Constraint Programming approaches to the Protein Folding Problem.

Agostino Dovier
DIMI, University of Udine (IT)
www.dimi.uniud.it/dovier
www.dimi.uniud.it/dovier/PF

Outline of the talk

- Basic notions on Proteins
- Introduction to Protein Folding/Structure Prediction Problem
- The PFP as a constrained optimization problem ($CLP(\mathcal{FD})$)
  - Abstract modeling (HP) and solutions
  - Realistic modeling and solutions
- Simulation (CCP) approach to the problem
- Other approaches
- Conclusions

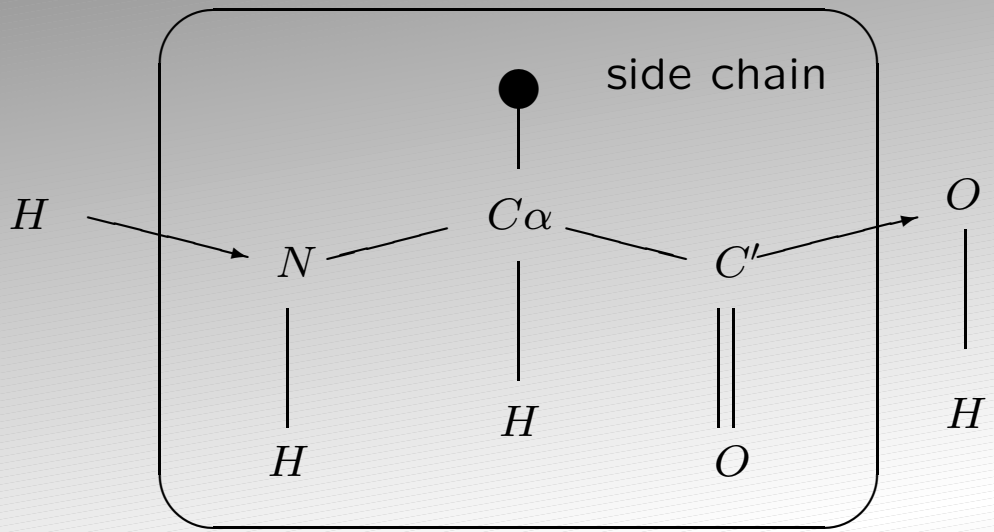
Proteins

- Proteins are abundant in all organisms and fundamental to life.
- The diversity of 3D protein structure underlies the very large range of their function:
  - Enzymes—biological catalysts
  - Storage (e.g. ferritin in liver)
  - Transport (e.g. haemoglobin)
  - Messengers (transmission of nervous impulses—hormones)
  - Antibodies
  - Regulation (during the process to synthesize proteins)
  - Structural proteins (mechanical support, e.g. hair, bone)

Primary Structure

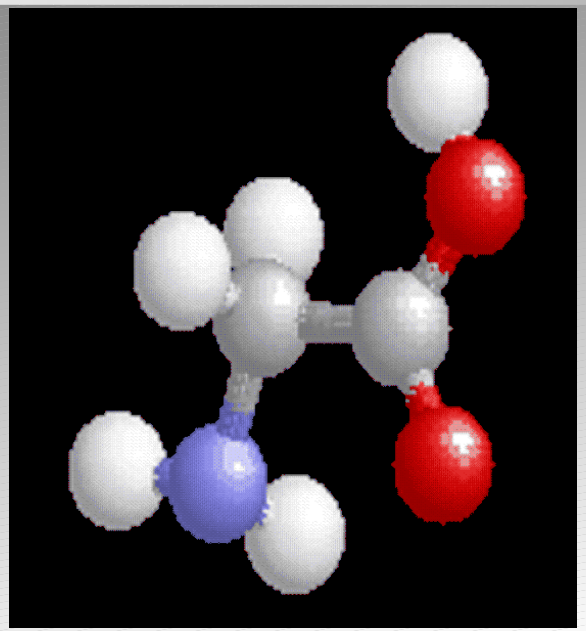
- A Protein is a polymer chain (a *list*) made of monomers (*aminoacids*).
- This list is called the *Primary Structure*.
- The typical length is 50–500.
- Aminoacids are of twenty types, called **Ala**nine (A), **Cys**teine (C), **Asp**artic Acid (D), **Glu**tamic Acid (E), **Phe**nylalanine (F), **Gly**cine (G), **His**tidine (H), **I**so**le**ucine (I), **Lys**ine (K), **Leu**cine (L), **Met**hionine (M), **As**paragi**n**e (N), **Pro**line (P), **Gl**utami**n**e (Q), **Arg**inine (R), **Ser**ine (S), **Thr**eonine (T), **Val**ine (V), **Tryp**tophan (W), **Tyr**osine (Y).
- *Summary:* The primary structure of a protein is a list of the form $[a_1, \ldots, a_n]$ with $a_i \in \{A, \ldots, Z\} \setminus \{B, J, O, U, X, Z\}$.

Aminoacid Structure



- The *backbone* is the same for all aminoacids.
- The *side chain* characterizes each aminoacid.
- Side chains contain from 1 (Glycine) to 18 (Tryptophan) atoms.

Example: Glycine and Arginine



$C_2H_5NO_2 \rightarrow$ 10 atoms

$C_6H_{14}N_4O_2 \rightarrow$ 26 atoms

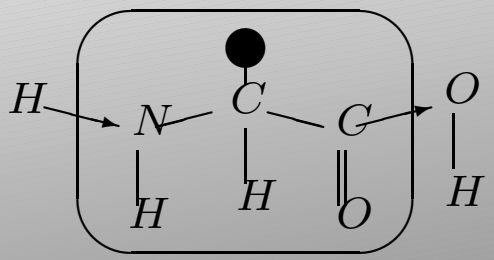
Remember the base scheme (9 atoms) $\Rightarrow$

