ELSEVIER

Discrete Applied Mathematics 127 (2003) 145–161

DISCRETE APPLIED MATHEMATICS

The algorithmics of folding proteins on lattices

Vijay Chandru[a,b], Abhi DattaSharma[c], V.S. Anil Kumar[d,*]

[a] *Computer Science & Automation, Indian Institute of Science, Bangalore 560 012, India*
[b] *Strand Genomics Pvt Ltd. India*
[c] *Kombinatorische Geometrie, Zentrum Mathematik, Gab, Technische Universität München, D-80290 München, Germany*
[d] *Basic and Applied Simulation Sciences (D-2), MS M997, Los Alamos National Laboratory, Los Alamos, NM 87545, USA*

Received 1 January 2001; accepted 28 January 2002

Abstract

It should be possible to predict the fold of a protein into its native conformation, once we are given the sequence of the constituent amino acids. This is known as the protein structure prediction problem and is sometimes referred to as the problem of deciphering the second half of the genetic code. While large proteins fold in nature in seconds, computational chemists and biologists have found that folding proteins to their minimum energy conformations is a challenging unsolved optimization problem. Computational complexity theory has been useful in explaining, at least partially, this (Levinthal's) paradox. The pedagogic cross-disciplinary survey by Ngo, Marks and Karplus (Computational Complexity, Protein Structure Prediction and the Levinthal Paradox, Birkhauser, Basel, 1994) provides an excellent starting point for non-biologists to take a plunge into the problem of folding proteins. Since then, there has been remarkable progress in the algorithmics of folding proteins on discrete lattice models, an account of which is presented herein.
© 2002 Elsevier Science B.V. All rights reserved.

1. Introduction

Proteins are among the most important biological molecules, and are responsible for correct bodily functioning in all living organisms. Each protein has well-defined functions, which range from building up DNA and RNA molecules to controlling

* Corresponding author.
E-mail addresses: chandru@csa.iisc.ernet.in (V. Chandru), abhi@mathematik.tu-muenchen.de (A. DattaSharma), anil@lanl.gov (V.S. Anil Kumar).

0166-218X/03/$ - see front matter © 2002 Elsevier Science B.V. All rights reserved.
PII: S0166-218X(02)00381-5

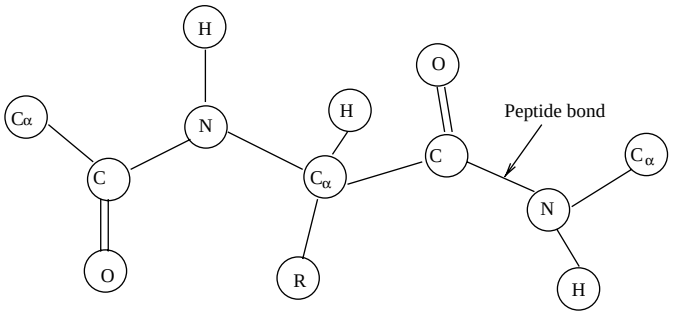

Fig. 1. Backbone of a protein: amino acids connected by peptide bonds. R refers to a side chain

different parameters in living cells. It is amazing that all proteins are built of very simple building blocks, known as amino acids. There are 20 different amino acids, each of which consists of a central carbon atom—called the alpha carbon—bonded to an amino group (NH_2), a carboxyl group (COOH) and a side chain. The side chain is different in each amino acid, and is solely responsible for the characteristics of the amino acids like shape, size and polarity. The structure and properties of a protein in turn depend on the these basic features of the constituent amino acids.

Amino acids are linked to each other by means of *peptide bonds*. They result in a constrained backbone of the alpha carbons (see Fig. 1), with limited values for bond angles, which can be derived by steric constraints (akin to configuration space diagrams for collision-free motion of kinematic assemblies and robot arms).

Proteins fold into compact and very varied structures, but these are usually a combination of some simple motifs called *secondary* structures. Two important types of motifs are α helices and β sheets. The final protein structure can be viewed as being built up of the secondary structures connected in specific ways.

Determining the structure of proteins is a very important problem. The three-dimensional (3D) structure of a protein is believed to be a very important determinant of the properties of the protein. This becomes crucial in drug design where the aim is to obtain proteins with specific functionalities. The most remarkable discovery in this area was made by Christian Anfinsen and his colleagues in the 1950s when they found that many simple proteins had a unique native structure, which just seems to depend on the sequence. This has been subsequently verified for a large number of proteins and it is now believed that the native structure is a minimum energy configuration (the Thermodynamic Hypothesis). This has led to an enormous interest in trying to develop methods to predict the 3D structure from the sequence information via optimization techniques. Determining a protein sequence has become feasible with current technology, but determining the exact 3D structure is still a very slow and expensive process.

Proteins usually fold into their native structure very fast, though the conformation space is very large. The way nature figures out the right folding pathway has remained a mystery. The view that an intermediate conformation evolves via a sequence of

somewhat random modifications leads to a paradox, called *Levinthal's paradox*. It is very likely that a better understanding of the nature of dominant forces would lead to better success in predicting the 3D structure.

The difficulty of working with the detailed atomic level model has motivated biologists to work on simple discrete models. These problems are essentially combinatorial optimization problems, and one purpose of this paper is to survey recent results.

Section 2 describes the basic lattice model and the experimental results of Dill et al. [11]. Section 3 discusses various aspects of the computational intractability of folding proteins. Section 4 gives an overview of the various approximation algorithms known for this problem. Section 6 gives a brief description of the work by Clote [8] that tries to explain the Levinthal paradox by means of Markov chain theory. We conclude in Section 7 with a discussion of the value of existing results, and interesting open questions.

2. Discrete models for protein folding

A very general model for protein folding could be derived by a detailed modeling of all possible interactions. This would be of little value, as the conformation space would be very large, and finding a minimum energy conformation would be quite hopeless. This has driven biologists to design simpler discrete models for studying this problem, which should be relatively simpler to analyze, and could suggest useful properties. Apart from searching for the minimum energy native structure, biologists are also interested in the *folding pathway*, by which the native state is reached, and an explanation to Levinthal's paradox, mentioned in the previous section

One way to discretize this problem is to restrict attention to embeddings on a lattice. The energy function also has to be defined appropriately in this new setting. Broadly, three discrete models have been proposed for protein folding [23]. These are described in the next three sub.section We follow the conventions and notations of [23] in the next three sub.section Then we describe a very simple model known as the Hydrophobic–Hydrophilic model and survey the known empirical results about it.

