengths and bond angles to be constants, and consider only torsional angles (phi and psi angles), which we also model as two revolute joints (2 dof). Thus, the model will consist of $n+1$ links and n revolute joints.

As mentioned before, protein folding has a few notable differences from usual PRM applications. First, as our problems are not posed in an environment containing external obstacles, the only collision constraint we impose is that our configurations be self-collision free, and, for the protein folding problem, our preference for low energy conformations leads to an additional constraint on the feasible conformations. Second, in PRM applications, it is usually considered sufficient to find *any* feasible path connecting the start and goal. For our folding problems, however, we are interested not only in whether there exists a path, but we are also interested in the *quality* of the path. For example, for the paper folding problems, one is interested in a path which makes a minimal number of folds, and for the protein folding we are interested in low energy paths. Keeping these differences in mind, let's proceed through the three stages of PRM node generation, roadmap construction, and query.

2.1 Node Generation

During node generation, after the joint angles are known, the coordinates of each atom in the system are calculated, and these are then used to determine the potential energy of the conformation. The node is accepted and added to the roadmap based on its potential energy. This filtering helps

us to generate more nodes in low energy regions, which is desirable since we are interested in finding the pathways that are most energetically favorable (low energy).

2.2 Roadmap Construction

For each node, we first find its k nearest neighbors in the roadmap (using Euclidean distance in C-space), for some small constant k, and then try to connect it to them using some simple local planner. Each attempt performs feasibility checks for N intermediate configurations between the two corresponding nodes as determined by the chosen local planner (the number of such configurations is, e.g., the resolution used for collision detection, which may be set by the user). If there are still multiple connected components in the roadmap after this stage (which is generally the case, and in fact is sometimes unavoidable, see, e.g., [2, 3]), other techniques will be applied to try to connect different connected components (see [1] for details).

When two nodes are connected, the corresponding edge is added to the roadmap. We associate a weight with each edge. By assigning the weights in this manner, we can find the shortest or most energetically feasible path when performing subsequent queries.

2.3 Query

The resulting roadmap can be used to find a feasible path between given start and goal configurations. Usually, attempts are made to connect the start and the goal configurations to the same connected component of the roadmap. If this succeeds, a path is returned, otherwise failure is reported. For the protein folding, if the potential of some intermediate node is too large (as compared to some predetermined maximum), a failure is reported, otherwise the path is returned.

2.4 Validation

To test, we considered two proteins, Protein GB1 and Protein A. In general, our results are very encouraging – in both cases, the formation order of the secondary structures seems to agree with the results of the pulse labeling experiments. Thus, while further investigation and tuning of the PRM technique for proteins is still needed, our preliminary findings show that this motion planning approach is a potentially valuable tool. For example, it could be used to study the secondary structure formation order for proteins where this has not yet been determined experimentally.

3 Current Work

Our current work involves improving on our path selection methods and developing techniques to for determining the how 'similar' two paths are. Path selection involves extracting paths from the roadmap on which to perform analysis. Path similarity involves investigating ways to determine if two paths can be considered to be of the same family. For this paper, we studied Protein G (see Figure 1).

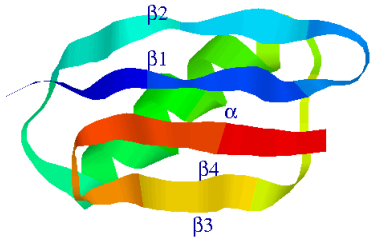

Figure 1: Protein G has four secondary structures: an α-helix and three β sheets.

3.1 Path Selection

Initially, we used a simple pruning technique to obtain a set of paths for study. We performed a

formation order	*old method*	*new method* $k = 0.80$	0.50	0.05
α, β3-4, β1-2, β1-4	181	2577	1852	69
α, β1-2, β3-4, β1-4	125	741	426	17
β3-4, β1-2, β3-4, α	0	462	186	4
β1-2, β3-4, α, β1-4	0	308	48	0
β1-2, β3-4, β1-4, α	0	16	10	0
β3-4, β1-4, β1-2, α	0	5	2	0

Table 1: The number of paths found by each path selection method.

breadth-first search of the tree, looking for the first nodes that satisfied these parameters: the node must have at least k children, and may not exceed a potential P. Then paths would be defined as the shortest path from these nodes to the goal.

We felt, however, that this method was inappropriate, since there is no guarantee that the nodes we choose as the start of the paths are in fact denatured. In its place, we thought to use a technique that would, by its very nature, find only denatured states. Our new method also performs a breadth-first search of the tree, but it looks for the first nodes that have no secondary structure. We determine a node to have formed a specific secondary structure (i.e. an alpha helix or beta sheet (see Figure 1) if the node has formed $k\%$ of the native contacts from that structure. A contact is a pair of residues close enough to be considered in contact with each other; a native contact is therefore a contact existing in the native fold. We do not define what k should be, and as expected, the number of paths selected depends greatly upon this variable (see Table 1). For the purpose of this paper, we set $k = 0.05$, in order to maintain a manageable collection of paths to test.

