Electronic Notes in Theoretical Computer Science 76 (2002)
URL: http://www.elsevier.nl/locate/entcs/volume76.html 16 pages

Using Secondary Structure Information for Protein Folding in $CLP(\mathcal{FD})$

Agostino Dovier [a,2] Matteo Burato [b] Federico Fogolari [b,3]

[a] *Dip. di Matematica e Informatica, Univ. di Udine. Via delle Scienze 206, 33100 Udine (Italy).*

[b] *Dip. Scientifico-Tecnologico, Univ. di Verona. Strada Le Grazie 15, 37134 Verona (Italy).*

Abstract

The *protein folding problem* is the problem of predicting the 3D structure of a protein when the linear sequence of aminoacids identifying it is known. In this paper we present a declarative implementation in *Constrain & Generate* style in $CLP(\mathcal{FD})$ of the protein folding problem, for models based on face-centered cubes. We use information concerning secondary structure (and other heuristics) to sensibly prune the search space. Preliminary results on real proteins are encouraging.

Key words: $CLP(\mathcal{FD})$, Bio-Computing, Protein-Folding.

1 Introduction

Proteins are linear polymers constituted by linked units (aminoacids). The linear sequence of n aminoacids may be represented by a string of n letters taken from an alphabet of 20 letters (one for each aminoacid type, which differs from all others in its physico-chemical properties), and it has a direction due to the asymmetry of its components. The process by which a protein in a random conformation reaches its thermodynamically stable and peculiar spatial arrangement, the so-called *native conformation*, is known as *protein folding*. Several experiments prove that in consequence of an induced randomization of conformation (a process known as denaturation) the protein returns immediately in the native conformation [1]. The *protein folding problem (PFP)* is the problem of predicting the native conformation (i.e., the 3D structure of

[1] The work is partially supported by MIUR project: *Automatic Aggregate —and number— Reasoning for Computing.*

[2] Email: dovier@dimi.uniud.it

[3] Email: fogolari@sci.univr.it

the protein) when the linear sequence of aminoacids identifying the protein (i.e., its chemical structure) is known.

It is widely accepted that the native conformation ensures a state of minimum free energy. Thus, the solution of the PFP can be split into two steps:

(i) Defining an energy function, for the adopted protein representation, whose basic values generally depend on *distances* between any pair of aminoacids and on their *type*.

(ii) Finding the 3D conformation that minimizes the value of the energy function.

The former subproblem is faced using statistical analysis of native conformations obtained by X-Ray or NMR methods, e.g. by defining a matrix of potentials such that, for each pair of aminoacids, returns the value of energy associated with each *contact*. Thus, we can use this information for defining an energy function associated to a given conformation. This allows us to face the latter subproblem which is exactly a minimization problem involving several constraints. The decision version of the problem (namely, *is there a folding of the sequence with energy less than k?*) with some simplifying topological assumptions is proved to be NP-complete [5,8]. If the energy function can be polynomially computed, also the decision version of the general PFP remains (it is not worse than) NP. In spite of its intractability, the problem deserves to be attacked. The problem is in fact of crucial importance in Biology and Biotechnology and we can be encouraged by the non-huge typical length of a protein. Moreover, the fact that proteins fold extremely fast witnesses the existence of a hidden mechanism. Some knowledge about this mechanism surely improves any previous attempt. One of the ingredients of this mechanism could be the fact that particular partial conformations such as helices and sheets are recognizable in most proteins. These structural elements are called *Secondary Structure* of a protein. The high accuracy obtained by secondary structure predictions [15] prompts for inclusion of this prediction for protein folding prediction. In particular, in this work we investigate how effective is, in terms of efficiency and accuracy, the introduction of constraints obtained by this prediction.

In this paper we formally define the protein folding problem. Then we describe our declarative implementation in *Constrain & Generate* style in $CLP(\mathcal{FD})$. We use a topological model based on face-centered cubes. Initially, constraints come from this structure. Then we test the effectiveness of using secondary structure information in the algorithm. These further constraints, together with some heuristics for pruning the search space, allow us to successfully apply the algorithm on proteins of length around 50. We study the effectiveness of the method on some small proteins whose structure is known.

2 Related Work

There exists a wide bibliography concerning the PFP. For detailed and up-to-date reviews, see [7,16]. All prediction methods make use of statistical information available from the more than 18000 structures deposited in the Protein Data Bank (PDB) [10]. The correct fold for a new sequence can be obtained when *homology* (sequence similarity) is detected with a sequence for which the structure is available. Another approach tries to superimpose (*thread*) a chain on a known structure and evaluates the plausibility of the fold. At variance with the latter methods, *ab-initio* methods try to find the native conformation without a direct reference to a structural model.

As far as the constraint community is concerned, a subproblem of the whole problem, based on the *HP-model* is studied and successfully solved for proteins of length 30–40 in [2]. This model splits aminoacids into two classes (H and P) and the problem reduces to that of finding the conformation that maximizes the contacts of H's. This approach is very interesting, but the high level abstraction of the model does not ensure that the result is the native conformation; in particular, the local sub-conformations of the form of α-helices or β-strands (cf. Sect. 3) are often lost. Several other interesting works on this model have been performed by the same group (see, e.g., [3,4]). In [17] the related side-chain problem is studied inside the CLP framework.