It is hard to objectively compare these models. The success of any model will probably depend on how much it can be analyzed practically.

2.1. The Protein Structure Prediction (PSP) model

This is a very general non-discrete model, and we mention it only for completeness. The model was defined formally by Ngo and Marks [22], who also give an NP hardness result for this model.

In this model, the protein is described by the complete list of the atoms in the molecules, their connectivities, bond lengths and angles and force constants between all pairs of atoms. The energy of a conformation is a non-convex function obtained by summing the contributions of different kinds of interactions: $U = \sum_{\mathrm{b}} K_{\mathrm{b}}^{\mathrm{bond}}(l_{\mathrm{b}} - l_{\mathrm{a}}^{0})^2 + \sum_{\mathrm{a}} K_{\mathrm{a}}^{\mathrm{angle}}(\Theta_{\mathrm{a}} - \Theta_{\mathrm{a}}^{0})^2 + \sum_{\mathrm{t}} K_{\mathrm{t}}^{\mathrm{torsion}}(1 - \cos[n_{\mathrm{t}}(\phi_{\mathrm{t}} - \phi_{\mathrm{t}}^{0})]) + \sum_{i>j} K_{ij}^{\text{non-local}} f(r_{ij}/r_{ij}^{0})$.

As one would suspect, finding a conformation that minimizes this function is NP hard [22]. The hardness is shown by a reduction from the Partition problem. No approximation algorithms are known for this problem.

2.2. The Lattice Polymer Embedding (LPE) model

This model was formulated by Unger and Moult [30]. The protein is modeled as a chain of beads, $S=s_1,\ldots,s_n$. The space is the collection of embeddings in the 3D cubic lattice. An embedding means that each bead must be placed at some lattice site, and successive beads must be adjacent on the lattice. In addition, the embedding must be non-self-intersecting. A coefficient $c(s_i,s_j)$, known as the affinity coefficient, is defined for each pair s_i,s_j. And finally, we are given a function $f:[S]\times[S]\times[S]\to R$.

The energy of any conformation is defined as $U=\sum_{i<j}c(s_i,s_j)f(|x(s_i)-x(s_j)|, |y(s_i)-y(s_j)|,|z(s_i)-z(s_j)|)$, where $(x(s_i),y(s_i),z(s_i))$ is the position of bead s_i in the embedding. The objective is to find the conformation that minimizes this energy. Unger and Moult show that this problem is NP hard, by a reduction from the Optimal Linear Arrangement for this problem.

2.3. The Charged Graph Embedding (CGE) model

This model also describes the protein as a sequence $S=s_1,\ldots s_n$ of beads. A charge $C(s_i)\in\{-1,0,1\}$ is associated with each s_i. For each pair s_i,s_j, the contribution $U(s_i,s_j)$ to the energy is $C(s_i)C(s_j)/d(s_i,s_j)$ if $d(s_i,s_j)\leqslant d_{\max}$, and otherwise it is 0. $d_{\max}$ is defined to be a distance cutoff, beyond which interactions are not assumed to exist. The function $d(s_i,s_j)$ is the euclidean distance between $(x(s_i),y(s_i),z(s_i))$ and $(x(s_j),y(s_j),z(s_j))$. One important condition is that bonds are allowed to cross, as long as there is at most one bead per site. Fraenkel [12] showed that this problem is also NP hard by reduction from 3D matching.

An interesting feature of the CGE model is that it incorporates charges on the residues. On the other hand the bonds permitted are not realistic.

2.4. The Hydrophobic–Hydrophilic model

We now consider a popular model of protein folding called the Hydrophobic–Hydrophilic model, introduced by Dill [10] and studied extensively in [13,18–20,30–32]. This is the simplest possible abstraction of the folding problem, which is still non-trivial and retains the hardness features of the original problem. The success of Dill et al. in explaining several different phenomena regarding folding pathways and the space of native conformations has made the study of discrete models more respectable with traditional biologists.

The model starts with classifying the twenty amino acids as H (hydrophobic or non-polar) or P (hydrophilic or polar). This classification is known from experimental results. A protein is viewed as a sequence $S=s_1,\ldots,s_n$, with $s_i\in\{H,P\},\forall i$. Allowed conformations are non-self-intersecting embeddings on a lattice, usually taken to be cartesian. The energy of any folding is inversely proportional to the number of pairs

of H's that are neighbors in the folding. Therefore, minimizing the energy is equivalent to maximizing the number of contacts, or the H–H pairs that become adjacent. Since the number of H–H pairs that occur in successive positions in S is fixed, the energy depends only on the number of non-consecutive H–H pairs that become adjacent, also called topological neighbors.

Natural generalizations of this model have also been considered. These involve either changing the underlying lattice, or changing the underlying alphabet and the energy function.

2.4.1. Theoretical results

This simple model was shown to be NP complete by Berger and Leighton [4] for 3D and by Crescenzi et al. [9] for 2D, which are briefly discussed in Section 3. Even before hardness results were known, Hart and Istrail [14] gave a simple approximation algorithm, which is described in Section 4.

2.4.2. Experimental results

A lot of empirical work has been done on this model. Dill et al. [11] have extensively studied the properties of this model by actual enumeration of all conformations for small sequences. Unger and Moult [31,32] used genetic algorithms to obtain compact foldings of fairly long sequences, but could not give any guaranteed bounds on their algorithms.

It is interesting to look at the empirical results of Dill et al. [11] and the original motivations for studying this model. There have been two divergent views about the dominant forces responsible for protein folding. The first view held that the folding process is dominated by local interactions, with hydrogen bonding being primarily responsible, which explained the formation of secondary structures like α helices and β sheets satisfactorily. The second view, which has only recently gained support, is that non-local interactions are the primary determinants of the folding process. As mentioned in [11], Kauzmann argued, way back in the 1950s, that a strong force is the tendency of non-polar amino acids to form hydrophobic cores.

2.5. The value of discrete models

Analyzing discrete models for protein folding are very interesting and challenging questions for computer scientists, but may be unsatisfactory for traditional biologists. It is therefore important to understand the pros and cons of studying discrete models.