White $=$ H

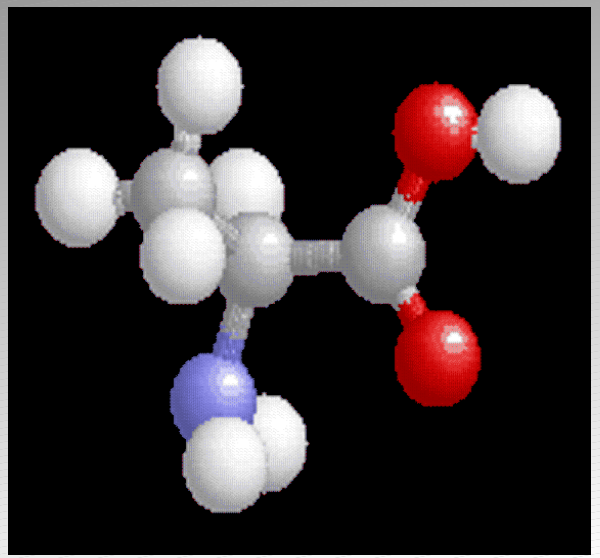
Blue $=$ N

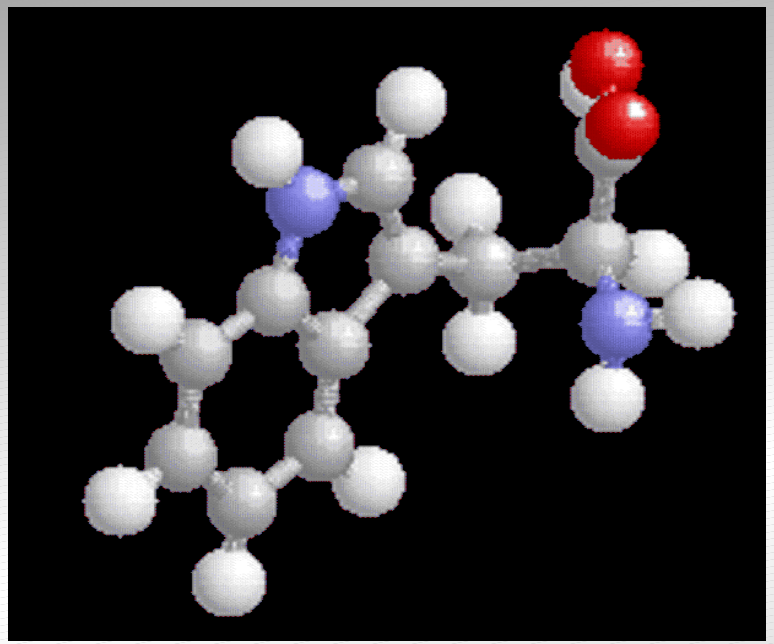
Red $=$ O

Grey $=$ C



Example: Alanine and Tryptophan

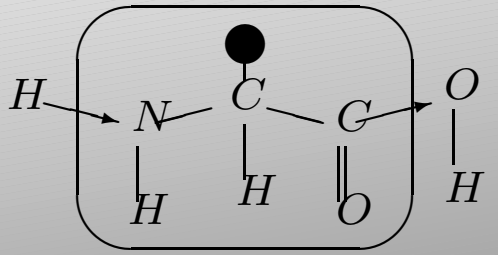




$C_3H_7NO_2 \rightarrow$ 13 atoms

$C_{11}H_{12}N_2O_2 \rightarrow$ 27 atoms

White $=$ H
Blue $=$ N
Red $=$ O
Grey $=$ C



Aminoacid's size

Name	Chemical	Side Chain
A	$C_3H_7NO_2$	4
C	$C_3H_7NO_2S$	4
D	$C_4H_7NO_4$	16
E	$C_5H_9NO_4$	10
F	$C_9H_{11}NO_2$	14
G	$C_2H_5NO_2$	1
H	$C_6H_9N_3O_2$	11
I	$C_6H_{13}NO_2$	13
K	$C_6H_{14}N_2O_2$	15
L	$C_6H_{13}NO_2$	13

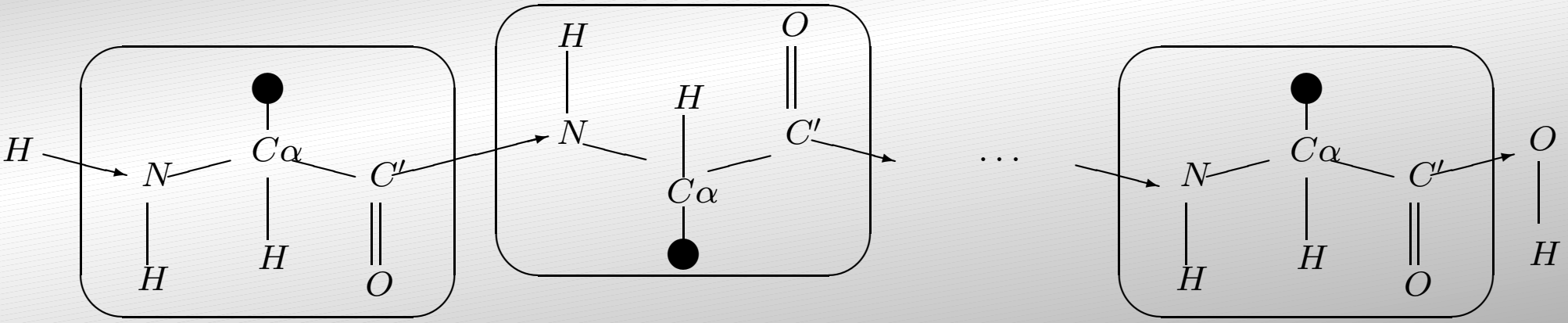
Name	Chemical	Side Chain
M	$C_5H_{11}NO_2S$	11
N	$C_4H_8N_2O_3$	8
P	$C_5H_9NO_2$	8(*)
Q	$C_5H_{10}N_2O_3$	11
R	$C_6H_{14}N_4O_2$	17
S	$C_3H_7NO_3$	5
T	$C_4H_9NO_3$	9
Y	$C_9H_{11}NO_3$	15
V	$C_5H_{11}NO_2$	10
W	$C_{11}H_{12}N_2O_2$	18

Images from:

`http://www.chemie.fu-berlin.de/chemistry/bio/amino-acids_en.html`

Primary Structure, detailed

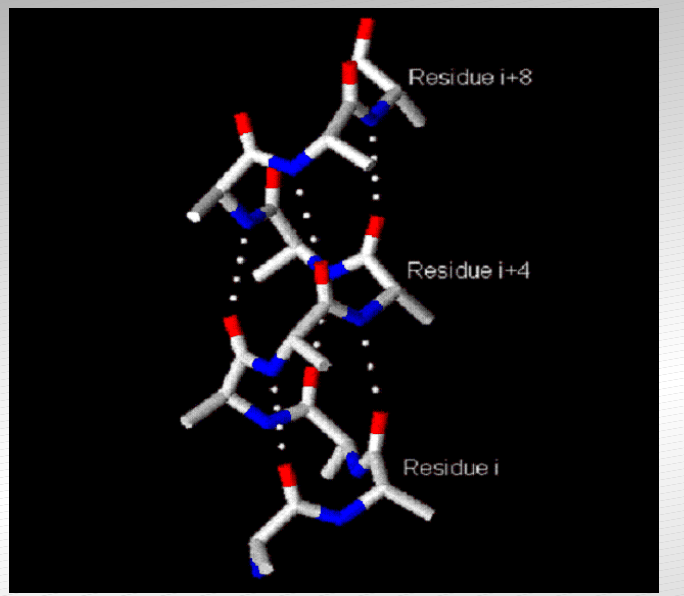
- The primary structure is a linked list of aminoacids.
- The terminals H (left) and OH (right) are lost in the linking phase.

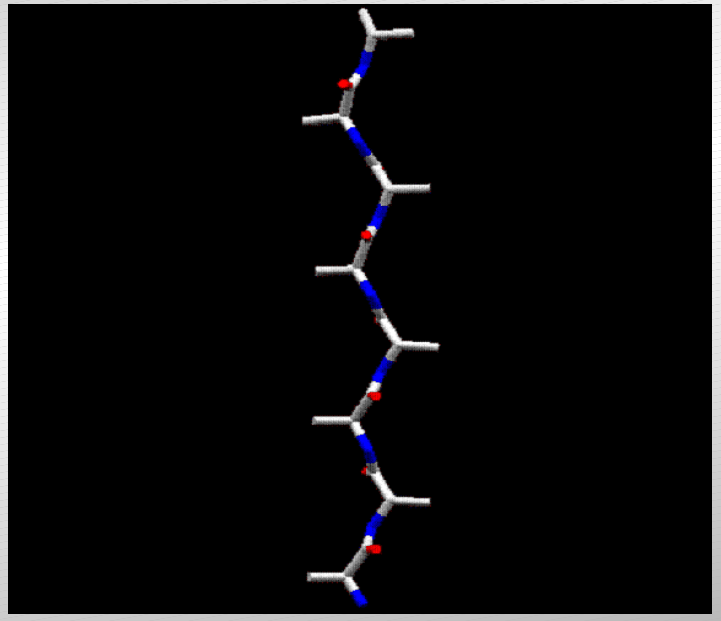


The Secondary Structure

- Locally, a protein *can* assume two particular forms:

  α-helix *β-sheet*





- This information is the *Secondary Structure* of a Protein.

