# An Unified Approach for Structures alignment

R. Andonov and Nicola Yanev

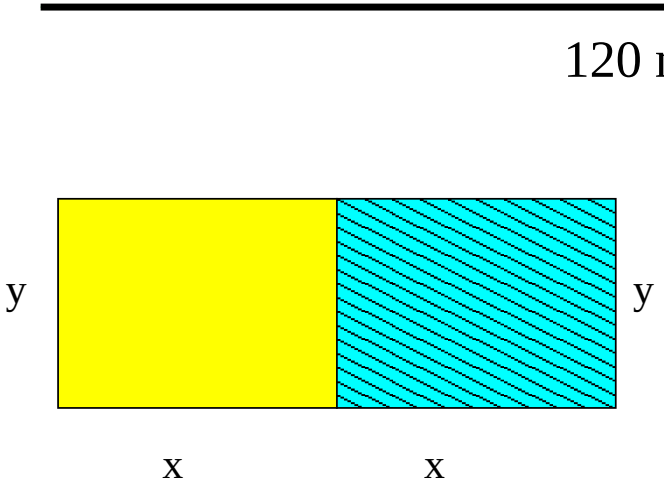
Ecole chercheurs Bioinformatique, Liffré, Novembre, 2005

# Outline

- Succinct introduction to Integer programming
- Two applications
  - Protein Folding Problem
  - Proteins Structure Comparison problem
- Building graph theoretical formalism
- Using Integer Programming for problems resolving
  - general purpose approaches
  - dedicated approaches
- Conclusions, perspectives, open problems

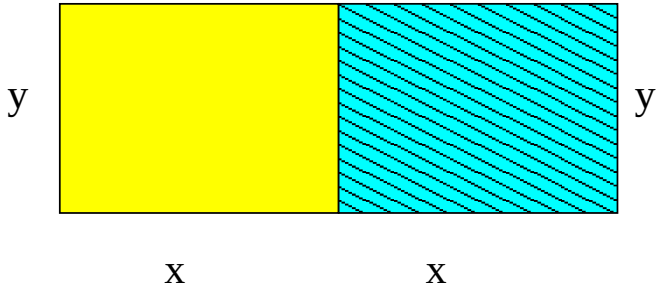
# *Continuous vs. Integer programming*

A simple example: Build 2 equal rectangular enclosures of maximal area size from 120 m.bar



