JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 5, Number 3, 1998
Mary Ann Liebert, Inc.
Pp. 423–465

On the Complexity of Protein Folding

PIERLUIGI CRESCENZI,[1] DEBORAH GOLDMAN,[2] CHRISTOS PAPADIMITRIOU,[3]
ANTONIO PICCOLBONI,[4] and MIHALIS YANNAKAKIS[5]

ABSTRACT

We show that the protein folding problem in the two-dimensional H-P model is NP-complete.

Key words: protein folding, computational complexity.

1. INTRODUCTION

PROTEINS ARE POLYMER CHAINS CONSISTING of monomers of twenty different kinds. Much of the genetic information in the DNA contains the sequence information of proteins, with three nucleotides encoding one monomer. In turn, proteins in an organism *fold* to form a very specific geometric pattern, known as the protein's *native state.* It is this geometric pattern that determines the macroscopic properties, behavior, and function of a protein. It is in general reasonably stable and unique.

The mapping from DNA sequences to monomer sequences is simple and very well-understood. In contrast, the mapping from the sequence of a protein to the geometric configuration of its native state—the "second half of the genetic code" (King, 1989)—is much more intricate and complex, and less understood; it has been the subject of intense investigation for decades. It seems clear that the forces underlying protein folding are the interactions between their monomers; recently, the view that *non-local* interactions dominate this process has been gaining ground (Dill *et al.*, 1995). To test this and other hypotheses concerning protein folding, researchers resorted to *simplified models* of proteins, mathematical abstractions of proteins that hide many aspects and exaggerate the effect of others; analysis and computer simulation of such models can then be compared to experimental results with actual proteins, to determine whether the emphasized aspects are indeed the dominant ones.

Perhaps the most successful and best-studied such model, and the one with apparently the best match with experiments,* is the *two-dimensional hydrophobichydrophilic model*, or HP model, proposed by Dill (1990). In this model it is assumed that the protein is a sequence of 0's and 1's, and folding entails embedding the sequence in the two-dimensional lattice (see Figure 1). Each such folding is evaluated with a *score*, equal to the number of pairs of 1's that are adjacent in the lattice without being adjacent in the sequence; for example, in Figure 1, the score is five, corresponding to the five pairs of 1's connected by dotted lines. The score

[1]Dipartimento di Sistemi e Informatica, Università di Firenze, Via C. Lombroso 6/17, 50134, Firenze, Italy.
[2]Department of Mathematics, University of California at Berkeley, Berkeley, California 94720.
[3]Computer Science Division, University of California at Berkeley, Berkeley, California 94720.
[4]Università di Milano, Dipartimento di Scienze dell'Informazione, Via Comelico 39/41, 20135 Milano, Italy.
[5]Bell Laboratories, 700 Mountain Avenue, Murray Hill, New Jersey 07974.

**Chan and Dill (1993) state that "for chain lengths for which exhaustive enumeration is possible (up to about 30 monomers), two-dimensional models more accurately represent the physically important surface-interior ratios of proteins than do 3-dimensional models."*

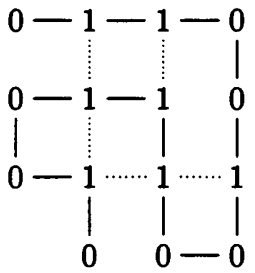


FIG. 1. Embedding in the two-dimensional lattice.

captures a simple model of energy minimization, in which the "hydrophobic" 1's tend to be close to each other and thus avoid exposure, while 0's are neutral. It is assumed in this model that the native folded state is the one that maximizes score. The protein folding problem, in this framework, can be stated as follows: "given a sequence of 0's and 1's, find the embedding in the lattice that maximizes score."

That proteins fold so as to minimize energy has been accepted for decades. This view quickly leads to a puzzling aspect of the problem, known as *Levinthal's paradox*, which can be paraphrased as follows: "How can a folding protein choose so quickly among so many possible foldings the one with minimum energy?" (Dill *et al.*, 1995). The result presented in the following can thus be seen as a more compelling restatement of that paradox, since it implies that finding the optimum folding in the two-dimensional HP-model—the simplest abstraction of the protein folding problem one finds in the literature, and presumably a vast simplification of the true detailed 3-dimensional energy minimization problem in actual proteins—is NP-complete, that is to say, among the provably hardest problems of the sort alluded to by the paradox, in which we must optimize among an astronomical population of states.

There have been several NP-completeness results related to protein folding in the literature. A few years ago, several authors pointed out that certain general restatements of the problem, in which monomers attract or repel each other in ways that are general and can be used in encoding, are NP-complete (Fraenkel, 1993; Ngo *et al.*, 1994; Unger and Moult, 1993). More interestingly, it was proved in Paterson and Przytycka (1996) that a combinatorial generalization of the HP model to an infinite alphabet, of which one symbol is neutral like HP's 0 symbol, and the score counts the number of adjacencies of elements with the same symbol, is NP-complete. More recently, Nayak *et al.* (1998) improved this to a finite, albeit very large alphabet. The present result is the first to settle the complexity of the simple two-dimensional HP model actually proposed in the literature as the ultimate simplification of the protein folding problem. The HP problem has been attacked from the point of view of approximation algorithms (Hart and Istrail, 1995); the present result sheds little light on this aspect of the problem, as our reduction is not in any interesting way approximation-preserving. Recently and independently, Berger and Leighton (1997) proved that the three-dimensional protein folding problem in the HP model is NP-complete; in fact, the approximability implications of their result are stronger than ours.

Our reduction is from the Hamilton cycle problem. As is common in previous proofs of weaker results, we start by showing that the folding problem for *sets of sequences* (that is, when many sequences are to be optimally folded) is NP-complete (Theorem 1 in Section 2). We then proceed to establish the result for a single sequence, by resorting to certain interesting variants of the planar Hamilton cycle problem (Theorem 17 in Section 3). In our proof we utilize an idea of Trevisan (1996), whereby graphs can be embedded in the hypercube so that adjacency is captured by Hamming distance.

Our proof captures one of the basic intuitions of the HP model, namely that hydrophobic monomers will tend to form a large "sphere" (in the two-dimensional lattice, a large hydrophobic square). Impurities in this sphere then must be aligned optimally to maximize score, and it is the complexity of this alignment that our proof exploits. Finally, in Section 4 we briefly discuss a version of our proof (in fact, without the planarity complication) which settles the NP-completeness of the three-dimensional version of the protein folding problem in the HP model—and in fact, the MAX-SNP-completeness of the problem of minimizing losses in three dimensions.

2. THE MULTISTRING FOLDING PROBLEM

The *two-dimensional lattice* is the graph, (Z^2, L), with node set Z^2 (all points in the Euclidean plane with integer coordinates), and edge set $L = \{((x, y), (x', y')) : |x - x'| + |y - y'| = 1\}$. Consider a set of strings

$S = \{s_1, \ldots, s_m\}$ from the alphabet $\{0, 1\}$. A *folding* of these strings is an embedding of S into the lattice, that is to say, a one-to-one mapping f from the set $\{(i, j) : 1 \leq i \leq m, 1 \leq j \leq |s_i|\}$ to Z^2 such that for all $1 \leq i \leq m, 1 \leq j \leq |s_i| - 1$ we have $(f(i, j), f(i, j + 1)) \in L$. Fix a folding f; the points $f(i, j)$ and $f(i, j + 1)$ are called *f-neighbors*. The score of a folding f is the number of edges $\{(x, y), (x', y') \in L\}$ such that (a) (x, y) and (x', y') are not f-neighbors, and (b) each of these points is the image under f of a pair (i, j) such that the jth symbol of s_i is a 1. An edge of the lattice $\{(x, y), (x', y')\} \in L$ is said to be a *loss* if (a) these points are not f-neighbors, and (b) exactly one of these two points is the image under f of a pair (i, j) such that the jth symbol of s_i is a 1. Each position in a string containing a 1, and which is not the first or the last, can participate in zero, one, or two losses.

The MULTISTRING FOLDING PROBLEM is the following: given a set of strings $s_1, \ldots, s_m \in \{0, 1\}^*$ and an integer E, is there a folding with E or fewer losses? If, as is the case in the strings we construct, no string starts or ends in a 1, then it is easy to see that the total score of a folding is equal to twice the number of 1's, minus the losses, divided by two; hence, minimizing losses is the same as maximizing score, the traditional way of stating the protein folding problem.

In this section we prove the following theorem:

Theorem 1. *The* MULTISTRING FOLDING PROBLEM *is NP-complete.*

In the next section we shall show that the problem remains NP-complete even if there is only one string.

2.1. *Description of the reduction*

We start from the following NP-complete problem: Given a graph $G = (V, E)$ with n nodes of degree four or less, and two nodes $v_1, v_n \in V$, is there a Hamilton path from v_1 to v_n?

As a preliminary step in our reduction, we first map the nodes in G to the hypercube according to a map used by Trevisan (1996). Using Hadamard codes, he showed that there exists a function, which we call T, mapping the n nodes of the graph to codewords in $\{0, 1\}^{8n}$ such that the images of two unconnected nodes have Hamming distance strictly greater than two nodes connected by an edge; in particular, applying T to the nodes of G we find, for $i \neq j$ in $\{1, 2, \ldots, n\}$:

1. If $\{v_i, v_j\}$ is an edge in G, then $d_H(T(v_i), T(v_j)) = \frac{7}{2}n$;
2. If $\{v_i, v_j\}$ is not an edge, then $d_H(T(v_i), T(v_j)) = 4n$.

We assume without loss of generality that n is a power of 2. Notice that if there is a Hamilton path from v_1 to v_n in G, then there is a Hamilton path from $T(v_1)$ to $T(v_n)$ in the Hamming space of length $\frac{7}{2}(n - 1)n$; otherwise, if there is no Hamilton path from v_1 to v_n in G, then any Hamilton path from $T(v_1)$ to $T(v_n)$ must have length strictly greater than $\frac{7}{2}(n - 1)n$. We note, finally, that the function T may be chosen so that $T(v_1)$ and $T(v_n)$ contain at most as many 0's as $T(v_i)$, for any $i \in \{1, \ldots, n\}$.

We now construct a set of strings S and an integer E, such that there is a Hamilton path from v_1 to v_n in G if and only if there is a folding of the strings in S with at most E losses. The allowed number of losses is $E = 7(n - 1)n$. As for the set of strings S, let $L = 180n^{14}$. S will contain L strings, $s_1, \ldots, s_L$, such that the first L/n strings correspond to node v_1, the second L/n to node v_2, and so on. All strings, with the exception of s_1 and S_L, will be constructed similarly and so that strings corresponding to the same city are identical. Define

$$q = \lceil \sqrt{LE} \rceil$$

and let c be an even positive integer to be specified later. Let d (suggesting *dense*) denote the string

$$d = \prod_{i=1}^{L/90} 1^8 0,$$

let m (suggesting *middle*) denote the string

$$m = \prod_{i=1}^{2E-16n} 1^{cq} 0,$$

and, for $i \in [n]$, let $t(i)$ (suggesting the *Trevisan code*) denote the string

$$t(i) = \prod_{l=1}^{8n} \left(1^{cq} T(v_i)_l\right)^2.$$

The description of the strings s_2 through s_{L-1} follows: for $k \in \{2, \ldots, L-1\}$ and $i = \lceil \frac{nk}{L} \rceil$,

$$s_k = 0^{L^4} d 1^{L/10} d 1^{L/5 - 2E(cq+1)} m t(i) 1^{2L/5} d 0^{L^4}.$$

Notice that each string contains a prefix and suffix of L^4 0's. There are three dense substrings of 1's and 0's in each string. Finally, toward the middle of the string, there is the substring $mt(i)$ of length

$$2E\left(c\lceil\sqrt{LE}\rceil + 1\right)$$

which we call the *sparse substring*, which consists of the sparse string m and the sparse string $t(i)$ containing two copies of the Trevisan code (interspersed between strings of 1's). The substring lying between the prefix and suffix, called the *internal substring*, has total length L.

The strings s_1 and s_L, called the *flanks*, are identical, respectively, to strings s_2 and s_{L-1} except for the fact that 4 0's are inserted between every other pair of two adjacent 1's in s_2 and s_{L-1}, beginning with the first pair of 1's in each maximal substring of adjacent 1's. Notice that, by placing two copies of each bit of the Trevisan code next to each other (separated by 1's), we have arranged for all maximal substrings of adjacent 1's to have even length. Formally, setting

$$d' = \prod_{i=1}^{L/90} (10^4 1)^4 0,$$

$$m' = \prod_{i=1}^{2E-16n} (10^4 1)^{cq/2} 0,$$

and, for $i \in [n]$,

$$t(i)' = \prod_{l=1}^{8n} \begin{cases} \left[(10^4 1)^{cq/2} T(v_i)_l\right]^2 & \text{if } T(v_i)_l = 0 \\ (10^4 1)^{cq/2} T(v_i)_l 0^4 1 (10^4 1)^{cq/2-1} 10^4 T(v_i)_l & \text{if } T(v_i)_l = 1 \end{cases},$$

for each $k \in \{1, L\}$ and $i = \lceil \frac{nk}{L} \rceil$,

$$s_k = 0^{L^4} d'(10^4 1)^{L/20} d'(10^4 1)^{L/10 - E(cq+1)} m' t(i)'(10^4 1)^{L/5} d' 0^{L^4}.$$

This completes the description of the reduction.

2.2. *The intended folding*

In this subsection we show that if there is a Hamilton path from v_1 to v_n in G, then there is a folding with at most E losses. This is rather easy; the hard direction is the opposite, proved in the next subsection.

Let $(v_{i_1}, v_{i_2}, \ldots, v_{i_n})$ be the order of the nodes contained in a Hamilton path, where $v_{i_1} = v_1$ and $v_{i_n} = v_n$. We construct a folding, called the *intended folding*, by arranging the non-flank strings, $s_2, \ldots, s_{L-1}$, vertically to form a $(2L^4 + L) \times (L - 2)$ rectangle as follows: all the strings corresponding to node v_{i_1} are placed adjacent with their first bits in the same horizontal line at the left side of the rectangle, then the strings corresponding to node v_{i_2} are placed next, and so on until the strings corresponding to node v_{i_n} complete the rectangle. The flank strings s_1 and s_L, which differ only from s_2 and s_{L-1} through the addition of groups of 4 0's, also have vertical orientation except for the fact that the 4-zero groups are bent to the left and right, respectively. In this way, the flanks s_1 and s_L can be placed, respectively, adjacent to the left and right sides of the rectangle so that their first bits are in line with the first bits of the other strings and so that, since the 4-zero groups have been excluded, the resulting patterns of bits adjacent to the rectangle are exactly the strings corresponding to nodes v_1 and v_n. A schematic drawing of the intended folding appears in Figure 2. Note that in the intended folding the central $L \times L$ square is composed primarily of 1's with some horizontal lines of 0's and horizontal

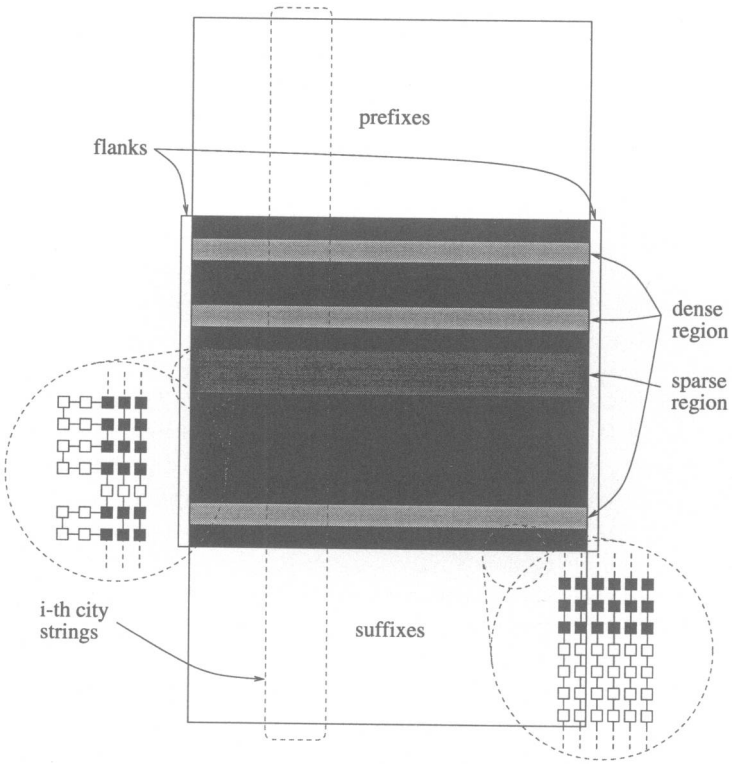


FIG. 2. The intended folding.

lines containing code bits running through it. It should be clear that the only place where a loss may occur is where two code bits $T(v_{i_j})_l$ and $T(v_{i_{j+1}})_l$ from the Trevisan code corresponding to two different nodes v_{i_j} and $v_{i_{j+1}}$ are adjacent. However, by the properties of the Trevisan code and because we have arranged the order of the identical groups of strings to be the same as the order $(v_{i_1}, v_{i_2}, \ldots, v_{i_n})$ of the Hamilton path, we are guaranteed that the two copies of the Trevisan code result in at most $2\frac{7}{2}(n-1)n = E$ adjacent but not neighboring 0-1 pairs. Therefore, the folding has at most E losses, and one direction has been proved.

