ORNL/CP-100039

Solving Globally-Optimal Threading Problems in "Polynomial-Time*

Ying Xu, Ying Xu, and Edward C. Uberbacher
Life Sciences Division
Oak Ridge National Laboratory
Oak Ridge, TN 37831-6480

To be published and presented with *The Third Internal Annual Conference on Computational Molecular Biology*, Lyon, France, April 12-17, 1999.

*Research was supported by the Office of Health and Environmental Research, U.S. Department of Energy under contract No. DE-AC05-96OR22464 with Lockheed Martin Energy Research Corporation.

Solving Globally-Optimal Threading Problems in "Polynomial-Time"
(Extended Abstract)

Ying Xu, Dong Xu, and Edward C. Uberbacher
Computational Biosciences Section, Life Sciences Division
Oak Ridge National Laboratory
Oak Ridge, TN 37831-6480, USA
Email: {xyn, xud, ube}@ornl.gov

Abstract

Computational protein threading is a powerful technique for recognizing native-like folds of a protein sequence from a protein fold database. In this paper, we present an improved algorithm (over our previous work) for solving the globally-optimal threading problem, and illustrate how the computational complexity and the fold recognition accuracy of the algorithm change as the cutoff distance for pairwise interactions changes. For a given fold of m residues and M core secondary structures (or simply *cores*) and a protein sequence of n residues, the algorithm guarantees to find a sequence-fold alignment (threading) that is globally optimal, measured collectively by (1) the singleton match fitness, (2) pairwise interaction preference, and (3) alignment gap penalties, in $O(mn + MnN^{1.5C-1})$ time and $O(mn + nN^{C-1})$ space. C, the *topological complexity* of a fold as we term, is a value which characterizes the overall "structure" of the considered pairwise interactions in the fold, which are typically determined by a specified cutoff distance between the beta carbon atoms of a pair of amino acids in the fold. C is typically a small positive integer. N represents the maximum number of possible alignments between an individual core of the fold and the protein sequence when its neighboring cores are already aligned, and its value is significantly less than n. When interacting amino acids are required to "see each other", C is bounded from above by a small integer no matter how large the cutoff distance is. This indicates that the protein threading problem is polynomial-time solvable if the condition of "seeing each other" between interacting amino acids is sufficient for accurate fold recognition. A number of extensions have been made to our basic threading algorithm to allow finding a globally-optimal threading under various constraints, which include consistencies with (1) specified secondary structures (both cores and loops), (2) disulfide bonds, (3) active sites, etc.

1 Introduction

Computational protein threading, as an effective tool for recognition of native-like folds, concerns finding an alignment between a protein fold template and a query protein sequence, which optimizes a specified scoring function. Typically, this function is a weighted sum of the following measures: (1) the fitness of aligning a particular amino acid to a particular environment in the fold template; (2) the interaction preference between a pair of amino acids assigned to the template positions that are spatially "close"; and (3) gap penalties for the unaligned amino acids and template positions. A number of computer programs for protein threading, using this type of scoring function or variations, have been developed [1, 2, 6, 7, 8, 10, 12, 13]. While some success has been achieved by these and similar computer programs, the protein threading problem is far from being solved

[3, 4]. Among many of the unsolved issues in the protein threading problem, lack of effective computational methods that guarantee to find an optimal[a] threading represents one of the key problems which need to be further addressed.

The computational protein threading problem, using the above or a similar scoring scheme, is widely considered to be computationally intractable. While this belief is supported by the NP-hardness proof of protein threading [9], it also makes people shy away from seeking a more direct solution to the threading problem. As the NP-hardness proof is done based on the assumption that interactions between every pair of amino acids in a protein structure need to be considered and included in the scoring scheme, this result may not apply to the protein threading problems that many of the existing computer programs attempt to solve. This is because most of the these programs use a cutoff distance in defining the pairwise interactions, which typically ranges from 7Å to 15Å. As we know, when the cutoff distance changes from the size of a protein fold's diameter to zero, the computational complexity of the threading problem changes from NP-hard to polynomial-time solvable. Some recent research suggests that considering pairwise interactions among spatially "close" residues, e.g., residues that can "see each other", is probably sufficient for accurate fold recognition [11]. Understanding how the cutoff distance of pairwise interactions affect the computational complexity and the fold recognition accuracy of the protein threading problem is one of the key goals of our research.

