Structure-approximating design of stable proteins in 2D HP model fortified by cysteine monomers

Alireza Hadj Khodabakhshi, Ján Maňuch, Arash Rafiey and Arvind Gupta

School of Computing Science
8888 University Drive, Simon Fraser University
Burnaby, BC, V5A 1S6, Canada
E-mail: alireza@cs.sfu.ca, jmanuch@sfu.ca, arafieyh@cs.sfu.ca, arvind@mitacs.cs

The inverse protein folding problem is that of designing an amino acid sequence which has a prescribed native protein fold. This problem arises in drug design where a particular structure is necessary to ensure proper protein-protein interactions. The input to the inverse protein folding problem is a shape and the goal is to design a protein sequence with a unique native fold that closely approximates the input shape. Gupta *et al.*[1] introduced a design in the 2D HP model of Dill that can be used to approximate any given (2D) shape. They conjectured that the protein sequences of their design are stable but only proved the stability for an infinite class of very basic structures. The HP model divides amino acids to two groups: hydrophobic (H) and polar (P), and considers only hydrophobic interactions between neighboring H amino in the energy formula. Another significant force acting during the protein folding are sulfide (SS) bridges between two cysteine amino acids. In this paper, we will enrich the HP model by adding cysteines as the third group of amino acids. A cysteine monomer acts as an H amino acid, but in addition two neighboring cysteines can form a bridge to further reduce the energy of the fold. We call our model the HPC model. We consider a subclass of linear structures designed in Gupta *et al.*[1] which is rich enough to approximate (although more coarsely) any given structure. We refine the structures for the HPC model by setting approximately a half of H amino acids to cysteine ones. We conjecture that these structures are stable under the HPC model and prove it under an additional assumption that non-cysteine amino acids act as cysteine ones, i.e., they tend to form their own bridges to reduce the energy. In the proof we will make an efficient use of a computational tool 2DHPSolver which significantly speeds up the progress in the technical part of the proof. This is a preliminary work, and we believe that the same techniques can be used to prove this result without the artificial assumption about non-cysteine H monomers.

Keywords: HP model; protein stability; protein design; 2D square lattice; cysteine.

1. Introduction

It has long been known that protein interactions depend on their native three-dimensional fold and understanding the processes and determining these folds is a long standing problem in molecular biology. Naturally occurring proteins fold so as to minimize total free energy. However, it is not known how a protein can choose the minimum energy fold amongst all possible folds.[2]

Many forces act on the protein which contribute to changes in free energy including hydrogen bonding, van der Waals interactions, intrinsic propensities, ion pairing, disulfide bridges and hydrophobic interactions. Of these, the most significant is hydrophobic interaction.[3] This led Dill to introduce the *Hydrophobic-Polar model*.[4] Here the 20 amino acids from which proteins are formed are replaced by two types of monomers: hydrophobic (H or '1') or polar (P or '0') depending on their affinity to water. To simplify the problem, the protein is laid out on vertices of a lattice with each monomer occupying exactly one vertex and neighboring monomers occupy neighboring vertices. The free energy is minimized when the maximum number of hydrophobic monomers are adjacent in the lattice. Therefore, the "native" folds are those with the maximum number of such HH contacts. Even though the HP model is the simplest model of the protein folding process, computationally it is an NP-hard problem for both the two-dimensional[5] and the three-dimensional[6] square lattices.

In many applications such as drug design, we are interested in the complement problem to protein folding: *inverse protein folding* or *protein design*. The *inverse protein folding problem* involves starting with a prescribed target fold or structure and designing an amino acid sequence whose native fold is the target (positive design). A major challenge in designing proteins that attain a specific native fold is to avoid proteins that have multiple native folds (negative design). We say that a protein is *stable* if its native fold is unique. In Gupta *et al.*,[1] a design in the 2D HP model that can be used to approximate any given (2D) shape was introduced and it was shown that approximated structures are native for designed proteins (positive design). It was conjectured that the protein sequences of their designed structures are also stable but only proved for an infinite class of very basic structures (arbitrary long "I" and "L" shapes), as well as computationally tested for over 48,000 structures (including all with up to 9 tiles). Design of stable proteins of arbitrary lengths in the HP model was also studied by Aichholzer *et al.*[7] (for 2D square lattice) and by Li *et al.*[8] (for 2D triangular lattice), motivated by a popular paper of Brian Hayes.[9] In this paper we aim to show stability for a subclass of the structures introduced by Gupta *et al.*[1] which is still rich enough to approximate (although more coarsely) any target shape.