# Continuous vs. Integer programming

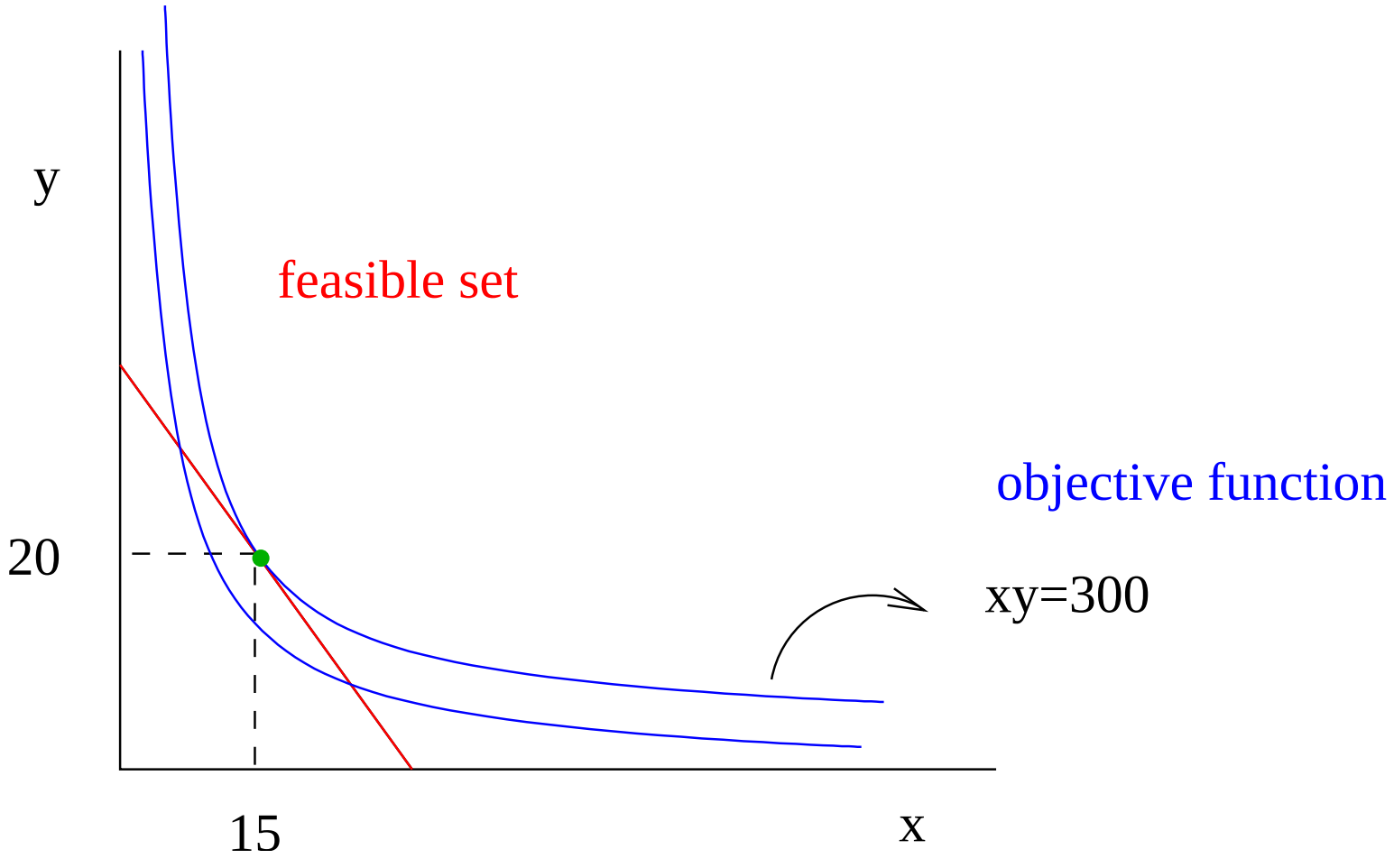
120 meters



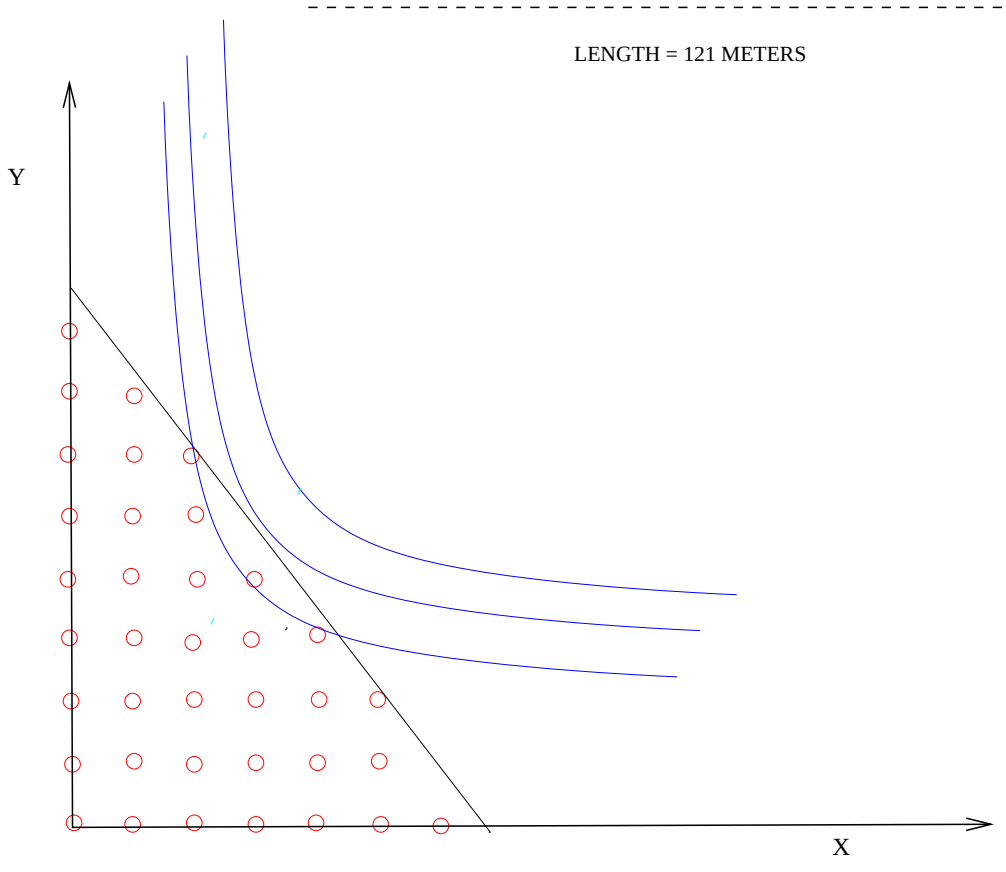

$xy \rightarrow \max$ ( objective function)