We have developed an algorithm which guarantees to find an optimal threading between a query protein sequence and a fold template. In formulating the threading problem, we follow a few widely-used assumptions. We assume that (i) each protein fold is represented by a series of cores (α-helices and β-strands) with the connecting loops being removed; and (ii) alignment gaps are confined to the regions between cores. Different than the other threading algorithms, the computational complexity, $O(mn + MnN^{1.5C-1})$, of our algorithm depends not only on the length n of the query sequence, the length m and the number of cores of the fold template, but more importantly also depends on the topological complexity C of the fold template, where N is $<< n$. The topological complexity, which will be defined in Section 2, characterizes the overall structure of the considered pairwise interactions of the fold template, and it is a monotonic function of the cutoff distance of pairwise interactions. It is bounded from above by a small constant when the concept of "visible volume" [11] is used in defining the pairwise interactions, implying that the protein threading problem is polynomial-time solvable if the "seeing-each-other" condition on interacting pairs is sufficient for accurate fold recognition.

We have implemented this globally-optimal threading algorithm with a number of extensions as a computer system, called **PRO**tein **S**tructure **P**rediction and **E**valuation **C**omputer **T**oolkit (**PROSPECT**). These extensions are added to facilitate easy incorporation of known biological information, like (1) known (or predicted) loop regions and other types of secondary structure information, (2) known disulfide bonds, (3) active sites, about the query protein sequence into our threading program.

[a]Throughout this paper, an optimal threading means a globally optimal threading.

2 An improved algorithm for globally optimal threading

2.1 Problem formulation

Let $s = s_1 s_2 ... s_n$ be a protein sequence, and (t, T) be a protein fold template with loops being removed[b], where $t = t_1 t_2 ... t_m$ is a sequence of template positions (with an array of physical properties attached to each of them), and $T = T_1, ..., T_M$ is the sequence of core secondary structures t is partitioned into. Let pairs(t, T) be the set of all pairs of positions in the template (t, T), considered to have pairwise interactions. Further let loop(T_i, T_{i+1}) represent the loop length[c] between the cores T_i and T_{i+1}. We use $f(x, y)$ to represent the fitness of aligning the amino acid x to the template position y, $p(x_1, x_2)$ to denote the interaction preference between of a pair $(x_1, x_2) \in$ pairs(t, T), and $L(r_1, r_2)$ to represent a penalty for the length difference between two ("aligned") loops of lengths r_1 and r_2. A *protein threading problem* can be defined as to find a partition[d] $\{S_1, ..., S_{2M+1}\}$ of s such that $\|S_{2i}\| = \|T_i\|$ for all $1 \leq i \leq M$, and the following function is minimized:

$$\sum_{i \in [1,M]} \sum_{j \in [1, \|T_i\|]} f(S_{2i}[j], T_i[j]) + \sum_{(T_i[j], T_{i'}[j']) \in \text{pairs}(t,T)} p(S_{2i}[j], S_{2i'}[j']) + \sum_{i \in [1, M+1]} L(\|S_{2i-1}\|, \text{loop}(T_{i-1}, T_i)) \tag{1}$$

where $X[k]$ represents the k^{th} element of X, and $\| \cdot \|$ represents the cardinality of a set.