This new method is advantageous for at least two reasons. First, it is logically more appropriate for this task. When analyzing paths, we are interested in paths starting with a denatured protein. Only then may we analyze the entire folding process. Second, this process can find a significantly larger number of paths. While this may not be an improvement from a computational point of view, it gives us more data to work with, which implies that it is more complete. Indeed, for Protein G, this method selects paths whose secondary structure formation order had not been encountered before. However, the two orders found by the original method form an overwhelming majority, which is encouraging as it does not refute our previous work. We concluded that this method is more thorough, and more accurately reflects the variety of folding pathways possible. For the rest of our work, we use this method in place of the original.

3.2 Path Similarity

To understand protein folding behavior, it is important to understand the potential landscape of a folding protein. We already divide the paths into groups based on their secondary structure formation order. We are interested in studying two things. First, if these groups may themselves be broken further into subgroups. This would indicate an even higher level of separation between paths. Second, we would like to study the degree of similarity between paths from different groups. We expect that there is very little intergroup connectivity, but if there is, that may suggest that our division between paths with different secondary structure formation order is artificial.

To determine if two paths are similar, we break it down into two more basic tasks. First, we try to match the nodes of one path to the nodes of the other. Then, we check each pair for connectivity, based on previously defined constraints.

3.2.1 Node Matching

We attempt to do intelligent node matching by searching within the paths for the best match, as opposed to a simple proportional matching. First, we determine which path, $P1$, has fewer nodes. Let's call the second path $P2$. For each node in $P1$, starting with the one representing the goal, we search within a set R of nodes for the best

match. R is defined to be all nodes of $P2$ between the most recent matched node and the node $k * \lfloor P2.length/P1.length \rfloor + 1$ places from the goal (see Figure 2). This k is defined to be the number of matches attempted, regardless of their success. We say that two nodes may be paired if the difference in their number of native contacts does not exceed a given c, and if the difference in their potentials is not greater than a given p. If there are no matches that meet our criteria, no pair is found. If more than one match is found, then the 'best' pair is defined as the one which minimizes c.

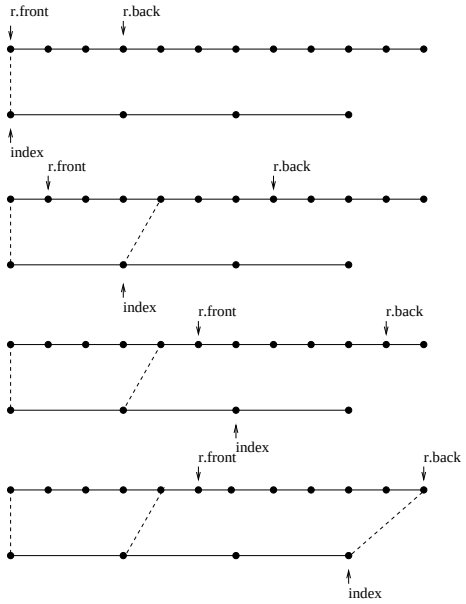

Figure 2: An example of the node matching process. For the first iteration, $R = (1, 4)$. Naturally, we match the goal to itself. In the second iteration, $R = (2, 8)$, and we find the best match to be 5. Thus, for the third iteration, $R = (6, 12)$, and we find no match. For the last iteration, $R = (6, 13)$, and we find the best match at 13.

One of the major advantages of this method, over a simpler proportional matching algorithm, is that if two paths share the same prefix, this method will find those matches. However, this method is not ideal for cases in which the plot of c for $P1$ is much different than that for $P2$. For instance, we might match two nodes with close c, but where the general tend for c is rising in one path and falling in the other. In the future, we may want to improve on our current technique by considering sets of nodes, instead of individual nodes, when attempting a match. That way, we can look at the trends in the paths as well.

3.3 Path Connectivity

Given a pair of paths and a set of matched nodes between them, we must decide if these paths are connectable. Because the roadmap is directed, we must consider movement between the paths in both directions. We define path $P1$ as connectable to path $P2$ if there are a sufficient precentage k% of connectable pairs of nodes from $P1$ to $P2$. We consider node $c1$ connectable to $c2$ if a straight line between $c1$ and $c2$ can be formed without self-collision, and if this line does not cross over any potential peaks.

For the task of interpreting this new information, we chose to represent the possible connections between all paths with a weighted digraph, where the paths are the vertices, and the edges represent the cost of moving between one path to another. This cost can either be equal to the maximum weight or average weight of moving between paths.