We will refer to the intended folding in the following subsection, so we end this subsection by stating a few definitions. The boundary of the $L \times L$ square will be referred to as the *intended boundary*, any point contained in it as an *intended boundary point* and its four corner points as *intended corners*. Any point contained in the square will be referred to as an *internal point* and internal points which are not on the intended boundary will be called *inside points*.

2.3. *The converse*

We begin by presenting an overview of the (rather involved) proof of the converse. We consider a folding of the strings with at most E losses; we have to show that it is the intended folding corresponding to a Hamilton path in G. We define the region R to consist of all internal points in the $L \times L$ square of the intended folding, as well as all points surrounded by such points. We first prove that the largest connected component of this region has area at least $L^2 - O(E)$ and perimeter at most $4L + O(E)$ (Lemmas 2 through 7), and therefore the smallest surrounding rectangle has sides of length

$$L \pm O(\sqrt{LE}),$$

and there is a square with sides of length

$$L - O(\sqrt{LE})$$

contained in R (Lemmas 8 and 9). We then consider a string passing through the center of this rectangle, and prove that it is "relatively straight," i.e., proceeding without too many bendings, from one end of the square to the opposite end (Lemma 10 and Corollary 11). We then prove that any string which passes through a narrow horizontal strip traverses the square from its top to the bottom side, and that in fact almost all strings traverse

the square this way (Lemmas 12 and 13 and Corollary 14). It follows that the folding is the intended one, and corresponds to a Hamilton path in G.

Now we present the full proof that if there is a folding of the strings in S with at most E losses, then there is a Hamilton path from v_1 to v_n in G. Let f be a folding of the strings in Z^2 such that the resulting number of losses is less than or equal to E. Since the embedding is injective, we will identify bits in the strings with their images in Z^2, and call a point in Z^2 a *0* or a *1* if it has a preimage which is, respectively, a 0 or a 1; otherwise, the point will be called *empty*.

Consider the region R in Z^2 which consists of all internal points (points in the $L \times L$ square in the intended folding) as well as all points surrounded by a discrete closed curve of internal points, where, for the purposes of a discrete curve, two points (x_1, y_1) and (x_2, y_2) in Z^2 may be jointed if they are adjacent (jointed by a vertical or horizontal edge in the lattice Z^2), or if $|x_1 - x_2| = 1$ and $|y_1 - y_2| = 1$—that is, we allow diagonal edges. We include points surrounded by a discrete curve of internal points in R so that R will not contain any "holes" (this definition is well-given, since a discrete analog of the Jordan curve theorem holds in this context, see (Khalimsky *et al.*, 1990)). Let R_C be the largest connected subset of R, where for connectivity we allow only vertical or horizontal edges. We will prove that R_C looks very much like an $L \times L$ square and that, for the most part, strings passing through R_C are approximately straight and parallel.

It will be helpful to visualize R not only as a collection of points but also as a collection of continuous regions. By visualizing each point contained in R as a unit square centered at the point and parallel to the axes, then subsets of R may be said to have perimeter and area, denoted by the functions $Perim(\cdot)$ and $Area(\cdot)$. Our immediate goal will be to prove lemmas bounding the perimeter and area of R_C.

We begin by bounding the perimeter. Define the boundary of a region A, denoted $Bdary(A)$, to consist of all points in A adjacent to at least one point not in A. Then the perimeter of any set of points consisting of connected components of R is equal to the number of boundary points it contains plus the number of convex corners formed by points on the boundary of the region. We make a few further definitions before we state our first lemma. We call an internal point *straight* if its two neighbors lie in the same vertical or horizontal line; otherwise, the point is *bent*. A point $q \in Z^2$ is said to be *within distance d* (*vertical distance d*) of a point $p \in Z^2$ if q is reachable from p using a path of at most d vertical and horizontal edges (at most d vertical edges and any number of horizontal edges). A point $p \in Z^2$ is said to have a *loss within distance d* (*vertical distance d*) if there exists a loss involving two points within distance d (vertical distance d) of p. Finally, a set of points is said to *have a loss within distance d* (*vertical distance d*) if it contains a point which has a loss within distance d (vertical distance d). The first lemma follows:

Lemma 2. *There exists a positive constant, c_1, such that*

$$|Bdary(R_C)| \leq |Bdary(R)| \leq 4L + c_1 E - 4.$$

Proof. Since any boundary point of R_C is also a boundary point of R, the first inequality is clear. To prove the second inequality, we first note that all points on the boundary of R must be internal points; for a point of R which is not internal must be surrounded by a curve of internal points and hence cannot be on the boundary. Suppose, then, that there is an internal point, p, on the boundary of R which is not one of the $4L - 4$ intended boundary points. We prove that there must either be a loss within distance 6 of p or an intended corner within distance 2 of p. There are three cases to consider:

Case 1: p is a 1. Point p must be adjacent to a point, q, which is not an internal point. If q is empty, then there is a loss within distance 1. If q is a 0, then there must also be a loss within distance 1 because no 1 which is not on the intended boundary has a non-internal 0 as a neighbor in any of the strings.

Case 2: p is a straight internal 0. Since p is a non-flank 0 and if there is no loss within distance 6, the picture of the neighborhood around p (up to rotation) must be as follows:

$$\begin{array}{ccc} 1 & & 1 \\ & \overline{\qquad} & \\ 1 & \text{—} p \text{—} & 1, \end{array}$$

where the horizontal thick segment represents the region boundary. However, the two possibilities for the point between the two uppermost 1's, empty or a non-internal 0, both lead to losses within distance 2 of p.

Case 3: p is a bent internal 0. Since p is on the boundary and, thus, cannot be adjacent to two other internal 0's, it follows from Lemma 3, stated below, that p must have a loss within distance 6 or an intended corner within distance 2.

We have completed showing that there must either be a loss within distance 6 or p or an intended corner within distance 2. Since there are at most a constant number of points within distance 6 of the two points involved in a loss and there are at most E points (for $n \geq 3$) within distance 2 of an intended corner, we can set c_1 equal to the constant plus four and the lemma follows. ■

We next state a lemma involving non-flank internal 0's referred to above. Its proof appears in Appendix A.

Lemma 3. *If z is a bent non-flank internal 0 which is neither adjacent to two other non-flank internal 0's nor within distance 2 of an intended corner, then there must be a loss within distance 6 of z.*

The final step toward limiting *Perim* (R_C) is to bound the number of convex corners on the boundary of any set of points which consists of connected components of R. We obtain such a bound using the following two lemmas. Recall that an *inside point* is an internal point which does not belong to the intended boundary.

Lemma 4. *There exists a positive constant, c_2, such that the number of convex corners on the boundary of any set of points consisting of connected components of R is at most $c_2 E$.*

Proof. Note that only four 1's may serve as convex corners without having a loss in distance 1. Non-flank 0's that are corners have a loss in distance 6 or an intended corner in distance 2 by Lemma 3. Thus, there are at most $c_3 E$ 1's and non-flank 0's which can be corners, where c_3 is a constant. Suppose an internal flank 0 serves as a corner. Then it is not adjacent to any other internal points than its two 1 neighbors. Note, however, that in the intended folding an internal flank 0 is adjacent to an inside point as well. We show below that at most $c_4 E$ intended boundary points (including internal flank 0's) can be adjacent to no inside points. Thus, there are at most $c_4 E$ internal flank zeros which can be corners, and, taking $c_2 = c_3 + c_4$, there are at most $c_2 E$ convex corners in R and therefore any set consisting of connected components of R.

To show that at most $c_4 E$ intended boundary points can be adjacent to no inside points, consider the $(L-2)^2$ inside points (they form an $(L-2) \times (L-2)$ square in the intended folding). These points are involved in at least $4(L-2)$ *contacts*, where a contact is an adjacent pair of points involving an inside point and a point which is not an inside point. A contact consisting of an inside point and an empty or non-internal point implies there is either a loss within distance 6 or an intended corner within distance 2 of the inside point by the proofs of Lemmas 2 and 3. Therefore, there are at most $c_3 E$ inside points involved in contacts with empty or non-internal point. The remaining contacts must involve inside points and intended boundary points. In the lemma following this one we prove that an intended boundary point can be involved in more than one contact only if it is within distance 8 of an intended corner or a loss. Thus, there are at most $c_5 E$ intended boundary points adjacent to more than one inside point, where c_5 is a constant. It then follows that at least $4(L-2) - 2c_3 E - 3c_5 E$ intended boundary points must be involved in contacts, and, thus, that, taking $c_4 = 2c_3 + 3c_5 + 4$, at most

$$c_4 E \geq (4L-4) - [4(L-2) - 2c_3 E - 3c_5 E] = 2c_3 E + 3c_5 E + 4$$

intended boundary points can be adjacent to no inside points. This completes the proof of the lemma. ■

The proof of the following lemma appears in Appendix B.

Lemma 5. *An intended boundary point which is adjacent to at least two inside points implies there is either an intended corner or a loss within distance 8 of the intended boundary point.*

We may now obtain the desired bound on the perimeter.

Corollary 6. *There exists a positive constant, c_6, such that*

$$Perim(R_C) \leq 4L + c_6 E.$$

Proof. The corollary follows from the observation that the perimeter of R_C is equal to the number of boundary points plus the number of convex corners it contains. Using Lemmas 2 and 4, we may take $c_6 = c_1 + c_2$. ■

We now prove a lower bound on $Area(R_C)$ and, in particular, we prove that, by limiting ourselves to the connected region R_C, we do not lose many internal points.

Lemma 7. *There exists a positive constant, c_7, such that*

$$Area(R_C) \geq L^2 - c_7 E.$$

More specifically, R_C contains all the internal substrings of the non-flank strings and all but at most $c_7 E$ internal points contained in the flanks.

Proof. Let R_N consist of the connected components of R which contain the internal points of the $L-2$ non-flank strings. Since all the internal points from a non-flank string must lie in the same connected component, each component of R_N must contain at least L points, and there are at most $L-2$ such components. We show $R_N \subseteq R_C$ by proving that, in fact, R_N contains only one connected component. Suppose R_N consists of $x \leq L-2$ connected components, for a positive integer x. First, we find a lower bound on $Perim(R_N)$ by making use of the fact that any component of area A must have perimeter greater than or equal to $4\sqrt{A}$. Using this inequality to find a lower bound on the perimeter, the limiting situation is when $x-1$ components are as small as possible and the remaining points are contained in the last component. A lower bound on $Perim(R_N)$ is, thus, given by

$$4(x-1)\sqrt{L} + 4\sqrt{L^2 - (x+1)L}.$$

On the other hand, we can find an upper bound on $Perim(R_N)$ by finding upper bounds on $|Bdary(R_N)|$ and the number of convex corners contained in R_N. Clearly $|Bdary(R_N)| \leq |Bdary(R)|$, since any boundary point of R_N is also a boundary point of R; therefore, using Lemmas 2 and 4, we see that $4L + c_1 E - 4 + c_2 E$ is an upper bound on $Perim(R_N)$. Comparing upper and lower bounds, it must be true that

$$\begin{aligned} 4L + (c_1 + c_2)E - 4 &\geq 4(x-1)\sqrt{L} + 4\sqrt{L^2 - (x+1)L} \\ &\geq 4(x-1)\sqrt{L} + 4[L - (x+1)] \\ &= \left(4\sqrt{L} - 4\right)x + 4L - 4\sqrt{L} - 4. \end{aligned}$$

Consequently,

$$\left(4\sqrt{L} - 4\right)x \leq 4\sqrt{L} + (c_1 + c_2)E.$$

Since $4\sqrt{L} - 8 > (c_1 + c_2)E$ for $n \geq 10$, we must have $x = 1$. Therefore, since R_C is the largest connected component of R, we must have $R_N \subseteq R_C$, meaning the internal substrings of the non-flank strings are contained in R_C, and thus $Area(R_C) \geq L^2 - 2L$.

Suppose more than $(c_6 + 1)E$ internal flank points are not contained in R_C. Then

$$Perim(R_C) < 4L + c_6 E - (c_6 + 1)E = 4L - E,$$

since we have lost the use of more than $(c_6 + 1)E$ intended boundary points. However, we still must have

$$\begin{aligned} Perim(R_C) &\geq 4\sqrt{L^2 - 2L} \\ &\geq 4L - 8 \end{aligned}$$

which leads to a contradiction since $E \geq 8$ for $n \geq 2$. The lemma follows by taking $c_7 = c_6 + 1$. ■

We next show that R_C is, in fact, very similar to an $L \times L$ square by bounding the dimensions and area of the smallest rectangle containing R_C and the dimensions of a square strictly contained in R_C.

Lemma 8. *The smallest rectangle containing R_C has sides of length $L + a$ and $L + b$, where $c_7\sqrt{LE} \geq a, b \geq -c_7\sqrt{LE}$. Further, its area is at most $L^2 + c_s LE$, where c_8 is a positive constant.*

Proof. Let $a = \max\{|x_1 - x_2| + 1 : (x_1, y_1), (x_2, y_2) \in R_C\} - L$ and let $b = \max\{|y_1 - y_2| + 1 : (x_1, y_1), (x_2, y_2) \in R_C\} - L$. From the definitions it should be clear that the smallest rectangle containing R_C has sides of length exactly $L + a$ and $L + b$. Comparing the area and perimeter of the bounding rectangle to those of (the continuous version of) R_C, we derive the following inequalities:

$$(L + a)(L + b) \geq L^2 - c_7 E \Rightarrow (a + b)L + ab \geq -c_7 E$$
$$2(L + a + L + b) \leq 4L + c_6 E \Rightarrow a + b \leq \frac{c_6}{2} E.$$

If a and b are both positive or both negative, and since $c_7 > c_6$, it is certainly true that $c_7 E \geq a, b \geq -c_7 E$. We note for later use that $ab \leq c_7^2 E^2$, regardless of the signs of a and b. Now let us assume without loss of generality that a is positive and b negative. Substituting bounds for $a + b$ and b in inequality 1 using inequality 2, we derive:

$$\frac{c_6}{2} EL + a\left(\frac{c_6}{2} E - a\right) \geq -c_7 E \Rightarrow a\left(a - \frac{c_6}{2} E\right) \leq \frac{c_6}{2} LE + c_7 E.$$

Since

$$\left(\frac{c_6}{2}\sqrt{LE} + \frac{c_6}{2} E\right)\left(\frac{c_6}{2}\sqrt{LE}\right) = \frac{c_6^2}{2} LE + \frac{c_6^2}{4} E\sqrt{LE} \geq \frac{c_6}{2} LE + c_7 E,$$

we must have

$$a \leq \frac{c_6}{2}\sqrt{LE} + \frac{c_6}{2} E \leq c_7\sqrt{LE},$$

which implies $b \geq -c_7\sqrt{LE}$ using inequality 1.

To verify the bound on the area of the rectangle, we note that

$$(L + a)(L + b) = L^2 + (a + b)L + ab \leq L^2 + \frac{c_6}{2} LE + c_7^2 E^2$$

using inequality 2 and the bound on ab given above. Taking $c_8 = \frac{c_6}{2} + c_7^2$, the bound stated in the lemma follows. ■

We may now state a lemma which finds a large square strictly contained in R_C.

Lemma 9. *For a positive constant c_9, there exists a square with sides of length $L - c_9\sqrt{LE}$ contained in R_C.*

Proof. We proved in the previous lemma that the smallest rectangle containing R_C has area less than or equal to $L^2 + c_8 LE$, for a constant c_8. Since R_C contains at least $L^2 - c_7 E$ internal points by Lemma 7, clearly any region contained in the bounding rectangle of area $2c_8 LE$ must contain an internal point contained in R_C.

At this point, it will help to have the bounding rectangle and R_C oriented in the coordinate plane. Let us assume without loss of generality that one of the up to 4 potential central points of the bounding rectangle is (0, 0) (this is just a translation of the embedding f). Now consider the square, $SQ1$, with sides of length $L - c_7\sqrt{LE}$ which is centered at (0, 0). Since $SQ1$ is contained in the bounding rectangle, we know that, if we consider the 4 squares with sides of length $\sqrt{2c_8 LE}$ (and area $2c_8 LE$) which share a corner with $SQ1$, then there must be a point of R_C contained in each of those 4 squares. That is, there exist 4 points, (x_1, y_1), (x_2, y_2), (x_3, y_3), and (x_4, y_4), in R_C such that, for each $i \in [4]$, (x_i, y_i) is in quadrant i and

$$|x_i|, |y_i| \geq \frac{1}{2}(L - c_7\sqrt{LE}) - \sqrt{2c_8 LE}.$$

$SQ1$ and the 4 points are pictured in Figure 3.