Because of our assumption that alignment gaps are confined to regions between cores, we can use cores as the basic alignment units and give the following more compact definition of the protein threading problem, which we will use throughout the rest of the paper. Let

$$\text{PAIRS}(t, T) = \{(T_i, T_j) | \text{there exists } (x, y) \in \text{pairs}(t, T) \text{ and } x \in T_i, y \in T_j\}.$$

The protein threading problem can be restated so as to minimize the following function:

$$\sum_{i \in [1,M]} F(S_{2i}, T_i) + \sum_{(T_i, T_{i'}) \in \text{PAIRS}(t,T)} P(S_{2i}, S_{2i'}) + \sum_{i \in [1, M+1]} L(\|S_{2i-1}\|, \text{loop}(T_{i-1}, T_i)) \tag{2}$$

where

$$P(S_{2i}, S_{2i'}) = \sum_{(T_i[j], T_{i'}[j']) \in \text{pairs}(t,T)} p(S_{2i}[j], S_{2i'}[j']),$$

and

$$F(S_{2i}, T_i) = \sum_{j \in [1, \|T_i\|]} f(S_{2i}[j], T_i[j]).$$

Note that each partition $\{S_1, ..., S_{2M+1}\}$ of s defined above gives an alignment between $T_1 ... T_M$ and s. We call the starting position of S_{2i} in s an *aligned position* of core T_i.

[b]Only loop length conservation but not the actual loop alignment is modeled in our formulation of the protein threading problem.

[c]loop(T_0, T_1) and loop(T_N, T_{N+1}) represent the lengths of loops before the first core T_1 and after the last core T_N, respectively.

[d]Each S_i is a substring of s, and some of the S_i's could be $\emptyset$.

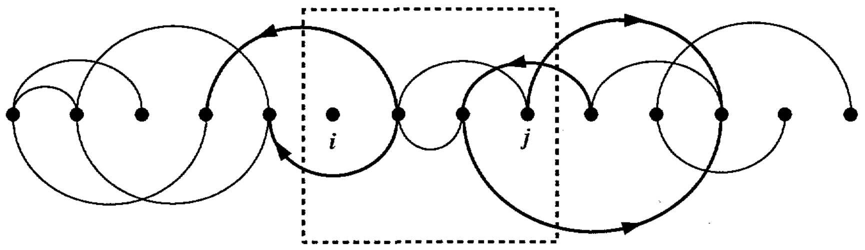

Figure 1: A graph representation of a template. Each node represents a core. An edge represents pairwise interactions between residues of the corresponding cores. The portion inside the dotted box represents a partial template, where each open link has an assigned orientation.

2.2 A divide-and-conquer algorithm

The problem described in Section 2.1 is a variation of the one described in [15], and we outline an improved algorithm for solving the problem. We first introduce a few terminology. We call each $(T_i, T_j) \in \mathrm{PAIRS}(t, T)$ a *link* between cores T_i and T_j. Each link (T_i, T_j) has an orientation, i.e., one end is the *source* and the other end a *sink*. The orientation of an link is determined by our threading algorithm to optimize the computational efficiency of the algorithm. For any sequence x, we use $x_{i,j}$ to denote the subsequence of x from positions i to j. Each partial template $T_{i,j}$ determines a set of *open links* $\{(T_v, T_w)\}$, with $v \in [i, j]$ and $w \notin [i, j]$. An open link $T_{i,j}$, for a given set of link orientation assignments, is called an *in-link* of $T_{i,j}$ if its source $\notin T_{i,j}$; otherwise it is an *out-link* of $T_{i,j}$. The *open link complexity* of $T_{i,j}$ under a particular set of link-orientation assignments is defined to be the number of different sources that its open links have. An *aligned position* of a link (T_v, T_w) is the aligned position of its source.

Let $score(T_{i,j}, s_{v,w}, D_{i,j}, A_{i,j})$ represent the score of a lowest-scoring alignment between the partial template $T_{i,j}$ and the partial sequence $s_{v,w}$ under the constraints that $T_{i,j}$'s open links have orientation assignments $D_{i,j}$ and $T_{i,j}$'s open links' aligned positions are $A_{i,j}$. The following statement can be proved by an inductive argument, which we omit in this abstract. For any $k \in [i, j-1]$, there exists a $u \in [v, w-1]$ such that