3 The Protein Folding Problem

In this section we formally define the protein folding problem. We use a recently computed matrix of contact potentials (Table 1) and we focus on the mathematical formalization of the problem.

3.1 Preliminary notions

The *Primary* structure of a protein is a sequence of linked units (or *residues*) of length, typically, less than 500. Each residue is an aminoacid, from a set $\mathcal{A}$ of 20 types:

Alanine (A)	**Cys**teine (C)	**Asp**artic Acid (D)	**Glu**tamic Acid (E)
Phenylalanine (F)	**Gly**cine (G)	**His**tidine (H)	**Isole**ucine (I)
Lysine (K)	**Leu**cine (L)	**Met**hionine (M)	**As**paragi**n**e (N)
Proline (P)	**Gl**utami**n**e (Q)	**Arg**inine (R)	**Ser**ine (S)
Threonine (T)	**Val**ine (V)	**Tryp**tophan (W)	**Tyr**osine (Y)

Applying statistical methods on structures obtained by X-Rays and NMR experiments, Table 1, that points out the energy associated to a pair of non consecutive aminoacids when they are in contact, has been developed [13,6]. With $\mathsf{Pot}(x, y)$ we denote the value of the table addressed by aminoacids x and y (the order is immaterial).

	CYS	MET	PHE	ILE	LEU	VAL	TRP	TYR	ALA	GLY
CYS	-3.477	-2.240	-2.424	-2.410	-2.343	-2.258	-2.080	-1.892	-1.700	-1.101
MET	-2.240	-1.901	-2.304	-2.286	-2.208	-2.079	-2.090	-1.834	-1.517	-0.897
PHE	-2.424	-2.304	-2.467	-2.530	-2.491	-2.391	-2.286	-1.963	-1.750	-1.034
ILE	-2.410	-2.286	-2.530	-2.691	-2.647	-2.568	-2.303	-1.998	-1.872	-0.885
LEU	-2.343	-2.208	-2.491	-2.647	-2.501	-2.447	-2.222	-1.919	-1.728	-0.767
VAL	-2.258	-2.079	-2.391	-2.568	-2.447	-2.385	-2.097	-1.790	-1.731	-0.756
TRP	-2.080	-2.090	-2.286	-2.303	-2.222	-2.097	-1.867	-1.834	-1.565	-1.142
TYR	-1.892	-1.834	-1.963	-1.998	-1.919	-1.790	-1.834	-1.335	-1.318	-0.818
ALA	-1.700	-1.517	-1.750	-1.872	-1.728	-1.731	-1.565	-1.318	-1.119	-0.290
GLY	-1.101	-0.897	-1.034	-0.885	-0.767	-0.756	-1.142	-0.818	-0.290	0.219
THR	-1.243	-0.999	-1.237	-1.360	-1.202	-1.240	-1.077	-0.892	-0.717	-0.311
SER	-1.306	-0.893	-1.178	-1.037	-0.959	-0.933	-1.145	-0.859	-0.607	-0.261
ASN	-0.788	-0.658	-0.790	-0.669	-0.524	-0.673	-0.884	-0.670	-0.371	-0.230
GLN	-0.835	-0.720	-0.807	-0.778	-0.729	-0.642	-0.997	-0.687	-0.323	0.033
GLU	-0.616	-0.409	-0.482	-0.402	-0.291	-0.298	-0.613	-0.631	-0.235	-0.097
ASP	-0.179	-0.209	-0.419	-0.439	-0.366	-0.335	-0.624	-0.453	-0.039	0.443
HIS	-1.499	-1.252	-1.330	-1.234	-1.176	-1.118	-1.383	-1.222	-0.646	-0.325
ARG	-0.771	-0.611	-0.805	-0.854	-0.758	-0.664	-0.912	-0.745	-0.327	-0.050
LYS	-0.112	-0.146	-0.270	-0.253	-0.222	-0.200	-0.391	-0.349	0.196	0.589
PRO	-1.196	-0.788	-1.076	-0.991	-0.771	-0.886	-1.278	-1.067	-0.374	-0.042