In natural proteins, sulfide bridges between two cysteine monomers play an important role in improving stability of the protein structure.[10] We believe that enriching the HP model with the third type of monomers, cysteines, and incorporating sulfide bridges between two cysteines into energy model results in a model with even more stable designs. This added level of stability can help in proving formally that the designed proteins are indeed stable. We call this new model, the HPC model (hydrophobic-polar-cysteine). The cysteine monomers act as hydrophobic, but in addition two neighboring cysteines can form a bridge to further reduce the energy of the fold.

The class of structures which we use is a subset of *linear structures* introduced by Gupta *et al.*[1] They are formed by a sequence of "plus" shape tiles, cf. Figure 1(a), connected by overlapping two pairs of polar monomers (each coming from a different

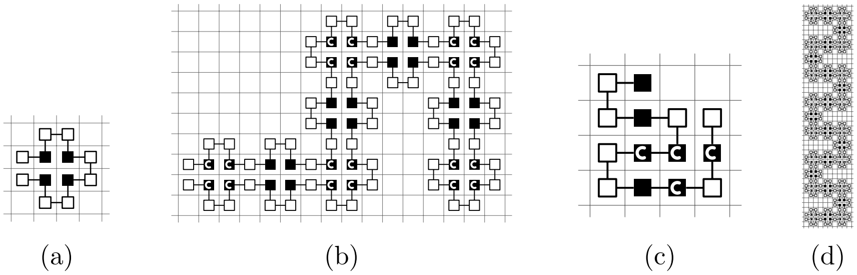

Fig. 1. (a) The basic building tile for constructible structures: black squares represent hydrophobic and white polar monomers. The lines between boxes represent the peptide bonds between consecutive monomers in the protein string. (b) An example of snake structure. The bending tiles use cysteines (black squares marked with C). (c) Example of energy calculation of a fold in HPC model. There are 5 contacts between hydrophobic monomers, thus the contact energy is -5. There are three potential sulfide bridges sharing a common vertex, hence only one can be used in the maximum matching. Thus the sulfide bridge energy is -2 and the total energy is -7.

tile). The structures are linear which means that every tile except the first and the last is attached to exactly two other tiles. In addition, we assume that the sequence of tiles has to change direction ("bend") in every odd tile. The hydrophobic monomers of these "bending" tiles are set to be cysteines, and all other hydrophobic monomers are non-cysteines, cf. Figure 1(b). We call these structures the *snake structures.* Note that approximately 40% of all monomers in snaked structures are hydrophobic and half of those are cysteines. Thus approximately 20% of all monomers are cysteines. Although, the most of naturally occurring proteins have much smaller frequency of cysteines, there are some with the same or even higher ratios: 1EZG (antifreeze protein from the beetle[11]) with 19.5% ratio of cysteines and the protein isolated from the chorion of the domesticated silkmoth[12] with 30% ratio.

Note that the snake structures can still approximate any given shape, although more coarsely than the linear structures. The idea of approximating a given shape with a linear structure is to draw a non-intersecting curve consisting of horizontal and vertical line segments. Each line segment is a linear chain of basic tiles depicted in Figure 1(a). At first glance, the snake structures seem more restricted than linear structures, as the line segments they use are very short and have the same size (3 tiles long). However, one can simulate arbitrary long line segments with snake structures forming a zig-zag pattern, cf. Figure 1(d).

We conjecture that the proteins for the snake structures are stable in the HPC model and that this can be proved using the techniques presented in this paper. These techniques are (i) the case analysis (also used in Gupta *et al.*[1]) and (ii) the induction on diagonals. Furthermore, to increase the power of the case analysis technique, we developed a program called "2DHPSolver" for semi-automatic proving of hypothesis about the folds of proteins of the designed structures. In this preliminary paper, we demonstrate the power of our techniques by showing that all snake structures are stable in the "strong" HPC model. The strong HPC model adds an artificial assumption that non-cysteine monomers form bridges as well to minimize the energy. We are currently working on extending our proof for the "proper" HPC model. Note that 2DHPSolver can be used for all three models: HP, HPC and strong HPC by setting the appropriate parameters.

2. Definitions

In this section we introduce the HPC model and fix some terminology used in the paper.

