Protein Folding in the 2D HP Model

Rune B. Lyngsø
Christian N. S. Pedersen

BRICS Report Series RS-99-16

ISSN 0909-0878 June 1999

Protein folding in the 2D HP model

Rune B. Lyngsø[*] Christian N. S. Pedersen[†]

Abstract

We study folding algorithms in the two dimensional Hydrophobic-Hydrophilic model (2D HP model) for protein structure formation. We consider three generalizations of the best known approximation algorithm. We show that two of the generalizations do not improve the worst case approximation ratio. The third generalization seems to be better, and the analysis of its approximation ratio leads to an interesting combinatorial problem.

1 Introduction

Proteins are polymer chains of amino acids. An interesting feature of nature is that even though there are an infinite amount of amino acids, only twenty different amino acids are used in the formation of proteins. The amino acid sequence of a protein can thus be abstracted as a string over an alphabet of size twenty. In nature proteins are of course not one dimensional strings but fold into three dimensional structures. The three dimensional structure of a protein is not static, but vibrates around an equilibrium known as the *native state*. Famous experiments by Anfinsen et al. [1] showed that a protein in its natural environment folds into, i.e. vibrates around, a unique three dimensional structure, the *native conformation*, independent of the starting conformation. The native conformation of a protein plays an essential role in the functionality of the protein, and it is widely believed that the native conformation of a protein is determined by the amino acid sequence of the protein. As experimental determination of the native conformation is difficult and time consuming, much work has been done to predict the native conformation computationally.

To predict the structure of a protein computationally it is necessary to model protein structure formation in the real system, i.e. in the proteins natural environment. A model is relevant if it reflects some of the properties of

[*] Department of Computer Science, University of Aarhus, Ny Munkegade, 8000 Århus C, Denmark. E-mail: `rlyngsoe@daimi.au.dk`.

[†] Basic Research In Computer Science (BRICS), Center of the Danish National Research Foundation, Department of Computer Science, University of Aarhus, Ny Munkegade, 8000 Århus C, Denmark. E-mail: `cstorm@brics.dk`.

protein structure formation in the real system. One obvious property could be *visual equivalence* between the native conformations in the model and the native conformations in the real system. Another more subtle, but useful property, could be *behavioral equivalence* between protein structure formation in the model and protein structure formation in the real system. As the laws of thermodynamics state that the native state of a protein is the state of least free energy, the real system is often modeled by a *free energy model* that specifies an energy function that assigns a free energy to every conformation in a set of legal conformations. The native conformation of a protein is then predicted to be a conformation that minimizes the energy function over the set of legal conformations.

The hydrophobic-hydrophilic model proposed by Dill [4] is a free energy model that models the belief that a major contribution to the free energy of the native conformation of a protein is due to interactions between hydrophobic amino acids that tend to form a core in the spatial structure shielded from the surrounding solvent by hydrophilic amino acids. In the model the amino acid sequence of a protein is abstracted as a binary sequence of hydrophobic and hydrophilic amino acids. Even though some amino acids cannot be classified clearly as being either hydrophobic or hydrophilic, the model disregards this fact to achieve simplicity. The model is usually referred to as the HP model where H stands for hydrophobic and P stands for polar.

The HP model is a *lattice model*, so called because the set of legal conformations is embeddings of the abstracted amino acid sequence in a lattice, in this case the two or three dimensional square lattice. In legal conformations amino acids that are adjacent in the sequence occupy adjacent grid points in the lattice, and no grid point in the lattice is occupied by more than one amino acid. Depending on the dimension of the square lattice we refer to the model as the 2D or 3D HP model. The free energy of a conformation depends on the number of non-adjacent hydrophobic amino acids that occupy adjacent grid points in the lattice. Figure 1 shows a conformation in the 2D HP model where 9 non-adjacent hydrophobic amino acids occupy adjacent grid points.

Despite the simplicity of the HP model, the folding process in the model have behavioral similarities with the folding process in the real system [5], and the model has been used by chemists to evaluate new hypothesis of protein structure formation [9]. The success of the HP model as a tool for chemists partly stems from the fact that the discrete set of legal conformations makes it possible to enumerate and consider all conformations of small proteins. Many attempts have been made to predict the native conformation, i.e. the conformation of lowest free energy, of a protein in the HP model [10, 11]. Most interestingly, the HP model was the first relevant model for protein folding for which approximation algorithms for the structure prediction problem, i.e. algorithms that find a conformation with free energy guaranteed close to the free energy of the native conformation, were formulated [6]. For a while it was believed that the structure prediction problem in the HP model would be solvable in polynomial time, but recently it was shown NP-complete [2, 3].

