Using Bit-Vector Decision Procedures for Analysis of Protein Folding Pathways

Christopher James Langmead[1,2] and Sumit Kumar Jha[1]

[1] School of Computer Science, Carnegie Mellon University
{cjl,jha}@cs.cmu.edu
[2] Department of Biological Sciences, Carnegie Mellon University

Abstract. We explore the use of bit-vector decision procedures for the analysis of protein folding pathways. We argue that the protein folding problem is not identical to the classical probabilistic model checking problem in verification. Motivated by the different nature of the protein folding problem, we present a translation of the protein folding pathways analysis problem into a bounded model checking framework with bit vector decision procedures. We also present initial results of our experiments using the UCLID bit-vector decision procedure.

1 Introduction

Protein folding is a phenomenon of fundamental interest that appeals to both experimental and computational biologists alike. Physically, folding corresponds to the process by which a given protein spontaneously moves from an unfolded (aka denatured) three-dimensional structure to its folded (or native) structure. The study of protein folding is important for a number of reasons. *First*, it is important to basic research because there is much that remains unknown about how proteins fold. In particular, it is not possible at the present time to accurately predict a protein's native structure in a reliable fashion. *Second*, a number of diseases, including Alzheimer's and bovine spongiform encephalopathy (aka "mad cow disease"), are associated with *misfolded* proteins. That is, proteins that end up in some configuration other than their native structure. Understanding how and why certain proteins misfold is essential to the prevention and the treatment of these diseases. *Finally*, folding is just one type of large-scale conformational change exhibited by proteins. Many enzymes, for example, change structure upon binding specific molecules. These structural changes are often the mechanism by which the enzymatic reaction is catalyzed. Insights revealed in the process of studying protein folding are immediately applicable to studying other kinds of conformational changes and, consequently, to applications such as drug design and protein engineering.

While there are many experimental measurements that are relevant to studying the process of folding (e.g., circular dichroism), none of these reveal a detailed picture of the folding processes from start to finish. For this reason, computational simulations of the folding process are generally used to elucidate the specific *pathways* between the unfolded and folded structures. Motion Planning [10]

has been widely applied to perform simulation based analysis of protein folding pathways [13]. Recently, Langmead and Jha [12] have proposed the use of temporal logic for the symbolic state space analysis of protein folding pathways with some success. This paper studies the use of decision procedure based verification methods for studying the protein folding process. We also present initial results on applying a decision procedure based algorithm for analyzing protein folding pathways.

2 Background: Protein Folding and Probabilistic State Transition Systems

As is widely known, DNA is the repository that stores the blueprint of life. The process of transcription reads a linear sequence of nucleotides and produces a linear chain of amino acids called proteins. Proteins use their complex three-dimensional and dynamic structure to satisfy a wide range of functional requirements. The process by which a linear sequence of amino acids acquires its three dimensional structural configuration is called protein folding. An unfolded protein is a protein where no residues have formed bonds with non-neighboring residues. A folded protein is a structure where the free energy of the protein has been minimized - the energy obtained by the formation of long range weak bonds between residues has offset the entropy loss due to conformance to a particular structure.

The configuration of a partially folded protein can be described by the fraction of residues that have already folded. Thus, the configuration of a protein can be represented by a bit-vector $b_0 b_1 \dots b_{n-1}$, where each bit represents the folded or unfolded state of a residue. Each configuration $c_i = b_0 b_1 \dots b_{n-1}$ of the state space represents a partially folded protein of n residues; if b_i is true, the i^{th} residue is folded otherwise the residue is unfolded. An energy function $E : 2^n \rightarrow \mathcal{R}$ maps each protein configuration $(c_i =)\ b_0 b_1 \dots b_{n-1}$ to its free energy $E(c_i)$.

The partially folded configurations of the proteins naturally define a state transition system. Each probabilistic transition in the state space of protein configurations is associated with the folding or unfolding of a residue in the protein. The probability of transition from the configuration c to the configuration c' is a function of the energy difference of the configurations $E(c') - E(c)$. A protein can move directly from a configuration c to another configuration c' only if the two configurations are not too structurally different. In our state space model, a configuration c is connected to another configuration c' by a transition if and only if the configurations differ in less than H bits, where H is a small number [3]. So, these state space models are Bounded Hamming Distance Kripke structures (BHDKS) [9].