$$score(T_{i,j}, s_{v,w}, D_{i,j}, A_{i,j}) = \min_{D^0_{i,k}, A^0_{i,k}} \left\{ \begin{array}{l} score(T_{i,k}, s_{v,u}, D^{1,k}_{i,j} \cup D^0_{i,k}, A^{1,k}_{i,j} \cup A^0_{i,k}) + \\ score(T_{k+1,j}, s_{u+1,w}, D^{2,k}_{i,j} \cup D^0_{i,k}, A^{2,k}_{i,j} \cup A^0_{i,k}) + \\ L(\mathrm{posn}(T_{k+1}) - \mathrm{posn}(T_k) - \|T_k\|, \mathrm{loop}(T_k, T_{k+1})) \end{array} \right\} \tag{3}$$

and for the boundary condition,

$$score(T_{i,i}, s_{v,w}, D_{i,i}, A_{i,i}) = \min_{u \in [v, w - \|T_{i,i}\| + 1]} \left\{ \begin{array}{l} F(\mathrm{aligned}(T_i, u), T_i) + \sum_{\text{in-link } (T_i,T_j) \in \mathrm{PAIRS}(t,T)} \\ P(\mathrm{aligned}(T_i, u), \mathrm{aligned}(T_j, \mathrm{posn}(T_j))) \end{array} \right\}, \tag{4}$$

where $D^0_{i,k}$ (and similarly $A^0_{i,k}$) denotes the orientation (aligned position) assignments to the open links of $T_{i,k}$ that connect to $T_{k+1,j}$; $A^{1,k}_{i,j}$ (similarly $A^{2,k}_{i,j}$) represents the subset of $A_{i,j}$ that is associated to $T_{i,k}$ ($T_{k+1,j}$); and $D^{1,k}_{i,j}$ and $D^{2,k}_{i,j}$ are similarly defined; $\mathrm{posn}(T_i)$ denotes the current aligned position of T_i and aligned(T_i, x) denotes the aligned portion of s with T_i when the aligned position of T_i is x; and $\overline{X}$ represents the inversed "in"/"out" sequence of X.

Based on (3) and (4), we know that $score(T_{1,M}, s_{1,n}, \emptyset, \emptyset)$ gives the score of an alignment that minimizes function (2) (and equivalently function (1)). The following pseudo-code gives a direct implementation of recurrence (3) and calculates the score of an optimal alignment between T and s.

```
THREADING ($T_{i,j}, s_{v,w}, D_{i,j}, A_{i,j}, score$)
1. if  $i < j$
2.     $score \leftarrow +\infty$;
3.     pick a $k \in [i, j-1]$ and a set of orientation assignments $D^0_{i,k}$;
4.     for  each $u \in [v, w-1]$ and each possible set of aligned positions $A^0_{i,k}$ do
5.       $score_0 \leftarrow L(\text{posn}(T_{k+1}) - \text{posn}(T_k) - \|T_k\|, \text{loop}(T_k, T_{k+1}))$;
6.       call THREADING ($T_{i,k}, s_{v,u}, D^{1,k}_{i,j} \cup D^0_{i,k}, A^{1,k}_{i,j} \cup A^0_{i,k}, score_1$);
7.       call THREADING ($T_{k+1,j}, s_{u+1,w}, D^{2,k}_{i,j} \cup D^0_{i,k}, A^{2,k}_{i,j} \cup A^0_{i,k}, score_2$);
8.       if ($score > score_0 + score_1 + score_2$) then  $score \leftarrow score_0 + score_1 + score_2$;
9. else a simple procedure for calculating function (4).
```

The algorithm is correct for any $k \in [i, j-1]$ and any set of orientation assignments (line 3); but its computational efficiency changes as these values change. To recover an optimal sequence-fold alignment from the optimal alignment score, some simple bookkeeping needs to be done, which can be done without increasing the asymptotic complexity of the algorithm.

3 Computational complexity analysis

By exploiting a number of (nontrivial) data structures, the algorithm **THREADING** can be implemented in $O(mn + Mn^{1.5C})$ time and $O(mn + n^C)$ space, where C is the maximum open link complexity throughout the recursion of the algorithm (for the selected k's and orientation assignments). To further reduce the computational complexity hinges on two things: (a) making C as small as possible; (b) making the number of possible aligned positions for each individual core, and hence the number of combined aligned positions (line 4), as small as possible. We have achieved this by (i) discovering a new and significant parameter, the topological complexity of a fold template, which leads to the minimization of C; and (ii) utilizing biological constraints to reduce the number of possible aligned positions for each core.