Consider the square, $SQ2$ (also pictured in Figure 3), which has sides of length $L - (c_7 + 2\sqrt{2c_8})\sqrt{LE}$ and is centered at the origin. Since R_C is connected and we have found the points $(x_i, y_i) \in R_C, i \in [4]$, we must have that $Perim(R_C) \geq Perim(SQ2) = 4L - 4(c_7 + 2\sqrt{2c_8})\sqrt{LE}$. Moreover, if there exists a point in the interior of $SQ2$ but not in R_C which is at a distance of more than

$$4(c_7 + 2\sqrt{2c_8})\sqrt{LE} + c_6 E$$

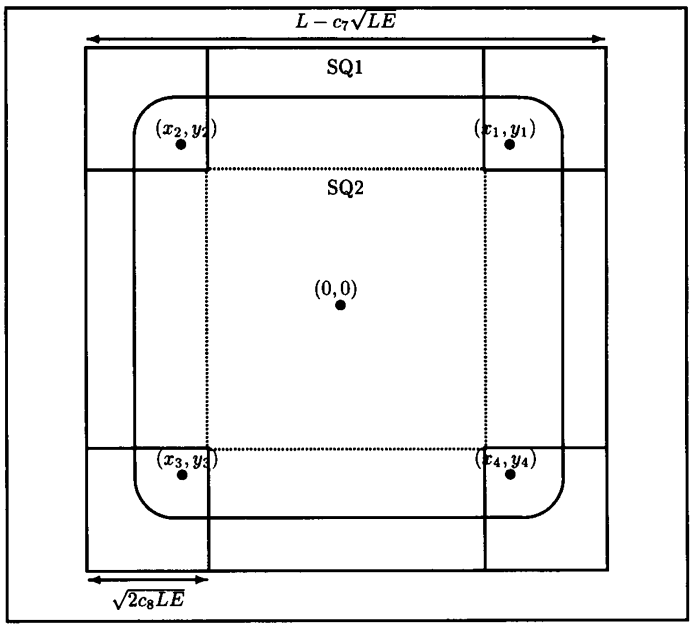


FIG. 3. $SQ1$, $SQ2$, and 4 points of R_C.

from all the sides of $SQ2$, then, since there are no holes in R_C, we must have $Perim(R_C) > 4L + c_6E$. This is clearly impossible due to Corollary 6. Thus, any square with sides of length less than

$$L - (c_7 + 2\sqrt{2c_8})\sqrt{LE} - 2[4(c_7 + 2\sqrt{2c_8})\sqrt{LE} + c_6E]$$
$$= L - (9c_7 + 18\sqrt{2c_8})\sqrt{LE} - 2c_6E$$

centered at the origin must be contained in R_C. In particular, the square, SQ, centered at (0, 0) with sides of length $L - c_9\sqrt{LE}$, where $c_9 = 29c_8$, is contained in R_C. ■

We now move closer to our goal of showing that many strings pass through R_C in an approximately straight and parallel fashion by first showing that there exists a non-flank string passing near the origin which is approximately straight. Note that it is impossible for any prefix or suffix 0 to be in R_C since that would imply the whole prefix or suffix is in R_C, forcing its perimeter to be too high. Thus the string must exit R_C and, in particular, SQ, at two points. We show the two points at which the string near (0, 0) exists SQ (not to return) are near the centers of two opposite sides of SQ. As a corollary to the placement of this string, we derive that the flanks must lie at opposite sides of R_C.

Lemma 10. *There exists a non-flank string passing within an $O(\sqrt{LE})$ distance of (0, 0) which exists SQ within $O(\sqrt{LE})$ distance of the centers of two opposite sides. Since a side of SQ has length $L - c_9\sqrt{LE}$, the string must then be relatively straight, containing $O(\sqrt{LE})$ bent points.*

Proof. It should be clear that the second sentence follows from the first since the internal substring of a non-flank string has length L. Of the L bits, $L - c_9\sqrt{LE}$ must be used to get from one side of the square to the other, leaving a slack of size only $c_9\sqrt{LE}$.

Consider the square with side length $\sqrt{2c_8LE}$ which is centered at the origin. This square must contain an internal point, p, of R_C. The string, s, containing p cannot be a flank, because we would lose the use of too many intended boundary points inside SQ, so we have found a non-flank string passing near (0, 0). Point p is at a distance of at least $\frac{1}{2}(L - c_9\sqrt{LE}) - \sqrt{2c_8LE}$ from each side of SQ. Thus s cannot exist SQ at any point farther than $(c_9 + 3\sqrt{2c_8})\sqrt{LE}$ from the centers of the sides of SQ.

There are then 3 possibilities for the way in which string s exits SQ: it either exits at a single side (forms a U-shape), exits at adjacent sides (forms an L-shape), or exits at opposite sides as desired. We prove the first two cases are impossible. Suppose, first, that s forms a U-shape. For purposes of illustration we will assume s exits through the top side of SQ. Recall the structure of the non-flank strings: there are 3 dense substrings of length $\frac{L}{10}$, the middle of which begins with bit number $\frac{3L}{10} + 1$ of the internal substring. We claim that no

matter how s is folded into a U-shape, the bits on the opposite side of the U, for illustration, the right side, facing bits $\frac{3L}{10}+1$ through $\frac{3L}{5}$ on the left must be 1's. The string must have at least $L-(c_9+2\sqrt{2c_8})\sqrt{LE}$ bits inside SQ, and by evaluating the two limiting configurations of the U (where the at most $(c_9+2\sqrt{2c_8})\sqrt{LE}$ bits are outside SQ), we see that bits $\frac{3L}{10}+1$ through $\frac{2L}{5}$ may only be opposite bits in the range

$$\left[\frac{7L}{10}+1-(c_9+2\sqrt{2c_8})\sqrt{LE},\ \frac{4L}{5}+(c_9+2\sqrt{2c_8})\sqrt{LE}\right].$$

We may assume

$$(c_9+2\sqrt{2c_8})\sqrt{LE}\le\frac{L}{180}-E-1,$$

which is true for $n\ge 10$. Clearly, then,

$$(c_9+2\sqrt{2c_8})\sqrt{LE}\ll L/10$$

and the bits in the above range are 1's. Next, again because the string has slack only at most $(c_9+2\sqrt{2c_8})\sqrt{LE}$, at most that many 0's out of the $\frac{L}{90}$ in the middle dense substring may be horizontal or bent. Thus we may assume there are at least $E+1$ vertical 0's. Now, if there is no loss in the window containing a vertical 0 and its neighbors and the three points adjacent to them on the right, then the point adjacent to the vertical 0 on the right must also be a vertical 0. Assuming the adjacent vertical 0 is also a non-flank 0 without a loss in a similar window, then the pattern will continue and there will be a horizontal line of vertical 0's:

$$\begin{array}{ccc}1&1&1\\|&|&|\\0&0&0\\|&|&|\\1&1&1\end{array}$$

Thus, there are $E+1$ lines of vertical 0's extending from the left side of the U toward the right which may not be terminated unless they reach a flank, impossible due to the resulting loss of intended boundary points, or unless one of the 0's in the line has a loss within its window containing six points. Since a loss terminating a line may be uniquely associated with that line, one of the $E+1$ lines of 0's must have no associated losses. However, this line of 0's must run into a 1 on the right side of the U, resulting in a loss and a contradiction. We conclude the string cannot form a U-shape.

Let us, next, suppose string s forms an L-shape. We prove this is impossible by using a similar argument involving lines of 0's. One half of the L (for illustration we assume it exits through the top side of SQ) contains the middle, dense substring and, as above, we are guaranteed there is at least one horizontal line of vertical 0's without an associated loss in distance 2. Consider the horizontal second half of the L, say exiting through the right side of SQ, which contains the rightmost dense substring. This second half contains a portion of at least

$$\frac{L}{10}-(c_9+2\sqrt{2c_8})\sqrt{LE}\ge\frac{3L}{40}$$

of the rightmost dense substring, and since at most $(c_9+2\sqrt{2c_8})\sqrt{LE}$ of the first $\frac{L}{180}$ 0's in the dense substring are vertical or bent, at least $E+1$ 0's are horizontal (and at a distance at least $\frac{L}{40}$ from the right side of SQ). These 0's must form vertical lines of horizontal 0's, and, using a symmetrical argument, we are guaranteed that at least one of the lines will be without an associated loss in distance 2. However, this line must encounter the horizontal line of vertical 0's originating from the first half of the L, and no flank may terminate either of these lines between the sides of the L and the point where they meet. Since the straight 0's in the lines are surrounded by 1's, one of the two lines must suffer a loss, a contradiction. Therefore, the string cannot form an L. ■

We may assume without loss of generality that string s exits at the top and bottom of SQ and thus its orientation is vertical. (This is just a rotation of the folding f.) We may now place the flanks:

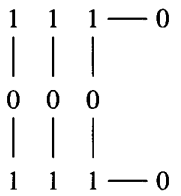
Corollary 11. *The flanks both intersect the lines*

$$y=\pm\left[\frac{1}{2}(L-c_9\sqrt{LE})-c_{10}\sqrt{LE}\right],$$

where c_{10} is a positive constant. Further, in the vertical interval stretching between the two lines, one flank lies to the left of and one flank lies to right of the lines

$$x = \pm\left[\frac{1}{2}\left(L - c_9\sqrt{LE}\right) - \frac{3}{2}c_7E\right].$$

Proof. We use horizontal lines of vertical 0's to place the flanks. In fact, the only way for a horizontal line of vertical 0's to end without a loss in vertical distance 1 is for it to end in a vertical flank 0:



Using the first and last dense substrings contained in the vertical string, we can find horizontal lines of vertical 0's with no associated losses at distances of no more than $8c_9\sqrt{LE} + 9(E+1) \leq c_{10}\sqrt{LE}$, where $c_{10} = 17c_9$, from the top and bottom of SQ. The central string s prohibits the flanks from stretching across the top or bottom of R_C to complete these lines; the prefix and/or suffix of s would cause the loss of too many intended boundary points. Consequently, one flank must complete both of the lines on the left and the other flank must complete both of the lines on the right. Thus, both flanks intersect the lines

$$y = \pm\left[\frac{1}{2}\left(L - c_9\sqrt{LE}\right) - c_{10}\sqrt{LE}\right].$$

Suppose that in the vertical interval

$$\left[-\frac{1}{2}\left(L - c_9\sqrt{LE}\right) + c_{10}\sqrt{LE}, \frac{1}{2}\left(L - c_9\sqrt{LE}\right) - c_{10}\sqrt{LE}\right]$$

a flank is further than $\frac{3}{2}c_7E$ from the closest side of SQ inside the interior of SQ. Then $3c_7E$ points of the flank must be inside SQ, but this implies that at least c_7E intended boundary points have been lost, which is impossible. Therefore, the flank completing the lines of 0's on the left must lie to the left of the line

$$x = -\frac{1}{2}\left(L - c_9\sqrt{LE}\right) + \frac{3}{2}c_7E$$

in the vertical interval and the flank completing the lines of 0's on the right must lie to the right of the line

$$x = \frac{1}{2}\left(L - c_9\sqrt{LE}\right) - \frac{3}{2}c_7E$$

in the vertical interval. ■

Let RT be the region bounded by the two flanks and the lines

$$y = \pm\left[\frac{1}{2}\left(L - c_9\sqrt{LE}\right) - c_{10}\sqrt{LE}\right].$$

We now show that greater than $L - \frac{L}{n}$ non-flank strings pass through (R_C and) RT and exit through its top and bottom sides. (We will show that no prefix or suffix bit may be contained in RT.) In addition, we show we may assume all 0's contained in the sparse substrings of the non-flank strings are straight. This is the large collection of approximately straight and parallel strings we have been searching for. An important fact to note is that the y-coordinate (vertical coordinate) of any point contained in the sparse substring of these strings is confined to lie in an interval of length $(c_9 + 2c_{10})\sqrt{LE} + 1$; that is, bit number l of the internal substring must lie in the vertical interval of length $(c_9 + 2c_{10})\sqrt{LE} + 1$ centered at $\frac{L}{2} - l$. Setting $c = 4(c_9 + 2c_{10})$, so that the shortest number of 1's between two 0's in the sparse substring is

$$4(c_9 + 2c_{10})\lceil\sqrt{LE}\rceil,$$

it is clearly impossible for a 0 in the sparse substring to be adjacent to a 0 from the same string. Further, and very importantly, if the 0 is bit number l of the internal substring of one of our collection of strings, then it may not be adjacent to any other 0 in another string in the collection other than the 0 (if it is a 0) in bit number l of the other string. These facts will allow us to find a Hamilton path using the collection of strings. They will also be important in the two lemmas and two corollaries below where we find the collection of strings. Note, finally, that the 0's and all the code bits in the sparse substrings of the strings in the collection must lie in the vertical interval,

$$\left[-2E\left(c\lceil\sqrt{LE}\rceil+1\right)-\frac{c}{4}\sqrt{LE}, 2E\left(c\lceil\sqrt{LE}\rceil+1\right)+\frac{c}{4}\sqrt{LE}\right]$$
$$\subseteq\left[-\left(2cE+\frac{c}{4}\right)\lceil\sqrt{LE}\rceil-2E, \left(2cE+\frac{c}{4}\right)\lceil\sqrt{LE}\rceil+2E\right].$$

Lemma 12. *Any non-flank string which passes through the region, RT', which is bounded by the two flanks and the lines*

$$y=\pm\left[(2cE+c_9+2c_{10})\lceil\sqrt{LE}\rceil+2E\right]$$

must exit RT through its top and bottom sides.

Proof. We first show that no prefix or suffix bit may be contained in RT. Recall that lines of vertical 0's without associated losses and completed by flanks are guaranteed to lie on the top and bottom sides of RT or just above and below these sides. No prefix or suffix bit may lie on these lines. Further, since no prefix or suffix bit may lie in SQ, the whole prefix or suffix must be contained in the region bounded by the closest flank and side of SQ, and the two lines. However, this would cause the loss of too many intended boundary points contained in the closest flank. Thus, no prefix or suffix bit may be contained in RT (and we also note for future reference that there can be no intended corners inside RT).

Let s' be a non-flank string passing through RT'. String s' must exit RT through two points contained in its top and/or bottom sides. We next eliminate the possibility of s' forming a U-shape with respect to either the top or bottom side using the same argument as used in Lemma 10. We need only the assumption that

$$(c_9+2c_{10})\sqrt{LE}+(4cE+2c_9+4c_{10})\lceil\sqrt{LE}\rceil+4E\le\frac{L}{10}$$

which is true for $n\ge 10$. Thus s' exits RT through its top and bottom sides as desired. ■

Lemma 13. *There are at least*

$$L-c_9\lceil\sqrt{LE}\rceil-6c_7E$$

strings which pass through (R_C and) RT and exit through its top and bottom sides.

Proof. We prove that $L-c_9\sqrt{LE}-6c_7E$ strings pass through the region $RT'\cap SQ$ contained in RT by again making use of horizontal lines of vertical 0's. It suffices to show that s has sparse vertical 0 without a loss in vertical distance 1. For then, a horizontal line of vertical 0's must be formed and we are guaranteed by the previous lemma that all strings containing the 0's which pass through $RT'\cap SQ$ must exit RT through its top and bottom sides. Further, by the observations made before the previous Lemma, the strings in $RT'\cap SQ$ containing the 0's are distinct. Finally, since no flank string may pass through SQ further than $\frac{3}{2}c_7E$ from the corresponding side of SQ (as noted in Corollary 11) to end the line, there are certainly at least

$$L-c_9\lceil\sqrt{LE}\rceil-6c_7E$$

strings passing through RT. (We removed an extra factor of $3c_7E$-2 because later it will be useful to assume these strings are far from the flanks.)