In this paper we describe three attempts to improve the best known approximation algorithm for the structure prediction problem in the 2D HP model. We show that two generalizations of this algorithm, the U-fold algorithm and S-fold algorithm, do not improve on the best known 1/4 worst case approximation ratio. The approximation ratio of the third generalization, the C-fold algorithm, seems to be better. We prove that the worst case approximation ratio of the C-fold algorithm is at most 1/3 and observe that it is closely related to an interesting combinatorial problem which we examine experimentally. Most of the work described in this paper was done in the Spring 1996 as part of a graduate course [7]. Independently of our work Mauri et al. [8] observe experimentally that the approximation ratio of an algorithm similar to our C-fold algorithm seems to be around 3/8.

The rest of this paper is organized as follows. In Section 2 we formally describe the 2D HP model and bound the free energy of the native conformation of a protein in the model. In Section 3 we describe three attempts to improve the currently best approximation algorithm for the structure prediction problem in the 2D HP model. In Section 4 we describe and examine experimentally an interesting problem that is related to the approximation ratio of one of the approximation algorithms described in Section 3.

2 The 2D HP model

In the 2D HP model a protein, i.e. an amino acid sequence, is abstracted as a string describing the hydrophobicity of each amino acid in the sequence. Throughout this paper we will use S to denote the abstraction of an amino acid sequence of length n, that is, S is a string of length n over the alphabet $\{0, 1\}$ where $S[i]$, for $i = 1, 2, \ldots, n$, is 1 if the ith amino acid in the sequence is hydrophobic and 0 if it is hydrophilic. We will use the term "hydrophobic amino acid" to refer to a 1 at some position in S, and say that the parity of the 1 is even if its position in S is even, and odd if its position in S is odd.

A folding of a protein in the 2D HP model is an embedding of its abstraction S in the 2D square lattice such that adjacent characters in S occupy adjacent grid points in the lattice, and no grid point in the lattice is occupied by more than one character. We say that two 1's in S form a non-local 1-1 bond if they occupy adjacent grid points in the lattice but are not adjacent in S. Figure 1 shows a folding of the string 111010100101001001 in the 2D HP model with nine non-local 1-1 bonds. The free energy of a folding of S is the number of non-local 1-1 bonds in the folding multiplied by some constant $\epsilon < 0$. The free energy function models the belief that the driving force of

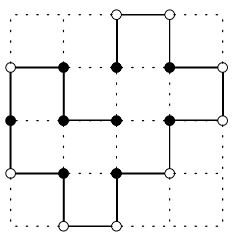


Figure 1: A conformation in the 2D HP model with 9 non-local 1-1 bonds.

protein structure formation is interactions between hydrophobic amino acids.

We say that the *score* of a folding of S is the number of non-local 1-1 bonds in it, and that the *optimal score* of a folding of S, $\mathrm{OPT}(S)$, is the maximum score of a folding of S. The simple energy function implies that the native conformation of a protein in 2D HP model is a folding of its abstraction with optimal score. The *structure prediction problem* in the 2D HP model is thus to find a folding of S in the 2D square lattice with optimal score. This problem has recently been shown to be NP-complete [2, 3], which makes it interesting to look for approximation algorithms that find a folding of S with score guaranteed to be some fraction of the optimal score of a folding of S. To issue such a guarantee for a folding algorithm, we need an upper bound on $\mathrm{OPT}(S)$. To derive an upper bound on $\mathrm{OPT}(S)$ we make two observations.

The first observation is that a hydrophobic amino acid can form at most two non-local 1-1 bonds in the 2D square lattice except if it is the first or the last amino acid in the sequence, in which case it can form at most three non-local 1-1 bonds. The second observation is that two hydrophobic amino acids, $S[i]$ and $S[j]$, can occupy adjacent grid points in the 2D square lattice, i.e. form a non-local 1-1 bond, if and only if i is even j is odd or vice versa. If we define $\mathrm{EVEN}(S)$ as the set of even positions in S containing a hydrophobic amino acid, i.e. $\{i \mid i \text{ is even and } S[i] = 1\}$, and $\mathrm{ODD}(S)$ as the set of odd positions in S containing a hydrophobic amino acid, i.e. $\{i \mid i \text{ is odd and } S[i] = 1\}$, then the two observations gives

$$\mathrm{OPT}(S) \leq 2 \cdot \min\{|\mathrm{EVEN}(S)|, |\mathrm{ODD}(S)|\} + 2. \tag{1}$$

This upper bound was first derived by Hart and Istrail [6], who used it in the performance analysis of a simple folding algorithm that guarantees a folding with score 1/4 of the optimal score. This algorithm and various attempts to improve it is the topic of the next section.