	THR	SER	ASN	GLN	ASP	GLU	HIS	ARG	LYS	PRO
CYS	-1.243	-1.306	-0.788	-0.835	-0.616	-0.179	-1.499	-0.771	-0.112	-1.196
MET	-0.999	-0.893	-0.658	-0.720	-0.409	-0.209	-1.252	-0.611	-0.146	-0.788
PHE	-1.237	-1.178	-0.790	-0.807	-0.482	-0.419	-1.330	-0.805	-0.270	-1.076
ILE	-1.360	-1.037	-0.669	-0.778	-0.402	-0.439	-1.234	-0.854	-0.253	-0.991
LEU	-1.202	-0.959	-0.524	-0.729	-0.291	-0.366	-1.176	-0.758	-0.222	-0.771
VAL	-1.240	-0.933	-0.673	-0.642	-0.298	-0.335	-1.118	-0.664	-0.200	-0.886
TRP	-1.077	-1.145	-0.884	-0.997	-0.613	-0.624	-1.383	-0.912	-0.391	-1.278
TYR	-0.892	-0.859	-0.670	-0.687	-0.631	-0.453	-1.222	-0.745	-0.349	-1.067
ALA	-0.717	-0.607	-0.371	-0.323	-0.235	-0.039	-0.646	-0.327	0.196	-0.374
GLY	-0.311	-0.261	-0.230	0.033	-0.097	0.443	-0.325	-0.050	0.589	-0.042
THR	-0.617	-0.548	-0.463	-0.342	-0.382	-0.192	-0.720	-0.247	0.155	-0.222
SER	-0.548	-0.519	-0.423	-0.260	-0.521	-0.161	-0.639	-0.264	0.223	-0.199
ASN	-0.463	-0.423	-0.367	-0.253	-0.344	0.160	-0.455	-0.114	0.271	-0.018
GLN	-0.342	-0.260	-0.253	0.054	0.022	0.179	-0.290	-0.042	0.334	-0.035
GLU	-0.382	-0.521	-0.344	0.022	0.179	0.634	-0.664	-0.584	-0.176	0.189
ASP	-0.192	-0.161	0.160	0.179	0.634	0.933	-0.324	-0.374	-0.057	0.257
HIS	-0.720	-0.639	-0.455	-0.290	-0.664	-0.324	-1.078	-0.307	0.388	-0.346
ARG	-0.247	-0.264	-0.114	-0.042	-0.584	-0.374	-0.307	0.200	0.815	-0.023
LYS	0.155	0.223	0.271	0.334	-0.176	-0.057	0.388	0.815	1.339	0.661
PRO	-0.222	-0.199	-0.018	-0.035	0.189	0.257	-0.346	-0.023	0.661	0.129

Table 1
Potential Matrix

Native conformations are largely built from *Secondary Structure elements* (i.e., helices and sheets) often arranged in defined motifs. α-helices are constituted by 5 to 40 contiguous residues arranged in a regular right-handed helix with 3.6 residues per turn. β-sheets are constituted by extended strands of 5 to 10 residues. Each strand is made of contiguous residues, but strands participating in the same sheet are not necessarily contiguous in sequence. There are algorithms based on neural networks that can predict with high accuracy (75% [7]) the secondary structure of a protein. Another important structural feature of proteins is the capability of cysteine residues of covalently binding through their sulphur atoms, thus forming disulfide bridges, which impose important *contact* constraints. This kind of information is often available, either by experiments or predictions.

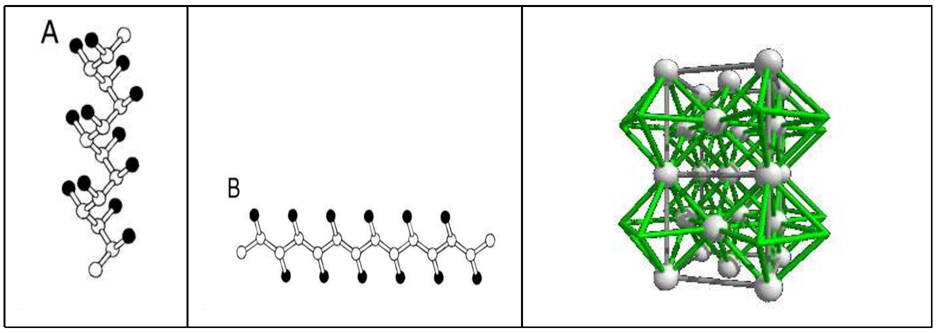

Fig. 1. α-helix, β-strand, and fcc cube

3.2 Problem definition

Several models are proposed for reasoning about the 3D properties of proteins, basically concerning the admissible spatial positions of each aminoacid (also called the *Tertiary Structure* of the protein). Given a sequence $S = s_1 \cdots s_n$, with $s_i \in \mathcal{A}$, the position p_i of a point representing each aminoacid s_i is a triple of values $\langle x, y, z\rangle$ (3D models). Moreover, the variables x, y, and z can be real numbers (in—rather untractable—models in which proteins are left free to take any position) or integer numbers (in models where aminoacids can take a finite number of positions of a suitable lattice).

We use the predicate next stating that two positions are admissible consecutive positions for two aminoacids that appear consecutively in the sequence. next requires as parameters the two points, but also the whole sequence of positions, since in some lattices the fact that two points are in sequence depends also on other points (for instance, the previous one). It is assumed that a fixed (often chosen as unitary value) distance separates two consecutive aminoacids. We also employ the binary predicate contact: two non-consecutive aminoacids s_i and s_j in the position p_i and p_j are in contact (in this case we write $\mathsf{contact}(p_i, p_j)$) when their distance is less than a certain value. Lattice models simplify the definition of these two predicates. The following definition of the PFP is general enough to apply to several models.