Since real experiments are slow and expensive, biologists have begun to rely on computer simulations to understand protein folding. The complete biological model is too complex to simulate, and therefore lots of researchers have started trying to use results from simulations on simple discrete models to deduce general properties of the folding process.

There are certainly many shortcomings in these simple models:

(1) The resolution of the original problem is lost. Bond angles are in reality not right angles, but lie in some other allowed regions (as in, e.g. the Ramachandran plot).

(2) Details of protein structure, bond energies and charges cannot be represented accurately in such models.
(3) Bond lengths are not captured well enough.

In spite of these limitations, discrete lattice models have been studied for their many virtues:

(1) Lattice models allow simulations of a large number of conformational changes. Detailed atomic models would not be able to explore more than small changes that occur over very small timescales.
(2) Simulation of atomic models involve many parameters and approximations, making their validity just as doubtful as for the simple models. The lattice models can be checked for specific phenomena and also allow almost exhaustive enumeration for small sequences. This is useful in deducing statistical properties of conformations.

See [11] for a wealth of predictions based on simulations on this model.

3. NP hardness results

We survey the known hardness results about various lattice models of protein folding in this section, and their implications for designing good algorithms. The proofs of NP hardness for the various models are quite involved and we only sketch the proof here.

Informally, problems whose solutions can be verified in polynomial time, are said to be in the class NP (refer to [13] for details). An NP-complete problem is one that is in NP and is at least as hard as any other problem in NP. Some well-known examples are Travelling Salesman Problem and satisfiability. No efficient algorithm has been found so far for any NP complete problem, and it is widely believed that it might not even exist. The NP hardness of protein folding in various models implies that it might be hopeless to look for efficient algorithms to find the exact optimum in general.

It is usually very simple to show that a given problem is in NP. Showing the other part, i.e., it is as hard as any other problem in NP involves proving that an algorithm for the given problem can be used to solve some other known NP complete problem.

Worst case hardness: It is important to keep in mind that the notion of NP completeness is a statement about the *worst* case hardness of a problem. It is, therefore, still possible that the protein folding problem is not hard on the average, an issue that deserves attention. It is also possible that additional constraints about the problem may make it tractable. Also, since solving the folding problem exactly is hard, it immediately raises the question of approximation, discussed in the next .section

Folding with larger alphabet size: The hardness results for the HP model were preceded by results for slightly more general models, which used larger alphabet sizes. The first result about hardness of string folding problems was by Paterson and Przytycka [26], where they use a score function that counts the number of times identical letters become adjacent in the lattice. One shortcoming of their reduction was that they used an unbounded alphabet size, which came from a reduction from Satisfiability. This was remedied in a result by Nayak et al. [24], where they use bounded alphabet size, and also show an approximation hardness result. This means that getting arbitrarily good

approximations is also unlikely. Still, their alphabet size was larger than a realistic size. Atkins and Hart [2] gave a hardness reduction for a smaller alphabet size. Hart and Istrail [15] introduced the notion of robustness of hardness for protein folding. They showed that by parameterizing the underlying lattice, one could show hardness for the energy function of the LPE model, for any reasonable lattice.

For the HP model, Crescenzi et al. [9] proved NP hardness for the 2D model. The hardness for the 3D model was shown by Berger and Leighton [4], and this is briefly discussed in the next section.

3.1. The hardness of the 3D-HP model

We discuss here the hardness of the 3D-HP model, proved by Berger and Leighton [4]. The reduction will be sketched informally and most of the details of the proof will be omitted here. A subproblem in the reduction is a variant of a classical 3D VLSI wire routing problem. We shall highlight the main points in this interesting connection, as this might be a useful tool in other string folding problems.

The starting point for the reduction of Berger and Leighton is a seemingly simpler version of the folding problem, which they call the Perfect HP String-Fold problem: Given a sequence containing n^3 Hs, is there a folding in which all the Hs can be packed into a $n \times n \times n$ cube? By a simple argument, it can be shown that if a sequence has n^3 H's, the only way to get a score of $3n^2(n-1)$ is to pack all the H's into a cube, and this implies that the Perfect HP String-Fold problem is a special case of the HP folding problem.

The NP-hardness of the Perfect HP String-Fold problem is shown by reduction from a variant of the Bin Packing. The input to this problem consists of a set U of items, with $s(u)$ denoting the size of an item $u \in U$, and integers B and K. Each $s(u)$ is even, and the items satisfy $\sum_u s(u) = BK$. The objective is to decide whether U can be partitioned into sets $U_1, \ldots, U_K$ such that the sum of the items in each U_i equals B.

The Reduction: We are given an instance, $\mathscr{B}$, of Bin Packing, as described above. The reduction constructs an instance $S_{\mathscr{B}}$ of the folding problem with the following property.

Theorem 1 (from Berger and Leighton [4]). *The sequence $S_{\mathscr{B}}$ can be folded with a score of n^3 if and only if $\mathscr{B}$ has a solution.*

The instance $S = S_{\mathscr{B}}$ is now described here. The sequence has many parts, whose functionality is described informally here. The items in the bin packing problem are encoded as the suffix of the string S. While describing S, we will also describe the folding of S that packs all H's into a cube, assuming that $\mathscr{B}$ has a solution. This will also be a proof that if $\mathscr{B}$ has a solution, S can be folded with a score of n^3. The following observation is needed in the construction:

Observation 1. *Suppose S can be folded so that all the H's can be packed into a cube. Then, a substring of the form $(PHHP)^*$ can be mapped to a connected path*