Given a set of initial states in the probabilistic state space, it is of interest to find if a particular configuration in the state space is reachable with at least a

[3] In this paper, we use $H = 1$

certain probability value along some path. If a particular configuration is reachable with a high probability, then this configuration will play an important role in determining the functional properties of the protein. Moreover, if there are two or more high-probability configurations, such a protein may be subject to misfolding which, as previously mentioned, is relevant to a variety of diseases. In principle, it may be possible to design a drug that targets pathways to misfolded configurations and in that way, prevents certain classes of disease.

3 Motivation for Application of SMTs to Protein Folding

The experiences of Langmead et al [12] clearly indicate that existing symbolic probabilistic model checkers do not scale well to state spaces of size with more than 2^{20} states. On the other hand, discrete state model checking has been applied to more than 2^{60} states [5]. Recently, efficient decision procedures [2, 3, 7, 11] for reasoning over bit-vectors and real arithmetic have been developed and applied to problems in areas as diverse as software security [6] and hybrid systems [8].

Several aspects of the protein folding problem make it more amenable to a decision procedure based approach. Unlike traditional probabilistic verification problems, the values of the probability obtained during the analysis of protein dynamics need not be precise - it is really not interesting to know if a state is reachable with probability 0.0011 or 0.0012. In traditional verification questions, some states are unreachable. Hence, it is required to know if the probability value is precisely zero or close to zero. However, for protein folding, all states are actually reachable and the interesting queries would like to know if a particular state is reachable with high enough probability.

In traditional probabilistic model checking, the probability values originate from prior chosen constants of probabilistic protocols and algorithms. In protein folding, the probability values are an involved function of the bit pattern of the source and the destination state of the transition system. These probability values are naturally given as a function of the bit pattern of the source and the destination states. Thus, bit-vector expressions are a compact way of representing the transition probabilities.

While traditional verification questions often want to know if a state is *ever* reachable, in the context of the present model of protein folding, every state is reachable. What is relevant are those pathways from the unfolded to the folded state that are highly probable and thus biologically significant. The probability of a given pathway is related to the magnitude of the energies of the configurations along the path; low-energy pathways are more probable than high-energy pathways. Thus, bit-vector bounded model checking tools like UCLID are the natural choice for analyzing protein folding pathways.

4 Translating Protein Folding Pathways into SMT

The probability of reaching a state through a given path is given by the product of the transition probabilities. Also, the probability of a transition is exponentially proportional to the difference in the energy of the states before and after the transition. SMT solvers [4] for linear bit-vector arithmetic are much more efficient than those for nonlinear arithmetic. We, hence, take the logarithm of the probability values and use linear arithmetic to compute the logarithm of the probability values of the paths. We note that α is a normalization factor while β is derived from the gas constant and the ambient temperature.

$$P_{i+1} = P_i \cdot \alpha e^{\beta(E-E')} \Leftrightarrow \log P_{i+1} = \log P_i + \log(\alpha) + \beta \cdot (E - E')$$

Given a bit vector $b_0 b_1 \ldots\ldots b_{n-1}$ corresponding to a state configuration c of the protein, there is an energy function which computes the energy of the configuration. Let $E(b_0, b_1 \ldots b_{n-1})$ be the energy function. Then the probability of moving from the configuration $c' := b'_0 b'_1 \ldots\ldots b'_{n-1}$ to the configuration $c'' := b''_0 b''_1 \ldots\ldots b''_{n-1}$ is given by:

$$\text{Transition } (c', c'') \Rightarrow \log(P_{i+1}) := \log(P_i) + \log(\alpha) + \beta \cdot (E(c') - E(c''))$$

The energy function for a protein configuration is a function of the bit pattern of the configuration. In particular, we have used an energy function involving pairwise contact potentials ($\phi_{i,j}$) and entropy values (ψ_i) in our experiments.

$$E(b_0, b_1 \ldots b_{n-1}) = \sum_{0 \le i,j \le n-1} \phi_{i,j} \cdot b_i \cdot b_j + \sum_{0 \le i \le n-1} \psi_i \cdot b_i$$

Though the energy potential of each configuration is nonlinear in the bits involved in the configuration, the transitions in our probabilistic state space only depend on the change in the energy $E(c') - E(c'')$, which is a linear function in the bits of the configurations. Assuming the bit b_k is flipped from 1 to 0,

$$E(b_0, b_1 \ldots b_k \ldots b_{n-1}) - E(b_0, b_1 \ldots \neg b_k \ldots b_{n-1}) = \sum_{k,j} \phi_{k,j} \cdot b_j + \psi_k$$