Given a sequence $S = s_1 \cdots s_n$, with $s_i \in \mathcal{A}$, the *protein folding problem (PFP)* is the problem of finding the sequence of positions $P = p_1 \cdots p_n$ satisfying the constraints:

(1) $(\forall i \in \{1, \ldots, n-1\})\, \mathsf{next}(P, p_i, p_{i+1})$

(2) $(\forall i, j \in \{1, \ldots, n\})(i \neq j \rightarrow p_i \neq p_j)$

and minimizing the energy:

(3) $$E(P) = \sum_{\substack{1 \leq i < n \\ i+2 \leq j \leq n}} f(P, S, i, j)$$

$$\text{where } f(P,S,i,j) = \begin{cases} \mathsf{Pot}(s_i, s_j) & \text{if } \mathsf{contact}(p_i, p_j) \\ 0 & \text{otherwise} \end{cases}$$

3.3 Lattice Models

Lattice models have long been used for protein structure prediction [11]. A possible model is a lattice whose points are a subset of $\mathbb{N}^3$. Nevertheless, the only angles between three consecutive aminoacids allowed in this case are 90° and 180°. This lattice is too rigid for a realistic modelization of the problem.

In [14] it is shown that the *Face-Centered Cubic Lattice* (fcc) model is a well-suited, realistic model for 3D conformations of proteins. The model is based on cubes of size 2, but the central point of each face is also admitted (together with the vertices). Points at distance $\sqrt{2}$ are connected; their distance is said *lattice unit*. In this way each point is adjacent to 12 neighboring points (see Fig. 1).

Every residue may be linked to an adjacent residue by one of the possible twelve lattice vectors. The angle between three residues may therefore assume values 60°, 90°, 120°, and 180°. Steric and energetic restraints in proteins make values 60° and 180° unfeasible. Therefore only links at 90° and 120° will be retained. No similar restriction exists on torsional angles among four adjacent residues. A *contact* is defined among two non adjacent residues when their separation is two lattice units. The more natural choice of a single lattice unit is ruled out because it does not take into account aminoacid steric hindrance. Physically, a lattice unit corresponds to 3.8 Å, which is roughly the van der Waals contact distance between two carbon atoms. Consequently, we impose the constraint that two non consecutive residues must be separated by more than one lattice units.

4 Definition in $CLP(\mathcal{FD})$

In this section we describe the main predicates used to implement declaratively the PFP. We have used the library clpfd of SICStus PROLOG 3.9.1 [9]. Complete code and other related material can be found in:

`http://www.dimi.uniud.it/~dovier/PF/pf_clp.html`

The main clause is a classical *Constrain & Generate* [12] clause of the form:

```
fcc_pf(ID, Energy):-
    protein(ID, Primary, Secondary),
    constrain(Primary, Secondary, Indexes, Tertiary_flat, Energy, _),
    labeling(Primary, Secondary, Indexes, Tertiary_flat, Energy),
    pretty_print(Primary, Tertiary_flat).
```

The `protein` predicate, extensionally defined in an auxiliary file `data.pl`, allows us to access to the `Primary` and `Secondary` structures of a protein given its name. For instance,

```
protein(ID, Primary, Secondary):-
     ID = '1LE0',
     Primary = [s,w,t,w,e,g,n,k,w,t,w,k],
     Secondary = [strand(2,4), strand(9,11)].
```

The `constrain` predicate deterministically adds the (finite domain) constraints for the variables involved, while `labeling` looks for the solution in the search space. `pretty_print` prints the output in a format suitable for validating the results with biological software (c.f. Section 6).

`Tertiary_flat` is the output list of positions (triples of integers) of the various aminoacids (the conformation) and Energy is the output value of energy associated to it (values of the Table 1 are multiplied by 1000 in order to deal with integer values). For efficiency reasons in the labeling phase, we use a *flat* representation of this list. Also a non-flat version of it, called `Tertiary`, is used in other predicates.

`Indexes` is an auxiliary variable that we discuss, together with the variable `Secondary`, in the next section.

Just as an example, consider a possible computation:

```
| ?- fcc_pf('1LE0', Energy).
1       s       12      12      12
2       w       13      13      12
3       t       14      13      13
4       w       15      12      13
5       e       16      13      13
6       g       17      12      13
7       n       17      11      12
8       k       18      10      12
9       w       17      9       12
10      t       16      9       13
11      w       15      10      13
12      k       14      10      12
Energy = -4085
```

4.1 Constrain

The predicate `constrain` is defined as follows:

```
constrain(Primary, Secondary, Indexes,
          Tertiary_flat, Energy, PotList):-
    generate_tertiary(N, Primary, Tertiary, Tertiary_flat),
    domain_bounds(N, Tertiary),
    avoid_symmetries(N, Tertiary),
    avoid_self_loops(Tertiary),
    next_constraints(Tertiary),
    distance_constraints(Tertiary),
    energy_constraints(Primary, Tertiary, Energy, PotList).
```