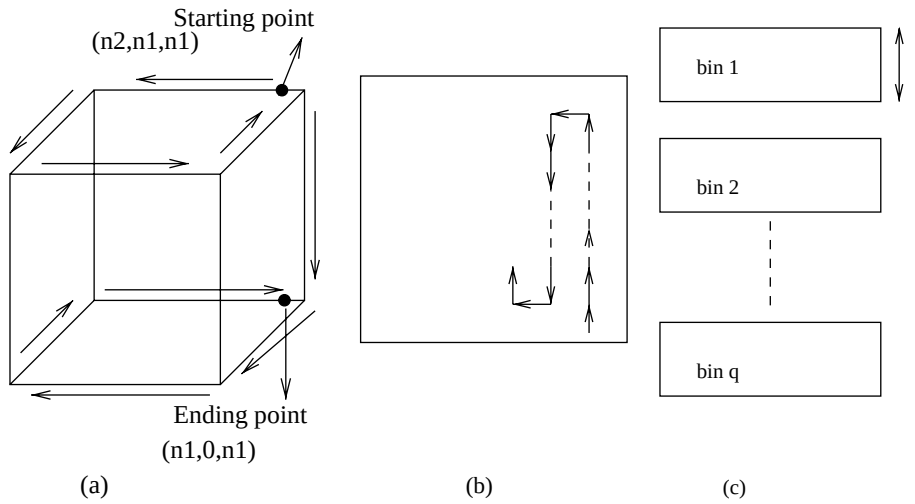

Fig. 2. The reduction of Berger and Leighton [4]: (a) The edges of the cube being covered from the *starting point* to the *ending point*. (b) The snake-like packing on each face. (c) Formation of bins on the front face.

on the surface of the cube and a substring of the form $(PHP)^*$ *can be mapped to a path along the edges of the cube.*

The prefix of S can be functionally broken down into four parts.

(1) The first part has the form $(PHP)^{9n-10}$ and it forms a path (from the above observation) that goes through all the corners and nine of the 12 edges (Fig. 2(a)). The points in the cube have coordinates $(i,j,k), 0 \leqslant i,j,k \leqslant n-1$. The path starts at $(n-2, n-1, n-1)$ and ends at $(n-2, 0, n-1)$, as shown in the figure.

(2) The second part of the prefix is designed to cover five of the six faces of the cube, starting from the back face and leaving out the front face and has the form $(PHHP)^{(n-2)^2/2}(PHP)^{n-2}P^4((PHHP)^{(n-2)^2/2}P^7)^3(PHHP)^{(n-2)^2/2-1}$. Each of the five faces is covered by the substring $(PHHP)^{(n-2)^2/2}$, which is possible by the observation above. After this prefix of S has folded, all the faces of the cube, except the front face, are covered by H's. The packing on each face is done in a snake-like fashion (as in Fig. 2(b)).

(3) The third part is $H^{(n-2)^3+2}\alpha$, where α is a sequence of Ps, long enough to ensure connectedness. This substring winds through the interior in a snake-like pattern filling it up, and ends at $(n-1, n-1-q, -2)$, where $q = 2\max(K, B^{1/3})$. At this point, only the front face is uncovered.

(4) The final part of the prefix sets up q bins on the front face and has the form $((PHP)^q(PHHP)^{(n-2)/2}(PHP)^{2q+1}P^4(PHHP)^{(n-2)/2}(PHP)^qP^{2q})^q\beta$. This substring is first embedded on the front face to create the walls of q bins, as shown in Fig. 2(c). The string β is designed to first fill $q-K$ bins completely, and fill all but B points in each of the remaining K bins.

The suffix is of the form $\Pi_{u\in U}P^{cn}(PHHP)^{s(u)/2}$, where c is some large constant, chosen by the considerations described below (Π denotes a concatenation of strings). So this part encodes the sizes of the items in the lengths of the blocks of $(PHHP)$'s. For the region corresponding to bin U_i, the goal is to pack the strings corresponding to the items packed in bin U_i. These strings can be placed in any order, but must be packed contiguously in a snake-like winding manner. The real difficulty is to ensure that connect the substrings $(PHHP)^{s(u)/2}$ with the sequence of P's between them. This corresponds to a VLSI wire routing problem where a set of non-intersecting substrings in 2D have to be connected by wires in the third dimension, with the additional constraint that the connecting wires have the same length. Berger and Leighton construct a nice gadget (which they call a crossbar switch) which takes in inputs at points $(i, j, -4)$, and produces outputs at $k = -10n + 2$. Within the switch, the inputs can be connected to any permutation of outputs, using vertex disjoint paths. The permutation is chosen so that item strings which are subsequent in S become adjacent at the output level.

For the converse of the proof, it can be shown that a solution to the Perfect Folding problem in which all the H's are packed into a cube must have a bin like structure, because of the sequence constraints. Refer to [4] for the complete details.

4. Approximation algorithms

Since solving the protein folding problem *exactly* is NP-hard in all the models discussed earlier, it is natural to look for an *approximate* solution, i.e., a folding whose energy is only slightly higher than that of the optimum. An α factor approximate solution is defined as a folding whose energy is at most αE_{OPT}, where E_{OPT} is the minimum energy over all foldings, with $\alpha \leqslant 1$. We are interested in polynomial time approximation algorithms that produce such a solution. Unfortunately, very few significant approximation algorithms known, which is probably an indicator of the hardness of this problem. All the current algorithms are for variants of the HP model. The heart of these approximation algorithms involves good *folding rules*. These are compact paradigms for folding, which result in constant factor approximations.

For the HP model, an α approximation corresponds to obtaining a folding in which the number of H–H contacts is at least an α fraction of that in the optimal, since the energy is inversely proportional to the number of H–H contacts. The first such approximation algorithm was designed by Hart and Istrail [14], who obtained approximation factors of $\frac{1}{4}$ and $\frac{3}{8}$ for the 2D and 3D HP models, respectively. We give an overview of their algorithm in Section 4.1. We also discuss a recent improvement for 2D by Newman [21]. Section 4.2 describes results known for the HP model in other lattices and under more generalized notions of hydrophobicity. Finally, in Section 4.3, we discuss the relevance of such results.

4.1. The Hart–Istrail algorithms

We now discuss the algorithm due to Hart and Istrail [14]. The input to the algorithm is a sequence $S = s_1 \ldots s_n$, where $s_i \in \{H, P\}, \forall i$. The score of a folding is the number of

non-consecutive Hs that become adjacent in the folding. The goal is to find a folding of S in the cartesian lattice which maximizes the score.

The starting point of this algorithm is the important observation that s_i and s_j can be adjacent in any folding if and only if $|i-j|$ is odd, as a consequence of the cartesian lattice. In order to simplify this discussion, let us assume that $s_1 = s_n = P$. Let us call any $s_i = H$ to be *odd* if j is odd and *even* otherwise. Clearly, an odd H can have only an even H as a topological neighbor. For a subsequence s of S, define $N_o(s)$ and $N_e(s)$ to be the number of odd H's and even H's in s, respectively. Also, let $N_o = N_o(S)$ and $N_e = N_e(S)$. Let $N = N_o + N_e$ be the total number of H's in S.