Similarly, if the bit b_k is flipped from 0 to 1,

$$E(b_0, b_1 \ldots b_k \ldots b_{n-1}) - E(b_0, b_1 \ldots \neg b_k \ldots b_{n-1}) = -\sum_{k,j} \phi_{k,j} \cdot b_j - \psi_k$$

5 Refinement based Analysis of Protein Folding Pathways

The transition system of the protein folding problem is defined naturally by the configurations of the protein and the transitions among them. Given a protein with n residues, the associated Kripke structure $K = (S,\ T, S_0,\ AP, L, Prob)$, where

- $S = \{0,1\} \times \{0,1\} \times \cdots \times \{0,1\}$.

– $T = S \times S$. In order to prevent proteins from jumping across structural energy barriers, we only allow transitions to occur among structurally similar configurations. Hence, $(s, s') \in T$ iff s and s' have no more than one residue folded in a different configuration.
– $S_0 = \{0, 0, 0 \ldots, 0\}$. While we could start from any state or set of states, our experiments are based on starting from the unfolded configuration and then studying the folding process.
– $AP = \{R_0, R_1, \ldots, R_n\}$ is the set of atomic propositions labeling the state. R_i indicates whether the i^{th} residue is folded or unfolded in the given state.
– $L : S \rightarrow 2^{AP}$. We label each state with the folded or unfolded status of each residue in the unique configuration corresponding to the state. Hence,in state s, $R_i = true$ iff the i^{th} residue is folded in state s i.e. iff s(i) = 1.
– $Prob : T \rightarrow [\mathbf{0}, \mathbf{1}]$. Each transition (s, s') is labeled with a probability which indicates the probability of going from state s to state s'.

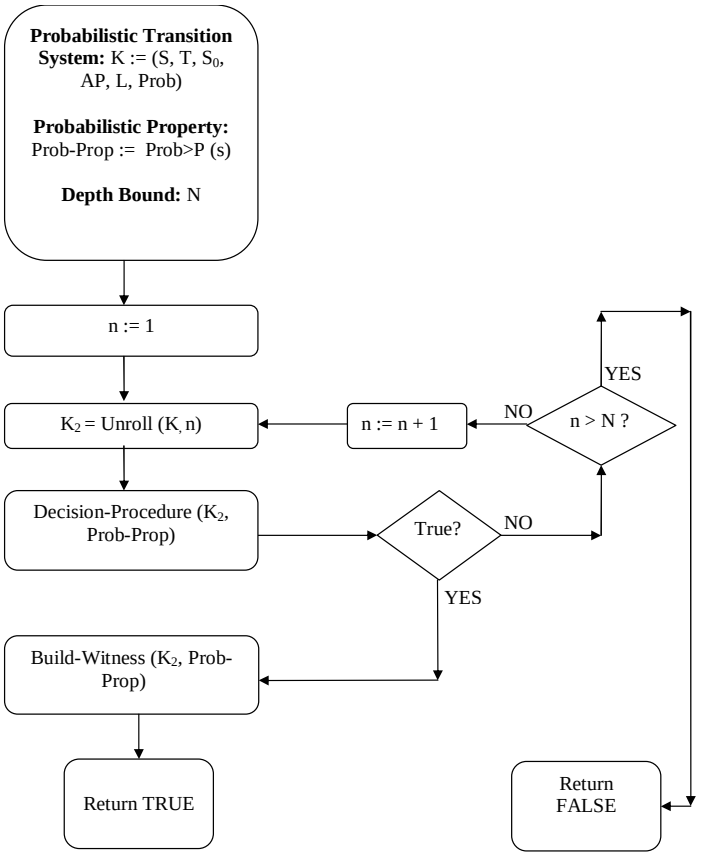

Fig. 1. The Bounded Analysis Algorithm

Our analysis of protein folding pathways is based on bounded model checking using a finite bit vector to represent probabilities. Our reachability queries are simple i.e. whether it is possible to reach a state t with probability at least p.

As shown in Fig. 1, we first unroll the state transition system K for n steps and ask if it is possible to reach the state s with probability at least P along any path of length at most n. If the decision procedure actually finds such a path, the result of the decision procedure can be used to build a more explicit witness. The witness can then be presented as a protein folding pathway satisfying the given probabilistic property to the end-user (even visually, like the movie of a folding protein). If the decision procedure does not find such a path in n steps, then a formula is constructed for $n + 1$ steps unless the value of n exceeds a given threshold N on the length of the pathway.