3.1 The topological complexity of a fold

For partial template $T_{i,j}$, we use $c(T_{i,j}, D)$ to denote $T_{i,j}$'s open link complexity under the link-orientation assignments D. Let $\mathcal{T}$ represent a binary tree with the root representing the whole template $T_{1,M}$, the two children of each non-leaf node v representing a bi-partition of the (partial) template that v represents, and each leaf node representing a (different) core. We call

$$\min_{\mathcal{T},D}\{\max_{\text{tree node } T_{i,j} \in \mathcal{T}} c(T_{i,j}, D)\} \tag{5}$$

the *topological complexity* of fold T. An efficient algorithm has been developed for calculating the topological complexity of a fold and its associated bi-partition points and link-orientation assignments. By using these bi-partition points and link-orientation assignments in line 3 of **THREADING**, the maximum open link complexity C becomes the topological complexity of the fold, and hence minimized. We refer the reader to [15] for more details.

Table 1 shows the distribution of the topological complexity with respect to the cutoff distance (measured between beta-carbon atoms) for pairwise interactions, over a set of 750 folds[e] in our database. The first number of each entry denotes the number of cases among 750 folds that has a particular topological complexity (row) when the cutoff (Å) for pairwise interactions is set at a particular distance (column). The second number of each entry is the same as the first one except that the "seeing-each-other" condition is applied. We have observed that the topological complexity of our current data set is bounded from above by 8, no matter how large the cutoff distance when the "seeing-each-other" condition is used in defining interacting pairs.

Table 1: Distribution of topological complexity

C/cutoff (Å)	5	6	7	8	9	10	11	12	13	14	15
0	52	52	52	52	52	52	52	52	52	52	52
	52	52	52	52	52	52	52	52	52	52	52
1	87	87	87	87	87	87	87	87	87	87	87
	87	87	87	87	87	87	87	87	87	87	87
2	249	161	133	116	110	105	101	98	116	125	121
	255	162	134	118	111	107	103	102	128	135	125
3	246	251	184	133	114	99	91	85	70	86	82
	250	252	194	150	116	105	86	79	76	92	84
4	116	179	231	197	159	140	129	116	120	102	93
	106	177	241	187	160	150	138	133	117	105	103
5	0	20	61	140	144	147	121	113	106	109	102
	0	20	61	135	152	142	127	116	102	106	102
6	0	0	2	24	77	100	119	123	111	98	96
	0	0	1	20	66	96	121	121	114	97	103
7	0	0	0	1	7	18	47	69	74	78	86
	0	0	0	1	6	10	35	60	66	66	76
8	0	0	0	0	0	2	2	6	13	12	28
	0	0	0	0	0	1	1	2	6	10	18
9	0	0	0	0	0	0	1	1	1	1	4
	0	0	0	0	0	0	0	0	0	0	0

3.2 Application of biological constraints

To reduce the number of possible aligned positions in the query sequence for each individual core is another key in reducing the total number of possible sets of aligned positions, the computational bottleneck of our threading algorithm. For each core T_i, we use $\mathcal{D}_i$ to denote the set of all possible aligned positions of T_i. Two types of information can be used to reduce the size of a $\mathcal{D}_i$: (1) general biological constraints; and (2) fold-specific knowledge.

[e]The template length ranges from 31 - 793 amino acids and the number of their core secondary structures ranges from 1 to 34.

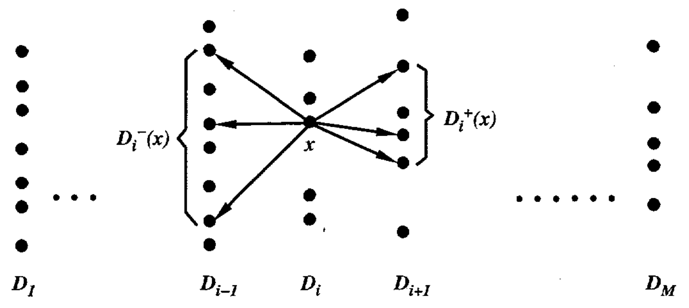

Figure 2: Valid aligned positions $\mathcal{D}_i^-(x)$ and $\mathcal{D}_i^+$ for T_{i-1} and T_{i+1}, respectively, when T_i is aligned at position x. Each node on layer i represents a possible aligned position core T_i may have (independent of other cores alignments). Each $y \in \mathcal{D}_i^-(x)$ and x forms a directed edge from x to y; Similarly for each $y \in \mathcal{D}_i^+(x)$.