The first group of predicates adds constraints to the variables of the lists

$$\texttt{Tertiary} = [[X_1, Y_1, Z_1], \ldots, [X_N, Y_N, Z_N]]$$
$$\texttt{Tertiary\_flat} = [X_1, Y_1, Z_1, \ldots, X_N, Y_N, Z_N]$$

where N is the length of the list `Primary`. `domain_bounds` bounds to $0 \ldots 2{*}N$ all the variables X_i, Y_i, Z_i. Moreover, exploiting some lattice properties (as done in [2]) we also force $X_i + Y_i + Z_i$ to be even. `avoid_symmetries` introduces some constraints aimed at removing redundant admissible conformations, equivalent to others modulo some symmetries. In particular, we set

$$[X_1, Y_1, Z_1], [X_2, Y_2, Z_2] = [N, N, N], [N+1, N+1, N]$$

(if N is odd, add 1 to all variables, for satisfying the *even* constraint).
`avoid_self_loops` forces all triples to be distinct. We use the built-in predicate `all_different` on the list $[I_1, \ldots, I_n]$ where I_i `#=` $(X_i{*}P{*}P)+(Y_i{*}P)+Z_i$, for a suitable value P. `next_constraints` imposes the fact that $[X_i, Y_i, Z_i]$ and $[X_{i+1}, Y_{i+1}, Z_{i+1}]$ are adjacent points in the lattice. We use the variables DX, DY, and DZ, constrained in $0 \ldots 1$, and we force

$$\mathtt{DX \; \#= \; abs(X_i - X_{i+1}), DY \; \#= \; abs(Y_i - Y_{i+1}), DZ \; \#= \; abs(Z_i - Z_{i+1}),}$$
$$\mathtt{DX + DY + DZ \; \#= \; 2.}$$

Moreover, we also force that three consecutive points can only form angles of 90° or 120°. Given three consecutive points A, B, and C, we set

```
     vector_sum(A,D1,B),
     vector_sum(B,D2,C),
     vector_sum(D1,D2,[Dx, Dy, Dz]),
     abs(Dx) #= 2 #<=> Flag1,
     abs(Dy) #= 2 #<=> Flag2,
     abs(Dz) #= 2 #<=> Flag3,
     Flag1 + Flag2 + Flag3 #= 1.
```

where `vector_sum` performs the sum of two vectors. Observe the use of reified constraints: we require that exactly one direction has a variation of 2 units in 2 steps. `distance_constraints` forces two non-consecutive points to be at a distance of at least 2. For $j > i+1$ we add:

```
     DX #= abs(X_i - X_j),
     DY #= abs(Y_i - Y_j),
     DZ #= abs(Z_i - Z_j),
     DX #> 1 #\/ DY #> 1 #\/ DZ #> 1  #\/ DX + DY + DZ #> 2.
```

In this way, we avoid Euclidean distances that would have moved us outside $CLP(\mathcal{FD})$.

The predicate `energy_constraints` is the sum of the variables C defined for each pairs of aminoacids A_i, A_j occurring in `Primary`, with $j > i+1$, as follows:

```
table(A_i,A_j, Pot),
C in {0,Pot},
DX #= abs(X_i - X_j),
DY #= abs(Y_i - Y_j),
DZ #= abs(Z_i - Z_j),
2 #= DX + DY + DZ #<=> C #= Pot.
```

where `table` reports the value $\mathsf{Pot}(A_i, A_j)$ from Table 1 and the '2' in the last line states that the contact distance required is 2 (observe that `DX`, `DY`, and `DZ` are not constrained in $0 \ldots 1$ as happens for the definition of `next_constraints`). `PotList` is the list of all the `C`s defined above. We pass it as parameter since it is useful for our ad-hoc labeling that we discuss in Section 5.2. It is not needed in the main program (a mute variable is in fact used) but we use it in internal calls to the predicate `constraint`.

The number of constraints globally introduced is $O(n^2)$.

4.2 Generate

We can easily define the predicate `labeling` using the built-in `labeling` as follows:

```
labeling(Primary, Secondary, Indexes, Tertiary_flat, Energy):-
     labeling([ff,minimize(Energy)], Tertiary_flat).
```

The program is highly declarative, and thus it can be used to verify the correctness of the model. However, according to our tests, it finds solutions in reasonable time (few minutes) only for input lists of length $N \leq 11$.

5 Use of Secondary Structure Information

In this section we show how to exploit the secondary structure information to optimize the declarative prototype presented in the previous section. Moreover, we introduce some reasonable heuristics for pruning the search space.

As said in Section 3, the secondary structure of a protein can be predicted with high accuracy (e.g., by the program 'PhD' [15]). Information is of the kind:

`helix`(i,j): stating that elements $i, i+1, \ldots, j$ of the input sequence form an helix,

`strand`(i,j): that states that elements $i, i+1, \ldots, j$ are in a β-strand.

Often also information on disulfide bridges is available, either by experiments or predictions, which can be cast in the form:

`ssbond`(i,j): that witnesses the presence of a disulfide bridge between element number i and j (namely, that they are in contact or at least, very closed each other).