However, bit vector Decision Procedures are very sensitive to the number of bits that are involved in the computation. Our analysis needs us to compute the values of the logarithm of the probability. If we use k bits to represent the value of the logarithm of the probability, the error obtained due to rounding is given by

$$
\begin{aligned}
&\log P' \text{ - } \log P \leq 2^{-k} \\
&\Rightarrow P'/P \leq 2^{2^{-k}} \\
&\Rightarrow P' \leq 2^{2^{-k}} P \\
&\Rightarrow P' - P \leq (2^{2^{-k}} - 1)P
\end{aligned}
$$

In order to estimate the value of the error $P' - P$, we present the numerical values of the bound in Table 1.

Number of Bits	Error Bound
10	0.000677130693
20	$6.61036882 \times 10^{-7}$
30	$6.45543619 \times 10^{-10}$
40	$6.30384633 \times 10^{-13}$
50	$4.88498131 \times 10^{-15}$

Table 1. Upper bound on the estimated value of the error

Suppose we want to ask a question $Prob(s) < P$, i.e. whether the probability of a reachability property in n steps of the bounded model checking algorithm

is less than P. We first pick a small bit-width for the value of $\log P$ and try to ask the question if $Prob(s) < P - n\epsilon$. Here, the error term epsilon is derived from the bit-width of the value of $\log P$ used and the number of steps n used for the bounded model checking. If we succeed, we have shown that $Prob(s) < P$. Otherwise, we ask the question if $Prob(s) > P + n\epsilon$. If we succeed, we have

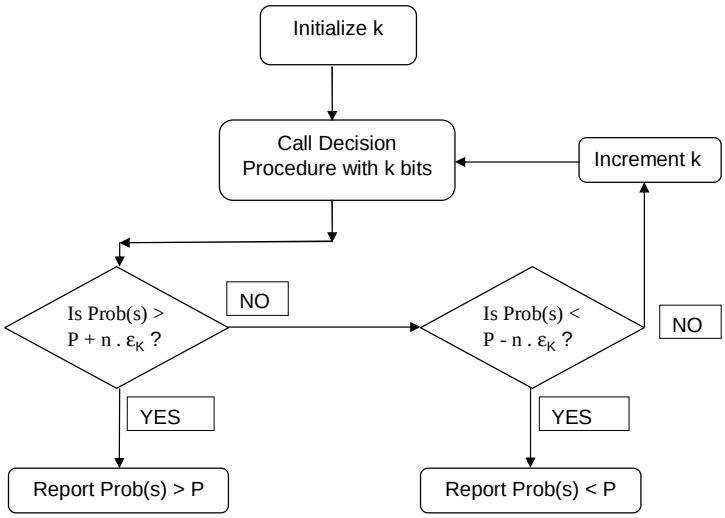

Fig. 2. Iterative probability approximation using abstraction-refinement.

If both our attempts fail, then we increment the size of the bit vector representing the logarithm of the probability term and repeat the above steps again. Our approach is based on the abstract refinement paradigm which is an effective technique to perform a lazy analysis of a given instance of a verification problem. It makes analysis only as expensive as is needed for analyzing the particular problem instance. The algorithm is illustrated in Fig. 2.

6 Experimental Results

We implemented our bounded model checking algorithm using the UCLID bit-vector decision procedure. We modeled the first m (m=10,20,30,50) residues of a protein called 1fkb and used the following probability function:

$P = \alpha \cdot e^{\beta \cdot (E - E')}$

Number of Bits	Time Taken	Result ($\lvert \log P \rvert > 4$?)
10	9.2s	No
20	26.44s	Yes
30	55.94s	Yes
50	218.18s	Yes

Table 2. Result for probability value with accuracy of 10 bits - is $\lvert \log P \rvert > 4$ for a state where the first 5 bits are folded (BMC steps = 5) ?

Our experiments were aimed at discovering whether the analysis of protein folding using SMT solvers proposed in this paper can be effectively deployed in practice. We chose $\alpha = 1, \beta = 1/kT$ for our experiments. This does not provide a probabilistic interpretation to our model. A truly probabilistic model would require normalizing the values of α and making a suitable choice for the value of β.