3 The folding algorithms

A simple strategy for folding a string in the 2D square lattice is to find a suitable folding point that divides the string into two parts, a prefix and a suffix, that we fold against each other. This creates a "U" structure in which non-local 1-1 bonds can be formed between 1's on opposite stems of the "U". Loops protruding from the two stems of the "U" can be used to increase the number of non-local 1-1 bonds between the stems by contracting parts of the stems. We say that a folding created this way is a U-fold. Figure 2 shows a schematic U-fold and the left part of Figure 3 shows a U-fold of the string 10010010100101010000011 with four non-local 1-1 bonds between the stems and five non-local 1-1 bonds in total.

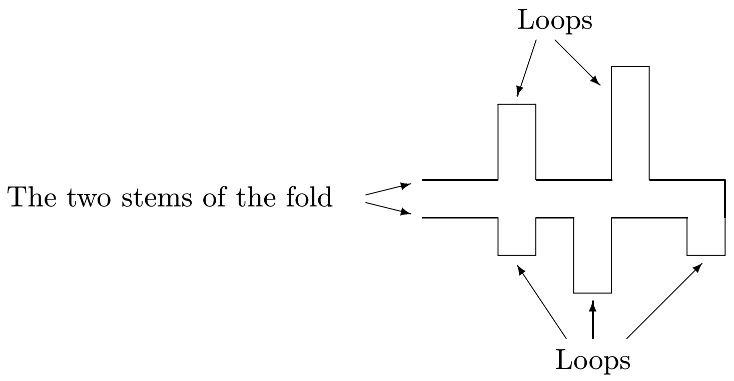
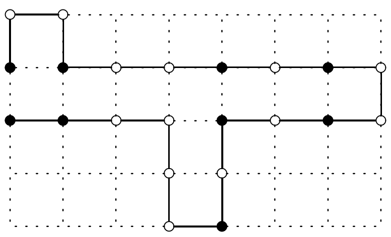


Figure 2: A schematic U-fold.

Hart and Istrail [6] present a folding algorithm that computes a U-fold of S with a guaranteed number of non-local 1-1 bonds between the stems. By a simple argument they show that the folding point can always be chosen such that at least half of the 1's with position in EVEN(S) are on one stem and at least half of the 1's with position in ODD(S) are on the other stem. Since there is an odd number of characters between any two characters in S with positions in either EVEN(S) or ODD(S), loops can be used to contract each stem such that every second character on the contracted stem is a 1 with even or odd parity depending on the stem. As each contracted stem contains at least $\min\{|\text{EVEN}(S)|, |\text{ODD}(S)|\}/2$ 1's with equal parity placed in every second position along stem, the number of non-local 1-1 bonds between the stems of the created U-fold is at least $\min\{|\text{EVEN}(S)|, |\text{ODD}(S)|\}/2$, so except for a constant term the created U-fold scores at least 1/4 of the upper bound on OPT(S) given by (1). We say that the asymptotic approximation ratio of the algorithm is 1/4. By being a little bit more careful in the choice of folding point Hart and Istrail are able to formulate the folding algorithm such that

1 0 0 1 0 0 − − − − 1 0 1 0
1 − − 1 0 0 0 0 1 0 1 0 1 0



1 − − 1 0 0 − − 1
1 0 0 1 − − 0 0 1

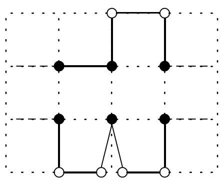
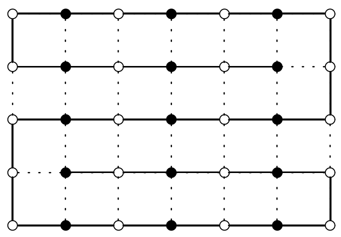


Figure 3: Left: Alignment of the prefix 1001001010 of the string 100100101001010101000011 with the rest of the string and the corresponding U-fold. Right: An example of an alignment with illegal gaps. The transformation to a folding implies that two loops protrude from the same element.

the create a U-fold, for every string S, scores at least 1/4 of the upper bound on OPT(S). We say that the absolute approximation ratio of the algorithm is 1/4. The folding algorithm runs in time $O(n)$ where n is the length of S.

Our first attempt to improve the approximation ratio of the folding algorithm by Hart and Istrail, is to count all non-local 1-1 bonds between the two stems of the U-fold, and not only those where the 1's on each stem have equal parity. More precisely, we want to compute a U-fold of S with the maximum number of non-local 1-1 bonds between the stems, i.e. a U-fold of S with optimal score between the stems. Computing such a U-fold is not difficult. As illustrated in Figure 3, the trick is to observe that a U-fold of S, with folding point k, that maximizes the number of non-local 1-1 bonds between the stems, corresponds to the an alignment of the prefix $S[1 \mathinner{..} k-1]$ with the reversed suffix $S[k+2 \mathinner{..} n]^R$ that maximizes the number of matches between 1's, and allows gaps to be folded as loops.