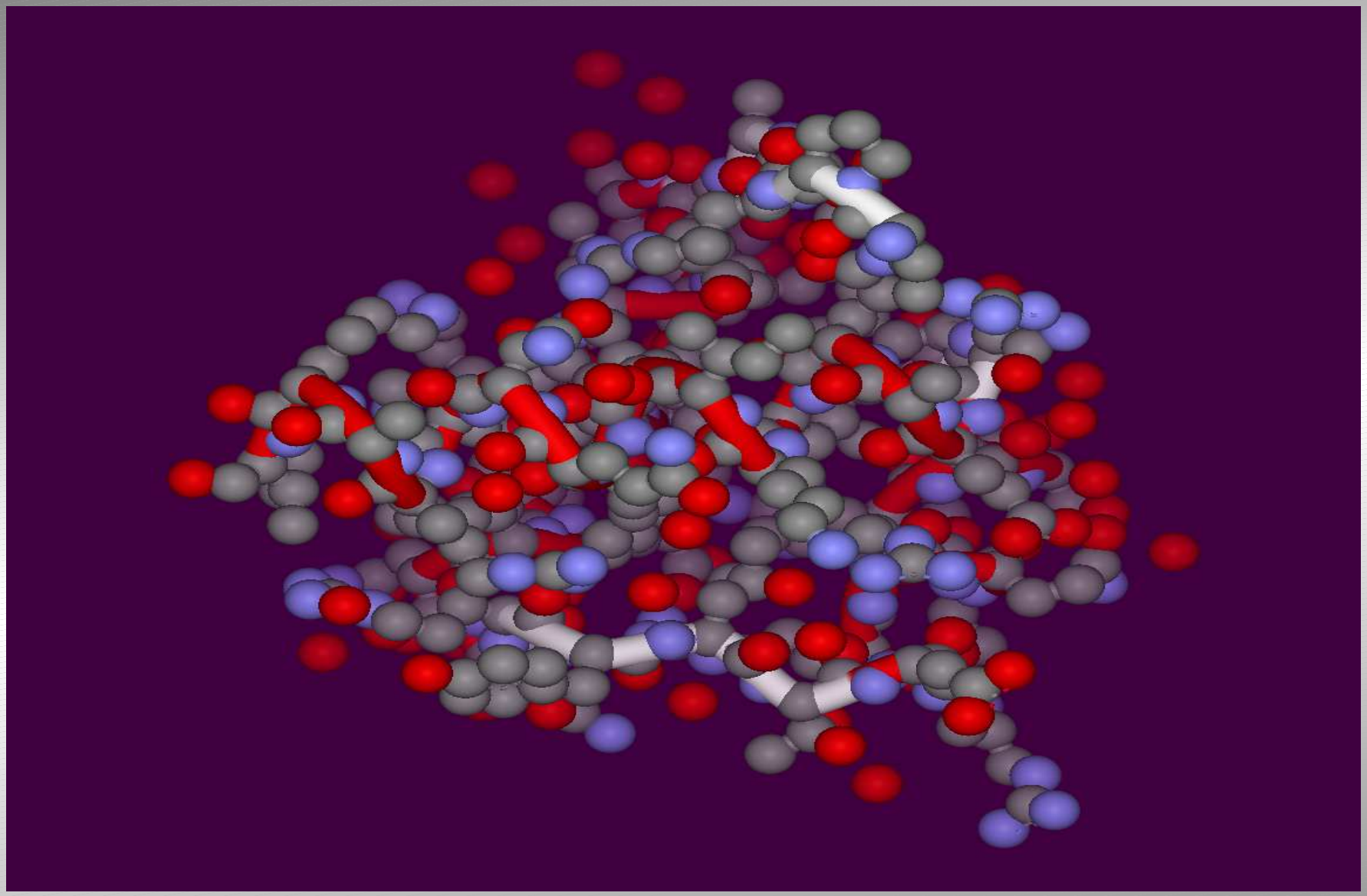
The Tertiary Structure

- The complete 3D conformation of a protein is called the *Tertiary Structure*.
- Proteins *fold* in a determined environment (e.g. water) to form a very specific geometric pattern (*native state*).
- The native conformation is relatively stable and unique and (*Anfinsen*'s hypothesis) is the state with minimum free energy.
- The tertiary structure determines the *function* of a Protein.
- $\sim$ 26000 structures (most of them redundant) are stored in the PDB.
- The number of possible proteins of length ≤ 500 is

$$20^1 + 20^2 + \cdots + 20^{500} = O(20^{501}) \sim 10^{651}$$

- The secondary structures is believed to form before the tertiary.

Example: Tertiary Structure of 1ENH



The Protein Folding Problem

- The *Protein Structure Prediction (PSP) problem* consists in predicting the Tertiary Structure of a protein, given its Primary Structure.
- The *Protein Folding (PF) Problem* consists in predicting the whole folding process to reach the Tertiary Structure.
- Sometimes the two problems are not distinguished.
- A reliable solution is fundamental for medicine, agriculture, Industry.
- Let us focus on the PSP problem, first.

The PSP Problem

- Anfinsen: the native state minimizes the whole protein energy. Two problems emerge.

1 Energy model:
   - What is the energy function $\mathbb{E}$?
   - It depends on what?

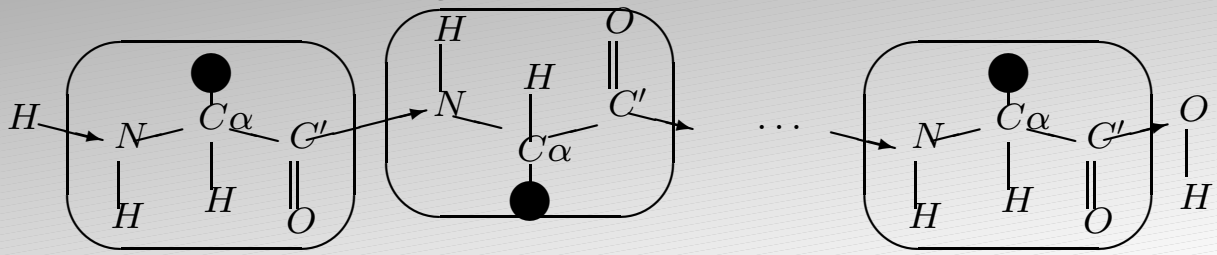
2 Spatial Model: Assume $\mathbb{E}$ be known, depending on the aminoacids $a_1, \ldots, a_n$ and on their positions, what is the search's space where looking for the conformation minimizing $\mathbb{E}$?
   - Lattice (discrete) models.
   - Off-lattice (continuous) models.

- After a solution/choice for (1) and (2) is available, we can try to study and solve the minimization problem

- If the solution's space is finite, a brute-force algorithm can be written.

The PSP as a minimization problem

- We give a general formal definition of the problem, under the *assumption* that each aminoacid is considered as a whole: a sphere centered in its $C\alpha$-atom.



- It emerges from experiments on the known proteins, that the distance between two consecutive $C\alpha$ atoms is fixed (3.8Å).

- Let $\mathcal{L}$ be the set of admissible points for each aminoacid.

- Given the sequence $a_1 \dots a_n$, a *folding* is a function

$$\omega : \{1, \dots, n\} \longrightarrow \mathcal{L}$$

such that:

  - next$(\omega(i), \omega(i+1))$ for $i = 1, \dots, n-1$, and
  - $\omega(i) \neq \omega(j)$ for $i \neq j$.

Objective function

- *Assumption*: the energy is the sum of the energy contributions of each pair of non-consecutive aminoacids.
- It depends on their distance and on their type. The contribution is of the form en_contrib(ω, i, j).
- The function to be minimized is therefore:

$$E(\omega) = \sum_{\substack{1 \leq i \leq n \\ i+2 \leq j \leq n}} \text{en\_contrib}(\omega, i, j)$$

- It is a constrained minimization problem (recall that: next$(\omega(i), \omega(i+1))$ and $\omega(i) \neq \omega(j)$).
- It is parametric on $\mathcal{L}$, next, and en_contrib.
- next and en_contrib are typically non linear.

A first proposal for the Energy: DILL

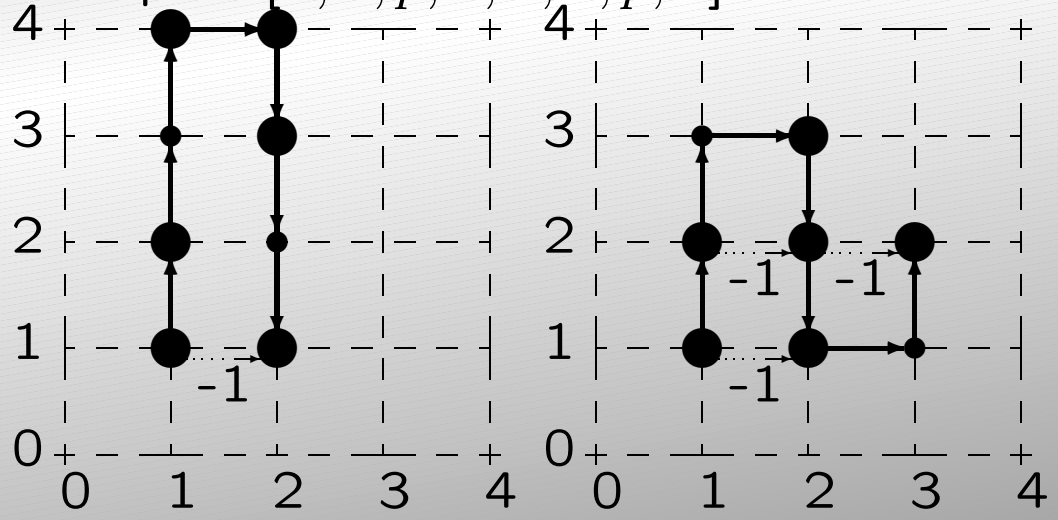
- The aminoacids: Cys (C), Ile (I), Leu (L), Phe (F), Met (M), Val (V), Trp (W), His (H), Tyr (Y), Ala (A) are *hydrophobic* (H).
- The aminoacids: Lys (K), Glu (E), Arg (R), Ser (S), Gln (Q), Asp (D), Asn (N), Thr (T), Pro (P), Gly (G) are *polar* (P).
- The protein is in water: hydrophobic elements tend to occupy the center of the protein.
- Consequently, H aminoacids tend to stay close each other.
- polar elements tend to stay in the frontier.