Depth of BMC	Time Taken	Result ($\lvert \log P \rvert > 4$?)
5	9.2s	No
10	18.331s	No
20	37.599s	No
40	75.358s	No

Table 3. Result for P value with accuracy of 10 bits - is $\lvert \log P \rvert > 4$ for a state where the first 5 bits are folded (state space = 2^{10}) ?

7 Conclusion

Bit-vector decision procedures are efficient at bounded analysis of protein folding pathways with huge state spaces. They perform better than symbolic probabilistic model checking approaches when the entire state space is actually reachable.

8 Future Work

This work used an off-the-shelf decision procedure. The formula generated for the energy term is ugly and unnecessarily complex, and makes the problem harder for the decision procedure. A decision procedure in which the energy function has been implemented as a bit-vector function is bound to outperform these experiments.

The error term involved in fixed width representations of probability P using k bits is $(2^{2^{-k}} - 1) \times P$. While we have used a fixed number of bits for the representation, the error term is actually a function of the probability itself. It is, hence, desirable to use more bits in the first few transitions of the bounded model checking algorithm and then use fewer bits. A decision procedure implementing such adaptive encoding can scale to larger state spaces without any loss of precision.

While analyzing a single protein is interesting, protein dynamics of a cluster of molecules is much more interesting for drug design. Decision procedures naturally allow one to quantify over positions of a pair of molecules and hence study their possible interactions.

This study clearly identifies a need and usefulness of a decision procedure for analyzing protein folding pathways and protein dynamics.

References

1. Rajeev Alur and Doron Peled, editors. *Computer Aided Verification, 16th International Conference, CAV 2004, Boston, MA, USA, July 13-17, 2004, Proceedings*, volume 3114 of *Lecture Notes in Computer Science*. Springer, 2004.
2. Clark W. Barrett and Sergey Berezin. CVC Lite: A new implementation of the cooperating validity checker. In Alur and Peled [1], pages 515–518.
3. Sergey Berezin, Clark Barrett, Igor Shikanian, Marsha Chechik, Arie Gurfinkel, and David L. Dill. A practical approach to partial functions in CVC Lite.
4. Randal E. Bryant, Daniel Kroening, Joel Ouaknine, Sanjit A. Seshia, Ofer Strichman, and Bryan Brady. Deciding bit-vector arithmetic with abstraction. In *Proceedings of TACAS 2007*, volume 4424 of *Lecture Notes in Computer Science*, pages 358–372. Springer, 2007.
5. Jerry R. Burch, Edmund M. Clarke, Kenneth L. McMillan, David L. Dill, and L. J. Hwang. Symbolic model checking: 10^{20} states and beyond. *Inf. Comput.*, 98(2):142–170, 1992.
6. Mihai Christodorescu, Somesh Jha, Sanjit A. Seshia, Dawn Song, and Randal E. Bryant. Semantics-aware malware detection. In *Proceedings of the 2005 IEEE Symposium on Security and Privacy (Oakland 2005)*, pages 32–46, Oakland, CA, USA, May 2005. ACM Press.
7. Vijay Ganesh and David L. Dill. A decision procedure for bit-vectors and arrays. In Werner Damm and Holger Hermanns, editors, *CAV*, Lecture Notes in Computer Science. Springer, 2007.
8. Susmit Jha, Bryan Brady, and Sanjit Seshia. Symbolic reachability analysis of lazy linear hybrid automata. In *Proceedings of FORMATS 2007*, Lecture Notes in Computer Science. Springer, in press.
9. Susmit Jha and R. K. Shyamasundar. Adapting biochemical Kripke structures for Distributed Model Checking. In Corrado Priami, Anna Ingolfsdottir, Bud Mishra, and Hanne Riis Nielson, editors, *Transactions on Computational Systems Biology*, pages 107–122, 2006.
10. Lydia Kavraki, Petr Svestka, Jean-Claude Latombe, and Mark Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. Technical Report CS-TR-94-1519, 1994.

11. Shuvendu K. Lahiri and Sanjit A. Seshia. The UCLID decision procedure. In Alur and Peled [1], pages 475–478.
12. C.J. Langmead and S. K. Jha. Predicting protein folding kinetics via model checking. In *Proceedings of Workshop on Algorithms in Bioinformatics (WABI) 2007*, Lecture Notes in Bioinformatics. Springer, in press.
13. Guang Song and Nancy M. Amato. Using motion planning to study protein folding pathways. In *RECOMB '01: Proceedings of the fifth annual international conference on Computational biology*, pages 287–296, New York, NY, USA, 2001. ACM Press.