Such an alignment corresponds to an optimal similarity alignment between $S[1 \mathinner{..} k-1]$ and $S[k+2 \mathinner{..} n]^R$, where a match between two 1's score 1, and all other matches and gaps score 0. To allow gaps to be folded out as loops, all gaps must have even length and between any two gaps in the same string there must be at least two matched characters. These additional rules on gaps can be enforced without increasing the running time of the alignment algorithm, so a U-fold of S with folding point k and optimal score between the stems can be computed in the time required to compute an optimal similarity alignment, i.e. in time $O(n^2)$ where n is the length of S. By considering every folding point this immediately gives an algorithm, the U-fold algorithm, that computes a U-fold with optimal score between the stems in time $O(n^3)$. By observing that the best folding point k corresponds to an entry $(k-1, n-k-1)$ with



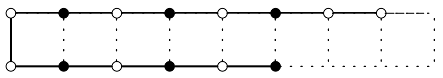


Figure 4: A string of the form $(10)^i0(10)^i00(10)^i00(10)^i(01)^i$. For these strings the U-fold with optimal score between the stems is only 1/4 of the score of the optimal folding.

maximum value in the alignment matrix resulting from an alignment of S and S^R with the above parameters, i.e. matches between 1's score 1, everything else score 0, and gaps have to be expressible as loops, we can reduce the running time of the U-fold algorithm to $O(n^2)$.

As the foldings considered by the folding algorithm by Hart and Istrail are a subset of the foldings considered by our U-fold algorithm, the approximation ratio of the U-fold algorithm is at least 1/4. Unfortunately it is no better in the worst case. As illustrated in Figure 4, this follows because any string of the form $(10)^i0(10)^i00(10)^i(01)^i$, $i > 0$, when folded as a U-fold with optimal score between the stems only scores 1/4 of the score of an optimal folding. The 1/4 approximation ratio of our U-fold algorithm and the folding algorithm by Hart and Istrail is thus tight. An obvious way to try to improve the approximation ratio of the U-fold algorithm would be to also count and maximize the number of non-local 1-1 bonds occurring between 1's on the loops. Unfortunately, as above, a set of strings can be constructed such that when folded this way they only score 1/4 of the score of an optimal fold.

Another way to try to improve the approximation ratio of the U-fold algorithm is to consider a larger set of foldings than U-folds. Figure 5 illustrates two ways to do this. The first way is to allow multiple bends of the string and loops on the outer stems. This gives rise to what we call S-folds. The second way is to allow two bends of the string that fold the two ends of the string towards each other and loops on the two stems. This gives rise to what we call C-folds. Both the S-fold and the C-fold with optimal score between the stems can be computed in time $O(n^3)$ using dynamic programming. For the C-fold it is easy to see how. A C-fold of S is a U-fold of a prefix, $S[1\,..\,k]$, and a U-fold of a suffix, $S[k+1\,..\,n]$, glued together to form a C-fold. As there are less than n ways to divide the string, the best C-fold can be found by computing and gluing together $2n$ U-folds. As each of these U-folds can be computed in time $O(n^2)$, the best C-fold can be computed in time $O(n^3)$. The

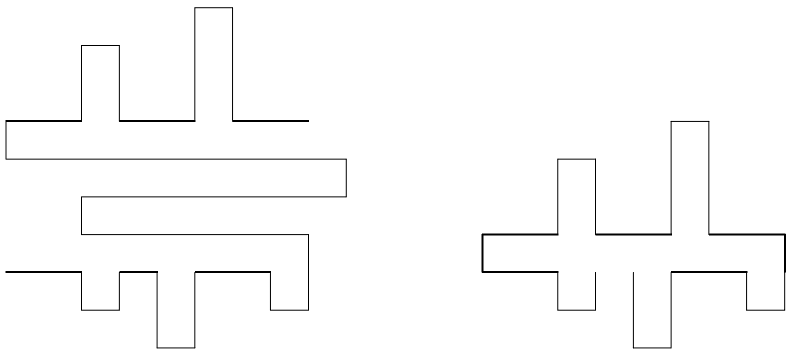

(a) S-fold (b) C-fold

Figure 5: Two ways to generalize the U-fold.

computation of the best S-fold in time $O(n^3)$ is somewhat more technical. We choose to omit the details of the S-fold algorithm as it, as explained below, unfortunately turns out that its approximation ratio is no better than 1/4.