An upper bound: Let N_o and N_e be the number of odd and even H's, respectively. In a d-dimensional cartesian lattice, the degree of any lattice vertex is $2d$. Since each H has two connected neighbors (by our assumption above), it can have at most $2d-2$ topological neighbors. As a result, the score of any folding (including the optimum fold, in particular) is at most $M = (2d-2)\min(N_o, N_e)$.

Notice that we do not know the optimum score. We shall prove that the algorithm achieves a factor of α by showing that it has a score of at least αM.

We need the following important combinatorial fact from [14]:

Lemma 1. *There is an index i (called the folding point) such that either $N_o(S_1) \geqslant N_o/2$, $N_e(S_2) \geqslant N_e/2$ or $N_e(S_1) \geqslant N_e/2, N_o(S_2) \geqslant N_o/2$, where S_1 is the subsequence $s_1, \ldots, s_{i-1}$ and S_2 is the subsequence $s_i, \ldots, s_n$.*

Without loss of generality, assume that the first condition of the lemma holds.

4.1.1. The algorithm in 2D

For the 2D lattice, the upper bound is $M = 2\min(N_o, N_e)$. The algorithm produces a folding with score at least $M/2$, which implies an approximation factor of $\frac{1}{4}$.

The basic idea of this algorithm, and all related subsequent work is to design folding rules, which result in a careful placement of the H's. Let the folding point i be as defined in Lemma 1. Also, S_1, S_2 are as defined in the lemma. The folding rule in the algorithm arranges all the odd H's in S_1 (and the even H's in S_2) at regular intervals on a straight line, called its *face*, in the lattice. This allows each odd H in S_1 to become adjacent to an even H in S_2, resulting in a score of $N_o(S_1) \geqslant M/2$.

The folding rule is described by the following algorithm. Order the even H's in S_2 from s_i to s_n and the odd H's in S_1 from s_{i-1} to s_1. Let $s_{f(1)}, s_{f(2)}, \ldots$ be the successive odd H's in S_1 and $s_{g(1)}, s_{g(2)}, \ldots$ be the successive even H's in S_2. Fig. 3(a) shows the folding point and the labeled odd and even H's for an example. The subsequence $s_{f(1)}, \ldots, s_{g(1)}$ is arranged as in Fig. 3(b), so that $s_{f(1)}, s_{g(1)}$ are adjacent, and the subsequence between them is above both of them in the lattice. Now, assume that the first $j-1$ odd H's in S_1 and first $j-1$ even H's in S_2 have been arranged. The subsequences $s_{f(j-1)+1}, \ldots, s_{f(j)}$ and $s_{g(j-1)+1}, \ldots, s_{g(j)}$ are now arranged as in Fig. 3(c). This ensures that $s_{f(j-1)}, s_{f(j-1)+1}, s_{f(j)}$ are in the same vertical face, as are $s_{g(j-1)}, s_{g(j-1)+1}, s_{g(j)}$. This construction is done repeatedly for $j = 2, \ldots N_o(S_1) = N_e(S_2)$. Fig. 3(d) shows the final folding for the sample sequence.

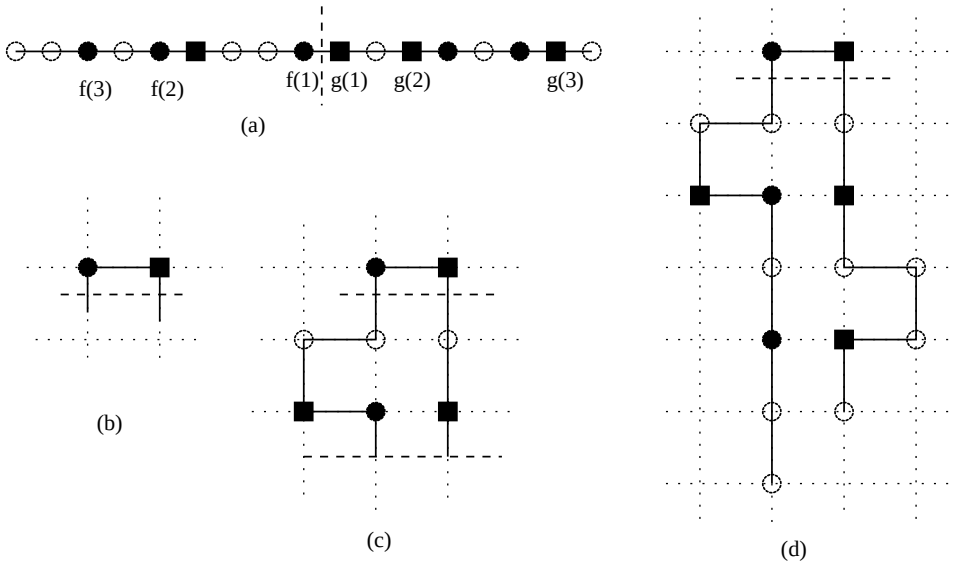

Fig. 3. (a) Sequence PPHPHHPPHHPHHPHH. Odd H's are shown as dark circles and even H's as dark squares and P's as light circles. The odd and even H's to participate in contacts are labeled. (b),(c) Intermediate folding steps. (d) Final fold.

This folding has the nice property that for each j, $s_{f(j)}$ and $s_{g(j)}$ are adjacent, and contribute 1 to the total score. As a result, the total score of the resulting folding is $N_o(S_1) = N_e(S_2)$.

4.1.2. Folding in 3D

The upper bound in 3D is $M = 4\min(N_o, N_e)$. The algorithm ensures that at least half of these H's have three contacts, giving a total score of $3\min(N_o, N_e)/2$, which implies an approximation factor of $\frac{3}{8}$.