$4x + 3y = 120,\ x \geq 0,\ y \geq 0$ ( constraints)

# *some geometry in action*



# *integer lengths case: the feasible set*




THE CASE OF INTEGER ENCLOSURES

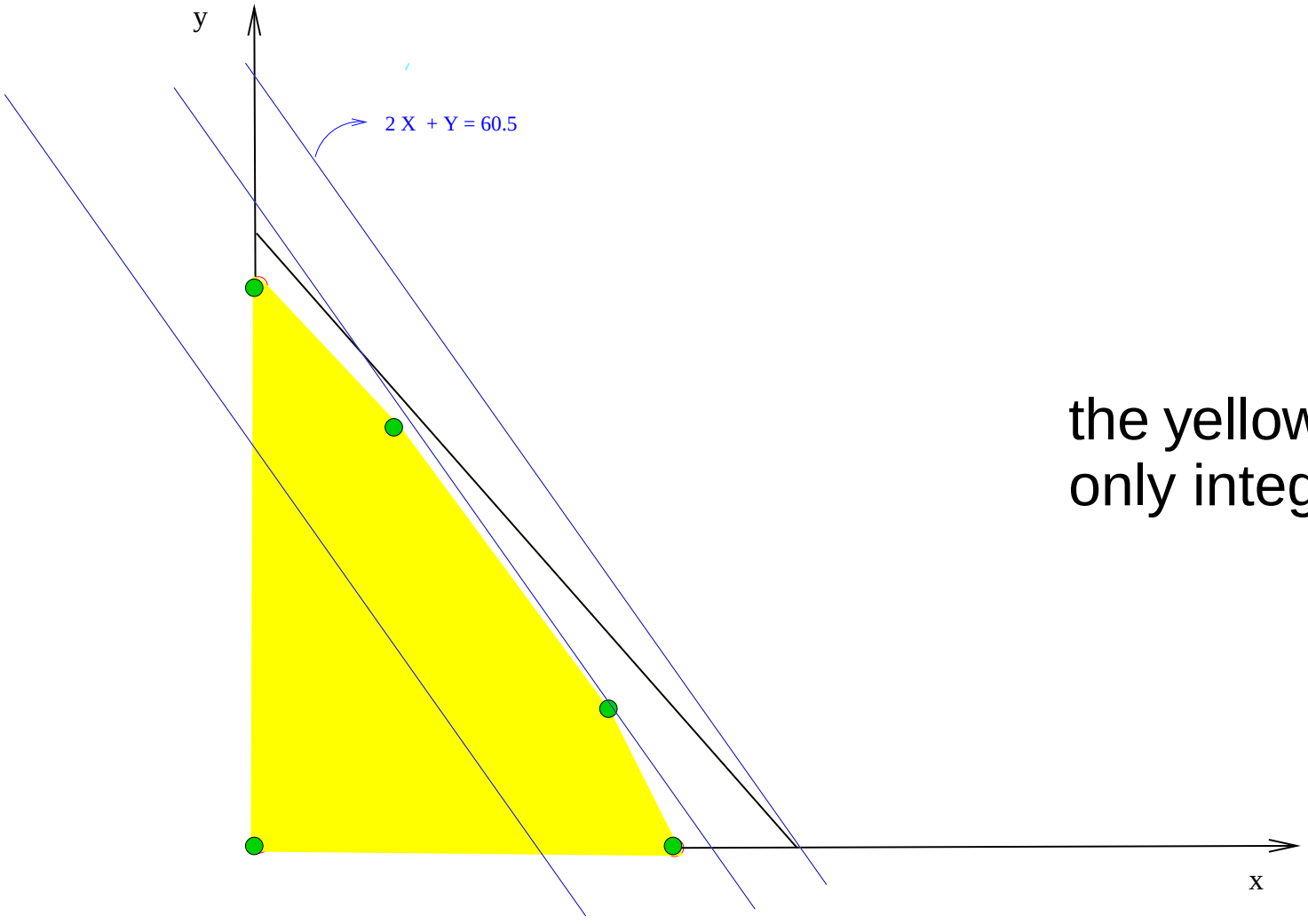
the enclosures to be built from up to 121 pieces of unit length