As S- and C-folds are supersets of U-folds, the approximation ratio of both the S- and C-fold algorithm is at least 1/4. Unfortunately this approximation ratio is tight for the S-fold algorithm because any string of the form $(10)^i(0^{2i+1}1)^{4i}(10)^i$, $i > 0$, when folded as a S-fold with optimal score between the stems only scores 1/4 of the score of an optimal folding. Similar to U-folds, we can show that counting and maximizing the number of non-local 1-1 bonds occurring between 1's on the loops of the S-fold does not improve the worst case approximation ratio of the folding algorithm. In contrast to U- and S-folds, we have not been able to find a set of strings that show that the 1/4 approximation ratio of the C-fold algorithm is tight. In fact experiments indicates, as explained in the next section, that the approximation ratio of the C-fold algorithm is somewhat better than 1/4. This is also observed in [8].

In our analysis of the approximation ratio of the C-fold algorithm we came up with a relation to an interesting matching problem. This is the topic of the next section. We end this section by summarizing the presented results.

Theorem 1 *The score of the best U- and S-fold of string S is at least, and at most in the worst case,* 1/4 *of the score of an optimal fold of S. The score of the best C-fold of string S is at least* 1/4 *of the score of the optimal fold of S.*

4 The circle problem

Let $P \in \{+,-\}^*$ be a string that contains equally many $+$'s and $-$'s. We say that P is a balanced string of length $n = |P|$. Consider P wrapped around the

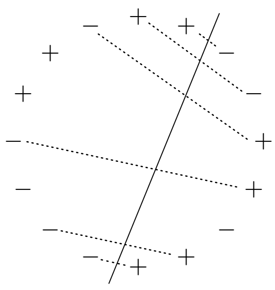


Figure 6: An example of a matching in a balanced string

perimeter of a circle. A matching in P is obtained by dividing the circle by a line and connecting $+$'s with $-$'s using non-crossing lines that all intersect the dividing line. The size of the matching is the number of non-crossing lines connecting $+$'s with $-$'s that intersect the dividing line. Figure 6 shows an example of a matching of size 6. A maximum matching in P is a matching in P of maximum size. We use $M(P)$ to denote the size of a maximum matching in P and we use $M(n)$ to denote the minimum of $M(P)$ over all balanced strings P of length n, that is

$$M(n) = \min_{P:|P|=n} M(P).$$

The matching problem in balanced strings, or the circle problem as we call it, is closely related to the approximation ratio of our C-fold algorithm. To see the relation, we introduce the parity labelling of a string.

The *parity labelling* of a string $S \in \{0,1\}^*$ is a string $P_S \in \{+,-\}^*$ in which the ith character indicates the parity of the ith 1 in S, e.g. the parity labelling of 100101110101 is $-++-+++$. A *balanced* parity labelling of S is a maximum length subsequence of P_S that contains equally many $+$'s and $-$'s. From the definition of EVEN(S) and ODD(S) follows that P_S contains $|\text{EVEN}(S)|$ $+$'s and $|\text{ODD}(S)|$ $-$'s, so a balanced parity labelling of S is obtained by removing $\big||\text{EVEN}(S)| - |\text{ODD}(S)|\big|$ $+$'s or $-$'s from P_S. The length of a balance parity labelling of S is $2 \cdot \min\{|\text{EVEN}(S)|, |\text{ODD}(S)|\}$, but the labelling is not unique as there can be several ways to choose the $+$'s or $-$'s to remove from P_S, e.g. the parity labelling $-++-+++$ gives $-++-$, $-+-+$ and $--++$ as possible balanced parity labellings.

Upper bounding the C-fold approximation ratio

To get the relation to C-folds, we observe that a C-fold of S with k non-local 1-1 bonds between the stems corresponds to a matching of size k in a balanced parity labelling of S. This implies that an upper bound on $M(n)$ is

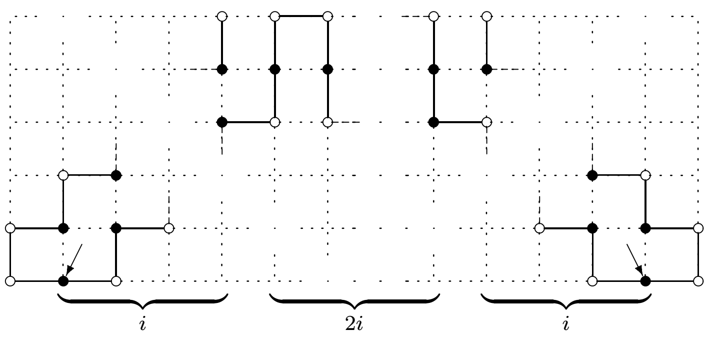


Figure 7: A folding of a string of the form $(01)^i000(01)^i(010)^{2i}(10)^i000(10)^i$. Only the two 1's indicated with arrows have less than the optimal two non-local bonds. The total number of non-local bonds in this folding is $2 \cdot \min\{|\text{EVEN}(S)|, |\text{ODD}(S)|\} - 1 = 4i - 1$ and thus, by the balanced parity labelling argument, the optimal score between the stems in a C-fold of this string is approximately 1/3 of the score of the optimal folding.