We have developed a method for calculating the fitness profile of each individual core in the query protein sequence based on the combination of singleton fitness and secondary structure predictions [16], and have applied this method to preclude part of the query sequence as the possible aligned positions. Based on our extensive study, we have 0% false preclusion of correct aligned positions in the fold recognition process when related thresholds are set so that 10 - 20% of the sequence positions are precluded as possible aligned positions for each core. Knowledge about specific folds could help to further reduce the number of possible aligned positions for each individual core. Multiple sequence alignment or other protein sequence analysis could reveal positions of loop regions in a sequence, which can be precluded for consideration for possible aligned positions for any core.

Typically a loop region has 5 - 30 amino acids. By using the minimum and maximum loop lengths as constraints, we can significantly reduce the size of each $\mathcal{D}_i$. We outline an algorithm to achieve this. Let each $\mathcal{D}_i$ be sorted in the increasing order of their potential aligned positions. For each $x \in \mathcal{D}_i$, define $\mathcal{D}_i^+(x)$ be the sublist of $\mathcal{D}_{i+1}$ such that both the minimum loop length and the maximum loop length are satisfied whe T_i is aligned at position x and T_{i+1} is aligned with any element of this sublist, for each $i \in [1, M-1]$. $\mathcal{D}_i^-(x)$ is similarly defined with respect to T_{i-1}, for each $x \in \mathcal{D}_i$ and each $i \in [2, M]$. Using the graph representation of Figure 2, we have the following key observation: an $x \in \mathcal{D}_i$ is a part of a possible set of aligned positions in line 4 of **THREADING**, for any i, if and only if x is on a simple direct path from an element of $\mathcal{D}_1$ to an element of $\mathcal{D}_M$. The following algorithm removes all the elements that are not on any such direct path.

```
FILTER ($\mathcal{D}$)
forward step:
1.   for $i = 2$ till $M$ step $+1$ do
2.     if $x \in \mathcal{D}_i$ has $\mathcal{D}_i^-(x) = \emptyset$ then remove $x$;
reverse step:
3.   for $i = M - 1$ till $1$ step $-1$ do
4.     if $x \in \mathcal{D}_i$ has $\mathcal{D}_i^+(x) = \emptyset$ then remove $x$.
```

The correctness of the algorithm is straightforward to verify. Let $N = \max_{i,x\in\mathcal{D}_i}\{\|D_i^-(x)\|, \|D_i^+(x)\|\}$. With a careful implementation and analysis, the computational complexity of **THREADING** can be reduced to $O(mn + MnN^{1.5C-1})$ time and $O(mn + nN^{C-1})$ space.

4 Constrained optimal protein threading

This section gives a number of extensions to the basic **THREADING** algorithm of Section 2. The main goal is to provide a mechanism to facilitate easy incorporation of known biological constraints and knowledge during the fold recognition process. Knowledge about a query protein sequence will not only help to reduce the computational cost of the threading problem (as we discussed in Section 3), it could also help to significantly increase the fold recognition accuracy. All the following constrained optimal protein threading problems can be solved by slightly generalized versions of **THREADING**.