This information is contained in the list `Secondary`; for our tests, we have picked it from the Protein Data Bank [10]. It is of course possible to modify the code so as to use automatically the prediction given by the program PhD and/or other programs.

In the fcc model it is useful to adopt also another representation for describing the positions of the aminoacids. We already discussed that, given two points, only 6 possible directions for the next point are allowed. Thus, with a list of $N-2$ numbers ranging from 1 to 6 we can precisely identify the space position of each aminoacid. This representation is not well-suited for computing energy, but it is perfect for imposing constraints regarding helices and strands. As a matter of fact, helices are sequences of the form: 1-3-4-6-1-3-4-6-$\cdots$ (they can also start from 3, 4, or 6) while β-strands are associated to sequences of the form: 2-3-2-3-$\cdots$ (or 3-2-3-2$\cdots$)—cf. Fig. 1.

5.1 New Constraints

The predicate `constrain` is modified by adding the following literals in its body (N is the length of the input list `Primary`):

```
generate_indexes(N,Indexes),
secondary_info(Secondary, Indexes, Tertiary),
indexes_to_coordinates(Indexes, Tertiary).
```

`generate_indexes` returns the list `Indexes` $= [I_3, \ldots, I_N]$ of the variables constrained between $1 \ldots 6$. The predicate `secondary_info` adds the constraints concerning the secondary structure information. In particular, `helix`(i,j) forces the variables of `Indexes` associated to elements $i, i+1, \ldots, j$ to have values $1, 3, 4, 6, \ldots$.[4] Similarly, `strand`(i,j) forces the variables of `Indexes` associated to elements $i, i+1, \ldots, j$ to have values $2, 3, \ldots$.[5] `ssbond`(i,j) introduces the constraint:

$$\texttt{abs}(\texttt{X}_\texttt{i} - \texttt{X}_\texttt{j}) + \texttt{abs}(\texttt{Y}_\texttt{i} - \texttt{Y}_\texttt{j}) + \texttt{abs}(\texttt{Z}_\texttt{i} - \texttt{Z}_\texttt{j}) \ \texttt{\#=<}\ 6.$$

The predicate `indexes_to_coordinates` relates the constraints on the indexes to the variables denoting positions. The definition is rather technical and we omit it here due to lack of space. Basically, we need to relate a variable I_i with domain $1 \ldots 6$ to the sequence of points

$$[X_{i-2}, Y_{i-2}, Z_{i-2}], [X_{i-1}, Y_{i-1}, Z_{i-1}], [X_i, Y_i, Z_i]$$

[4] Or, using more relaxed constraints, one of the sequences starting from 1, 3, 4, or 6.

[5] Again, we could use 2,3,... or 3,2,...

that can form only 6 possible angles.

5.2 New Generate

Even with the pruning ensured by the constraints coming from secondary structure information, the search space remains in general too large to be used for real-size proteins. We have sets of working examples of length around 20, but this does not suffice. Thus, we have replaced the `labeling` predicate with the predicate `local_labeling` that we explain below in order to further pruning the search space. Two ideas are used to justify the heuristics employed for this aim:

(i) It is not frequent that elements that are distant in the primary structure affect sensibly the contacts on the closer elements. This allows us to compute first some local optimal conformations and then propagate this information.

(ii) Each time a fixed number of position values have been instantiated, we evaluate the partial energy and, possibly, decide to cut the search tree. The rationale behind this heuristic is the fact that the global minimum energy conformation must be compatible with locally low energy conformations. Thus, conformations that have high partial energies can be discarded.

As far as point (i) is concerned, we manage the instantiation procedure so as to locally optimize subsequences in which known secondary structures have been recognized. In this way, we force the relative position of the elements of the subsequence analyzed. This partial structure is rigid and as such it can be inserted in the whole minimization process.

More in detail, let `Indexes` be the list of variables that will store the various angles. At a certain point of the computation some of them are instantiated (due to domain reduction or a previous partial labeling), other are still not instantiated. The situation is therefore of the form:

$$\texttt{Indexes} = [(\bar{v}_0), \bar{c}_1, \bar{v}_1, \ldots, \bar{c}_h, (\bar{v}_h)]$$

for some $h \geq 0$, where $\bar{v}_i$ denotes a list of unbounded variables, $\bar{c}_i$ a list of ground values. $\bar{v}_0$ and $\bar{v}_h$ can be the empty list (e.g., when all the variables in `Indexes` are instantiated). The predicate `admissible_sub` finds all the sublists of the form:

$$[\bar{c}_i, \bar{v}_i, \bar{c}_{i+1}, \bar{v}_{i+1}, \bar{c}_{i+2}]$$

If no tuples of this form occur in `Indexes` (e.g., when `Indexes` $= [\bar{v}_0]$), some particular cases have been studied. Then the sublist `SubIndexes` that maximizes the value: $|\bar{c}_i| + |\bar{v}_i| + |\bar{c}_{i+1}| - 2 \cdot |\bar{v}_i|$ is chosen (in other words, the most constrained sublist). If in the most constrained sublist there are too much variables (e.g., more than 10) we cut the sublist in order to leave only 10 free variables. Then we find the best folding of the local sublist. The folding of a subsequence is called a *phase*.