also an upper bound on the score between the stems of C-folds of strings with balanced parity labellings of length n. In other words, if $M(n) \leq \alpha n$ then the score between the stems of a C-fold of S is upper bounded by α multiplied by the length of a balanced parity labelling of S, i.e. upper bounded by $\alpha \cdot 2 \cdot \min\{|\text{EVEN}(S)|, |\text{ODD}(S)|\}$. Since the length of a balanced parity labelling of S is equal to the upper bound on $\text{OPT}(S)$ given by (1), $M(n) \leq \alpha n$ implies that the approximation ratio of the C-fold algorithm, with respect to the upper bound on $\text{OPT}(S)$ given by (1), is at most α.

It is easy to prove that $M(+^i \, -^i \, (+-)^i \, -^i \, +^i) = 2i + 1$ for any $i > 0$. Hence, $M(n) \leq n/3 + 1$, so the asymptotic approximation ratio of our C-fold algorithm is at most 1/3 if analyzed with respect to the upper bound on $\text{OPT}(S)$ given by (1). Fortunately, as illustrated in Figure 7, for any $i > 0$ there exists a string $(01)^i000(01)^i(010)^{2i}(10)^i000(10)^i$ with balanced parity labelling $+^i \, -^i \, (+-)^i \, -^i \, +^i$ for which $\text{OPT}(S)$ deviates from the upper bound of (1) by at most a constant term. This example shows that the asymptotic approximation ratio of the C-fold algorithm is at most 1/3.

Lower bounding the C-fold approximation ratio

To use a matching in a balanced parity labelling of S to improve on the 1/4 approximation ratio of the C-fold algorithm, two requirements must be met. First, we need to be able to transform a matching in a balanced parity labelling of S into a C-fold of S with a number of non-local bonds proportional by some factor β to the size of the matching. Secondly, we need to lower bound the

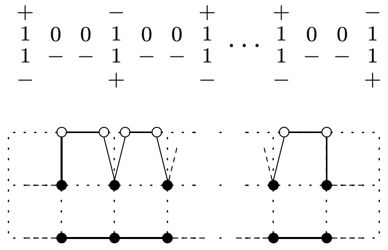


Figure 8: An example where the obvious transformation from a matching of a balanced parity labelling of a string to a C-fold, trying to place two 1's with connected labels opposite each other on the stems of a C-fold, fails.

asymptotic ratio of $M(n)/n$ by some constant $\gamma > 1/(4 \cdot \beta)$. This would yield an asymptotic approximation ratio of the C-fold algorithm of $\beta \cdot \gamma > 1/4$. We have not yet solved these problems but will in the following report on some promising approaches and experiments.

The task of transforming a matching in a balanced parity labelling of S to a C-fold of S is not as straightforward as transforming the non-local bonds between the stems of a C-fold of S to a matching in one of the balanced parity labellings of S. Though one can identify the labels with 1's it will not always be the case that there is a legal C-fold of S where the non-local bonds between the stems corresponds to the connections between the corresponding labels in a matching in a balanced parity labelling of S.

To observe this, consider the two strings $S' = 1^{2i}$ and $S'' = (100)^{2i-1}1$, both with balanced parity labellings $P_{S'} = P_{S''} = (-+)^i$. Assume that S contains S' and S'' as substrings and that the labels of these two substrings have been connected with each other in the matching in a balanced parity labelling of S. As illustrated in Figure 8, we get the same problem as in the right-hand example in Figure 3 with two loops protruding from the same element if we try to make the obvious transformation of this matching to a C-fold of S. We observe that the obvious transformation only fails when we have stretches of consecutive 1's in one of the stems. One approach to solve the problem of transforming a matching in the balanced parity matching of S to a C-fold of S would thus be to 'eliminate' or at least 'shorten' consecutive stretches of 1's by removing 1's while ensuring compensatory non-local bonds.