A first proposal for the Energy: DILL

- This fact suggest an energy definition: if two aminoacids of type H are *in contact* (i.e. no more distant than a certain value) in a folding they contribute negatively to the energy.
- The aminoacid is considered as a whole: a unique sphere centered in its $C\alpha$ atom.
- The notion of being *in contact* is naturally formalized in *lattice models*: one (or more) *lattice units*.

The simplest PFP formalization

- The spatial model is a subset of $\mathbb{N}^2$.
- A *contact* is when $|X_1 - X_2| + |Y_1 - Y2| = 1$.
- The primary list is a sequence of h and p.
- Each contact between pairs of h contributes as -1.
- We would like to find the folding(s) minimizing this energy
- Example: $[h, h, p, h, h, h, p, h]$



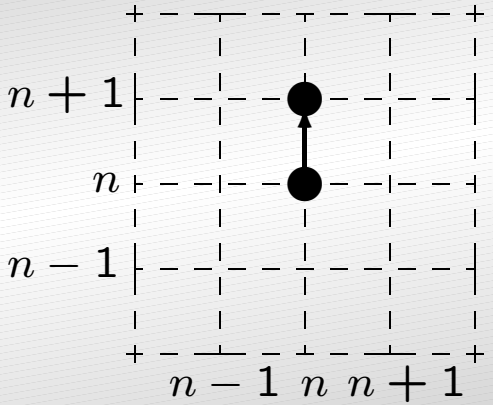
Unfortunately, the decision version:
Is there a folding with Energy $< k$?
is *NP-complete*

HP on $\mathbb{N}^2$

- If the primary structure is $[a_1, \ldots, a_n]$ with $a_i \in h, p$, then

$$\omega(i) \in \mathcal{L} = \{(i,j) : i \in [1..2n-1], j \in [1..2n-1]\}$$

- W.l.o.g, we can assume that $\omega(1) = (n, n)$.
- To avoid simple symmetries, w.l.o.g., we can assume that $\omega(2) = (n, n+1)$.



- We need to implement *next*, *en_contrib*, ...
- Let us see a simple (and working) $CLP(\mathcal{FD})$ code.

Encoding the PF, HP, on $\mathbb{N}^2$

```
pf(Primary, Tertiary) :-        %%% Primary = [a1,...,aN], ai in {h,p}
        constrain(Primary,Tertiary,Energy),
        labeling([ff,minimize(Energy)],Tertiary).

constrain(Primary,Tertiary,Energy) :-
        length(Primary,N),
        M is 2*N, M1 is M - 1,
        length(Tertiary,M),  %%% Tertiary = [X1,Y1,...,XN,YN]
        domain(Tertiary,1,M1),
        starting_point(Tertiary,N),
        avoid_loops(Tertiary),
        next_constraints(Tertiary),
        energy_constraint(Primary,Tertiary,Energy).
```

Encoding the PF, HP, on $\mathbb{N}^2$

```
starting_point([N,N,N,N1|_],N) :-  %%% X1=Y1=X2=N, Y2=N+1
      N1 is N + 1.

avoid_loops(Tertiary):-
       positions_to_integers(Tertiary, ListaInteri),
       all_different(ListaInteri).

positions_to_integers([X,Y|R], [I|S]):-
       I  #= X*100+Y,    %%% 100 is a "large" number
       positions_to_integers(R,S).
positions_to_integers([],[]).
```

This way, we do not introduce a disjunction

$$X_i \neq X_j \vee Y_i \neq Y_j$$

for each constraint

$$(X_i, Y_i) \neq (X_j, Y_j)$$

Encoding the PF, HP, on $\mathbb{N}^2$

```
next_constraints([_,_]).
next_constraints([X1,Y1,X2,Y2|C]) :-
        next(X1,Y1,X2,Y2),
        next_constraints([X2,Y2|C]).

next(X1,Y1,X2,Y2):-
        domain([Dx,Dy],0,1),
        Dx #= abs(X1-X2),
        Dy #= abs(Y1-Y2),
        Dx + Dy #= 1.
```

Note: a non linear constraint.

Encoding the PF, HP, on $\mathbb{N}^2$

```
energy_constraint(Primary,Tertiary,Energy):- ...
```

is defined recursively so as to fix

$$\texttt{Energy \#=} C_{1,3} + C_{1,4} + \cdots + C_{1,N} + C_{2,4} + \cdots + C_{2,N} + \vdots + C_{N-2,N}$$

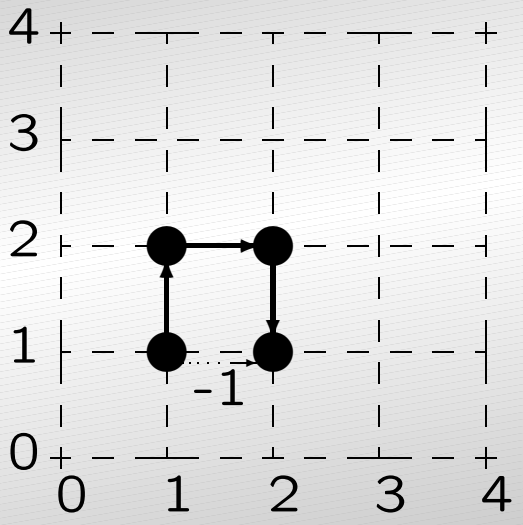
Where each $C_{A,B}$ is defined as follows:

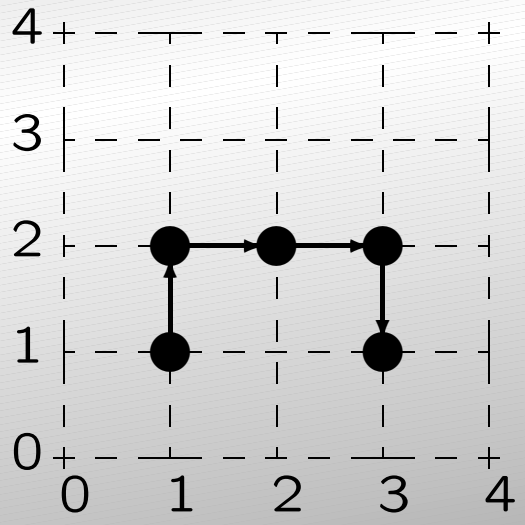
```
energy(h,XA,YA,h,XB,YB,C_AB) :-
        C_AB in {0,-1},
        DX #= abs(XA - XB),
        DY #= abs(YA - YB),
        1 #= DX + DY  #<=> C_AB #= -1.
energy(h,_,_,p,_,_,0). energy(p,_,_,h,_,_,0). energy(p,_,_,p,_,_,0).
```

Note: a reified, non linear constraint.

Constraint Optimization

- This basic code is a good starting point for Optimization.
- A first idea concerns the objective function `Energy`.
- Only aminoacids at an *odd* relative distance can contribute to the Energy.





- Proof: think to the offsets at each step.

Constraint Optimization

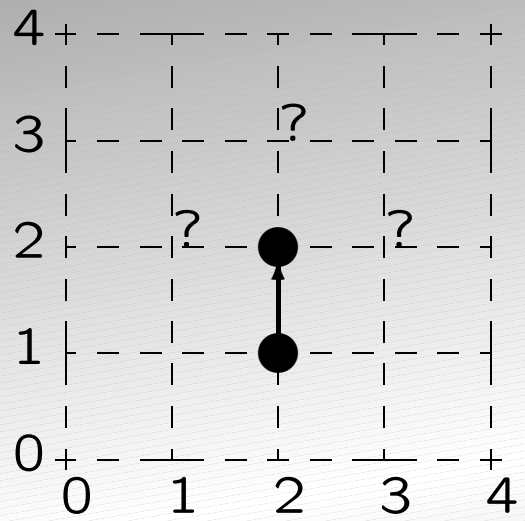
- Thus,

$$\texttt{Energy \#=} \underbrace{C_{1,3}}_{=0} + C_{1,4} + \underbrace{C_{2,4}}_{=0} + \cdots + C_{1,N} +$$
$$\underbrace{C_{2,4}}_{=0} + C_{2,5} + \cdots + C_{2,N} +$$
$$\vdots$$
$$+ \underbrace{C_{N-2,N}}_{=0}$$

- Speed up 3×.