We verify that there is always at least one vertical 0 without a loss in vertical distance 1 contained in the sparse substring of s. Any horizontal 0 in the sparse substring must have a loss within vertical distance

$$(c_9+2c_{10})\lceil\sqrt{LE}\rceil+1$$

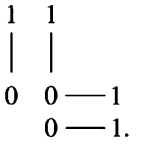
since, if there is no loss within vertical distance

$$(c_9 + 2c_{10})\lceil\sqrt{LE}\rceil,$$

then a vertical line of 0's extending upward of length

$$(c_9 + 2c_{10})\lceil\sqrt{LE}\rceil + 1$$

must be formed. However, because of the large distance between 0's noted in the paragraph before the previous Lemma, the next bit in the line must be a 1 or empty, resulting in a loss. Next, the only way for a sparse bent 0 in s not to have a loss within vertical distance 2 (or distance 2), is if it is adjacent to two other 0's:



However, the two 0's which are adjacent to the bent 0, neither of which may be flank 0's, must be contained in the same string since non-flank strings (contained in RT') exit RT through its top and bottom sides. Yet this is impossible due to the limitations on the proximity of 0's. Therefore, all sparse bent 0's must have a loss within vertical distance 2 (and distance 2). Since losses within vertical distance

$$(c_9 + 2c_{10})\lceil\sqrt{LE}\rceil + 1$$

are uniquely attributable to the sparse 0's in a string and there are at least $E + 1$ sparse 0's, there must be a vertical sparse 0 without a loss in vertical distance 1. ■

Corollary 14. *There exist more than* $L - \frac{L}{n}$ *non-flank strings which pass vertically through* RT *such that all their sparse 0's are vertical or horizontal.*

Proof. There ar at most 8 points within distance 2 of a loss, so there are at most $8E$ bent 0's. Thus at least

$$L - c_9\lceil\sqrt{LE}\rceil - 6c_7E - 8E$$

strings pass vertically through RT such that all their 0's are straight. Assuming

$$c_9\lceil\sqrt{LE}\rceil + 6c_7E + 8E < \frac{L}{n},$$

true for $n \geq 10$, we are done. ■

Therefore, there is at least one representative of every node passing vertically through RT. Let $(s_{k_1}, s_{k_2}, \ldots, s_{k_m})$ be the left-to-right order of the collection of strings in Corollary 14. We claim that for $l \in [m-1]$, the number of losses involving strings s_{k_l} and $s_{k_{l+1}}$ and points in between must be at least the Hamming distance of the two strings. Recall the Hamming distance of the two strings is equal to the number of points where a 0 bit from the Trevisan code of one node corresponds a 1 bit from the Trevisan code of the other node (the bits are at the same position in the Trevisan code). Suppose there is a code vertical 0 in s_{k_l} which corresponds to a code 1 in $s_{k_{l+1}}$, and let us say s_{k_1} is to the left of $s_{k_{l+1}}$. Then there must be a line of vertical 0's extending from the code 0 to the right, and this line must end before or at the time it reaches $s_{k_{l+1}}$. None of the 0's in the line can be in string $s_{k_{l+1}}$ and the loss ending the line must be contained in the window consisting of the last 0 in the line and its neighbors and the three adjacent points to the right. Therefore, since the loss must involve points in the strings s_{k_l} and $s_{k_{l+1}}$ and/or points in between the strings and the loss is within vertical distance 1 of the code 0's, the loss may be uniquely attributed to the pair of differing code bits in the two strings. Similarly, if the code 0 is horizontal, using similar windows, there must be a loss involving points between the strings within vertical distance at most

$$(c_9 + 2c_{10})\lceil\sqrt{LE}\rceil + 1.$$

The loss may also be uniquely attributed to the pair of differing code bits. Thus, if s_{k_l} and $s_{k_{l+1}}$ correspond to different nodes, there are at least $7n$ losses which can be attributed to the two strings due to the Trevisan code. However, since all n nodes are represented in the collection of strings, the total number of losses contained in RT is at least $7(n-1)n = E$. Consequently, there may only be $n-1$ transitions between different nodes, and the order $(v_{j_1}, v_{j_2}, \ldots, v_{j_n})$, of the nodes corresponding to the strings above, deleting adjacent occurrences of the same node, must correspond to a Hamilton path in G.

In fact, we have found our desired Hamilton path from v_1 to v_n. Assume without loss of generality that the flank on the left side of R_C corresponds to v_1, that is, the flank is s_1 (this is just a reflection of the folding f). We prove:

Lemma 15. *It must be true that $v_{j_1} = v_1$ and $v_{j_n} = v_n$.*

Proof. We show that $v_{j_1} = v_1$; it follows by symmetry that $v_{j_n} = v_n$. First, we note that all the sparse 0's contained in s_{k_1} must be vertical. The reason is that there are always two losses attributable to sparse horizontal 0's (one above the 0 and one below), and only loss may be lying between the strings s_{k_1} and s_{k_2}. Since we are already guaranteed E losses which lie between the strings s_{k_1} and s_{k_m}, there can be no other losses outside this region.

Next, we show that all the strings passing through RT' between s_1 and s_{k_1} also represent v_{j_1}. It suffices to verify that there are no sparse bent 0's in any of these strings. For all the strings except for the string closest to s_1 this is easy. We simply use the same argument as was used in Lemma 13 to show that there must be a loss within distance 1 of a string containing a sparse bent 0. Regarding the string next to s_1, call it s'', we may rule out any sparse bent 0's facing toward the right using the same argument. In terms of any sparse bent 0 which faces toward the left, there can be no intended corner within distance 1 of the bent 0's neighbors (this was noted in Lemma 12), and the 0 may not be adjacent to two other non-flank internal 0's. Thus, using the proof of Lemma 3, there must be a loss within distance 6 of the 0; however, this is impossible since s'' is far from s_{k_1}.

Finally, we show that a sparse 0 contained in s_1 which is (horizontally) adjacent to a point of s'' must lie in a vertical interval of length

$$(c_9 + 2c_{10})\lceil\sqrt{LE}\rceil$$

centered at $\frac{L}{2} - l$, if the 0 is the lth internal bit contained in s_1. This fact will complete the proof, for then s_1 must have 0's in exactly the same Trevisan code positions as s''. Since the Trevisan code contained in s_1 has at most as many 0's as the code in s'', the codes must then be identical. Therefore, $v_{j_1} = v_1$.

To show that a sparse 0 of s_1 which is (horizontally) adjacent to s'' must lie in the desired interval, it suffices to show that at least

$$L - (c_9 + 2c_{10})\lceil\sqrt{LE}\rceil$$

internal points of s_1 with distinct y-coordinates are horizontally adjacent to points of s'' and are contained in a vertical interval of length

$$L - (c_9 + 2c_{10})\lfloor\sqrt{LE}\rfloor.$$

Inside RT,

$$L - (c_9 + 2c_{10})\lfloor\sqrt{LE}\rfloor$$

internal points of s'' with distinct y-coordinates face s_1 horizontally (in a vertical interval of length $L - (c_9 + 2c_{10})\lfloor\sqrt{LE}\rfloor$). Again making use of Lemma 3, none of these points may be a bent 0. Since all the points must then be 1's or vertical 0's and no losses within distance 2 of s'' are allowable, the points must all be adjacent horizontally to distinct internal points of s_1. This completes the proof. ■

3. THE STRING FOLDING PROBLEM

In this section we show that the STRING FOLDING PROBLEM (the special case of the multistring problem with $|S| = 1$, which captures the protein folding problem in the 2-dimensional HP model) is also NP-complete.

Let us call a planar graph *special* if it consists of disjoint faces with nodes of degree three, connected together by paths of length two, and becomes triply connected if all nodes of degree two are collapsed. See Figure 4 for an example.

Theorem 16. *The Hamilton cycle problem remains NP-complete even if restricted to special planar graphs.*

Proof. The reduction from exact cover to planar Hamilton cycle in (Johnson and Papadimitriou, 1985) produces a special planar graph, if the 2-input and 3-input "or" gadgets are replaced by the ones shown in Figure 5. ■

Fix a planar graph G. Two Hamilton cycles are called *orthogonal* if they have the following property: their disjoint union (where we duplicate edges in their intersection) is a degree-4 planar graph with multiple edges which can be embedded in the plane in such a way that the edges around each node alternate between the two cycles. Figure 6 depicts the two Hamilton cycles of the *diamond* graph (plus another node G); they are orthogonal because, by duplicating the three paths of length two, one obtains a degree-four graph around each node of which edges of the two Hamilton cycles alternate.

Suppose that a graph contains the *diamond* graph depicted in Figure 6 (ignore the node G standing for the rest of the graph). The diamond graph has four endpoints, denoted N, S, E, W, whereby it is connected to the rest of the graph. Any Hamilton cycle of the overall graph must traverse the diamnd either from N to S, or from E to W (but not, e.g., from E to N).

Theorem 17. *The* STRING FOLDING PROBLEM *is NP-complete.*

Proof. We start from the Hamilton cycle problem for special planar graphs. Given any special planar graph G, we shall modify the graph so that it contains a "standard" Hamilton cycle H_0, such that any Hamilton cycle of the original graph corresponds to a cycle of the modified graph that is orthogonal to H_0. Starting from G

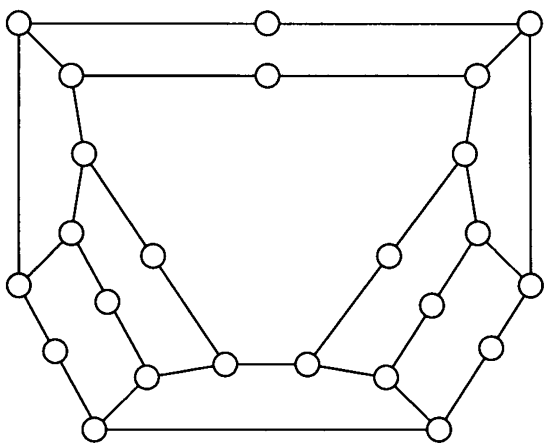


FIG. 4. A special planar graph.

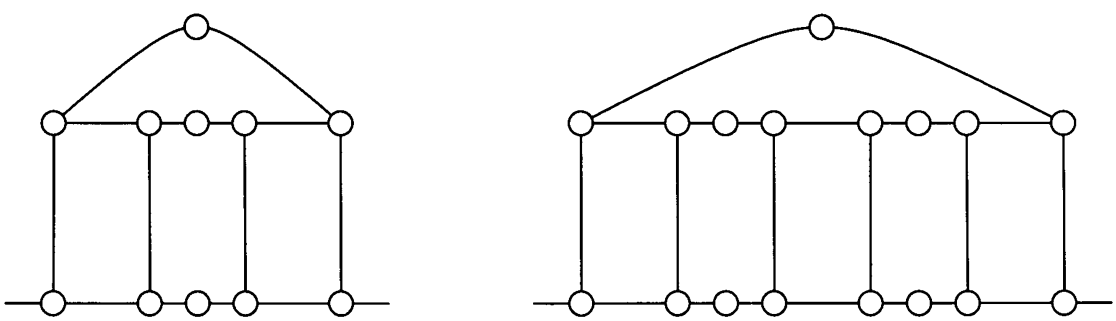


FIG. 5. New 2-input and 3-input gadgets.

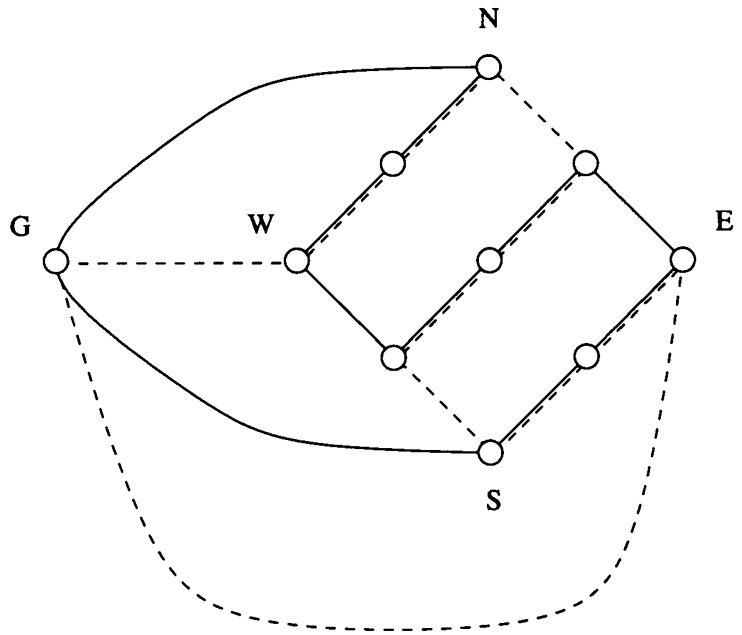


FIG. 6. The diamond graph.

and its embedding, take only the degree-2 nodes of G, and consider two such nodes adjacent if they are on the same face of the embedding. Since the original graph is special, the resulting graph G' is connected.

Consider thus a cycle C of G' (allowing repeated nodes but no self-loops) that visits all nodes of G' at least once. If the two nodes adjacent to an occurrence of v are on the same face, repeat that occurrence twice. Now for each node v, count the occurrences of v on C and let $a(v)$ be the resulting number.

Replace now each degree-2 node v of G, and its adjacent edges, by $a(v)$ copies of the diamond; the copies are disjoint, and the N and S nodes of the two outermost ones (or the unique one, if $a(v) = 1$) coincide with the nodes of G adjacent to v (see Figure 7). Let $C = (v_0, v_1, \ldots, v_k = v_0)$. For $i = 0, \ldots, k-1$, suppose the ith element of C is the b_ith occurrence of node v_i (let $b_k = 1$); for each $i = 1, \ldots, k$ join the E or W node of the b_{i-1}th copy of the diamond replacing node v_{i-1}, whichever has not been considered up to now, with the E or W node of the b_ith copy of the diamond replacing node v_i, whichever is in the same face with the previous node (for v_0, if $v_1 \neq v_0$, we start with the endpoint, E or W, that is on the same face as v_1, and if $v_1 = v_0$, we start with the endpoint which is not on the same face as v_2). Notice that these new edges do not harm the planarity of the graph, and they, together with the E–W traversal of the diamonds, form the standard Hamilton cycle, H_0, of the resulting graph G''.

H_0 is the only Hamilton cycle of G utilizing a E–W traversal of the diamonds. Any Hamilton cycle utilizing a N–S traversal must correspond to a Hamilton cycle of the original graph G. It is easy to see that any such cycle will be orthogonal to H_0—because the E–W and the N–S traversals of the diamond are orthogonal.

We shall now construct the instance of the string folding problem. We take any degree-2 node and replace it with two degree-1 nodes, and make these nodes the endpoints of the Hamilton path sought. H_0 becomes a Hamilton path as well. We now perform Trevisan's transformation *having deleted the E–W edges* of the graph (that is, the endpoints of these edges have large Hamming distance in the Trevisan code). We then perform the multistring reduction, with the following modifications:

- The number of strings corresponding to each city, L/n, is odd. This is trivial to accomplish by adding one string to each set.
- All strings corresponding to the same city are connected together in one string, by ordering them arbitrarily, and connecting the end of the suffix of string $2i-1$ to the end of the suffix of the string $2i$, and the beginning of the prefix of string $2i$ to the beginning of the prefix of string $2i+1$,

$$i = 1, \ldots, \frac{\frac{L}{n} - 1}{2}.$$

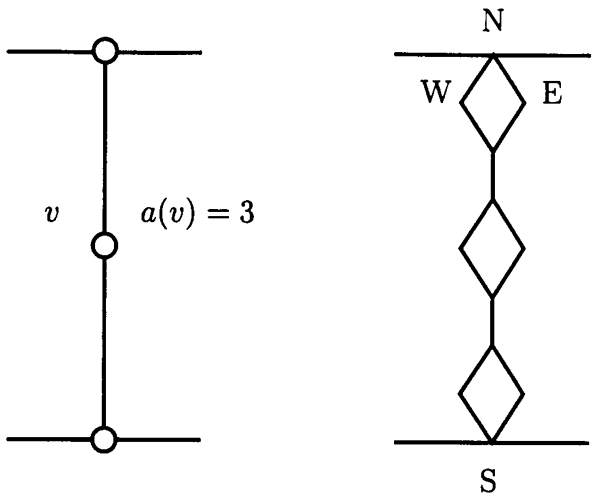


FIG. 7. Replacing a degree-2 node by diamonds.

- Finally, all of these n long strings are connected together in the order suggested by the Hamilton path H_0 by long (of length $2L^4 + 2L^2$) strings of 0's.

We claim that there is a folding with E losses if and only if the original special graph had a Hamilton cycle. Suppose that indeed there is a folding with E or fewer losses. By the precise same argument as in the proof of Theorem 1, there is a Hamilton path in the graph G'' that does not utilize the E–W edges, and hence there is a Hamilton cycle in the original graph.

Conversely, suppose that G did have a Hamilton cycle. Then G'' has a Hamilton cycle H other than H_0, and in fact one that is orthogonal to H_0. But this means that we can arrange the n strings corresponding to the cities as in the intended folding in the proof of Theorem 1, joined together as they are via their prefixes and suffixes in the order suggested by H_0, because H_0 is orthogonal to H. ■

4. CONCLUSION

Our NP-completeness result settles in the affirmative the widespread conjecture that the protein folding problem, even in its most simple yet realistic two-dimensional HP simplification, is NP-complete. By a simple modification of our techniques (and, in fact, one that is free of the planarity complications of the present proof) we can show that the three-dimensional version of the protein-folding problem in the HP model is NP-complete. The appropriate modification of our proof is this: the string consists of roughly $L \times L \times L$ 1's, which form a cube protected from all sides (except for the eight corners, where 8 losses *must* occur) by 0's. The Hamilton cycle problem is encoded in the part of the cube that lies just below one particular face of the cube. This part is traversed in one direction by "tubes" whose cross-section is a square of four 0's. Mismatches in the alignment of these tubes correspond to mismatches in the Trevisan code of the underlying graph and contribute the only extra losses, beyond the eight mandatory ones. Thus the intended folding has a total of $4E + 8$ losses, and the steps in establishing the converse are analogous to the ones in the present proof. The same result was proven independently, and a few months earlier, by Berger and Leighton (1997).