A crucial structural property of the 2D folding is that all the adjacent H's lie on a face, which is a straight line. This immediately suggests the following heuristic for 3D. Partition the whole sequence into smaller parts. Each part is folded in a separate x–y plane, and these are imagined to be stacked on top of each other in the z direction. The faces along which each part is folded are perfectly aligned for each layer, which gives each H on any face (except the top-most and the bottom-most parts) three contacts—one from its own plane, one above and one below. One has to ensure that the faces have same lengths in each plane, and also that the *parity* of faces in adjacent layers is different. This means that if a layer has the face containing odd H's on the left and even H's on the right, then the layers immediately before and after this layer must place even H's on the left and odd H's on the right. As Hart and Istrail [14] shows, this is easy to achieve.

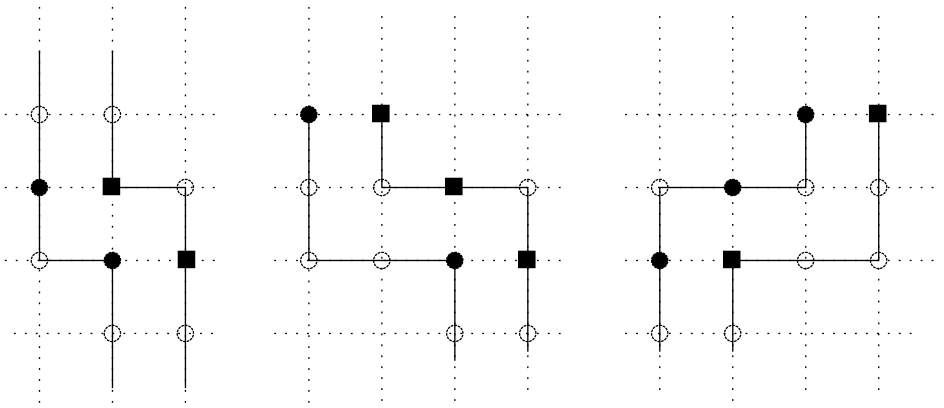

Fig. 4.

4.1.3. Improving the 2D algorithm

We briefly discuss the recent improvement by Newman [21] in 2D, which achieves a factor of $\frac{1}{3}$. The algorithm uses the same upper bound M, and is basically a variant of the folding rule of Hart and Istrail [14].

The motivation for the improvement comes from the following observation. Consider the algorithm described in Section 4.1.1. Suppose index j is such that $|f(j) - f(j+1)| = |g(j) - g(j+1)| = 2$. In this case, one could consider the folding of $s_{f(j)}, s_{f(j+1)}, s_{g(j)}, s_{g(j+1)}$ as in Fig. 4(a), which leads to three H–H contacts between these four elements. If the distances between $s_{f(j)}, s_{f(j+1)}$ or between $s_{g(j)}, s_{g(j+1)}$ are more than two, such an arrangement is possible only sometimes. If $|f(j)-f(j+1)| > 2$ and $|g(j) - g(j+1)| = 2$, the folding of Fig. 4(b) is done, involving $s_{f(j)}, s_{f(j+1)}, s_{g(j)}$ alone, while if $|f(j) - f(j+1)| = 2, |g(j) - g(j+1)| > 2$, the folding of Fig. 4(c) is done, involving $s_{f(j)}, s_{g(j)}, s_{g(j+1)}$ alone. The algorithm is basically again picking the odd and even H's in pairs, and perform the folding of the appropriate type. A folding of the type (b) or (c) by itself seems to be no good, but observe that an instance of a folding of type (b) together with that of type (c) involves three odd and three even H's, while yielding a total of four contacts. If the number of type (b) folds is the same as the number of type (c) folds, we have at least $\frac{4}{3}$ contacts for each of the H's involved, instead of 1 contact before, which improves the factor to $\frac{2}{3}$. Of course, they need not be equal, and one of them, say type (b) could occur more often, but as Newman shows, the difference can be cleverly bounded, yielding an approximation factor of $\frac{1}{3}$. This algorithm does not, unfortunately, improve the bound for 3D.

4.2. Folding in other lattices

One severe drawback of the square lattice is the parity problem: two residues cannot become adjacent if the string between them is of odd length, which is very unrealistic.

One way of remedying this is by altering the energy function and the lattice. Several researchers have looked at algorithms for folding in lattices other than cartesian. Agarwala et al. [1] consider folding in the triangular lattice model and give folding rules achieving much higher packing ratios in both two and three dimensions.

A natural generalization is to allow a larger alphabet size (to correspond to the 20 amino acids that make up actual proteins). Agarwala et al. also consider this variation. They generalize the notion of hydrophobicity, and design approximations for this notion in the triangular lattice.

4.3. A discussion of the approximation results

Since most of the algorithms work with a very similar kind of upper bound for the optimum score, the tightness of this bound is an important issue. Newman [21] shows that with this upper bound, one cannot expect to obtain an approximation better than $\frac{1}{2}$, because there are instances where the optimum could be close to $M/2$. Therefore, further improvements in the approximation ratio might come only after better upper bounds are found. Another interesting issue, which has not received any attention is the hardness of approximation.

4.4. The HP Side Chain model

The HP model only considers the main chain of a protein. A natural extension is a model proposed by Bromberg and Dill [6], that allows side chains to appear is to model the protein as a *caterpillar graph* instead of a linear chain. In this model, only the *legs* (side chains) are marked as hydrophobic or hydrophilic. Hart and Istrail [16] extended the original algorithm for the HP model and obtained an approximation factor of $\frac{1}{12}$ for the cubic lattice. They also extend this to the FCC lattice and obtain a factor of $\frac{31}{36}$. Heun [17] considers this model on extended cubic lattices, which are the cubic lattices along with their diagonals and presents approximation algorithms that achieve performance ratios of $\frac{59}{90}$ and $\frac{37}{40}$.

In an attempt to study an off-lattice model, Hart and Istrail [16] introduce the HP Tangent Sphere Model, which is an extension of the side chain model, with elements replaced by spheres of unit radius. They then show how to extend the analysis on the lattice to this model.

5. Other empirical approaches

Though there has been limited theoretical success in analyzing the discrete models, a lot of research has gone into empirical analysis.

The first application of genetic algorithms was probably by Unger and Moult [31,32]. Global optimization techniques have been used quite extensively (refer to papers in [5,25]). Many such papers talk about results on specific sequences and models, and it is hard to see a sound underlying theory in the known results.

There are also heuristic approaches based on constraint programming (e.g. [3]), machine learning and exhaustive enumeration (e.g. [33]).