Constraint Optimization

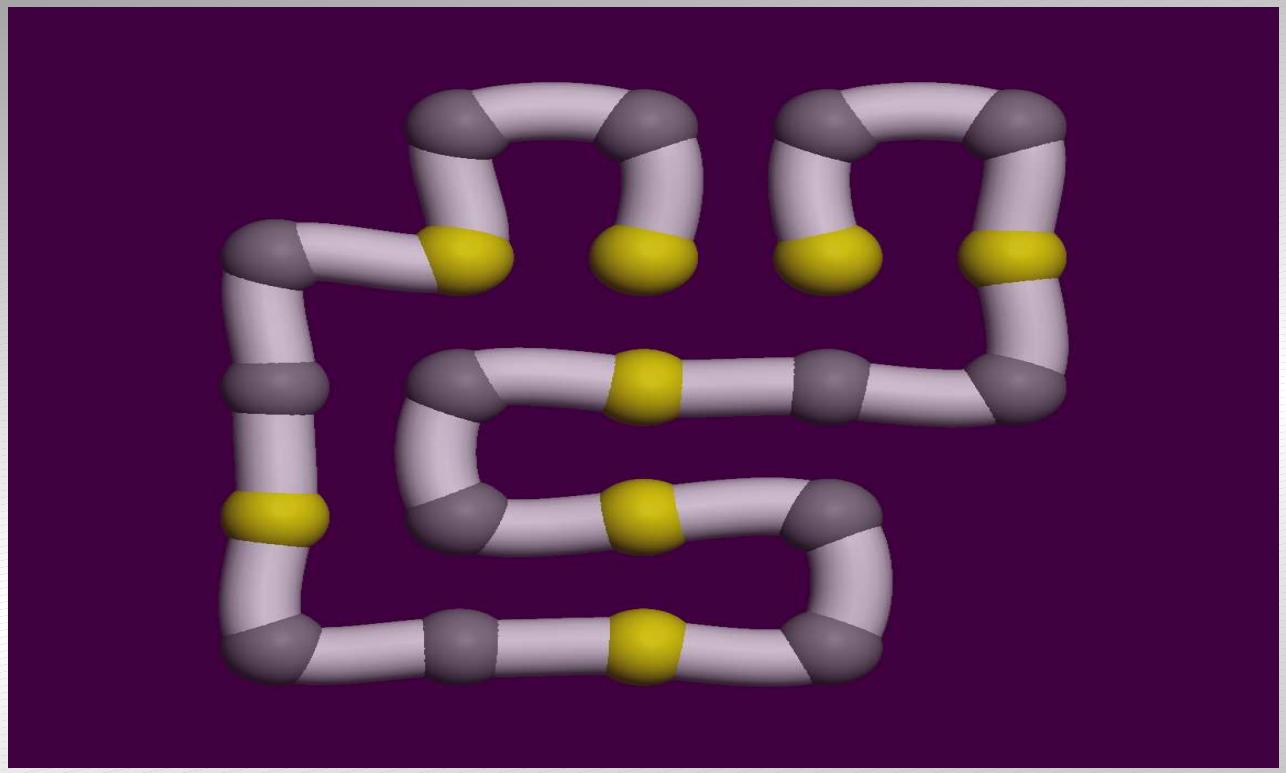
- The next element can (at the beginning) take 3 positions.
- Space looks as $\sim 3^n$
- Precisely, it is $\sim 1.28\frac{2.64^n}{n^{0.34}}$.
- We can reduce it by reducing the domains and/or adding linear distance constraints on pairs of aminoacids.



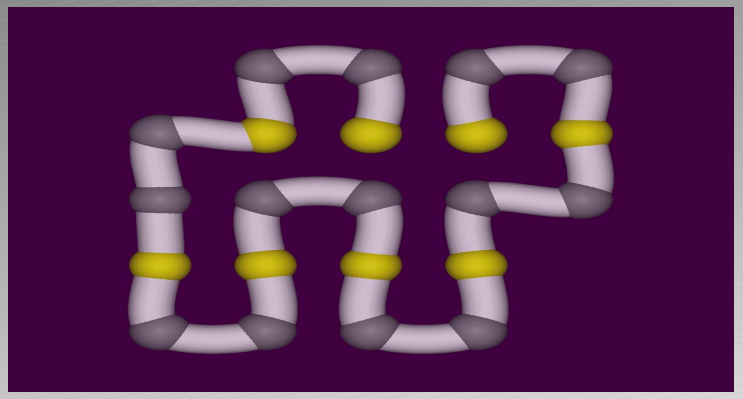
- Typically, in the solutions, the offsets are of the order of $2\sqrt{N}$ (for real proteins there are some more precise formulae).
- Speed up 20×.
- Further speed up? Avoid Symmetries (easy in this case). Smarter constraint propagation, using indexicals (in SICStus Prolog) or CHR.

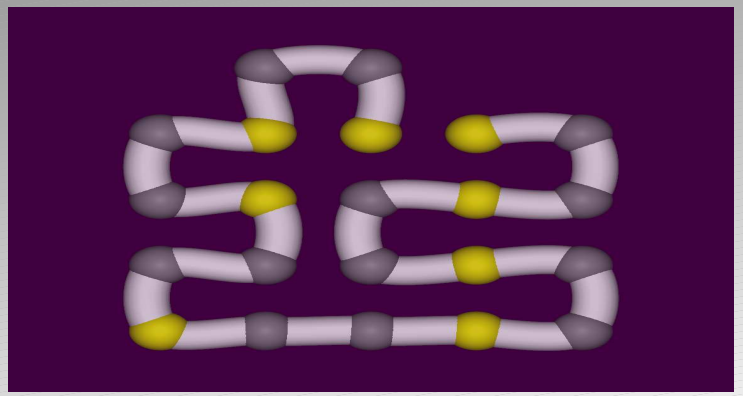
Example

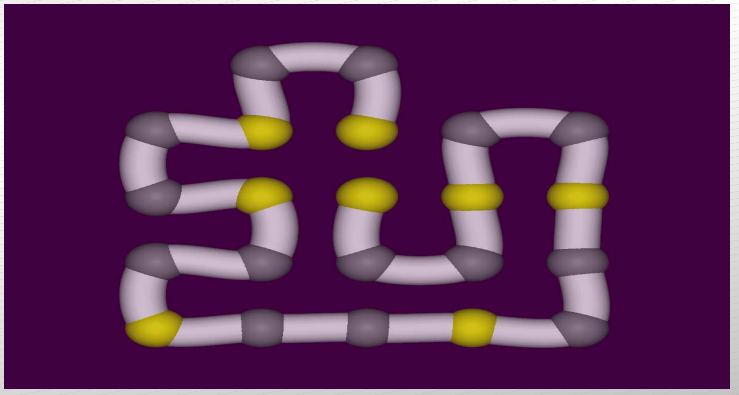
`:-pf([h,p,p,h,p,p,...,h,p,p,h],L)`, $n = 22$. 48 s., Energy = -6.