Secondary structure information

It is estimated that the best secondary structure prediction programs can give approximately 70% of secondary structure (α-helix, β-strand, and loop) prediction accuracy. Effectively using this type of information should help to improve the sequence-fold alignment accuracy. Typically, a secondary structure prediction program gives an estimate on the "probabilities" of a particular residue being a helix, a strand and a loop, represented as $(p_\alpha(x), p_\beta(x), p_o(x))$. We can model the optimal threading problem with known secondary structure predictions in the same way as in problem (1) except that the following term is added to the objective function (1).

$$\omega_s \sum_{i\in[1,M]} \sum_{j\in[1,\|T_i\|]} (1 - p_{\mathrm{core}(T_i[j])}(S_{2i}[j])) \tag{6}$$

where $\mathrm{core}(x) \in \{\alpha, \beta, o\}$ is the secondary structure of residue $T_i[j]$ in the fold template, and ω_s is a weight factor.

Disulfide bonds

Disulfide bonds between certain cysteine pairs of an unknown structure could possibly be determined before the full protein structure is solved. This type of knowledge can also help to improve the fold recognition accuracy. We model the optimal threading problem with known disulfide bonds $\mathcal{C}$ in the same way as in problem (1) under the constraint that each pair of $\mathcal{C}$ are aligned with template positions within certain distance d_{c-c}. This constraint can be implemented by adding the following term to the objective function (1).

$$\sum_{(S_{2i}[j], S_{2i'}[j'])\in\mathcal{C}} \mathrm{bond}(T_i[j], T_{i'}) \tag{7}$$

where $\mathrm{bond}(x, y) = \omega_d$, a large negative weight factor if the distance between x and y is $\leq d_{c-c}$, it is zero otherwise.

Active sites

Active sites of a query sequence could be determined experimentally or by multiple sequence alignment before the structure is fully determined. If the geometric relationship among the involved residues of an active site in the tertiary structure is known, this type of information can also be used to help to improve the sequence-fold alignment accuracy. The optimal threading problem with known active sites can be modeled as follows. Let $G(A)$ be a vector of geometric features describing the relationship among the involved elements of an active site A, and similarly $G(\text{aligned}(A))$ for A's aligned positions in the fold template. Ideally, $G()$ should reflect the multi-party relationship among the involved elements. But in our current implementation, $G()$ is represented as an array of pairwise relationship, describing geometric features along the line connecting two involved elements of A. This is because the general framework of our algorithm can deal with only pairwise "interactions". The constraint of putting an active site in an appropriate geometric environment of a fold template can be implemented by adding the following term to the objective function (1).

$$\omega_a \sum_{(a_i,a_j)\in\mathcal{A}} \text{match}(G(a_i,a_j), G(\text{aligned}(a_i), \text{aligned}(a_j))) \tag{8}$$

where $\mathcal{A}$ is a list of all the pairs of elements of an active site, and $\text{match}(x,y)$ measures how well two feature vectors match. Research is currently under way to extend this problem to deal with the most general form of $G()$.

Residue-residue distance constraints

Research is currently under way to explore using distance information between certain amino acids in an unknown structure from the NMR data as constraints for protein threading. Distance information between some pairs of amino acids in a sequence can be easily and unambiguously extracted from the NOESY spectra data. Effectively using this type of information could significantly improve the fold recognition accuracy. A generalized objective function to the one used for disulfide bonds is currently being studies for the purpose of incorporating NMR data in the fold recognition process.

5 Summary

The goals of our research are two-fold: (1) on the practical side, we want to develop an efficient algorithm that guarantees to find an optimal threading between a protein sequence and a fold template; (2) on the theoretical side, we are interested in understanding the intrinsic computational complexity of the protein threading problem under a rigorous formulation in which "accurate" fold recognition can be achieved. Our research suggests that topological complexity of a protein fold plays a significant role in determining the (intrinsic) computational complexity of the protein threading problem. The discovery of the topological complexity of a fold also helps us to better understand how to develop more efficient algorithms for the protein threading problem, by taking advantage of the properties of this measure.

By studying the topological complexity as a function of the cutoff distance between two interacting amino acids, we have put the protein threading problem into a more general framework of study. As we know, at the two extreme points lay the protein threading problem either as low-degree polynomial time solvable or NP-hard. Our study shows how the computational complexity changes as the cutoff distance increases, with and without applying the idea of visible volume. Extensive study is currently under way to understand how the fold recognition accuracy changes as the cutoff distance for interacting pairs changes. In a preliminary test on the sequence-fold alignment accuracy, **PROSPECT** aligned about 50% of the cores to the correct positions on 33 out of the 68 pairs of aligned structures from the UCLA benchmark set [5], and the average alignment accuracy is 3.9 base shifts per residue.