$4x+3y \leq 121,\ x \geq 0,\ y \geq$

x,y- integer

# *convex hull, linear objective*

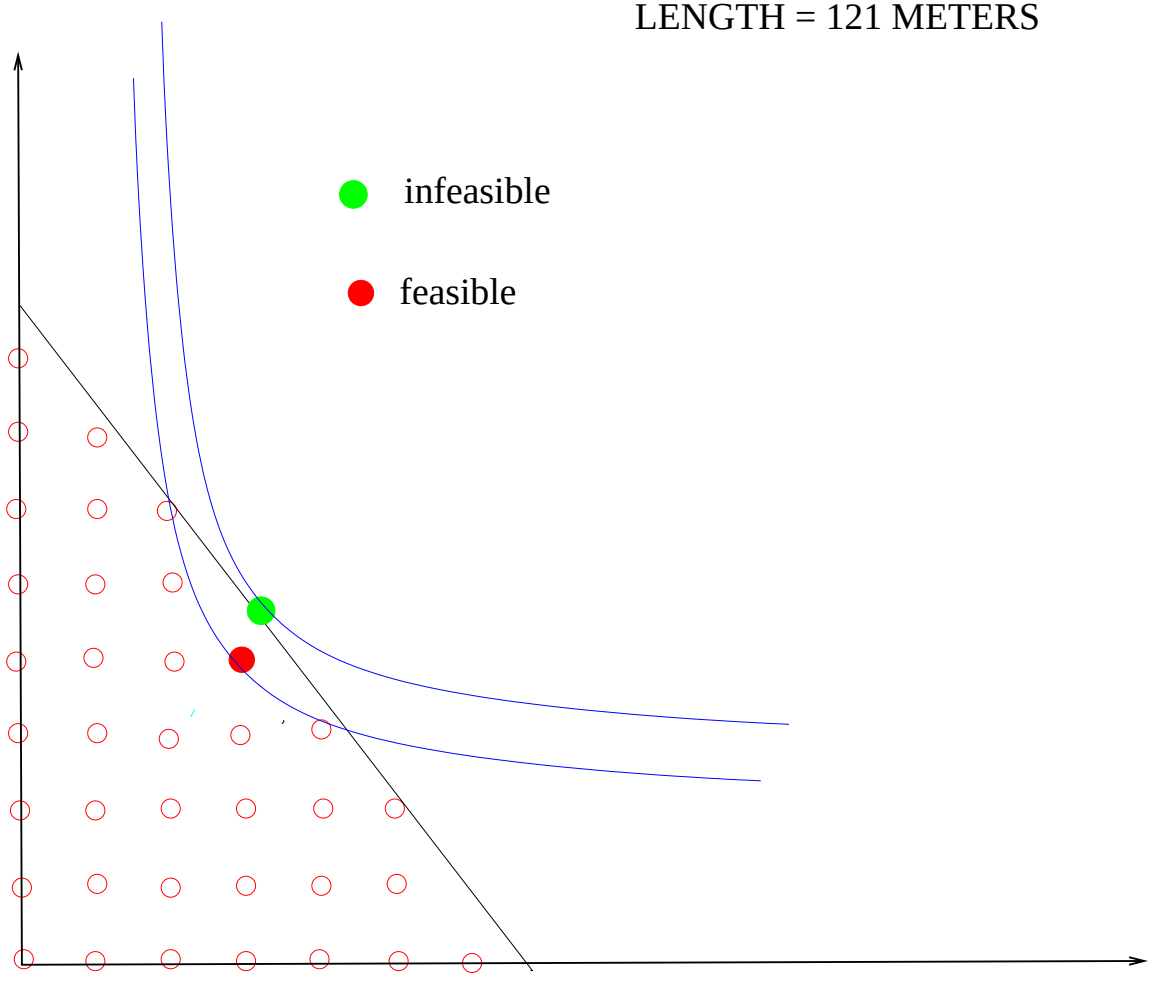
LENGTH = 121 METERS




the yellow polytope has only integer vertices

THE CASE OF INTEGER ENCLOSURES

# *the concept of bounds ( relaxation )*




$x = 15.125, y = 20.16, ub = 30$

$x = 15, y = 20, lb = 300$

ub -upper bound

lb - lower bound

gap (absolute) is $ub - lb$

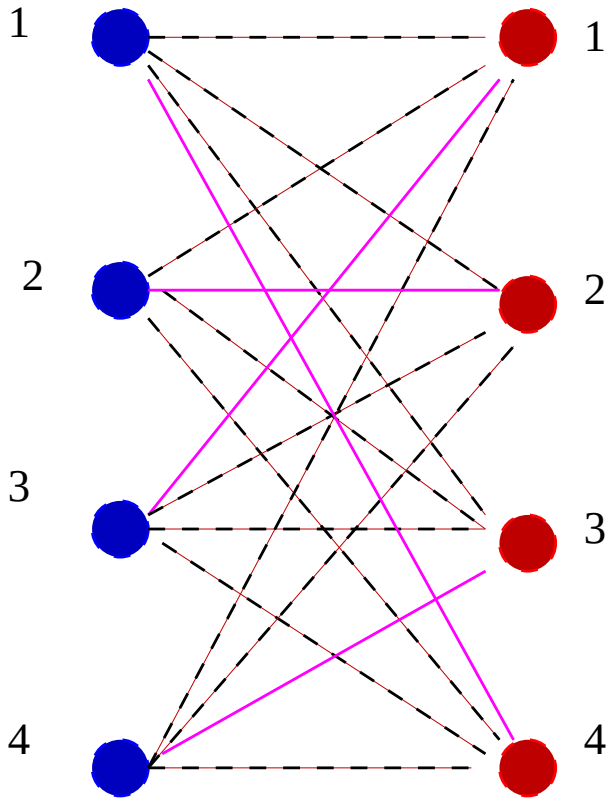
THE CASE OF INTEGER ENCLOSURES