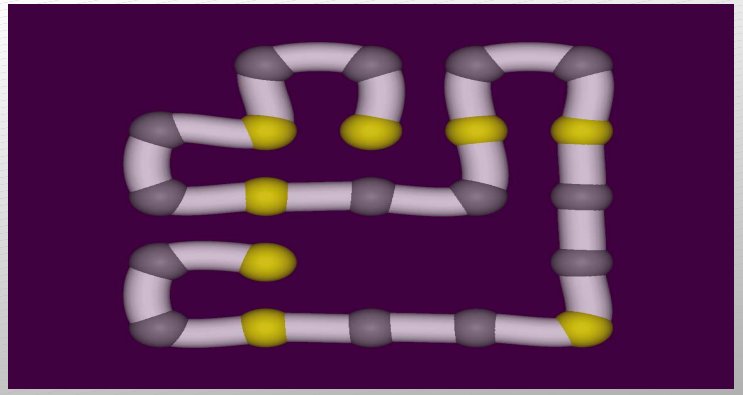


Example



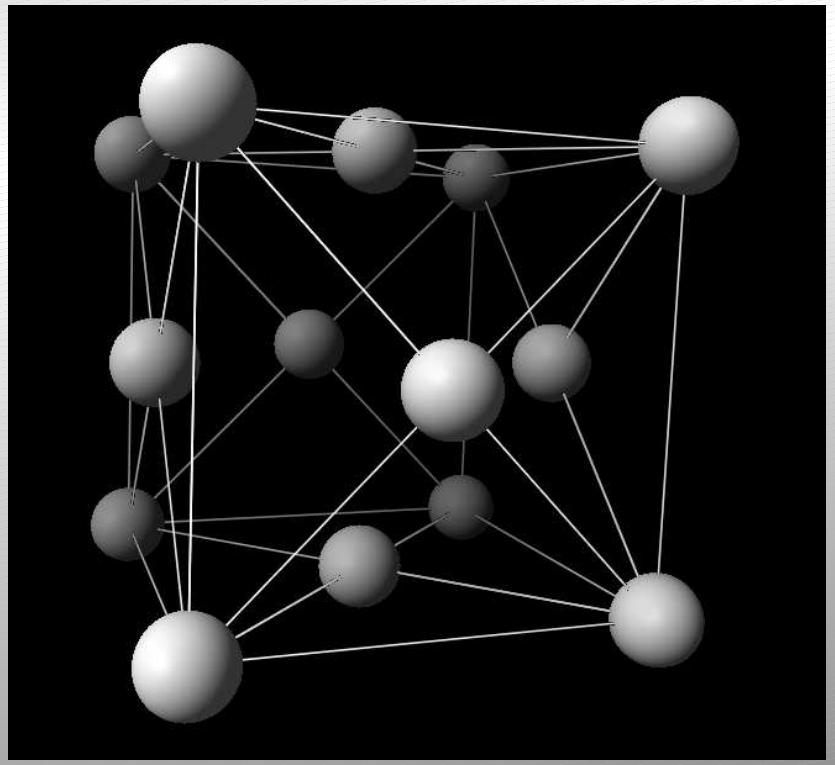






A more realistic space model

- The *Face Centered Cube lattice* models the discrete space in which the protein can fold.
- It is proved to allow realistic conformations.
- The cube has size 2.
- Two points are *connected* (next) iff $|x_i - x_j|^2 + |y_i - y_j|^2 + |z_i - z_j|^2 = 2$,
- Each point has 12 neighbors and $60°, 90°, 120°$ and $180°$ bend angles are allowed (in nature $60°$ and $180°$ never occur).



HP on FCC: Main Results

- A $Constraint(\mathcal{FD})$ program in Mozart by Backofen-Will folds HP-proteins up to length 150!
- Clever propagation, an idea of stratification and some geometrical results on the lattice.
- Drawbacks: It is only an abstraction. The solutions obtained are far from reality. For instance, helices and sheets are never obtained.
- Problems:
  - Energy function too simple.
  - Contact too strict.

A more realistic Energy function

- Same assumption: only pairs of aminoacids in *contact* contribute to the energy value.
- The notion of *contact* is easy on lattice models.
- There is a *potential matrix* storing the contribution for each pair of aminoacids.
- Values are either positive or negative.
- The global energy must be minimized.

PF, 20 aminoacids, on $\mathbb{N}^2$

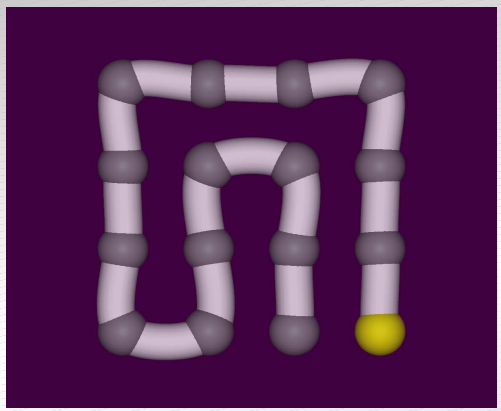
- Basically, the same code, with a call to `table(A,B,Cost)`.

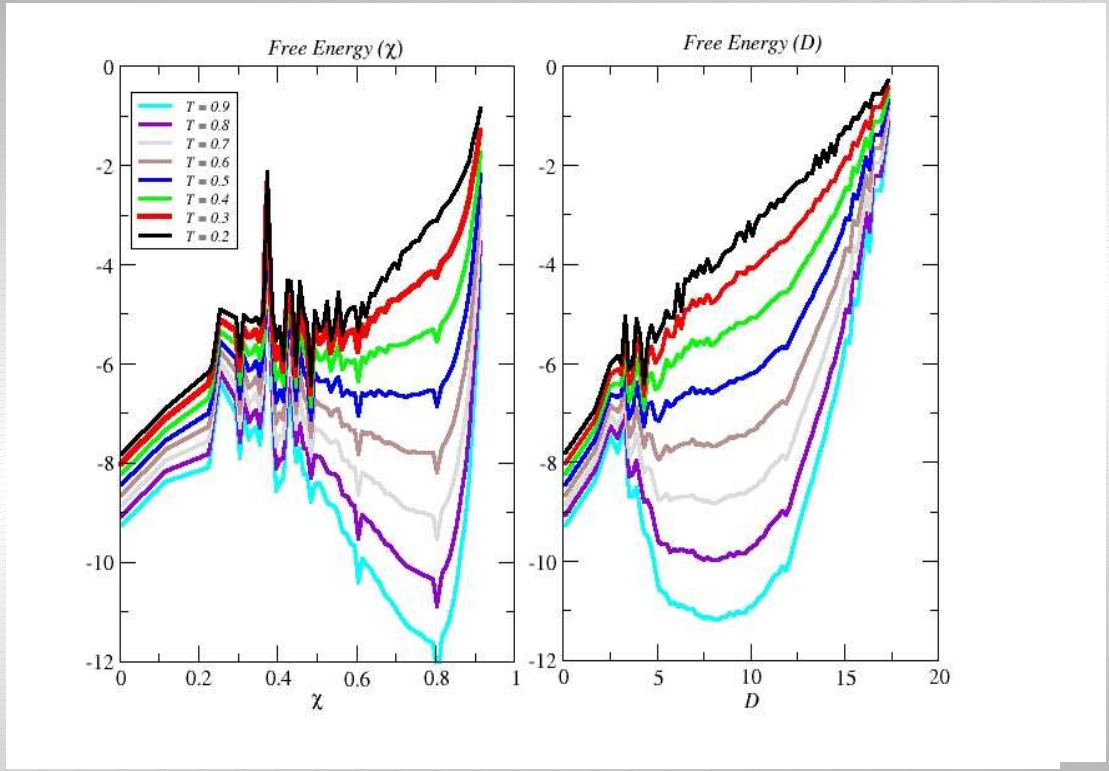
```
energy(A,XA,YA,B,XB,YB,C) :-
        table(A,B,Cost),
        (Cost #\= 0,!,
        C in {0,Cost},
        DX #= abs(XA - XB),
        DY #= abs(YA - YB),
        1 #= DX + DY  #<=> C #= Cost;
        C #= 0).
```

- Potentials by Kolinsky and Skolnick.
- Or by Miyazawa and Jernigan,
- refined by Berrera, Fogolari, and Molinari.

Example

`:- pf([s,e,d,g, d,l,p,i, v,a,s,f, m,r,r,d],L).`, $n = 16$.
0.9 s., Energy = -7.4. (search space: 6.416.596)



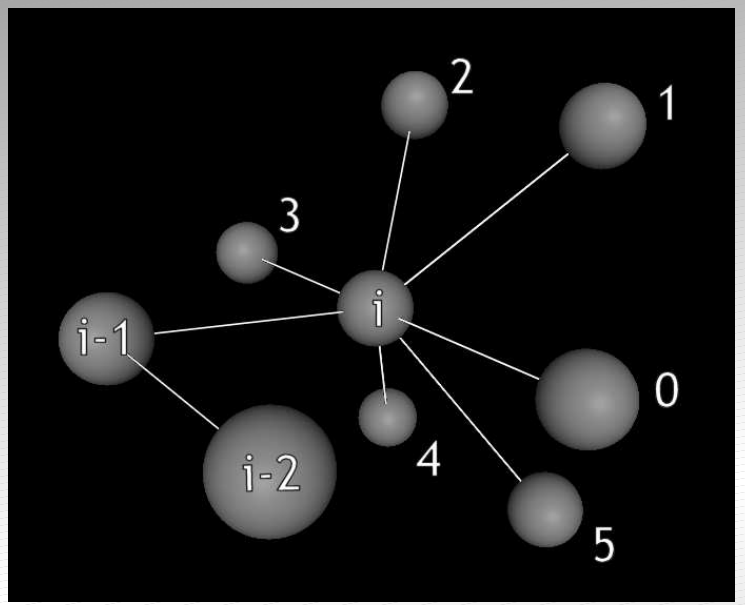


PF, 20 aminoacids, on FCC

- The solution's space is huge $\sim 1.26n^{0.162}(10,03)^n$.
- The potential table is 20×20.
- Contact is set when $|X_1 - X_2| + |Y_1 - Y_2| + |Z_1 - Z_2| = 2$
- New constraints (secondary structure) are needed.
- A careful tratment of the energy function as a matrix that statically (e.g. if two aminoacids s_i, s_j belong to the same α-helix, then $M[i,j] = 0$) and dynamically set to 0 most of its elements.
- Some heuristics for pruning the search tree (loosing completeness!)