2.1. *Hydrophobic-polar-cysteine (HPC) model*

Proteins are chains of monomers where each monomer is either hydrophobic or polar. Furthermore, we will distinguish two types of hydrophobic monomers: cysteines which can form sulfide bridges to decrease the energy of the fold and non-cysteines. We can represent a protein chain as a string $p = p_1 p_2 \dots p_{|p|}$ in $\{0, 1, 2\}^*$, where "0" represents a polar monomer, "1" a hydrophobic non-cysteine monomer and "2" a cysteine monomer.

The proteins are folded onto the regular lattice. A *fold* of a protein p is embedding of a path of length n into lattice, i.e., vertices of the path are mapped into distinct lattice vertices and two consecutive vertices of the path are mapped to lattice vertices connected by an edge (a peptide bond). In this paper we use the 2D square lattice.

A protein will fold into a fold with the minimum free energy, also called a *native fold*. In the HP model only hydrophobic interactions between two adjacent hydrophobic monomers which are not consecutive in the protein sequence (*contacts*) are considered in the energy model, with each contact contributing with -1 to the total energy. In addition, in the HPC model, two adjacent non-consecutive cysteines can form a sulfide bridge contributing with -2 to the total energy. (Note that the results in the paper are independent on the exact value of the energy of sulfide bridge, as long as it is negative, and therefore we did not research on determination of the correct value for this energy.) However, each cysteine can be involved in at most one sulfide bridge. More formally, any two adjacent non-consecutive hydrophobic monomers (cysteine or non-cysteine) form a contact and the contact energy is equal to -1 times the number of contacts; and any two adjacent non-consecutive cysteines form a *potential* sulfide bridge and the sulfide-bridge energy is equal to -2 times the number of matches in the maximum matching in the graph of potential sulfide bridges. The total energy is equal to the sum of the contact and sulfide bridge energies. For example, the energy of the fold in Figure 1(c) is $(-5) + (-2) = -7$. Note that there might be several native folds for a given protein. A protein with a unique native fold is called *stable* protein.

2.2. *Snake structures*

In Gupta *et al.*,[1] a wide class of 2D structures, called *constructible structures*, was introduced. They are formed by a sequence of "plus" shape tiles, cf. Figure 1(a), connected by overlapping two pairs of polar monomers (each coming from different tile). It was conjectured that these structures are stable and proved for two very simple subclasses of the linear structures, namely for L_0 and L_1 structures. The L_0 and L_1 structures consist of an arbitrary large sequence of tiles in the shape of a straight line and the letter L, respectively. Note that although L_1 structures are still quite simple, the proof of their stability involves analysis of a large number of cases.

In this paper, we consider a rich subclass of constructible structures. The structures in the subclass are *linear* which means that every tile t_i except the first t_1 and the last t_n is attached to exactly two other tiles t_{i-1} and t_{i+1} (and the first and the last ones are attached to only one tile, t_2 and t_{n-1}, respectively). In addition, we assume that the sequence of tiles has to change direction ("bend") in every odd tile. The hydrophobic monomers of these "bending" tiles are set to be cysteines, and all other hydrophobic monomers are non-cysteines, cf. Figure 1(b). We call these structures the *snake structures* and their proteins the *snake proteins.*

2.3. *The strong HPC model*

We conjecture that the snake proteins are stable in the HPC model, and furthermore that it can be proved with techniques presented in this paper. As a preliminary result, we present the proof that the snake proteins are stable in the artificial *strong* HPC model. In this model, the energy function consists of three parts (first two are the same as in the HPC model): (i) the bond energy, (ii) the sulfide bridge energy and (iii) non-cysteine bridge energy. The last part is equal to -2 times the number of matches (pairings) in the maximum matching of the graph of potential non-cysteine bridges, where there is a potential non-cysteine bridge between any two non-consecutive adjacent non-cysteine hydrophobic monomers. Thus, the fold in Figure 1(c) had energy -9 in the strong HPC model. This energy model can be interpreted as follows: we assume that we have two types of cysteine-like hydrophobic monomers each forming bridges, but no bridges are possible between "cysteines" of different types. Furthermore, in our design we only use cysteine-like hydrophobic monomers (in bending tiles we use the first type, in non-bending tiles the second type).

3. Proof techniques

In this section we review some basic proof techniques used in this paper.