APPENDICES

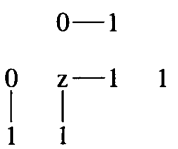
A. *Proof of lemma 3*

Lemma 3. *If z is a bent non-flank internal 0 which is neither adjacent to two other non-flank internal 0's nor within distance 2 of an intended corner, then there must be a loss within distance 6 of z.*

Proof. Suppose that there is a bent non-flank internal 0, z, which is neither adjacent to two other non-flank internal 0's nor within distance 2 of an intended corner. The assumption of no intended corners close to z

implies that if there is a 1 within distnace 2 of z which is contained in the substring 010, then it must either be contained in the larger substring 1010 and 0101. Suppose, in addition, that there is no loss within distance 6 of z. We show that in every potential configuration involving z these assumptions lead to a contradiction, for there must, in fact, be a loss within distance 6 of z.

Assuming there is no loss within distance 6 or z, the picture of its neighborhood under folding f (up to rotation) must be as follows:

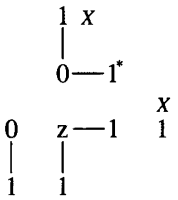


The reason for the appearance of the 1's other than the two neighbors of z is that there are always at least eight 1's which lie between internal 0's on any string, flank or non-flank (though on flanks they may be separated by some groups of 4 0's). The same separation is true for internal 0's and prefix or suffix 0's. This fact will play a role in almost all of the pictures of configurations which follow in the proof of this lemma and in the proof of Lemma 4.

Next, we consider the classifications of the 0's lying adjacent to z. Note that the two 0's cannot both be straight.

Case 1: One of the adjacent 0's is a bent flank internal 0:

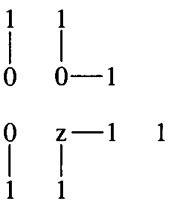
Assume without loss of generality that it is upper adjacent 0. Since there are no losses within distance 6, the picture must be as follows:



No matter where the 0 following the starred 1 is placed, there will be a loss (located at one of the two X's), a contradiction.

Case 2: One of the adjacent 0's is a bent non-flank internal 0 and the other adjacent 0 is not a bent non-internal 0.

Assume without loss of generality the bent non-flank internal 0 is the upper adjacent 0. The necessary picture follows:

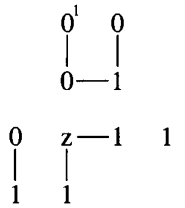


The adjacent 0 to the left of z cannot be straight (there are no maximal substrings of 0's of length two). Since we assumed it was not a bent non-internal 0 and it is not a bent internal 0, this case is impossible.

Only one case remains:

Case 3: One of the adjacent 0's is a bent non-internal 0.

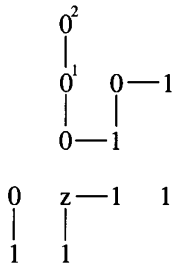
Assume it is the upper 0. The picture must be the following:



The upper 0 adjacent to z cannot be a prefix or suffix 0, due to our assumption about intended corners. It must be contained in one of the groups of 4 0's on a flank. As well, the 0 which is a neighbor to the 1-neighbor of the upper 0 may also not be a prefix or suffix 0.

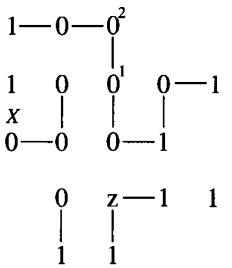
We next examine the potential locations for the 0 following the 0 marked with the superscript 1 (0^1 for short).

Case 3A: The 0 following 0^1 goes up, as appears below.



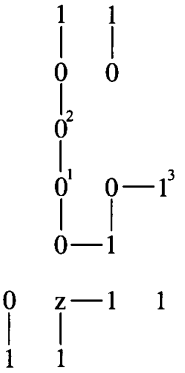
We examine the potential locations for the 0 following 0^2.

Case 3Ai: The 0 following 0^2 goes left:

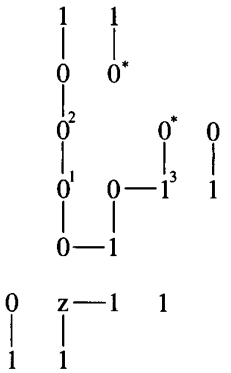


There must be a loss at the X.

Case 3Aii: The 0 following 0^2 goes up:

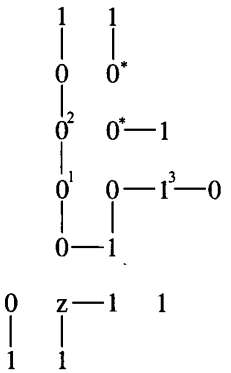


Case 3Aiia: The 0 following 1^3 goes up:



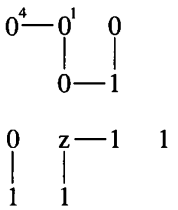
This configuration is impossible because no matter how the picture is completed, the two starred 0's must belong to the same string, but no string contains either a maximal substring of 0's of length three or the substring 10101.

Case 3Aiib: The 0 following 1^3 goes right:

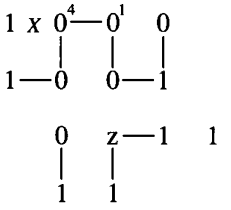


Similarly to the previous subcase, this configuration is impossible because the two starred 0's must be in the same string, but no string contains a maximal substring of 0's of length two.

Case 3B: The 0 following 0^1 goes left:

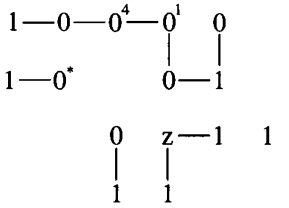


Case 3Bi: The 0 following 0^4 goes down:



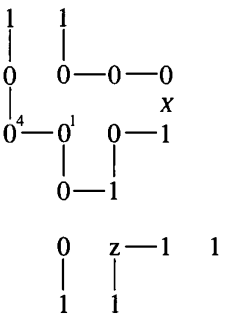
There must be a loss at the X.

Case 3Bii: The 0 following 0^4 goes left:



Here, again, the configuration is impossible because the two starred 0's must be contained in the same string in a maximal substring of 0's of length three.

Case 3Biii: The 0 following 0^4 goes up:



There must be a loss at the X, contradiction.

We have completed our case analysis. In all cases, we showed that under our assumptions the potential configurations must, in fact, either be impossible or contain a loss within distance 6 of z (all points shown in the figures were within distance 6 of z). Thus, a bent non-flank internal 0 which is neither adjacent to two other non-flank internal 0's nor within distance 2 of an intended corner must have a loss within distance 6. ■

B. Proof of Lemma 5

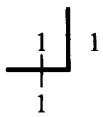
Lemma 5. *An intended boundary point which is adjacent to at least two inside points implies there is either an intended corner or a loss within distance 8 of the intended boundary point.*

Proof. Since every intended boundary point must be adjacent to at least one non-inside point, there are (without loss of generality) three distinct configurations in which an intended boundary point is adjacent to at least two inside points (we call these adjacent pairs contacts):

where x denotes the intended boundary point and the bars denote the positions of contacts. We will refer to these configurations as, respectively, patterns 1, 2, and 3.

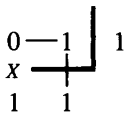
We break the intended boundary points into 4 types: prefix or suffix 1's, flank 1's contained in the substring 110, flank 1's contained in the substring 010, and flank internal 0's. Note that only prefix or suffix 1's may be adjacent to three inside points, so we will examine nine main cases. We show that in all the cases, assuming there is no intended corner in distance 8, there must be a loss within distance 8 of the intended boundary point. Some facts about substrings that are relevant to this proof are: first, as in the previous lemma, since there are no nearby intended corners, the substring 010 must be contained in the larger substring 1010 or 0101, and second and most importantly, an inside 1 adjacent to a 0 must be contained in the substring 11111111011111111. Some details of this proof have been skipped: two figures following each other without further comment mean that the second configuration follows necessarily from the first one.

Case 1: The intended boundary point is a prefix or suffix 1 in pattern 1. Assuming no nearby intended corners nor losses, the picture must be as follows:



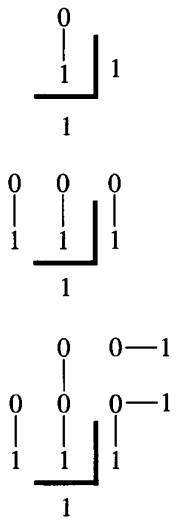
We may assume without loss of generality that the prefix or suffix 1 is adjacent to the lower inside 1.

Case 1a: The 0 following the prefix or suffix 1 is left:

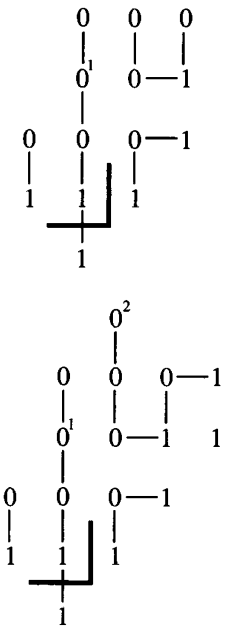


There must be a loss at the X.

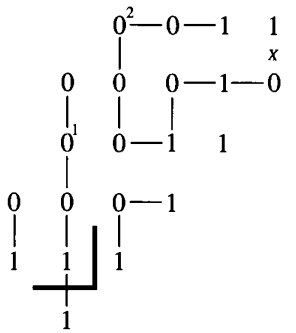
Case 1b: The 0 following the prefix or suffix 1 is up:



Case 1bi: The 0 following 0^1 is up:

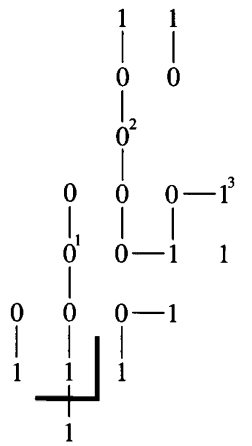


Case 1biA: The 0 following 0^2 is right:

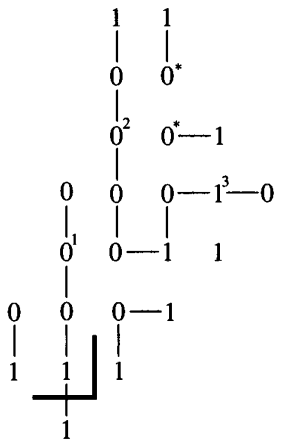


There must be a loss at the X.

Case 1biB: The 0 following 0^2 is up:

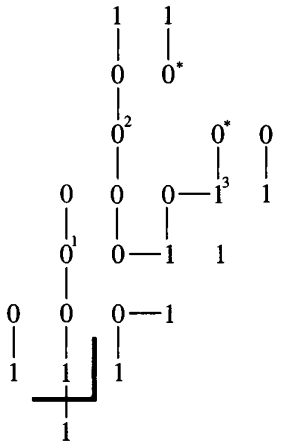


Case 1biB1: The 0 following 1^3 is right:



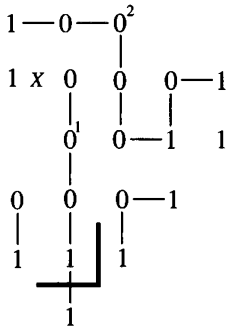
This configuration is impossible.

Case 1biB2: The 0 following 1^3 is up:



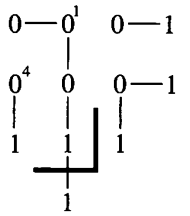
This configuration is impossible.

Case 1biC: The 0 following 0^2 is left:

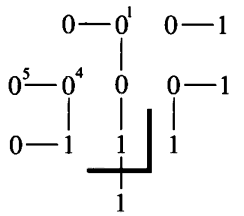


There must be a loss at the X.

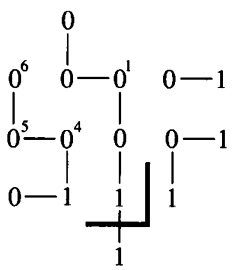
Case 1bii: The 0 following 0^1 is left:



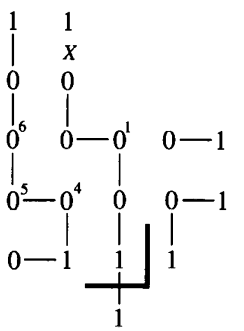
Case 1biiA: The bit following 0^4 is a 0:



Case 1biiA1: The 0 following 0^5 is up:

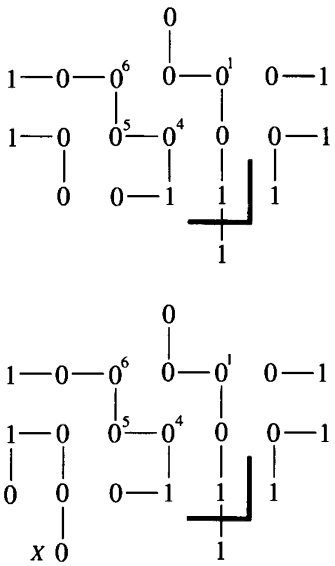


Case 1biiA1a: The 0 following 0^6 is up:



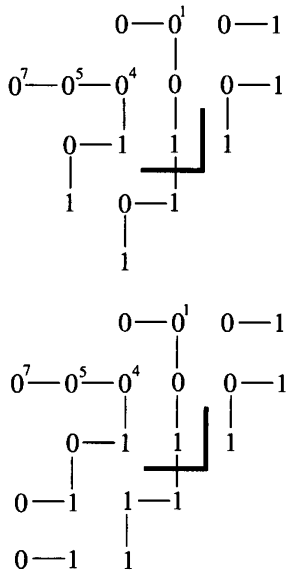
There must be a loss at the X.

Case 1biiA1b: The 0 following 0^6 is left:

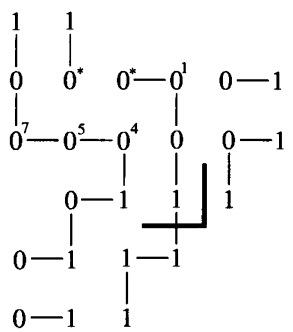


There must be a loss at the X.

Case 1biiA2: The 0 following 0^5 is left:

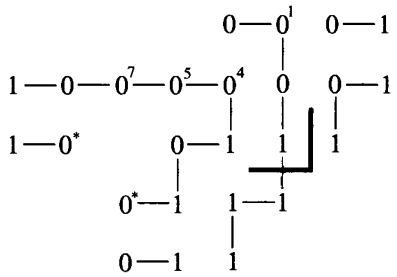


Case 1biiA2a: The 0 following 0^7 is up:



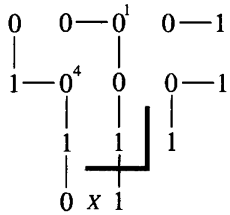
This configuration is impossible.

Case 1biiA2b: The 0 following 0^7 is left:



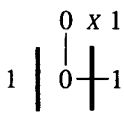
This configuration is impossible.

Case 1biiB: The bit following 0^4 is a 1:



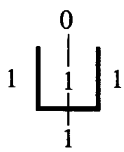
There must be a loss at the X.

Case 2: The intended boundary point is a prefix or suffix 1 in pattern 2:

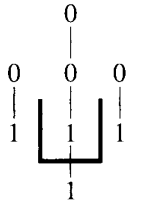


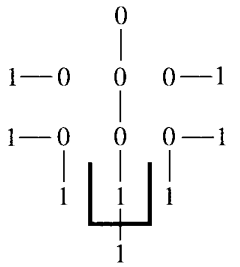
We may assume without loss of generality that the prefix or suffix 1 is adjacent to the one on the right and the neighboring 0 to the prefix or suffix 1 is up. There must be a loss at the X.

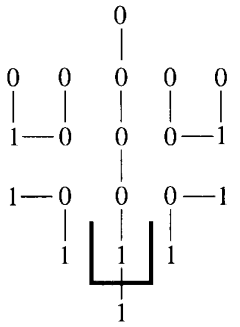
Case 3: The intended boundary point is a prefix or suffix 1 in pattern 3:

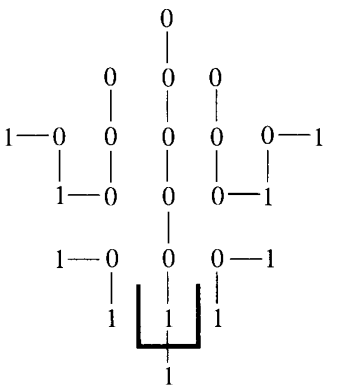


We may assume the prefix or suffix 1 is adjacent to the lower inside 1, for otherwise there would be a loss as in Case 2.