By carefully implementing the **THREADING** algorithm as a computer system **PROSPECT**, we can realistically find the most probable fold, based on a specified scoring scheme, for a query sequence of about 400 amino acids against a database of over 1000 folds with their topological complexity ≤ 6, on a workstation. With guaranteed global optimality and low computational cost, we expect **PROSPECT** to become a powerful tool in helping structural molecular biologists in their work of protein structure determination.

Acknowledgements

We thank Dr. Oakley Crawford and Dr. Victor Olman for providing us their program for calculating visible volumes, and for their very helpful discussions. We also want to thank Dr. Michael Unseren for critically reading the manuscript. This research was sponsored by the Office of Health and Environmental Research, U.S. Department of Energy, under Contract No. DE-AC05-96OR22464 with Lockheed Martin Energy Research Corporation.

References

[1] Bowie, J. U., Luthy, R., and Eisenberg, D. 1991. "A method to identify protein sequences that fold into a known three-dimensional structure", *Science*, 253, 164-170.

[2] Bryant, S. H. and Altschul, S. F. 1995. "Statistics of sequence-structure threading", *Curr. Opin. Struct. Biol.*, 5, 236 - 244.

[3] CASP I report. 1995. *Proteins: Structure, Function and Genetics*, 23, 295 - 462.

[4] CASP II report. 1997. *Proteins: Structure, Function and Genetics*, Suppl, 1 - 226.

[5] Fischer, D., Elofsson, A., Bowie, J. U., Eisenberg, D. 1996. "Assessing the performance of fold recognition methods by means of a comprehensive benchmark". In: *Biocomputing: Proceedings of the 1996 Pacific Symposium*, (Hunter, L. Klein, T., eds), 300–318. World Scientific Publishing Co. Singapore

[6] Godzik, A., Kolinski, A., and Skolnick, J. 1992. "Topology fingerprint approach to the inverse folding problem", *J. Mol. Biol.*, 227, 227 - 238.

[7] Jones, D. J., Taylor, W. R and Thornton, J. M. 1992. "A new approach to protein fold recognition", *Nature*, 358, 86–89.

[8] Johnson, M. S., Overington, J. P. and Blundell, T. L. 1993. "Alignment and searching for common protein folds using a data bank of structural templates", *J. Mol. Biol.*, 231, 735 - 752.

[9] Lathrop, R. H. 1994. "The protein threading problem with sequence amino acid interaction preferences is NP-complete", *Protein Engineering*, 7(9), 1059 - 1068.

[10] Lathrop, R. H. and Smith, T. F. 1996. "Global optimal protein threading with gapped alignment and empirical pair score functions", *J. Mol. Biol.*, 255, 641 - 665.

[11] Lo Conte, L. and Smith, T. F. 1997. "Visible volume: a robust measure for protein structure characterization", *J. Mol. Biol.*, 273, 338 - 348.

[12] Madej, T., Gibrat, J. F., and Bryant, S. H. 1995. "Threading a database of protein cores", *Proteins: Struct. Funct. Genet.*, 23, 356–369.

[13] Sippl, M. J. and Weitckus, S. 1992. "Detection of native-like models for amino acid sequences of unknown three-dimensional structure in a data base of known protein conformations", *Proteins: Struct. Funct. Genet.*, 13, 258–271.

[14] Xu, Y. and Uberbacher, E. C. 1996. "A polynomial-time algorithm for a class of protein threading problems", *Computer Applications in Biosciences*, 12, 511 - 517.

[15] Xu, Y., Xu, D. and Uberbacher, E. C. 1998, "An efficient computational method for globally optimal threading", *Journal of Computational Biology*, 5(3), 609 - 626.

[16] Xu, D., Unseren, M. A., Xu, Y., and Uberbacher, E. C. 1998, "Protein fold recognition at the secondary structure level", Submitted.