This can be done in much the same way as when contracting the stems of a C-fold by folding out loops. As illustrated in Figure 9 we can fold out a stretch of an even number of consecutive 1's in a hydrophobic loop such that only two 1's remains along the stem. In such a loop where $2i$ 1's have been removed, i of which are at positions in EVEN(S) and i of which are at positions in ODD(S), there will be i non-local bonds. As long as $\beta \cdot \gamma \leq 1/2$

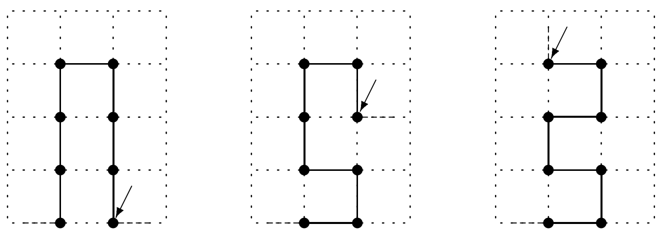


Figure 9: Possible hydrophobic loops of eight consecutive 1's. The positions of the embedding of the last 1 in the stretch is indicated with an arrow.

we can thus ensure compensatory non-local bonds. This allows us to remove the 1's that can be folded out in hydrophobic loops from S before finding a matching of a balanced parity labelling of the modified sequence.

Two problems still remain, though. First, the hydrophobic loops make the sequence less flexible since we cannot contract the stems immediately after a hydrophobic loop simply by folding out another loop. As indicated in Figure 9 we can however choose the position of the embedding of the last 1 in the stretch of 1's folded out rather freely which allows almost any contracting by an even number of amino acids immediately after a hydrophobic loop. Only when the loop removes $2i$ 1's with i odd there is a problem with contracting the stem by $i+1$ amino acids as we have to round the corner of the loop from the position furthest away from the stem where we can embed the last 1. Secondly, we have not eliminated stretches of consecutive 1's but merely limited them to being of length at most three. Though we find this approach promising we have not yet been able to carry through with the rigorous case-by-case analysis, an analysis that will require additional tricks besides the hydrophobic loops to handle special cases, of the various situations that can arise when trying to transform a matching of a balanced parity labelling of S to a C-fold of S.

To lower bound the asymptotic ratio of $M(n)/n$ it is easy to observe that $M(n) \geq n/4$. Unless we, unrealistically, hope to transform a matching in the balanced parity labelling of S into a C-fold of S with *more* non-local bonds than connections in the matching this lower bound does not say anything we do not already known, namely that the approximation of the C-fold algorithm is at least 1/4. Narrowing the gap between the trivial lower bound $M(n) \geq n/4$ and the upper bound $M(n) \leq n/3 + 1$ presented above has turned out to be a very difficult problem.

To get an impression of whether or not the trivial lower bound is tight, we did two experiments. First, we computed the value of $M(n)$ for all $n \leq 34$. As illustrated in Figure 10, this showed that $M(n) \geq n/3$ for all $n \leq 34$. Secondly, we computed $M(n)$ for a large number of randomly selected larger balanced strings. This random search did not produce a string in which the size of the

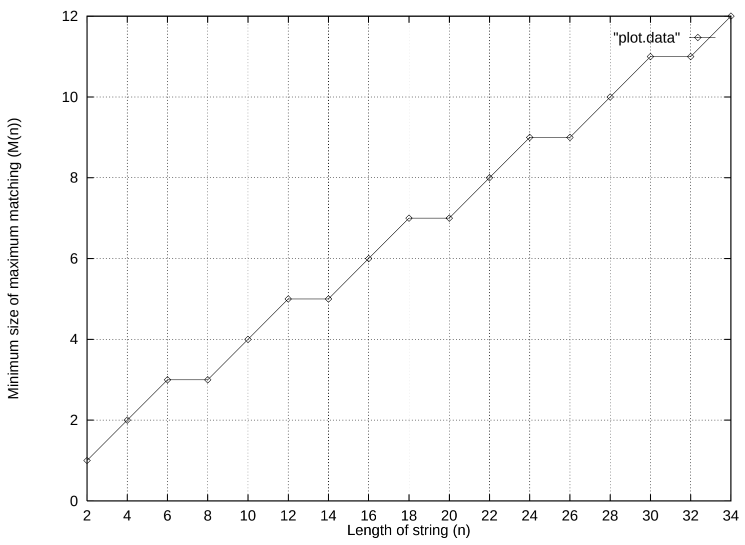


Figure 10: The minimum size of the maximum matching in balanced strings with length up to 34.

maximum matching was less than $n/3$. Combined these two experiments lead us to believe that $M(n) \geq n/3$.