Using the Secondary Structure

- A *local coordinate system* can be used to describe the torsional relationships θ for 4 consecutive aminoacids.



- Each torsion is represented by an *Index* $\theta(i)$.

- The set of indexes is independent from the orientation and it is a bridge between the Secondary and Tertiary Structures. Conversion between Indexes and coordinates is done using reified constraints.

- By analysis on deposited proteins, only 6 values are admitted.

Using the Secondary Structure

- An *α-helix* is represented by a θ sequence of 5-5-5-$\cdots$
- A *β-strand* is represented by a θ sequence of 3-3-3-$\cdots$
- Secondary structure can be predicted with high accuracy: we can use these constraints.
- Moreover, *ssbonds* are induced by aminoacids: **Cys**teine ($C_3H_7NO_2S$) and **Met**hionine ($C_5H_{11}NO_2S$).
- A ssbond constrains the two aminoacids involved to be close in the Tertiary Structure: $|X_i - X_j| + |Y_i - Y_j| + |Z_i - Z_j| \leq 6$.

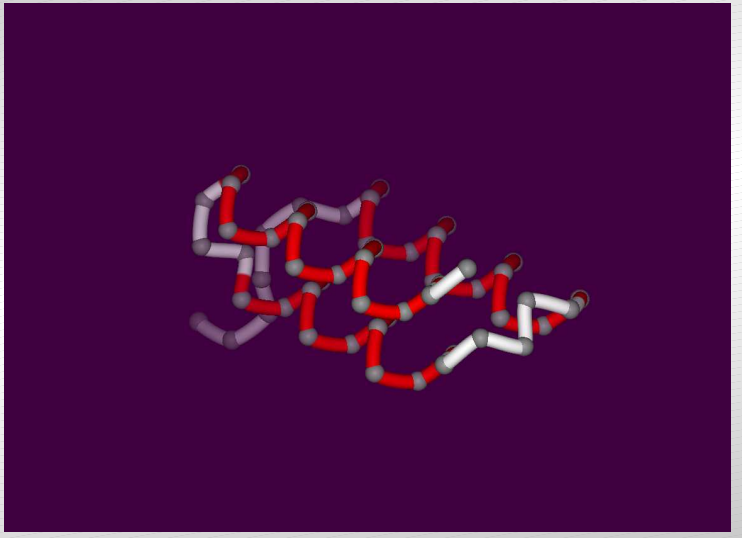
Labeling heuristics

- Working on *subsequences* of the protein, possibly including already folded runs of aminoacids, s.t. the problem size is tractable.
- *Local labeling* technique: every k instantiations (ff-leftmost), compare the current ground contributions of M to the best known ones.
- We allow a compact factor from user (default computed with an empirical formula) and we allow a time limit.
- We use again `clpfd` of SICStus Prolog—code in:
  `http://www.dimi.uniud.it/dovier/PF`

Example

- Proteins of length 60 can be predicted in some hours, with acceptable RMSD.

```
protein('1ENH', Primary, Secondary):-
   Primary = [r,p,r,t,a,f,s,s,e,q,
              l,a,r,l,k,r,e,f,n,e,
              n,r,y,l,t,e,r,r,r,q,
              q,l,s,s,e,l,g,l,n,e,
              a,q,i,k,i,w,f,q,n,k,
              r,a,k,i],
   Secondary = [helix(8,20),helix(26,36),
                helix(40,52), strand(22,23)].
```



To do

- Working on Constraint Propagation
- In particular, building constraint solver algorithms for lattice models.
- Of course parallelism can help and it is rather natural parallelize Prolog search.
- Other ideas (see next talk...)
- Integration with other approaches.

Constraint-based Simulation Approach

- Molecular dynamics analyzes atom-atom interactions and computes forcefields.
- Then uses the forcefield for a global simulation move.
- The number of atoms is huge (7–24 per aminoacid) and computations involve solutions of differential equations.
- There are working tools, but detailed simulations of real-size proteins are not yet applicable.
- Moreover, it is easy to fall into local minima, and
- It seems that the folding follow more macroscopical laws.

Concurrent Constraint Simulation Approach

- *Idea:* perform simulations at higher abstraction level (aminoacids) using concurrent constraint programming.
- Basically, each aminoacid is an independent agent that communicates with the others.
- Motion follows some rules, governed by energy.
- More refined energy model (w.r.t. optimization).
- Off-lattice spatial model.

Energy Function

- The Energy Function used is

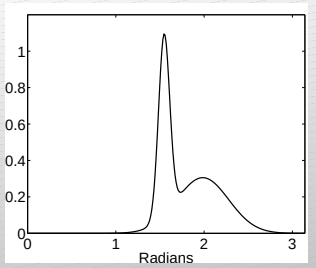
$$E(\vec{s}) = \eta_b\, E_b(\vec{s}) + \eta_a\, E_a(\vec{s}) + \eta_t\, E_t(\vec{s}) + \eta_c\, E_c(\vec{s})$$

- $E_b(\vec{s})$ is the Bond Distance term

$$E_b(\vec{s}) = \sum_{1 \le i \le n-1} \Big(r(s_i, s_{i+1}) - r_0 \Big)^2$$

- $E_a(\vec{s})$ is the Bond Angle Bend term

$$E_a(\vec{s}) = \sum_{i=1}^{n-2} -\log\left(a_1\, e^{-\left(\frac{\beta_i - \beta_1}{\sigma_1}\right)^2} + a_2\, e^{-\left(\frac{\beta_i - \beta_2}{\sigma_2}\right)^2} \right)$$



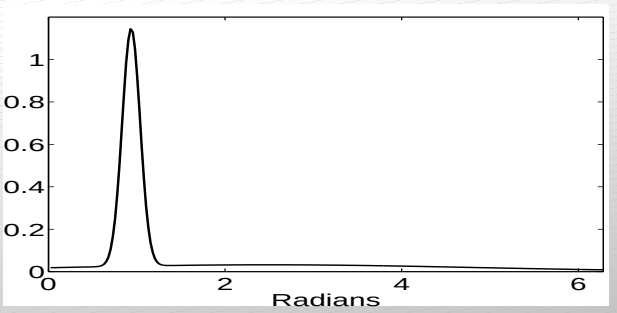
Energy Function

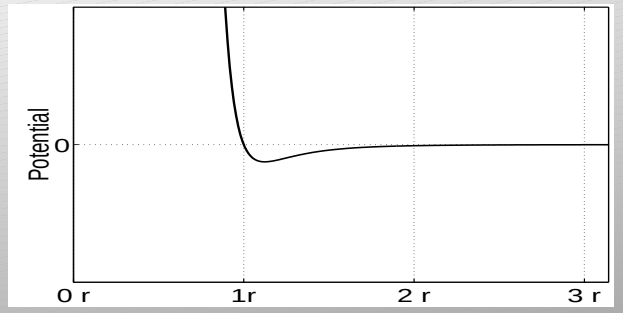
- $E_t(\vec{s})$ is the Torsional Angle term

$$E_t(\vec{s}) = \sum_{i=1}^{n-3} -\log\left(a_1\, e^{\frac{(\Phi_i-\phi_1)^2}{(\sigma_1+\sigma_0)^2}} + a_2\, e^{\frac{(\Phi_i-\phi_2)^2}{(\sigma_2+\sigma_0)^2}}\right)$$

- $E_c(\vec{s})$ is the Contact Interaction term

$$E_c(\vec{s}) = \sum_{i=1}^{n-3}\sum_{j=i+3}^{n}\left[|\mathrm{Pot}(s_i,s_j)|\left(\frac{r_0(s_i,s_j)}{r(s_i,s_j)}\right)^{12} + \mathrm{Pot}(s_i,s_j)\left(\frac{r_0(s_i,s_j)}{r(s_i,s_j)}\right)^{6}\right]$$