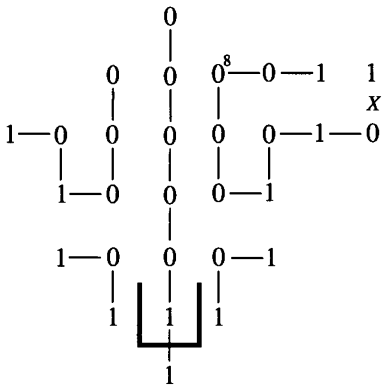






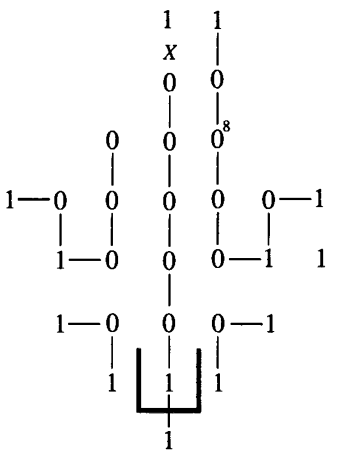


Case 3a: The 0 following 0^8 is right:



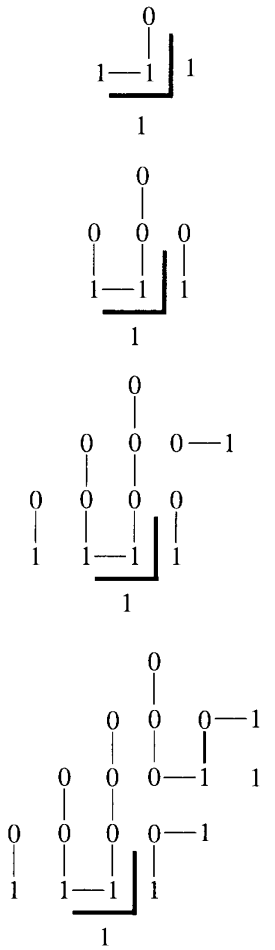
There must be a loss at the X.

Case 3b: The 0 following 0^8 is up:

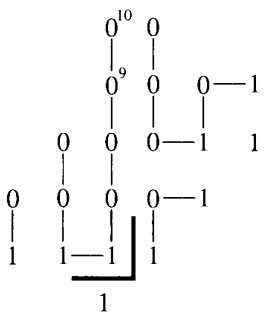


There must be a loss at the X.

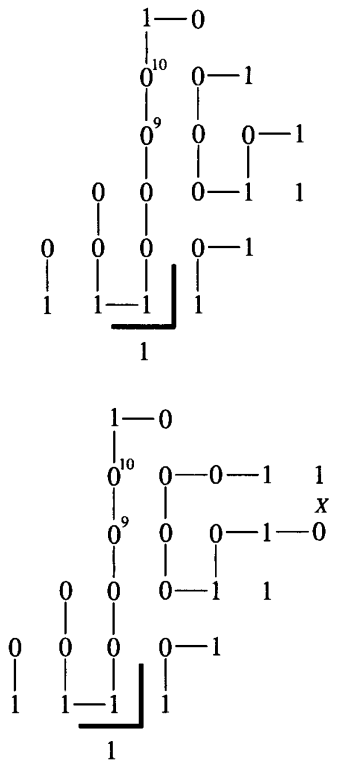
Case 4: The intended boundary point is a flank 1 contained in the substring 110 in pattern 1:



Case 4a: The 0 following 0^9 is up:

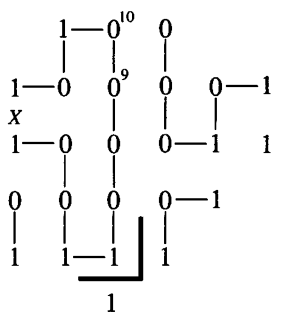


Case 4ai: The 1 following 0^{10} is up:



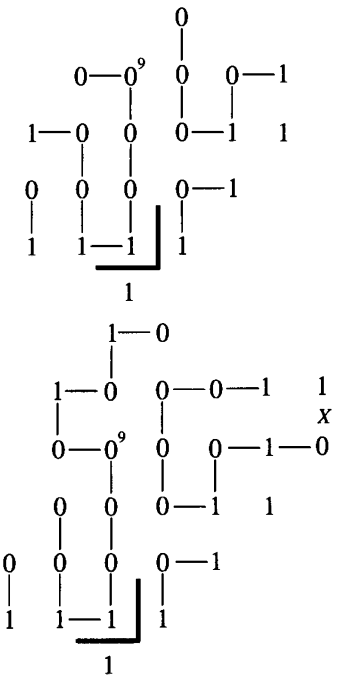
There must be a loss at the X.

Case 4aii: The 1 following 0^{10} is left:



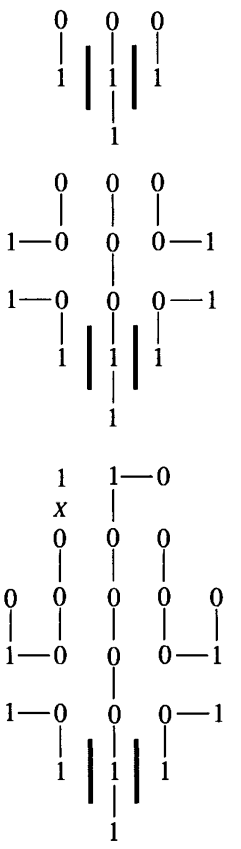
There must be a loss at the X.

Case 4b: The 0 following 0^9 is left:



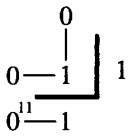
There must be a loss at the X.

Case 5: The intended boundary point is a flank 1 contained in the substring 110 in pattern 2:

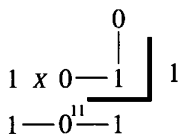


There must be a loss at the X.

Case 6: The intended boundary point is a flank 1 contained in the substring 010 in pattern 1. Since there are no intended corners nearby, the 1 is, in fact, contained in the larger substring 0010100. The picture follows, we will assume the upper 0 is followed by a 1:

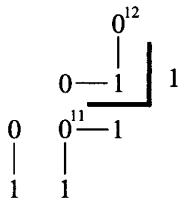


Case 6a: The 1 following 0^{11} is left:

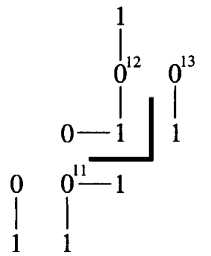


There must be a loss at the X.

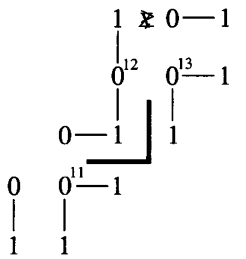
Case 6b: The 1 following 0^{11} is down:



Case 6bi: The 1 following 0^{12} is up:

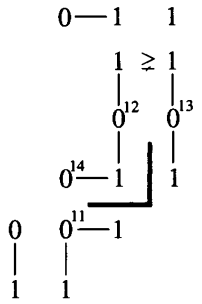


Case 6biA: The 1 following 0^{13} is right:

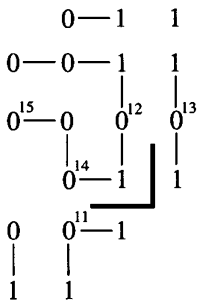


There must be a loss at the X.

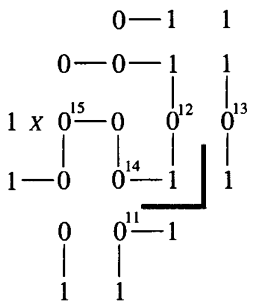
Case 6biB: The 1 following 0^{13} is up:



Case 6biB1: The 0 following 0^{14} is up:

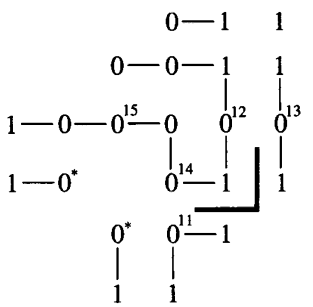


Case 6biB1a: The 0 following 0^{15} is down:



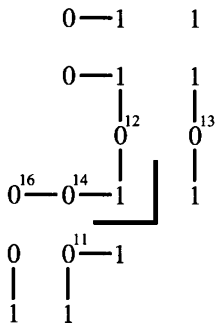
There must be a loss at the X.

Case 6biB1b: The 0 following 0^{15} is left:

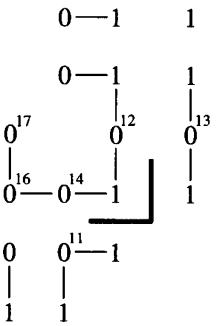


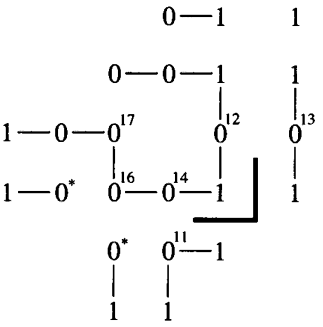
This configuration is impossible.

Case 6biB2: The 0 following 0^{14} is left:



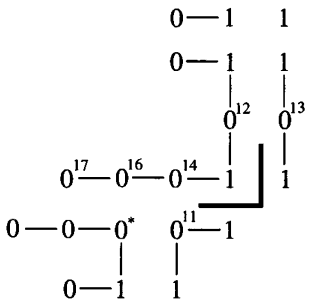
Case 6biB2a: The 0 following 0^{16} is up:



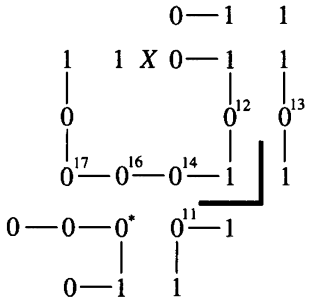


This configuration is impossible.

Case 6biB2b: The 0 following 0^{16} is left:

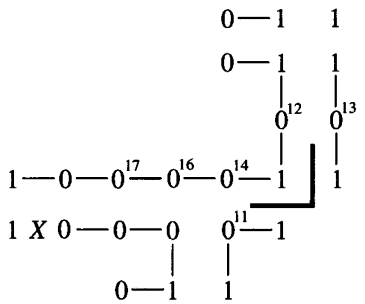


Case 6biB2bi: The 0 following 0^{17} is up:



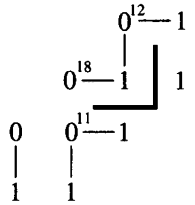
There must be a loss at the X.

Case 6biB2bii: The 0 following 0^{17} is left:

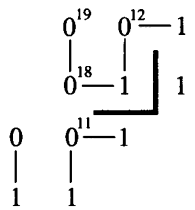


There must be a loss at the X.

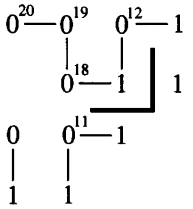
Case 6bii: The 1 following 0^{12} is right:



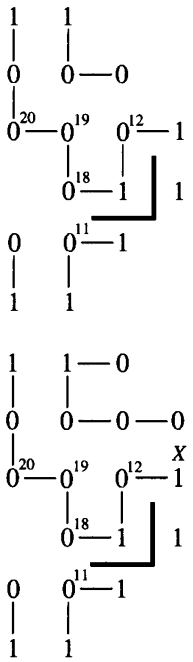
Case 6biiA: The 0 following 0^{18} is up:



Case 6biiA1: The 0 following 0^{19} is left:

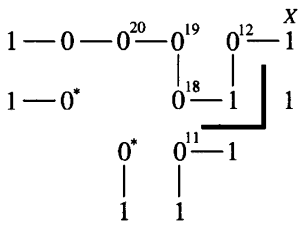


Case 6biiA1a: The 0 following 0^{20} is up:



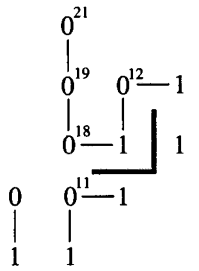
There must be a loss at the X.

Case 6biiA1b: The 0 following 0^{20} is left:

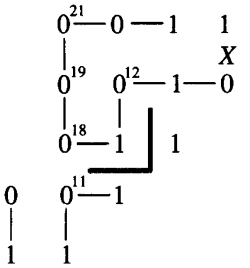


This configuration is impossible.

Case 6biiA2: The 0 following 0^{19} is up:

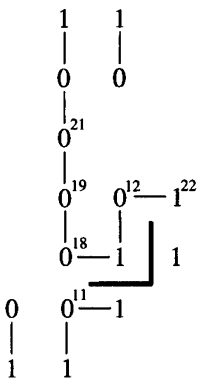


Case 6biiA2a: The 0 following 0^{21} is right:

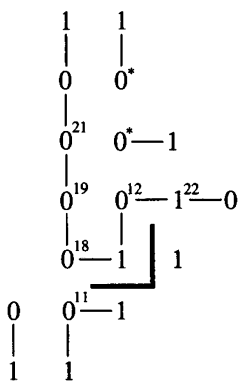


There must be a loss at the X.

Case 6biiA2b: The 0 following 0^{21} is up:

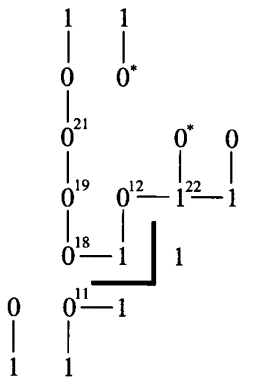


Case 6biiA2bi: The 0 following 1^{22} is right:



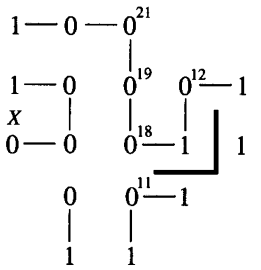
This configuration is impossible.

Case 6biiA2bii: The 0 following 1^{22} is up:



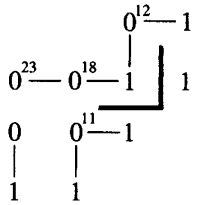
This configuration is impossible.

Case 6biiA2c: The 0 following 0^{21} is left:

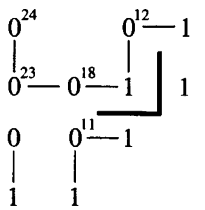


There must be a loss at the X.

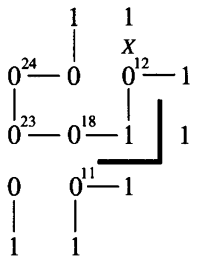
Case 6biiB: The 0 following 0^{18} is left:



Case 6biiB1: The 0 following 0^{23} is up:

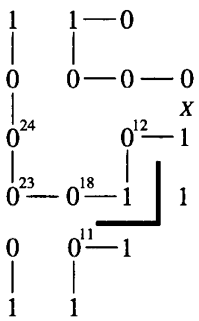


Case 6biiB1a: The 0 following 0^{24} is right:



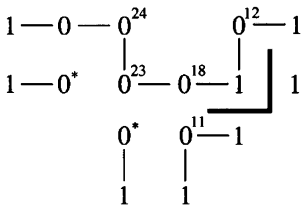
There must be a loss at the X.

Case 6biiB1b: The 0 following 0^{24} is up:



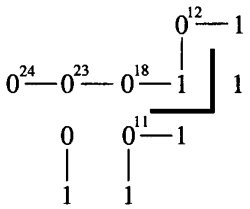
There must be a loss at the X.

Case 6biiB1c: The 0 following 0^{24} is left:

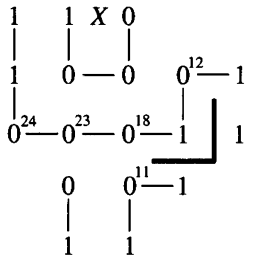


This configuration is impossible.

Case 6biiB2: The 0 following 0^{23} is left:

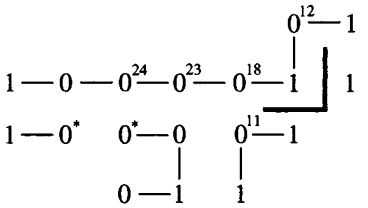


Case 6biiB2a: The 0 following 0^{24} is up:



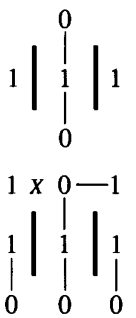
There must be a loss at the X.

Case 6biiB2b: The 0 following 0^{24} is left:



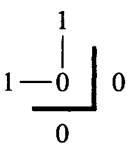
This configuration is impossible.