6. The Levinthal paradox and rapidly mixing Markov chains

In a pioneering work, Šali et al. [27,28], modeled the folding of a protein as a Markov chain and simulated folding of small proteins (in the Hydrophobic–Hydrophilic model). They kept the temperature fixed; so it was a metropolis simulation, and not simulated annealing. The contact energies in their model were normally distributed. They observed that the folding time seemed to be small *if and only if* the energy gap between the lowest energy and the second lowest energies of conformations on the lattice was large. Thus, their model suggests that thermodynamic factors might have an important role in driving the folding process, and this answers the Levinthal paradox in their model.

In a recent paper, Clote [8] shows a mathematical basis for the observation of Šali et al. Actually, Clote considers a variant of the moves used by Šali et al, which does not ensure self-avoidance. We sketch the main result here and indicate the direction of proof briefly.

Let i_0 and i_1 denote the conformations having the least and second least energies, among all conformations on the lattice. Let $f(i)$ denote the energy of conformation i, and $\Delta = f(i_1) - f(i_0)$ is the *energy gap* between the least and second least energy conformations. Let π_i denote the stationary probability of conformation i; $\pi_i = \exp(-f(i)/T)/Z$, where T is the temperature and Z is the normalization constant. Let S be the set of all conformations of a sequence $s=s_1,\ldots,s_n$. Let p be the transition probability matrix corresponding to p.

The first step involves a derivation of an upper bound for the mixing time t_0 (the time within which the distribution of the chain becomes ε-close to the stationary distribution, where distance is measured by variation distance) in terms of a function of $\pi_{i_0}\pi_{i_1}$. By using conductance arguments, developed by Sinclair [29], the following bound on the mixing time can be obtained.

Lemma 2 (from Clote [8]). $t_0 \leqslant (-\ln \varepsilon - \ln(\min \pi_i))((n|S|\pi_{i_0}\pi_{i_1}/c) + 1)$, *where* $c = \min \pi_i p_{i,j}$.

The proof of this lemma involves showing that the conductance of the chain (see [29]) is large. One standard method for this is via the canonical path approach: set up paths from w to w', for each $(w,w') \in S \times S$, so that the congestion on any edge is bounded.

Next, Clote considers another energy function $g(\ )$ such that $g(i) = f(i), \forall i \neq i_1$ and $g(i_1) > f(i_1)$, which implies the energy gap $\Delta_g = g(i_1) - g(i_0)$ is larger than $\Delta_f = f(i_1) - f(i_0)$. Then $\pi'_i = \exp(-g(i)/T)/Z'$ is the stationary distribution for the new function $g(\ )$, where Z' is the corresponding normalization constant. Let p' be the transition probability matrix corresponding to g. The following lemma relates the stationary distributions w.r.t. f and g.

Lemma 3 (from Clote [8]). *Let* f, g, π, π' *be as defined above. Then,* $\pi'_{i_0}\pi'_{i_1}/c' < \pi_{i_0}\pi_{i_1}/c$, *where* $c = \min \pi_i p_{i,j}$ and $c' = \min \pi'_i p'_{i,j}$.

This clearly implies that the mixing time decreases as $\pi_{i_0}\pi_{i_1}/c$ decreases, which in turn, decreases as the energy gap $f(i_1) - f(i_0)$ increases. Finally, the first passage time to the native state is upper bounded by the mixing time plus $2/\pi_{i_0}$, which consequently decreases with the energy gap. It should however be mentioned, that Clote's result really says that the upper bound on the first passage time reduces as the energy gap increases, but even this weaker statement is interesting because it is the first such theoretical justification.

7. Conclusions

In summary, discrete models for protein folding provide a wealth of interesting algorithmic questions. As observed earlier, NP-hardness is a worst case statement. An interesting question is to see whether there are constraints coming from real proteins, whose addition simplifies these models. These could be distributions on the residues within sequences, folding rules or any other properties of real sequences. Since approximation algorithms have been developed so far only for variants of the HP model, solving the other models remains a challenging open question. Hardness/approximation results on robustness of the sort obtained in [15] will also be valuable. added by anil Further results on off-lattice models, as in [16] will be very significant.

Many significant results have been obtained for the HP model since the survey by Ngo et al. [23], but these are still not known to be tight. For the HP model, more sophisticated folding rules have to be designed for getting better approximations. Analyzing them requires a more clever upper bound than the one used so far.

The use of other techniques like genetic algorithms, integer programming and nonlinear optimization is still in initial stages, and further exploration of these is likely to be promising. Since theoretical results are likely to be very hard, these techniques would be very useful in an empirical study of these models.

To encourage a fresh approach to the problem, we describe an integer program for the HP model below. Instead of maximizing the number of H–H bonds, we shall minimize the number of edges $e = (u, v)$ such that a H residue is placed only on one of u and v. This is possible because the sum of these two quantities is a constant, which is easy to verify.

Let $G(V, E)$ denote the lattice graph. Let $s = s_1 \ldots s_n$ denote the sequence to be folded on G. We shall fix one end point of the fold at vertex a. Let H denote the indices i such that s_i is 1.

There are $n|V| + |E|$ variables: $x_{v,i}, v \in V, i = 1 \ldots n$ and $y_e, e \in E$. All the $x_{v,i}$ variables will be binary. The variable $x_{v,i}$ is interpreted to be 1 if s_i is placed on vertex v and 0 otherwise. We have the following constraints.

(1) $x_{a,1} = 1$ (Starting of the embedding is fixed at $a \in V$),
(2) $\Sigma_{i=1}^{n} x_{v,i} \leqslant 1, v \in V$ (Non-self-intersection),
(3) $\Sigma_{v \in V} x_{v,i} = 1, i = 1 \ldots n$ (Each s_i is placed on some vertex),

(4) $\Sigma_{v\in V, i=1\ldots n} x_{v,i} = n$ (Length of the sequence is n),
(5) $x_{u,i} \leqslant \Sigma_{v\in N(u)} x_{v,i+1}, u \in V, i = 1 \ldots n-1$ (Forcing an embedding),
(6) $y_e \geqslant \Sigma_{i\in H} x_{u,i} - \Sigma_{i\in H} x_{v,i}, e = (u,v) \in E$,
(7) $y_e \geqslant \Sigma_{i\in H} x_{v,i} - \Sigma_{i\in H} x_{u,i}, e = (u,v) \in E$,
(8) $x_{u,i} \in \{0,1\}, u \in V, i = 1 \ldots n$,
(9) $y_e \geqslant 0, e \in E$.