Communication Strategy

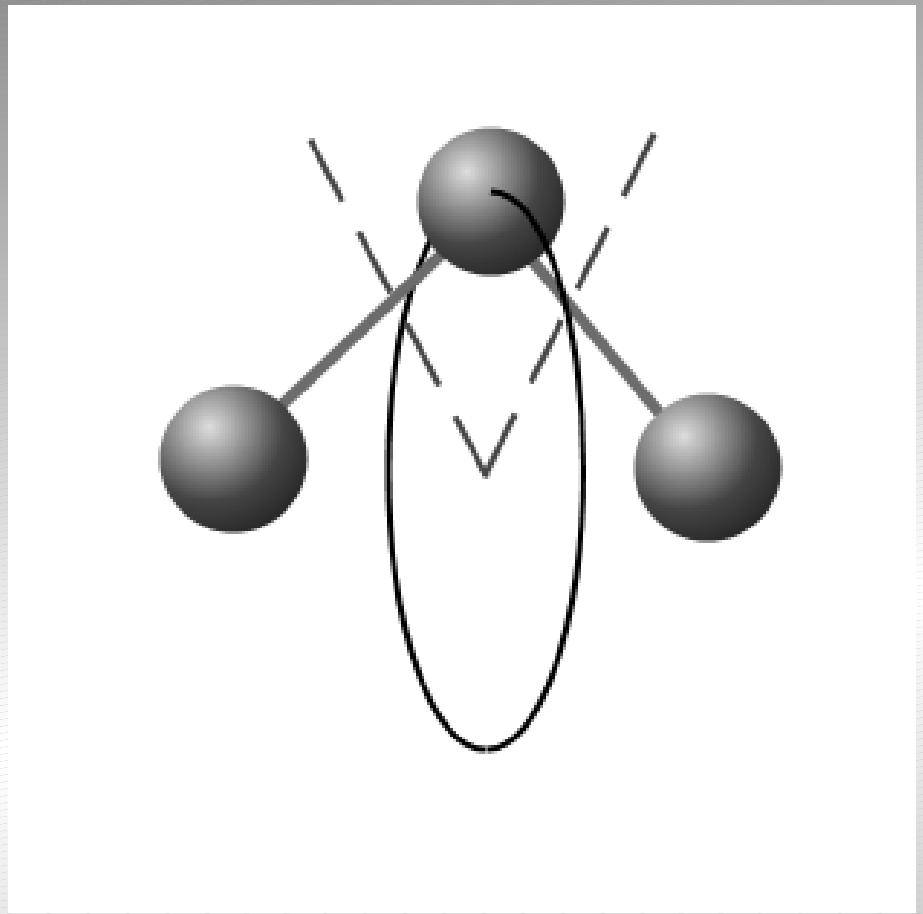
- Each agent, before performing a move, waits for the communication of a movement of some other aminoacid.
- The information of position changes of agent P_i is stored in a list L_i of logic terms (leaving the tail variable uninstantiated), thus keeping track of the entire history of the folding known to him.
- Each agent, while moving, uses the most recent information available to him, i.e. the last ground terms of the lists L_i.
- Each agent, once it has moved, communicates to all other agents its new position updating its list.

Moving Strategy

Each aminoacid performs a move in the following way:

- It randomly chooses a new position, close to the current one within a given step.
- Using the most recent information available about the spatial position of other agents, it computes the energy relative to the choice.
- It accepts the position using a Montecarlo criterion:
  - If the new energy is lower than the current one, it accepts the move.
  - If the new energy is greater than the current one, it accepts the move with probability $e^{-\frac{E_{new}-E_{current}}{T}}$.

Moving Strategy



- The new position is randomly selected in the following way:
  - He calculates the set of points which keep fixed the distance with the adjacent neighbours of the aminoacid (a circumference or a sphere).
  - He randomly select a point in this set, close to the current position.
  - He randomly select a small offset from this point.

The CCP simulation program

- ```
  simulation(S):-
        Init=[[I1|_],
              [I2|_],
              ...,
              [In|_]],
        run(1,S,Init) ||
        run(2,S,Init) ||
          ... ||
        run(n,S,Init).
  ```

- `Init` contains n lists with the initial positions `Ii`.

- The positions of the aminoacids are stored in a list, with a non-instantiated tail variable (_).

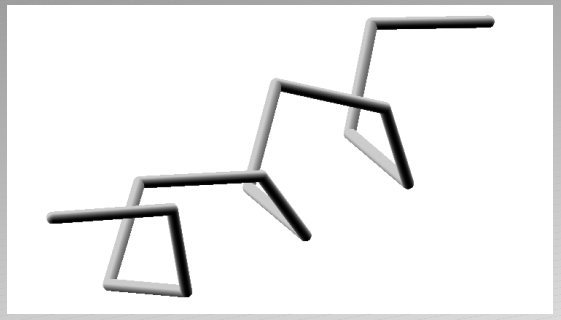
- The updating of the list is made substituting to this variable the new position and another unbounded variable (`_=[pos|_]`).
  ```

The CCP simulation program

- ```
  run(ID, S, [P1, P2, ..., Pn]):-
      getTails([P1, ..., Pn],[T1, ..., Tn]),
      ask(T1=[_|_]) -> skip + ... + ask(Tn=[_|_]) -> skip,
      getLast([P1, ..., Pn],[L1, ..., Ln]),
      updatePosition(ID,S,[L1,..,Ln],NP),
      tell(TID=[NP|_]),
      run(ID,S,[P1, ..., Pn]).
  ```

- `ID` is the identification code of the aminoacid
- getTails assigns to `Ti` the tails of the lists `Pi`
- Then the process waits for one of these variables to be instantiated by asking if `Ti` is a non-variable list.
- Once this happens it retrieves the last information with `getLast` and then updates its position with `updatePosition`.
- Finally, it communicates its move to all other processes
  ```

Results



- We the current parameters we are able to obtain an helix from 14 alanines residues in almost 10 seconds, although some tested proteins does not stay in their native state.
- A non–concurrent C simulation, with the same energy function and with the same moves, starting from the same position, takes very long time to fold into a helix.

To do

- new *communication strategy* to optimize the quantity of messages passed.
- a set of *cooperative* and *adaptive strategies*, to model the dynamic evolution of the system.
- We also want to implement a more detailed representation of the aminoacids, including the side chain as an ellipsoid and changing the energy function into a more tested one, i.e. the **UNRES** potential

Other approaches

- Of course the problem is faced with various methods (see [Neumaier97] for a review for mathematicians)
- Those with best results are based on *homology* and *threading*.
- In `http://www.rcsb.org/pdb/` 26.000 structures are deposited (not all independent!)
- One can look for a *homologous* protein (small changes/removals/insertions between the primary structures).
- If it is found, its tertiary structure is selected and weakly re-arranged (threading phase) for the new protein.

Other approaches ... can use constraints

- If an homologous protein is not found, but various subsequences are found in the PDB,
- each of those sequences is associated to a rigid local structure.
- The problem is reduced to the optimization of the energy assigning the positions to the unknown parts. This is similar of what is done using secondary structure constraints!
- Molecular Dynamics methods are precise but slow and can fall into local minima. They can start from a FCC Constraint-Based prediction!

Conclusions

- We have seen the definition of the Protein Structure Prediction Problem.
- We focused on simplified models (of space and energy).
- We have seen two uses of Constraint (logic/concurrent) programming for attacking it:
  - As a constrained minimization problem
  - As (abstract) simulation.
- The constraint approaches can be integrated in existing tools.
- A natural and useful application of declarative programming.
- There is a lot of work to do.

I am pleased to have worked on PFP with

- Rolf Backofen
- Luca Bortolussi
- Alessandro Dal Palù
- Federico Fogolari
- Sebastian Will
- Matteo Burato
- Sandro Bozzoli
- Fausto Spoto

Suggested Readings

- P. Clote and R. Backofen. *Computational Molecular Biology: An Introduction*. John Wiley & Sons, 2001.
- A. Neumaier. Molecular modeling of proteins and mathematical prediction of protein structure. *SIAM Review*, 39:407–460, 1997.
- 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.
- A. Kolinski and J. Skolnick. Reduced models of proteins and their applications. *Polymers* 45:511-524, 2004.
- T. Veitshans, D. Klimov, and D. Thirumalai. Protein folding kinetics: timescales, pathways and energy landscapes in terms of sequence-dependent properties. *Folding & Design*, 2:1–22, 1996.