The heuristic (ii) is implemented as follows. If the number of variables of the selected subsequence is less than 6, then we call the built-in

```
labeling([ff,minimize(Energy)], SubIndexes).
```

Otherwise, we have developed an ad-hoc labeling. The choice of the variable is first-fail and leftmost, as in the built-in case described above. After each instantiation of k variables in this phase we:

- Retrieve the minimum value Val already computed
- Retrieve the number To_do of variables to be instantiated in this stage
- Compute the number Done of variables instantiated in this phase
- Compute the energy value Valint forced by the contacts of the already known elements

We continue the solution search in this branch if:

$$\texttt{Valint} \#< \texttt{integer}(\xi * \texttt{Val} * (\texttt{Now\_Done/To\_do}))$$

Experimentally, we have set $k = 5$ and $\xi \leq 0.3$. This heuristics has the drawback of performing a partial computation for Valint. A constraint-based heuristics based on the same ideas that allows to avoid that computation uses the constrained global variable Energy:

$$\texttt{Energy} \#< \texttt{integer}(\eta * \texttt{Val} * (\texttt{Now\_Done/To\_do}))$$

to be used with $\eta \leq 1.2$ (in the tests: $\eta = 1.1$). We have used both the checks. We have implemented this heuristic using `assert` and `retract` for storing/retrieving the minimum energy Val temporarily found.

6 Experimental Results

We have tested our program on some small protein model systems. The names we have used are their identity codes in the Protein Data Bank [10] where their primary and secondary structures, as well as some other relevant information can be obtained. Computational results are reported in Table 2 (we have used SICStus PROLOG 3.9.1 [9] and a—rather old—PC Pentium III 500 MHz). More details on the computation time and results can be found at: `http://www.dimi.uniud.it/~dovier/PF/pf_clp.html`.

In Table 2, "b" stands for ssbond, "s" for strand, and "h" for helix. The quality of the prediction has been judged by the *root mean square deviation (RMSD)* of α-carbons in optimally superimposed structures extracted from the Protein Data Bank. In the protein model systems 1LE3, 1PG1, and 1ZDD terminal protecting groups have been neglected.

In addition to small peptide model systems we have studied *viscotoxin A3*

Name	N	Secondary Info	Time	Energy	RMSD
1LE0	12	[s(2,4),s(9,11)]	2m.30s	-4085	4.1 Å
1KVG	12	[b(2,11),s(2,4),s(9,11)]	2m.30s.	-6247	3.8 Å
1LE3	16	[s(2,6),s(11,15)]	12m.30s.	-4338	5.3 Å
1EDP	17	[b(1,15),b(3,11),h(9,15)]	33m.30s	-19416	4.7 Å
1PG1	18	[b(6,15),b(8,13), s(4,9),s(12,17)]	26s.	-2907	4.2 Å
1ZDD	34	[b(5,34),h(3,13),h(20,33)]	47m.	-19843	4.3 Å
1VII	36	[h(4,8),h(15,18),h(23,32)]	1h.31m.	-24725	10.0 Å 7.5 Å (4-32)
1E0M	37	[s(7,12),s(18,22),s(17,19)]	19h.17m.	-21944	7.2 Å 5.9 Å (7-22)
2GP8	40	[h(6,21),h(26,38)]	6h.37m.	-12361	5.0 Å
1ED0	46	[b(3,40),b(4,32),b(16,26), h(7,18),h(23,30), s(2,4),s(33,34)]	17h.50m.	-29916	7.3 Å 3.7 Å (7-30)
1ENH	54	[h(8,20),h(26,36),h(40,52), s(22,23)]	3h.24m.	-24859	10.0 Å 5.4 Å (8-52)

Table 2
Experimental Results

(1ED0) which is a 46 residue protein and *engrailed homeodomain* (1ENH), a DNA-binding domain of 54 residues, which display a somewhat more complex structure. In the case of 1ENH we have also executed a computation with a lower value of the parameters ξ and η (see Section 5.2) used to prune the search tree. Precisely, with $\xi = 0.1$ and $\eta = 0$ (actually, we have removed the test concerning η) we have obtained a solution with an energy of -24859 instead of the one with -20261 obtained with the other parameters. For other cases, no particular differences emerge by decreasing those parameters, but the computations become slower. A stereoview of the results obtained for 1ZDD is reported in Figure 2 in order to represent graphically the output of the computation (obtained using the program Whatif [18] on the Prolog output) and to compare it with the known structure deposited in the PDB (helices are depicted as cylinders).[6]