Case 7: The intended boundary point (not an intended corner) is a flank 1 contained in the substring 010 in pattern 2. We may assume without loss of generality that the lower 0 is followed by a 0 in the following picture:



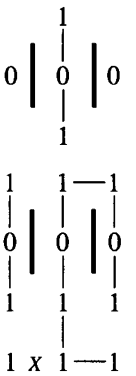
There must be a loss at the X.

Case 8: The intended boundary point is a flank internal 0 in pattern 1:



One of the inside 0's must be bent, thus Lemma 3 implies there is either an intended corner or loss within distance 8 of the flank 0.

Case 9: The intended boundary point is a flank internal 0 in pattern 2:



The inside 0's cannot be bent due to Lemma 3, thus there must be a loss at the X. This completes our case analysis.

In all cases, we showed that under our assumptions the potential configurations must either be impossible or contain a loss within distance 8 of the corner point (all points shown in the figures were within distance 8 of the corner point). ■

REFERENCES

Berger, B., and Leighton, F.T. 1997. Manuscript submitted. *Journal of Computational Biology*, July.

Chan, H.S., and Dill, K.A. 1993. The protein folding problem. *Physics Today* 24–32.

Dill, K.A., Bromberg, S., Yue, K., Fiebig, K., Yee, D.P., Thomas, P.D., and Chan, H.S. 1995. Principles of protein folding—a perspective from simple exact models. *Protein Science* 561–602.

Dill, K.A. 1990. Dominant forces in protein folding. *Biochemistry* 29, 7133–7155.

Fraenkel, A.S. 1993. Complexity of protein folding. *Bulletin of mathematical Biology* 55(6), 1199–1210.

Hart, W.E., and Istrail, S. 1995. Fast protein folding in the hydrophobic-hydrophilic model within three-eights of optimal. *Proceedings of the 27th Annual ACM Symposium on the Theory of Computing (STOC 95)*, 157–168.

Johnson, D.S., and Papadimitriou, C.H. 1985. Computational complexity. *In:* Lawler, E.L., Lenstra, J.K., Kan, A.H.G. Rinooy, and Shmoys, D.B. (eds.). *The Traveling Salesman Problem. A Guided Tour of Combinatorial Optimization.* Wiley Interscience.

Khalimsky, E., Kopperman, R., and Meyer, P.R. 1990. Computer graphics and connected topologies on finite oredered sets. *Topology and its Applications* 36, 1–17.

King, J. 1989, Deciphering the rules of protein folding. *Chemical Engineering News* 67, 32–54.

Nayak, A., Sinclair, A., and Zwick, U. 1998. Spatial codes and the hardness of string folding. *In: Proceedings of the acm-siam symposium on discrete algorithms.* To appear.

Ngo, T., Marks, J., and Karplus. 1994. *The Protein Folding Problem and Tertiary Structure Prediction.* Birkhauser. 435–508.

Paterson, M., and Przytycka, T. 1996. On the complexity of string folding. *Discrete Applied Mathematics* 71, 217–230.

Trevisan, L. 1996. When Hamming meets Euclid: The approximability of geometric TSP and MST. *In: Proceeding of the 28th Symposium on the Theory of Computing.*

Unger, R., and Moult, J. 1993. Finding the lowest free energy conformation of a protein in an NP-hard problem: Proof and implications. *Bulletin of Mathematical Biology* 55(6), 1183–1198.

Address reprint requests to:
Pierluigi Crescenzi
Dipartimento di Sistemi e Informatica
Università di Firenze
Via C. Lombroso 6/17
Firenze, Italy

Received for publication March 1, 1998; accepted as revised May 20, 1998.

This article has been cited by:

1. Andreas Holzinger. 2018. Introduction to MAchine Learning & Knowledge Extraction (MAKE). *Machine Learning and Knowledge Extraction* **1**:1, 1-20. [Crossref]
2. Alexey V. Melkikh, Andrei Khrennikov. 2018. Mechanisms of directed evolution of morphological structures and the problems of morphogenesis. *Biosystems* **168**, 26-44. [Crossref]
3. Guo Qianghui, Wang Jian, Xu Zhao. 2018. Approximation Algorithms for Protein Folding in the Hydrophobic-Polar Model on 3D Hexagonal Prism Lattice. *Journal of Computational Biology* **25**:5, 487-498. [Abstract] [Full Text] [PDF] [PDF Plus]
4. Amir Morshedian, Jafar Razmara, Shahriar Lotfi. 2018. A novel approach for protein structure prediction based on an estimation of distribution algorithm. *Soft Computing* **181**. . [Crossref]
5. Anylu Melo-Vega, Juan Frausto-Solís, Guadalupe Castilla-Valdez, Ernesto Liñán-García, Juan Javier González-Barbosa, David Terán-Villanueva. Protein Folding Problem in the Case of Peptides Solved by Hybrid Simulated Annealing Algorithms 141-152. [Crossref]
6. Bhavin J. Shastri, Alexander N. Tait, Thomas Ferreira de Lima, Mitchell A. Nahmias, Hsuan-Tung Peng, Paul R. Prucnal. Neuromorphic Photonics, Principles of 1-37. [Crossref]
7. Alexey V. Melkikh, Dirk K.F. Meijer. 2018. On a generalized Levinthal's paradox: The role of long- and short range interactions in complex bio-molecular reactions, including protein and DNA folding. *Progress in Biophysics and Molecular Biology* **132**, 57-79. [Crossref]
8. Menachem Stern, Matthew B. Pinson, Arvind Murugan. 2017. The Complexity of Folding Self-Folding Origami. *Physical Review X* **7**:4. . [Crossref]
9. Alexey V. Melkikh, Andrei Khrennikov. 2017. Molecular recognition of the environment and mechanisms of the origin of species in quantum-like modeling of evolution. *Progress in Biophysics and Molecular Biology* **130**, 61-79. [Crossref]
10. Yang Cheng-Hong, Lin Yu-Shiun, Chuang Li-Yeh, Chang Hsueh-Wei. 2017. A Particle Swarm Optimization-Based Approach with Local Search for Predicting Protein Folding. *Journal of Computational Biology* **24**:10, 981-994. [Abstract] [Full Text] [PDF] [PDF Plus]
11. Dai Zheng, Becerra David, Waldispühl Jérôme. 2017. On Stable States in a Topologically Driven Protein Folding Model. *Journal of Computational Biology* **24**:9, 851-862. [Abstract] [Full Text] [PDF] [PDF Plus] [Supplementary Material]
12. D. Ramyachitra, A. Ajeeth. 2017. MODCSA-CA: A multi objective diversity controlled self adaptive cuckoo algorithm for protein structure prediction. *Gene Reports* **8**, 100-106. [Crossref]
13. Leonardo de Lima Correa, Mario Inostroza-Ponta, Marcio Dorn. An evolutionary multi-agent algorithm to explore the high degree of selectivity in three-dimensional protein structures 1111-1118. [Crossref]
14. Bruno Grisci, Márcio Dorn. 2017. NEAT-FLEX: Predicting the conformational flexibility of amino acids using neuroevolution of augmenting topologies. *Journal of Bioinformatics and Computational Biology* **15**:03, 1750009. [Crossref]
15. . Neuromorphic Engineering 3-26. [Crossref]
16. Bruno Borguesan, Jonas Bohrer, Mariel Barbachan e Silva, Leonardo de Lima Correa, Marcio Dorn. Improving protein tertiary structure prediction with conformational propensities of amino acid residues 9-15. [Crossref]
17. Bruno Grisci, Marcio Dorn. Predicting protein structural features with NeuroEvolution of Augmenting Topologies 873-880. [Crossref]
18. Yong Wang, Ying Wang, Zikai Wu, Yuzhen Guo. 2016. Extended particle swarm optimisation method for folding protein on triangular lattice. *IET Systems Biology* **10**:1, 30-33. [Crossref]
19. Dipan Lal Shaw, A. S. M. Shohidull Islam, Shuvasish Karmaker, M. Sohel Rahman. Approximation Algorithms for Three Dimensional Protein Folding 274-285. [Crossref]
20. Thiago Lipinski-Paes, Michele dos Santos da Silva Tanus, José Fernando Ruggiero Bachega, Osmar Norberto de Souza. A Multiagent Ab Initio Protein Structure Prediction Tool for Novices and Experts 163-174. [Crossref]

21. Andreas Holzinger, Markus Plass, Katharina Holzinger, Gloria Cerasela Crişan, Camelia-M. Pintea, Vasile Palade. Towards interactive Machine Learning (iML): Applying Ant Colony Algorithms to Solve the Traveling Salesman Problem with the Human-in-the-Loop Approach 81-95. [Crossref]
22. Mark Pexton. 2015. Emergence and Fundamentality in a Pancomputationalist Universe. *Minds and Machines* **25**:4, 301-320. [Crossref]
23. Daniela Xavier, Berta Crespo, Rubén Fuentes-Fernández. 2015. A rule-based expert system for inferring functional annotation. *Applied Soft Computing* **35**, 373-385. [Crossref]
24. Busara Pattanasiri, Ying Wai Li, Thomas Wust, David P Landau. 2015. Effect of surface attractive strength on structural transitions of a confined HP lattice protein. *Journal of Physics: Conference Series* **640**, 012015. [Crossref]
25. Amarda Shehu. A Review of Evolutionary Algorithms for Computing Functional Conformations of Protein Molecules 31-64. [Crossref]
26. Mario Inostroza-Ponta, Camilo Farfán, Márcio Dorn. A Memetic Algorithm for Protein Structure Prediction based on Conformational Preferences of Aminoacid Residues 1403-1404. [Crossref]
27. Márcio Dorn, Mariel Barbachan e Silva, Luciana S. Buriol, Luis C. Lamb. 2014. Three-dimensional protein structure prediction: Methods and computational strategies. *Computational Biology and Chemistry* **53**, 251-276. [Crossref]
28. Farooq Ahmed. 2014. Profile of Christos Papadimitriou. *Proceedings of the National Academy of Sciences* **111**:45, 15858-15860. [Crossref]
29. Santos José, Villot Pablo, Diéguez Martin. 2014. Emergent Protein Folding Modeled with Evolved Neural Cellular Automata Using the 3D HP Model. *Journal of Computational Biology* **21**:11, 823-845. [Abstract] [Full Text] [PDF] [PDF Plus]
30. Andrea G. Citrolo, Giancarlo Mauri. 2014. A local landscape mapping method for protein structure prediction in the HP model. *Natural Computing* **13**:3, 309-319. [Crossref]
31. Thiago Lipinski-Paes, Osmar Norberto de Souza. 2014. MASTERS: A General Sequence-based MultiAgent System for Protein TERtiary Structure Prediction. *Electronic Notes in Theoretical Computer Science* **306**, 45-59. [Crossref]
32. Shih Chieh Su, Jyh Jong Tsay. A Novel Offspring Selection Strategy in GAs for Protein Structure Prediction 1171-1174. [Crossref]
33. Ryan Babbush, Alejandro Perdomo-Ortiz, Bryan O'Gorman, William Macready, Alan Aspuru-Guzik. Construction of Energy Functions for Lattice Heteropolymer Models: Efficient Encodings for Constraint Satisfaction Programming and Quantum Annealing 201-244. [Crossref]
34. Márcio Dorn, Luciana S. Buriol, Luis C. Lamb. 2014. MOIRAE: A computational strategy to extract and represent structural information from experimental protein templates. *Soft Computing* **18**:4, 773-795. [Crossref]
35. Diego Masone, Solène Grosdidier. 2014. Collective variable driven molecular dynamics to improve protein–protein docking scoring. *Computational Biology and Chemistry* **49**, 1-6. [Crossref]
36. Fábio Lima Custódio, Helio J.C. Barbosa, Laurent Emmanuel Dardenne. 2014. A multiple minima genetic algorithm for protein structure prediction. *Applied Soft Computing* **15**, 88-99. [Crossref]
37. Jingfa Liu, Yuanyuan Sun, Gang Li, Beibei Song, Weibo Huang. 2013. Heuristic-based tabu search algorithm for folding two-dimensional AB off-lattice model proteins. *Computational Biology and Chemistry* **47**, 142-148. [Crossref]
38. Emanuele Giaquinta, Laura Pozzi. 2013. An Effective Exact Algorithm and a New Upper Bound for the Number of Contacts in the Hydrophobic-Polar Two-Dimensional Lattice Model. *Journal of Computational Biology* **20**:8, 593-609. [Abstract] [Full Text] [PDF] [PDF Plus]
39. Yuzhen Guo, Yong Wang. Predicting the non-compact conformation of amino acid sequence by particle swarm optimization 119-122. [Crossref]
40. Christiane Regina Soares Brasil, Alexandre Claudio Botazzo Delbem, Fernando Luís Barroso da Silva. 2013. Multiobjective evolutionary algorithm with many tables for purely ab initio protein structure prediction. *Journal of Computational Chemistry* **34**:20, 1719-1734. [Crossref]
41. Mithun Radhakrishna, Joseph Grimaldi, Georges Belfort, Sanat K. Kumar. 2013. Stability of Proteins Inside a Hydrophobic Cavity. *Langmuir* **29**:28, 8922-8928. [Crossref]
42. Swakkhar Shatabda, M.A.Hakim Newton, Mahmood A Rashid, Abdul Sattar. An efficient encoding for simplified protein structure prediction using genetic algorithms 1217-1224. [Crossref]

43. Marcio Dorn, Mario Inostroza-Ponta, Luciana S. Buriol, Hugo Verli. A knowledge-based genetic algorithm to predict three-dimensional structures of polypeptides 1233-1240. [Crossref]
44. Gerhard König. 2013. Simulation and the Problem of Simplification. *Philosophy & Technology* **26**:1, 81-91. [Crossref]
45. Márcio Dorn, Luciana S. Buriol, Luis C. Lamb. 2013. A molecular dynamics and knowledge-based computational strategy to predict native-like structures of polypeptides. *Expert Systems with Applications* **40**:2, 698-706. [Crossref]
46. Ying Wai Li, Thomas Wüst, David P. Landau. 2013. Generic folding and transition hierarchies for surface adsorption of hydrophobic-polar lattice model proteins. *Physical Review E* **87**:1. . [Crossref]
47. Jyh-Jong Tsay, Shih-Chieh Su. 2013. An effective evolutionary algorithm for protein folding on 3D FCC HP model by lattice rotation and generalized move sets. *Proteome Science* **11**:Suppl 1, S19. [Crossref]
48. Alejandro Perdomo-Ortiz, Neil Dickson, Marshall Drew-Brook, Geordie Rose, Alán Aspuru-Guzik. 2012. Finding low-energy conformations of lattice protein models by quantum annealing. *Scientific Reports* **2**:1. . [Crossref]
49. Márcio Dorn, André L.S. Braga, Carlos H. Llanos, Leandro S. Coelho. 2012. A GMDH polynomial neural network-based method to predict approximate three-dimensional structures of polypeptides. *Expert Systems with Applications* **39**:15, 12268-12279. [Crossref]
50. Jonathan Birch. 2012. Robust processes and teleological language. *European Journal for Philosophy of Science* **2**:3, 299-312. [Crossref]
51. Thomas Wüst, David P. Landau. 2012. Optimized Wang-Landau sampling of lattice polymers: Ground state search and folding thermodynamics of HP model proteins. *The Journal of Chemical Physics* **137**:6, 064903. [Crossref]
52. BUSARA PATTANASIRI, YING WAI LI, DAVID P. LANDAU, THOMAS WÜST. 2012. WANG–LANDAU SIMULATIONS OF ADSORBED AND CONFINED LATTICE PROTEINS. *International Journal of Modern Physics C* **23**:08, 1240008. [Crossref]
53. Andre L. S. Braga, Janier Arias-Garcia, Carlos Llanos, Marcio Dorn, Alfredo Foltran, Leandro S. Coelho. Hardware implementation of GMDH-type artificial neural networks and its use to predict approximate three-dimensional structures of proteins 1-8. [Crossref]
54. Mithun Radhakrishna, Sumit Sharma, Sanat K. Kumar. 2012. Enhanced Wang Landau sampling of adsorbed protein conformations. *The Journal of Chemical Physics* **136**:11, 114114. [Crossref]
55. Paulo H. R. Gabriel, Vinícius V. de Melo, Alexandre C. B. Delbem. 2012. Algoritmos evolutivos e modelo HP para predição de estruturas de proteínas. *Sba: Controle & Automação Sociedade Brasileira de Automatica* **23**:1, 25-37. [Crossref]
56. Camelia Chira, Dragos Horvath, D Dumitrescu. 2011. Hill-Climbing search and diversification within an evolutionary approach to protein structure prediction. *BioData Mining* **4**:1. . [Crossref]
57. I. Dotu, M. Cebrian, P. Van Hentenryck, P. Clote. 2011. On Lattice Protein Structure Prediction Revisited. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* **8**:6, 1620-1632. [Crossref]
58. Ying Wai Li, Thomas Wüst, David P. Landau. 2011. Monte Carlo simulations of the HP model (the "Ising model" of protein folding). *Computer Physics Communications* **182**:9, 1896-1899. [Crossref]
59. T. Wüst, Y. W. Li, D. P. Landau. 2011. Unraveling the Beautiful Complexity of Simple Lattice Model Polymers and Proteins Using Wang-Landau Sampling. *Journal of Statistical Physics* **144**:3, 638-651. [Crossref]
60. Camelia Chira. A hybrid evolutionary approach to protein structure prediction with lattice models 2300-2306. [Crossref]
61. Mario Garza-Fabre, Eduardo Rodriguez-Tello, Gregorio Toscano-Pulido. Comparing alternative energy functions for the HP model of protein structure prediction 2307-2314. [Crossref]
62. Marcio Dorn, Luciana S. Buriol, Luis C. Lamb. A hybrid genetic algorithm for the 3-D protein structure prediction problem using a path-relinking strategy 2709-2716. [Crossref]
63. Sun-Yuan Hsieh, De-Wei Lai. 2011. A New Branch and Bound Method for the Protein Folding Problem Under the 2D-HP Model. *IEEE Transactions on NanoBioscience* **10**:2, 69-75. [Crossref]
64. Ivan Kassal, James D. Whitfield, Alejandro Perdomo-Ortiz, Man-Hong Yung, Alán Aspuru-Guzik. 2011. Simulating Chemistry Using Quantum Computers. *Annual Review of Physical Chemistry* **62**:1, 185-207. [Crossref]
65. Vincenzo Cutello, Giuseppe Morelli, Giuseppe Nicosia, Mario Pavone, Giuseppe Scollo. 2011. On discrete models and immunological algorithms for protein structure prediction. *Natural Computing* **10**:1, 91-102. [Crossref]