3.1. *Saturated folds*

The proteins used by Gupta *et al.*[1] in the HP model and the snake proteins in HPC or strong HPC models have a special property. The energy of their native folds is the smallest possible with respect to the numbers of hydrophobic cysteine and non-cysteine monomers contained in the proteins. We call such folds *saturated.* In saturated folds all parts of energy function produce minimum possible values. This means: (i) every hydrophobic monomer (cysteine or non-cysteine) has two contacts with other monomers; (ii) there is a sulfide bridge matching containing all or all but one cysteine monomers; and (iii) in the strong HPC model, there is a non-cysteine bridge matching containing all or all but one non-cysteine monomers. Obviously, a saturated fold of a protein must be native, and furthermore, if there is a saturated fold of a protein, then all native folds of this protein must be saturated.

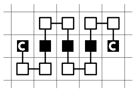

Fig. 2. Forbidden configuration in saturated fold under the strong HPC model

To illustrate the main difference between the HPC and the strong HPC models, consider a part of the fold in Figure 2 and assume that the number of non-cysteine hydrophobic monomers in the whole fold is even. In the HPC model, it is possible to extend the configuration in the figure to a complete saturated fold, while in the strong HPC model, this is not possible, as the non-cysteine hydrophobic monomers will never form a complete matching. Thus, the power of strong HPC model is in ability to faster eliminate a lot of cases, for instance, cases containing a configuration depicted in Figure 2, while in the HPC model the same proof will require a much deeper case analysis.

3.2. *2DHPSolver: a semi-automatic prover*

2DHPSolver is a tool for proving the uniqueness of a protein design in 2D square lattice under the HP, HPC or strong HPC models. 2DHPSolver is not specifically designed to analyze the snake structures or even the constructible structures. It can be used to prove the stability of any 2D HP design based on the induction on the boundaries. It starts with an initial configuration (initial field) which is given as the input to the program. In each iteration, one of the fields is replaced by all possible extensions at one point in the field specified by user. Note that in displayed fields red 1 represents a cysteine monomer, blue 1 a non-cysteine monomer and finally, uncolored 1 is hydrophobic monomer, but it is not known whether it is cysteine or not.

These extensions are one of the following type:

- extending a path (of consecutive monomers in the protein string);
- extending a 1-path (of a chain of hydrophobic monomers connected with contacts);
- coloring an uncolored H monomer.

There are 6 ways to extend a path, 3 ways to extend a one-path and 2 ways to color an uncolored H monomer. For each of these possibilities, 2DHPSolver creates a new field which is then checked to see if it violates the rules of the design. Those which do not violate the design rules will replace the original field.

However, this approach will result in producing too many fields, which makes it hard for the user to keep track of. Therefore, 2DHPSolver contains utilities to assist in automatically finding an extending sequence for a field which leads to either no valid configurations, in which case the field is automatically removed, or to only one valid configuration, in which case the field is replaced by the new more completed configuration. This process is referred to as a *self-extension.* The time required for searching for such extending sequence depends on the depth of the search, which can be specified by user through two parameters ”depth” and ”max-extensions”. Thus, leaving the whole process of proving to 2DHPSolver by setting the parameters to high values is not practical as it could take enormous amount of time. Instead, one should set parameters to moderate values and use intuition in choosing the next extension point when 2DHPSolver is unable to automatically find self-extending sequences. Note that these parameters can be changed at any time during the use of the program by the user.

2DHPSolver is developed using C++ and its source code is freely available to all users under the GNU Public Licence (GLP). For more information on 2DHP-Solver and to obtain a copy of the source codes please visit `http://www.sfu.ca/~ahadjkho/2dhpsolver/`.

4. Stability of the snake structures

In this section we prove that the protein of any snake structure is stable. Let S be a snake structure (fold), p its protein and let F be an arbitrary native (i.e., saturated) fold of p.

Define a path in F as a sequence of vertices such that no vertex appears twice and any pair of consecutive vertices in the path are connected by peptide bonds. A cycle is a path whose start and end vertices are connected by a peptide bond.

For $i \in \{\mathsf{0}, \mathsf{1}, \mathsf{2}\}$, an i-vertex in the fold F is a lattice vertex (square) containing a monomer i. For instance, a square containing a cysteine monomer in F is called a 2-vertex. An H-vertex is a vertex which is either 1-vertex or 2-vertex. Define a 1-path in F to be a sequence of H-vertices such that each H-vertex appears once and any pair of consecutive ones form an HH contact. A 1-cycle in F is a 1-path whose first and last vertices form an HH contact. A 1-cycle of length 4 is called a core in F.