The objective is to minimize $\Sigma_{e\in E} y_e$. Chandru et al. [7] observe that for some sequences the solution by a branch-and-bound technique is much better than the approximations described earlier. The immediate question is whether suitable relaxations of this formulation can be shown to yield better approximation bounds.

References

[1] R. Agarwala, S. Batzogloa, V. Dancik, S.E. Decatur, S. Hannenhalli, M. Farach, S. Muthukrishnan, S. Skiena, Local rules for protein folding on a triangular lattice and generalised hydrophobicity, RECOMB (1997) 1–2.
[2] J. Atkins, W.E. Hart, On the intractability of protein folding with a finite alphabet of amino acids, Algorithmica 25 (1999) 279–294.
[3] R. Backofen, The protein structure prediction problem: a constraint optimisation approach using a new lower bound, J. Constraints (2000).
[4] B. Berger, T. Leighton, Protein folding in the hydrophobic–hydrophilic model is NP complete, J. Comput. Biol. 5 (1998) 27–40.
[5] L.T. Biegler, T.F. Coleman, A.R. Conn, F.N. Santosa (Eds.), Large scale optimization with applications, Part III: molecular structure and optimization, IMA Vol. Math. Appl. 94 (1997).
[6] S. Bromberg, K. Dill, Side chain entropy and packing in proteins, Protein Sci. 3 (1994) 997–1009.
[7] V. Chandru, S. Ganesh, M.R. Rao, Folding proteins on lattices: an integer programming approach, in: M. Groetschel (Ed.), Festschrift for Professor Manfred Padberg, Springer, Berlin, 2002.
[8] P. Clote, Protein folding, the Levinthal paradox and rapidly mixing Markov chains, ICALP. Springer Lecture Notes in Computer Science 1644 (1999) 240–249.
[9] P. Crescenzi, D. Goldman, C. Papadimitrou, A. Piccolboni, M. Yannakakis, On the complexity of protein folding, J. Comput. Biol. 5 (1998) 423–446.
[10] K.A. Dill, Biochemistry 24 (1985) 1501.
[11] K.A. Dill, S. Bromberg, K. Yue, K.M. Fiebig, D.P. Yee, P.D. Thomas, H.S. Chan, Principles of protein folding: a perspective from simple exact models Protein Sci. 4 (1995) 561–602.
[12] A.S. Fraenkel, Complexity of protein folding, Bull. Math. Biol. 55 (1993) 1199–1210.
[13] M.R. Garey, D.S. Johnson, Computers and Intractability—A Guide to the Theory of NP-Completeness, Freeman, San Francisco, CA, 1979.
[14] W.E. Hart, S. Istrail, Fast protein folding in the hydrophobic–hydrophilic model within three-eights of optimal, J. Comput. Biol. 3 (1996) 53–96.
[15] W.E. Hart, S. Istrail, Robust Proofs of NP-hardness for protein folding: general lattices and energy potentials J. Comput. Biol. 4 (1) (1997) 1–22.
[16] W.E. Hart, S. Istrail, Lattice and off-lattice side chain models of protein folding: linear time structure prediction better than 86% of optimal J. Comput. Biol. 4 (3) (1997) 241–259.
[17] V. Heun, Approximate protein folding in the HP side chain model on extended cubic lattices, Proceedings of the European Symposium on Algorithms, 1999.
[18] K.F. Lau, K.A. Dill, A lattice statistical mechanics model of the conformation and sequence spaces of proteins, Macromolecules 22 (1989) 3986–3997.
[19] K.F. Lau, K.A. Dill, Theory for protein mutability and biogenesis, Proc. Natl Acad. Sci. USA 87 (1990) 638–642.
[20] D. Lipman, J. Wilber, Proc. Roy. Soc. London 245 (8) (1991).

[21] A. Newman, A new algorithm for protein folding in the HP model, in: Proceedings of the 13th ACM–SIAM, Symposium on Discrete Algorithms, 2002.
[22] J.T. Ngo, J. Marks, Computational complexity of a problem in molecular-structure prediction, Protein Eng. 5 (4) (1992) 313–321.
[23] J.T. Ngo, J. Marks, M. Karplus, Computational Complexity, Protein Structure Prediction and the Levinthal Paradox, Birkhauser, Basel, 1994.
[24] A. Nayak, A. Sinclair, U. Zwick, Spatial codes and the hardness of string folding problems, Proceedings of the 9th ACM–SIAM Symposium on Discrete Algorithms, 1998, pp. 639–648.
[25] P.M. Pardalos, D. Shalloway, G.L. Xue, (Eds.), Global minimization of nonconvex energy functions: molecular conformation and protein folding, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, 1996. Vol. 23, American Mathematical Society.
[26] M. Paterson, T. Przytycka, On the complexity of string folding, Discrete Appl. Math. 71 (1996) 217–230.
[27] A. Šali, E. Shakhnovich, M. Karplus, How does a protein fold? Nature 369 (1994) 248–251.
[28] A. Šali, E. Shakhnovich, M. Karplus, Kinetics of protein folding: a lattice model study of the requirements for folding to the native state J. Mol. Biol. 235 (1994) 1614–1636.
[29] A. Sinclair, Algorithms for Random Generation and Counting: A Markov Chain Approach, Birkhäuser, Basel, 1993.
[30] R. Unger, J. Moult, Finding the lowest free energy conformation of a protein is a NP-hard problem: proof and implications, Bull. Math. Biol. 55 (1993) 1183–1198.
[31] R. Unger, J. Moult, Genetic algorithms for protein folding simulations, J. Mol. Biol. 231 (1) (1993) 75–81.
[32] R. Unger, J. Moult, A genetic algorithm for three dimensional protein folding simulations, Proceedings of the 5th International Conference on Genetic Algorithms, 1993, pp. 581–588.
[33] K. Yue, K. Dill, Folding proteins with a simple energy function and extensive conformational searching, Protein Sci. 5 (1996) 254–261.