The motivation behind using the maximum weight to study this graph is the weakest link idea; movement from one path to another should be at least as difficult as the most difficult movement amongst the pairs of nodes. On the other hand, we might be interested in using the average weight as the node matching algorithm may not work perfectly. It is true that our node matching method minimizes the difference in potential for node pairs, but this difference does not give information about the potential of moving between them. In practice though, it does not matter which weight we use, probably because they are not too divergent. In the future, however, we might like to consider using these weights, and not our node connectibility method, to determine if the paths are connectable.

$k = .20$	group 1	group 2	group 3
group 1	5	1	1
group 2	-	26	1
group 3	-	-	5

Table 2: Number of connected components for $k = 0.2$

$k = .80$	group 1	group 2	group 3
group 1	5	6	4
group 2	-	26	4
group 3	-	-	6

Table 3: Number of connected components for $k = 0.8$

$k = 1.0$	group 1	group 2	group 3
group 1	8	29	8
group 2	-	50	29
group 3	-	-	8

Table 4: Number of connected components for $k = 1.0$

4 Results and Discussion

Here we present some preliminary results using the methods described in Sections 3.1 and 3.2 (see Tables 2, 3, and 4). As we anticipated, as k increases, the number of connected components (i.e. the distinct subgroups existing between each group) increases. At $k = 1.0$, the number of subgroups is maxed out, such that each subgroup contains exactly one path. However, we didn't expect that at $k = 0.2$, the number of subgroups within major groups remains almost the same, while there is only one subgroup between different groups. This suggests that there is more connectivity between groups than within, which seems unnatural.

However, as we have not thoroughly tested our methods yet, it is possible that there is simply something we have not taken into account, or that we are misinterpreting our graphs. Of course, we do not want to discount the possibility that these results are accurate, and therefore, our previous expectations were flawed. If this is the case, we might have some very interesting new information on protien folding behavior.

5 Conclusion and Future Work

In this paper, we expanded on our previous work. We propose a method for studying the potential landscape by reconstructing the roadmap from a group of paths selected from the original roadmap. By employing our path similarity tactics, we can eliminate irrelevant edges (i.e. edges between non-connectable paths), thus making the roadmap easier to query for further study.

We would like to continue to improve the methods presented here, by further testing with more proteins, such as Protein A, and by testing more combinations of variables. Until we do so, we cannot be certain that the new techniques we present in this paper are valid. We also need to investigate further our seemingly unnatural results from connectivity graphs. Once we have determined our new methods to be sturdy, we would like to test our new methods on larger proteins which have not been studied extensively.

6 Biography

Susan Lin is a fourth year undergraduate at UC Berkeley. She is slated to graduate in Spring 2002 with B.A.s in Mathematics and in Computer Science. When not working, studying, or sleeping, she likes gardening, crafts, and writing bad poetry. After graduation, she plans to take a year off to cross as many things off her 'To Do' list as possible, before proceeding to grad school.

References

[1] N. M. Amato, O. B. Bayazit, L. K. Dale, C. V. Jones, and D. Vallejo. OBPRM: An obstacle-based PRM for 3D workspaces. In *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, pages 155–168, 1998.

[2] T. Biedl, E. Demaine, M. Demaine, S. Lazard, A. Lubiw, J. O'Rourke, M. Overmars, S. Rob-

bins, I. Streinu, G. Toussaint, and S. Whitesides. Locked and unlocked polygonal chains in 3D. In *Proc. 10th ACM-SIAM Sympos. Discrete Algorithms*, pages 866–867, January 1999.

[3] J. Cantarella and H. Johnston. Nontrivial embeddings of polygonal intervals and unknots in 3-space. *J. Knot Theory Ramifications*, 7:1027–1039, 1998.

[4] W.A. Eaton, V. Muñoz, P.A.Thompson, C. Chan, and J. Hofrichter. Submillisecond kinetics of protein folding. *Curr. Op. Str. Bio.*, 7:10–14, 1997.

[5] B. Honig. Protein folding: From the Levinthal Paradox to structure prediction. *J. Mol. Bio.*, 293:283–293, 1999.

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

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

[8] L. Lu and S. Akella. Folding cartons with fixtures: A motion planning approach. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 1570–1576, 1999.

[9] J. O'Rourke. Folding and unfolding in computational geometry. In *Proc. Japan Conf. Discrete Comput. Geom. '98*, pages 142–147, December 1998. Revised version submitted to LLNCS.

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

[11] E.I. Shakhnovich. Theoretical studies of protein-folding thermodynamics and kinetics. *Curr. Op. Str. Bio.*, 7:29–40, 1997.

[12] G. Song and N. M. Amato. A motion planning approach to folding: From paper craft to protein folding. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 948–953, 2001.

[13] G. Song and N. M. Amato. Using motion planning to study protein folding pathways. In *Proc. Int. Conf. Comput. Molecular Biology (RECOMB)*, pages 287–296, 2001.