A core c is called *monochromatic* if all its H-vertices are either cysteines or non-cysteines. Let c_1 and c_2 be two cores in F. We say, c_1 and c_2 are adjacent if there is a path of length 2 or 3 between an H-vertex of c_1 and an H-vertex of c_2. We say c_1 and c_2 are correctly aligned if they are adjacent in one of the forms in Figure 3.

In what follows we prove that every H-vertex in F belongs to a monochromatic core and the cores are correctly aligned.

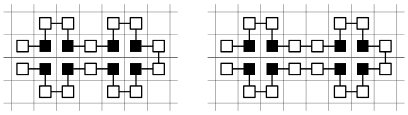

Fig. 3. Correctly aligned cores.

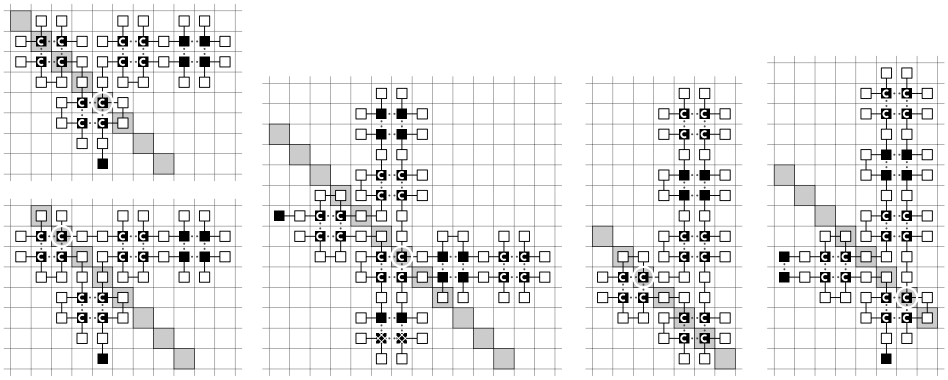

Fig. 4. Configurations with misaligned cores. The circled cysteine monomer is the one used as the starting point in induction proof by 2DHPSolver. The hatched black square depict hydrophobic monomers for which it was not yet determined whether they are cysteines or non-cysteines.

Lemma 4.1. *Every* H*-vertex in F belongs to a monochromatic core and all the cores are either correctly aligned or there is only one occurrence of one of the configurations depicted in Figure 4, in which* 3 *cores are not correctly aligned while others are correctly aligned.*

Proof. For any integer i, let SW_i be the set of lattice vertices $\{[x, y]; x+y = i\}$. Let m be the maximum number such that SW_i, $i < m$ does not contain any H-vertex, i.e., SW_m is a boundary of diagonal rectangle enclosing all H-vertices.

We start by proving the following claim.

Claim 4.1. *If there is an* H*-vertex w on SW_i then*

(1) w is on a monochromatic core c; and

(2) if c is adjacent to core c' which has a H*-vertex, on SW_j, $j < i$, then either c and c' are correctly aligned or one of the configurations depicted in Figure 4 occurs.*

(3) if c is adjacent to core c' which has a H*-vertex, on SW_j, $j > i$, then either c and c' are correctly aligned or one of the configurations depicted in Figure 4 occurs.*

Proof. We prove the (1) and (2) by induction on i. Note that one can prove (1) and (3) in a similar way.

For the base case, assume that w is an H-vertex on SW_m. It is enough to show that w is in a monochromatic core (case (1)). Since w lies on the boundary, this can be easily proved by short case analysis or by 2DHPSolver.

Now suppose $i > m$. Suppose none of the configuration in Figure 4 happens. By induction hypothesis, the part of the fold F that lies between SW_m and SW_{i-1} contains only correctly aligned monochromatic cores. We prove that any H-vertex w located on SW_i is on a monochromatic core c and if c is adjacent to a core c' which has a $\mathsf{1}$-vertex on $SW_{k'}$ for some $k' < i$ then c is correctly aligned to c'.

We show that if (1) and (2) does not happen for w then we see a subsequence in F which is not in p. This is done by enumerative case analysis of all possible extensions of this configuration and showing that each branch will end in a configuration has a subsequence not in p.

This process requires the analysis of many configurations which is very hard and time consuming to do manually. Therefore, we used 2DHPSolver to assist in analyzing the resulting configurations. The program generated proof of this step of the induction can be found on our website at `http://www.sfu.ca/~ahadjkho/2dhpsolver/`. Please be advised that this is a PDF document containing 2707 pages and 16543 images.