66. Md. Kamrul Islam, Madhu Chetty, A. Dayem Ullah, K. Steinhöfel. A Memetic Approach to Protein Structure Prediction in Triangular Lattices 625-635. [Crossref]
67. Shih-Chieh Su, Cheng-Jian Lin, Chuan-Kang Ting. 2011. An effective hybrid of hill climbing and genetic algorithm for 2D triangular protein structure prediction. *Proteome Science* **9**:Suppl 1, S19. [Crossref]
68. Shih-Chieh Su, Cheng-Jian Lin, Chuan-Kang Ting. An efficient hybrid of hill-climbing and genetic algorithm for 2D triangular protein structure prediction 51-56. [Crossref]
69. Roberto Santana, Pedro Larrañaga, José A. Lozano. 2010. Learning Factorizations in Estimation of Distribution Algorithms Using Affinity Propagation. *Evolutionary Computation* **18**:4, 515-546. [Crossref]
70. Benhui Chen, Jinglu Hu. 2010. A Hybrid EDA for Protein Folding Based on HP Model. *IEEJ Transactions on Electrical and Electronic Engineering* **5**:4, 459-466. [Crossref]
71. Dragos Horvath, Camelia Chira. Simplified chain folding models as metaheuristic benchmark for tuning real protein folding algorithms? 1-8. [Crossref]
72. Chenhua Huang, Xiangbo Yang, Zhihong He. 2010. Protein folding simulations of 2D HP model by the genetic algorithm based on optimal secondary structures. *Computational Biology and Chemistry* **34**:3, 137-142. [Crossref]
73. Fernanda Hembecker, Heitor Silverio Lopes. A Molecular Model for Representing Protein Structures and Its Application to Protein Folding 1-6. [Crossref]
74. César Manuel Vargas Benítez, Heitor Silvério Lopes. 2010. Protein structure prediction with the 3D-HP side-chain model using a master–slave parallel genetic algorithm. *Journal of the Brazilian Computer Society* **16**:1, 69-78. [Crossref]
75. Namsu Ahn, Sungsoo Park. 2010. Finding an Upper Bound for the Number of Contacts in Hydrophobic-Hydrophilic Protein Structure Prediction Model. *Journal of Computational Biology* **17**:4, 647-656. [Abstract] [PDF] [PDF Plus]
76. Abu Dayem Ullah, Kathleen Steinhofel. Upper bounds for foldings in the FCC-HP protein model 101-106. [Crossref]
77. Xiangqian Hu, David N. Beratan, Weitao Yang. 2009. A gradient-directed Monte Carlo method for global optimization in a discrete space: Application to protein sequence design and folding. *The Journal of Chemical Physics* **131**:15, 154117. [Crossref]
78. Fotini Markopoulou, Simone Severini. 2009. A Note on Observables for Counting Trails and Paths in Graphs. *Journal of Mathematical Modelling and Algorithms* **8**:3, 335-342. [Crossref]
79. Agostino Dovier, Andrea Formisano, Enrico Pontelli. 2009. An empirical study of constraint logic programming and answer set programming solutions of combinatorial problems. *Journal of Experimental & Theoretical Artificial Intelligence* **21**:2, 79-121. [Crossref]
80. Benhui Chen, Long Li, Jinglu Hu. A novel EDAs based method for HP model protein folding 309-315. [Crossref]
81. T. Wüst, D.P. Landau, C. Gervais, Y. Xu. 2009. Monte Carlo simulations of systems with complex energy landscapes. *Computer Physics Communications* **180**:4, 475-479. [Crossref]
82. Zahra Khani, Ali Ghaffari. Protein Prediction by Intelligent Algorithm 345-350. [Crossref]
83. MIGUEL M. F. BUGALHO, ARLINDO L. OLIVEIRA. 2009. CONSTANT TIME CLASH DETECTION IN PROTEIN FOLDING. *Journal of Bioinformatics and Computational Biology* **07**:01, 55-74. [Crossref]
84. Paulo H. R. Gabriel, Alexandre C. B. Delbem. Representations for Evolutionary Algorithms Applied to Protein Structure Prediction Problem Using HP Model 97-108. [Crossref]
85. Alessandro Dal Palù, Agostino Dovier, Enrico Pontelli. Logic Programming Techniques in Protein Structure Determination: Methodologies and Results 560-566. [Crossref]
86. Giuseppe Lancia. 2008. Mathematical Programming in Computational Biology: an Annotated Bibliography. *Algorithms* **1**:2, 100-129. [Crossref]
87. Alexandru-Adrian Tantar, Nouredine Melab, El-Ghazali Talbi. 2008. A grid-based genetic algorithm combined with an adaptive simulated annealing for protein structure prediction. *Soft Computing* **12**:12, 1185-1198. [Crossref]
88. Yu-Ying Shih, Michael Ho. Fast parallel bio-molecular logic computing algorithms: Protein folding 1-6. [Crossref]
89. BIN FU, SORINEL A. OPRISAN, LIZHE XU. 2008. MULTI-DIRECTIONAL WIDTH-BOUNDED GEOMETRIC SEPARATOR AND PROTEIN FOLDING. *International Journal of Computational Geometry & Applications* **18**:05, 389-413. [Crossref]
90. R. Santana, P. Larranaga, J.A. Lozano. 2008. Protein Folding in Simplified Models With Estimation of Distribution Algorithms. *IEEE Transactions on Evolutionary Computation* **12**:4, 418-438. [Crossref]

91. T. Wüst, D.P. Landau. 2008. The HP model of protein folding: A challenging testing ground for Wang–Landau sampling. *Computer Physics Communications* **179**:1-3, 124-127. [Crossref]
92. Alejandro Perdomo, Colin Truncik, Ivan Tubert-Brohman, Geordie Rose, Alán Aspuru-Guzik. 2008. Construction of model Hamiltonians for adiabatic quantum computation and its application to finding low-energy conformations of lattice protein models. *Physical Review A* **78**:1. . [Crossref]
93. Xinchao Zhao. 2008. Advances on protein folding simulations based on the lattice HP models with natural computing. *Applied Soft Computing* **8**:2, 1029-1040. [Crossref]
94. Heitor Silverio Lopes, Reginaldo Bitello. 2007. A Differential Evolution Approach for Protein Folding Using a Lattice Model. *Journal of Computer Science and Technology* **22**:6, 904-908. [Crossref]
95. Toni Mancini, Marco Cadoli. 2007. Exploiting functional dependencies in declarative problem specifications. *Artificial Intelligence* **171**:16-17, 985-1010. [Crossref]
96. Alessandro Dal Palu, Agostino Dovier, Enrico Pontelli. Enhancing the computation of approximate solutions of the protein structure determination problem through global constraints for discrete crystal lattices 38-44. [Crossref]
97. Hans-Joachim Böckenhauer, Dirk Bongartz. 2007. A weighted HP model for protein folding with diagonal contacts. *RAIRO - Theoretical Informatics and Applications* **41**:4, 375-402. [Crossref]
98. Carolina P. Almeida, Richard A. Goncalves, Marco C. Goldbarg, Elizabeth F. G. Goldbarg, Myriam R. Delgado. TA-PFP: A Transgenetic Algorithm to Solve the Protein Folding Problem 163-168. [Crossref]
99. Benjamin Parent, Alexandru Tantar, Nouredine Melab, El-Ghazali Talbi, Dragos Horvath. Grid-based evolutionary strategies applied to the conformational sampling problem 291-296. [Crossref]
100. Marco Cadoli, Toni Mancini. 2007. USING A THEOREM PROVER FOR REASONING ON CONSTRAINT PROBLEMS. *Applied Artificial Intelligence* **21**:4-5, 383-404. [Crossref]
101. Piotr Berman, Bhaskar DasGupta, Dhruv Mubayi, Robert Sloan, György Turán, Yi Zhang. 2007. The inverse protein folding problem on 2D and 3D lattices. *Discrete Applied Mathematics* **155**:6-7, 719-732. [Crossref]
102. Vincenzo Cutello, Giuseppe Nicosia, Mario Pavone, Jonathan Timmis. 2007. An Immune Algorithm for Protein Structure Prediction on Lattice Models. *IEEE Transactions on Evolutionary Computation* **11**:1, 101-117. [Crossref]
103. Julia Hockenmaier, Aravind K. Joshi, Ken A. Dill. 2007. Routes are trees: The parsing perspective on protein folding. *Proteins: Structure, Function, and Bioinformatics* **66**:1, 1-15. [Crossref]
104. Alexandru-Adrian Tantar, Nouredine Melab, El-Ghazali Talbi. A Comparative Study of Parallel Metaheuristics for Protein Structure Prediction on the Computational Grid 1-10. [Crossref]
105. Bin Fu, Wei Wang. 2007. Geometric Separators and Their Applications to Protein Folding in the HP-Model. *SIAM Journal on Computing* **37**:4, 1014-1029. [Crossref]
106. EUNICE E. SANTOS, EUGENE SANTOS. 2006. EFFECTIVE COMPUTATIONAL REUSE FOR ENERGY EVALUATIONS IN PROTEIN FOLDING. *International Journal on Artificial Intelligence Tools* **15**:05, 725-739. [Crossref]
107. Nilton Armstrong, Heitor Lopes, Carlos Erig Lima. Preliminary Steps Towards Protein Folding Prediction Using Reconfigurable Computing 1-7. [Crossref]
108. S. C. Kou, Jason Oh, Wing Hung Wong. 2006. A study of density of states and ground states in hydrophobic-hydrophilic protein folding models by equi-energy sampling. *The Journal of Chemical Physics* **124**:24, 244903. [Crossref]
109. David Chiang, Aravind K. Joshi, Ken A. Dill. 2006. A Grammatical Theory for the Conformational Changes of Simple Helix Bundles. *Journal of Computational Biology* **13**:1, 21-42. [Abstract] [PDF] [PDF Plus]
110. Jacek Błażewicz, Piotr Łukasiak, Maciej Miłostan. 2005. Application of tabu search strategy for finding low energy structure of protein. *Artificial Intelligence in Medicine* **35**:1-2, 135-145. [Crossref]
111. XIANG-SUN ZHANG, YONG WANG, ZHONG-WEI ZHAN, LING-YUN WU, LUONAN CHEN. 2005. EXPLORING PROTEIN'S OPTIMAL HP CONFIGURATIONS BY SELF-ORGANIZING MAPPING. *Journal of Bioinformatics and Computational Biology* **03**:02, 385-400. [Crossref]
112. Reinhard Schiemann, Michael Bachmann, Wolfhard Janke. 2005. Exact sequence analysis for three-dimensional hydrophobic-polar lattice proteins. *The Journal of Chemical Physics* **122**:11, 114705. [Crossref]

113. P. Clote. 2005. An Efficient Algorithm to Compute the Landscape of Locally Optimal RNA Secondary Structures with Respect to the Nussinov–Jacobson Energy Model. *Journal of Computational Biology* **12**:1, 83-101. [Abstract] [PDF] [PDF Plus]
114. Mao Chen, Wen-Qi Huang. 2005. A Branch and Bound Algorithm for the Protein Folding Problem in the HP Lattice Model. *Genomics, Proteomics & Bioinformatics* **3**:4, 225-230. [Crossref]
115. Richard Wroe, Erich Bornberg-Bauer, Hue Sun Chan. 2005. Comparing Folding Codes in Simple Heteropolymer Models of Protein Evolutionary Landscape: Robustness of the Superfunnel Paradigm. *Biophysical Journal* **88**:1, 118-131. [Crossref]
116. Zhenping Li, Xiangsun Zhang, Luonan Chen. 2005. Unique Optimal Foldings of Proteins on a Triangular Lattice. *Applied Bioinformatics* **4**:2, 105-116. [Crossref]
117. Esther M. Arkin, Michael A. Bender, Erik D. Demaine, Martin L. Demaine, Joseph S.B. Mitchell, Saurabh Sethia, Steven S. Skiena. 2004. When can you fold a map?. *Computational Geometry* **29**:1, 23-46. [Crossref]
118. Harvey J. Greenberg, William E. Hart, Giuseppe Lancia. 2004. Opportunities for Combinatorial Optimization in Computational Biology. *INFORMS Journal on Computing* **16**:3, 211-231. [Crossref]
119. Oswin Aichholzer, David Bremner, Erik D. Demaine, Henk Meijer, Vera Sacristán, Michael Soss. 2003. Long proteins with unique optimal foldings in the H-P model. *Computational Geometry* **25**:1-2, 139-159. [Crossref]
120. Vijay Chandru, Abhi DattaSharma, V.S Anil Kumar. 2003. The algorithmics of folding proteins on lattices. *Discrete Applied Mathematics* **127**:1, 145-161. [Crossref]
121. Volker Heun. 2003. Approximate protein folding in the HP side chain model on extended cubic lattices. *Discrete Applied Mathematics* **127**:1, 163-177. [Crossref]
122. James Aspnes, Julia Hartling, Ming-Yang Kao, Junhyong Kim, Gauri Shah. 2002. A Combinatorial Toolbox for Protein Sequence Design and Landscape Analysis in the Grand Canonical Model. *Journal of Computational Biology* **9**:5, 721-741. [Abstract] [PDF] [PDF Plus]
123. Berrin Yanikoglu, Burak Erman. 2002. Minimum Energy Configurations of the 2-Dimensional HP-Model of Proteins by Self-Organizing Networks. *Journal of Computational Biology* **9**:4, 613-620. [Abstract] [PDF] [PDF Plus]
124. Erich Bornberg-Bauer. 2002. Randomness, Structural Uniqueness, Modularity and Neutral Evolution in Sequence Space of Model Proteins. *Zeitschrift für Physikalische Chemie* **216**:2. . [Crossref]
125. Faming Liang, Wing Hung Wong. 2001. Evolutionary Monte Carlo for protein folding simulations. *The Journal of Chemical Physics* **115**:7, 3374-3380. [Crossref]
126. EUNICE E. SANTOS, EUGENE SANTOS. 2001. EFFECTIVE AND EFFICIENT CACHING IN GENETIC ALGORITHMS. *International Journal on Artificial Intelligence Tools* **10**:01n02, 273-301. [Crossref]
127. Sarah Pollock, Hershel M. Safer. Chapter 20. Bioinformatics in the drug discovery process 201-210. [Crossref]
128. Bernd Mayer, Giancarlo Marconi. 2000. Circular dichroic constrained structure optimization of homoalanine peptides. *Journal of Computational Chemistry* **21**:4, 270-281. [Crossref]
129. E.E. Santos, E. Santos. Reducing the computational load of energy evaluations for protein folding 79-86. [Crossref]
130. V. Cutello, G. Nicosia, M. Pavone. An immune algorithm with hyper-macromutations for the Dill's 2D hydrophobic-hydrophilic model 1074-1080. [Crossref]
131. D. Goldman, S. Istrail, C.H. Papadimitriou. Algorithmic aspects of protein structure similarity 512-521. [Crossref]
132. Leonardo de Lima Corrêa, Márcio Dorn. Multi-Agent Systems in Three-Dimensional Protein Structure Prediction 241-278. [Crossref]