To help prove a non-trivial lower bound, one might consider the restricted matching problem where the dividing line must be chosen such that it divides the circle into two halfs. This restriction does not seem to affect the lower bound, as rerunning the experiment presented in Figure 10 gives the same results. It might also be helpful to consider other formulations of the problem. We observe that a dividing line in the circular representation of P corresponds to a partition XYZ of P, where the one side of the divided circle is Y and the other side is ZX. The maximum size of a matching in P given a partition XYZ is the length of the longest common subsequence of Y and $\overline{ZX}^R$, so

$$M(P) = \max_{XYZ\,:\,P=XYZ} |LCS(Y, \overline{ZX}^R)|.$$

In this terminology the above restriction of the problem, i.e. that the circle should be divided into two halfs, corresponds to only maximizing over partitions XYZ of P where $|Y| = |ZX|$. Another formulation of the problem follows from the observation that part of $LCS(Y, \overline{ZX}^R)$ is a subsequence of a prefix Y and $\overline{X}^R$ and the rest is a subsequence of the rest of Y and $\overline{Z}^R$. We can thus split Y according to this and reformulate the calculation of $M(P)$ as

$$M(P) = \max_{X_1, X_2} \{|X_1| + |X_2| \mid X_1 \overline{X_1}^R X_2 \overline{X_2}^R \text{ is a subsequence of P}\}.$$

This lends an immediate generalization of the problem as we can define

$$M_k(P) = \max_{X_1,\dots,X_k} \{\sum_{i=1}^{k} |X_i| \mid X_1\overline{X_1}^R \dots X_k\overline{X_k}^R \text{ is a subsequence of P}\},$$

where $M(P) = M_2(P)$ and $M_1(P)$ is the corresponding problem for the U-fold (equivalent to fixing one end-point of the dividing line in the circle formulation of the problem). One can observe that $M_k(n) < n/2$ for any k because the string $P = +++--+--$ gives that $M_k(P) = M_1(P) = 3$, but apart from this we have not been able to come up with any non-trivial bounds for $M_k(n)$.

5 Conclusion

We have presented three generalizations of the best known approximation algorithm for structure prediction in the 2D HP model. We have shown that two of these generalization do not improve the worst case approximation ratio, while the third generalization might be better. The future work is clear. First, prove that a matching in a balanced string can be transformed to a C-fold with score equal to the size of the matching. Secondly, prove or disprove that $M(n) \geq \alpha n$ for some $\alpha > 1/4$. Combined this would give whether or not our C-fold algorithm improves the best known 1/4 approximation ratio for structure prediction in the 2D HP model. We conjecture that the approximation ratio of our C-fold algorithm where non-local bonds in the loops are considered is 1/3.

References

[1] C. B. Anfinsen, E. Haber, and F. H. White. The kinetics of the formation of native ribonuclease during oxidation of the reduced polypetide domain. *Proceedings of the National Academy of Science, USA*, 47:1309–1314, 1961.

[2] B. Berger and T. Leighton. Protein folding in the hydrophobic-hydrophilic (HP) model is NP-complete. *Journal of Computational Biology*, 5(1):27–40, 1998.

[3] P. Crescenzi, D. Goldman, C. Papadimitriou, A. Piccolboni, and M. Yannakakis. On the complexity of protein folding. *Journal of Computational Biology*, 5(3):423–465, 1998.

[4] K. A. Dill. Theory for the folding and stability of globular proteins. *Biochemistry*, 24:1501, 1985.

[5] K. A. Dill, S. Bromberg, K. Yue, K. M. Fiebig, D. P. Yee, P. D. Thomas, and H. S. Chan. Principles of protein folding – a perspective from simple exact models. *Protein Science*, 4:561–602, 1995.

[6] W. E. Hart and S. Istrail. Fast protein folding in the hydrophobic-hydrophilic model within three-eights of optimal. *Journal of Computational Biology*, Spring 1996, 1996.

[7] R. B. Lyngsø and C. N. S. Pedersen. Prediction of protein structures using simple exact models. Project in a Graduate Course. Available from http://www.daimi.au.dk/∼cstorm/papers, June 1996.

[8] G. Mauri, G. Pavesi, and A. Piccolboni. Approximation algorithms for protein folding prediction. In *Proceedings of the 10th Annual Symposium on Discrete Algorithms (SODA)*, pages 945–946, 1999.

[9] A. Sali, E. Shahknovich, and M. Karplus. How does a protein fold? *Nature*, 369:248–251, 1994.

[10] R. Unger and J. Moult. Genetic algorithms for protein folding simulations. *Journal of Molecular Biology*, 231:75–81, 1993.

[11] K. Yue and K. A. Dill. Forces of tertiary structural organization in globular proteins. In *Proceedings of the National Academy of Science, USA*, volume 92, pages 146–150, 1994.