One can see that in all of the configurations depicted in Figure 4, there are 3 cysteine cores c, c' and c'' which are adjacent pairwise and contain two occurrences of the subsequence $es = (\mathsf{020})^4$. The subsequence es occurs exactly twice in S_n and that is in t_1 and t_n. □

Analogously the SE_i is the set of vertices $\{[x, y]; x - y = i\}$ of the lattice. We have a similar claim for an H-vertex on SE_i. In each of the configurations in Figure 4 subsequence es occurs twice. Combining the two claims completes the proof of the lemma.

Theorem 4.1. *Every* H*-vertex in* F *belongs to a monochromatic core and all the cores are correctly aligned.*

Proof. By Lemma 4.1, every H-vertex is on a core. Consider a graph G defined as follows. For every core c of F, let x_c be a vertex in G. Furthermore, two vertices x_c, and $x_{c'}$ are connected in G if and only if cores c and c' are adjacent in F. We show that G is acyclic. For the contrary, let C be a cycle in G. If all the cores corresponded to vertices of C in F are correctly aligned we get a closed subsequence of Q which is not the entire Q. Thus C contains vertex x_c which c is one of the core shown in Figure 4. Each core c in Figure 4 is adjacent to at least three other cores in F. Therefore vertex x_c has degree at least three in G. If C is of length more than three then C contains only two of the three cores in Figure 4 and all other cores of F corresponded to C' are correctly aligned. However again we get a close subsequence of Q which is not the entire Q. Thus C has only three vertices, since x_c is of degree 2 and there is only one cycle in G, there is one vertex of degree 1. Now we have three occurrence of $(\mathsf{020})^4$ in F, a contradiction. Therefore G is acyclic. Similarly G

has no vertex of degree more than 2 as otherwise there would be three occurrences of $(020)^4$ in F. Thus all the cores are correctly aligned and each core is adjacent to at most two other cores, except the first and the last one. Note that since there is no vertex of degree 3 in G, every core in F is adjacent to other cores in a way that cores in S are connected. Now the first core c_1 in F (c_1 is adjacent to exactly one core) is correspond to t_1 of S. By continuing the sequence of p in core c_i of F and t_i of S for $i > 1$ we see that F has the same structure as S. Thus F is unique. □

5. Conclusions

In this paper we have enriched the HP model of Dill with the third type of amino acids, cysteines, and a new interaction acting between monomers, disulfide bridges. We consider a robust subclass of constructible structures introduced by Gupta *et al.*[1] able to approximate any given shape, and refine these structures for the new HP-cysteine model. We believe that introduction of cysteine monomers into structure design improves the stability of designed structures which in turn helps in proving the stability. To formally prove that the considered structures are stable, it is necessary to consider an enormous number of cases. For that reason, we have developed semi-automated prover 2DHPSolver. Using 2DHPSolver we are able to prove stability under one additional assumption on the HPC model. We are currently working on the proof of stability without this assumption.

We conjecture that use of cysteines in the design of proteins might help to improve their stability. To verify this, we would like to extend our results to 3D lattice models and test them using existing protein folding software.

References

1. A. Gupta, J. Maňuch and L. Stacho, *Journal of Computational Biology* **12**, 1328 (2005).
2. K. A. Dill, S. Bromberg, K. Yue, K. M. Fiebig, D. P. Yee, P. D. Thomas and H. S. Chan, *Protein Science* **4**, 561 (1995).
3. K. A. Dill, *Biochemistry* **29**, 7133 (1990).
4. K. A. Dill, *Biochemistry* **24**, 1501 (1985).
5. P. Crescenzi, D. Goldman, C. Papadimitriou, A. Piccolboni and M. Yannakakis, On the complexity of protein folding, in *Proc. of STOC'98*, 1998.
6. B. Berger and T. Leighton, *J. Comp. Biol.* **5**, 27 (1998).
7. O. Aichholzer, D. Bremner, E. Demaine, H. Meijer, V. Sacristán and M. Soss, *Computational Geometry: Theory and Applications* **25**, 139 (2003).
8. Z. Li, X. Zhang and L. Chen, *Appl. Bioinformatics* **4**, 105 (2005).
9. B. Hayes, *American Scientist* **86**, 216 (1998).
10. R. Jaenicke, *Eur. J. Biochem.* **202**, 715 (1991).
11. Y. Liou, A. Tocilj, P. Davies and Z. Jia, *Nature* **406**, 322 (2000).
12. G. C. Rodakis and F. C. Kafatos, *Proc. Natl. Acad. Sci. USA* **79**, 3551 (1982).