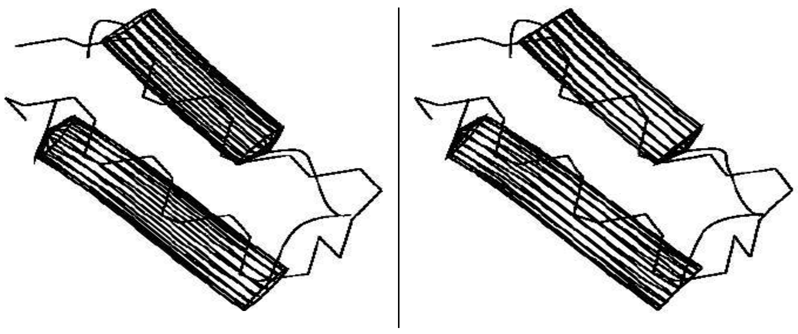
Fig. 2. Stereoview of 1ZDD protein

It is remarkable that for most small peptides ranging in length from 12 to 40 residues the RMSD is rather low, typically around 5.0 Å. For the homeodomain the superposition of residues 8–52 belonging to the core region between predicted and actual structure is as low as 5.4 Å. For viscotoxin A3 the different orientation of end parts of the molecule lead to a RMSD of 7.3 Å, while the RMSD obtained superimposing the helical region (residues 7–30) is extremely low (3.7 Å).

7 Future Work and Conclusions

In this paper we have formally defined the protein folding problem as a minimization problem using a recently developed table of potentials. We have implemented it in $CLP(\mathcal{FD})$ and we have sensibly optimized the code using information from secondary structure and two ad-hoc heuristics. The work done could be a starting point for further analysis of the same problem. The high declarativeness of the code allows an easy implementation of new ideas, certified by biological considerations, for introducing new constraints and/or heuristics. In the immediate future we plan to use the information coming from the folding in the HP model [2] as an initial suggestion for starting the solution search and/or for pruning the search tree. Other constraints can be inserted, such as side-chains constraints [17] or other topological constraints.

Acknowledgements

We would like to thank Marco Berrera, for making available to us Table 1 developed in his Master's Thesis [6], and for many useful discussions.

[6] To view the stereo image, put it close to your eyes, and then slowly move the paper away from your face, trying to keep the images superimposed until you can focus on them. You will see three images: the middle one should be a 3D image.

References

[1] C. B. Anfinsen. Principles that govern the folding of protein chains. *Science*, 181(4096):223–230, July 1973.

[2] R. Backofen. The protein structure prediction problem: A constraint optimization approach using a new lower bound. *Constraints*, 6(2/3):223–255, 2001.

[3] R. Backofen and S. Will. Excluding symmetries in constraint-based search. In J. Jaffar, ed., *CP'99*, LNCS vol. 1713, pages 73–87, 1999.

[4] R. Backofen and S. Will. Fast, constraint-based threading of HP-sequences to hydrophobic cores. In T. Walsh, ed., *CP'01*, LNCS vol. 2239, pages 494–508, 2001.

[5] B. Berger and T. Leighton. Protein folding in the hydrophobic-hydrophilic (HP) model is NP-complete. In *Proc. of Second Annual International Conferences on Computational Molecular Biology (RECOMB'98)*, pages 30–39, New York, 1998.

[6] M. Berrera. Determinazione di energie empiriche di contatto fra amminoacidi per la discriminazione di strutture proteiche native e non native. Master's thesis, Laurea in Biotecnologie Agro-Industriali, Univ. di Verona, 2002. In Italian.

[7] R. Bonneau and D. Baker. Ab initio protein structure prediction: progress and prospects. *Annual Review of Biophysics and Biomolecular Structure*, 30:173–89, 2001.

[8] P. Crescenzi, D. Goldman, C. Papadimitrou, A. Piccolboni, and M. Yannakakis. On the complexity of protein folding. In *Proc. of STOC*, pages 597–603, 1998.

[9] Swedish Institute for Computer Science. Sicstus Prolog Home Page. `http://www.sics.se/sicstus/`.

[10] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne. The Protein Data Bank. *Nucleic Acids Research* 28:235–242 (2000). `http://www.rcsb.org/pdb/`.

[11] E. Shakhnovich L. Mirny. Protein folding theory: from lattice to all-atom models. *Annual Review of Biophysics and Biomolecular Structure*, 30:361–96, 2001.

[12] K. Marriott and P. J. Stuckey. *Programming with Constraints.* The MIT Press, 1998.

[13] S. Miyazawa and R. L. Jernigan. Residue-residue potentials with a favorable contact pair term and an unfavorable high packing density term, for simulation and threading. *Journal of Molecular Biology*, 256(3):623–644, 1996.

[14] G. Raghunathan and R. L. Jernigan. Ideal architecture of residue packing and its observation in protein structures. *Protein Science*, 6:2072–2083, 1997.

[15] B. Rost and C. Sander. Prediction of protein secondary structure at better than 70% accuracy. *Journal of Molecular Biology*, 232:584–599, 1993.

[16] J. Skolnick and A. Kolinski. Computational studies of protein folding. *Computing in Science and Engineering*, 3(5):40–50, 2001.

[17] M. T. Swain and G. J. L. Kemp. A CLP approach to the protein side-chain placement problem. In T. Walsh, editor, *CP'01*, LNCS vol. 2239, pages 479–493, 2001.

[18] G. Vriend. WHAT IF: A molecular modeling and drug design program. *Journal of Molecular Graphics* (1990) 8:52–56.