# *integer polytope( assignment problem)*

| | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | 5 | 8 | 12 | **5** |
| 2 | 7 | **4** | 5 | 4 |
| 3 | **11** | 13 | 12 | 22 |
| 4 | 7 | 5 | **4** | 5 |

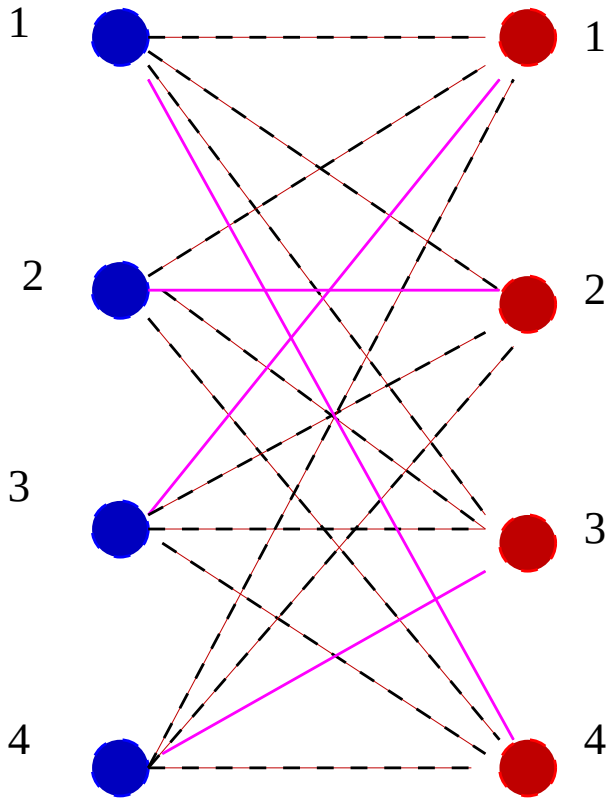
draw 4 independent numbers with a minimal sum

# *integer polytope (assignment problem)-bipartite graph*




| | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | 5 | 8 | 12 | **5** |
| 2 | 7 | **4** | 5 | 4 |
| 3 | **11** | 13 | 12 | 22 |
| 4 | 7 | 5 | **4** | 5 |

# integer polytope (assignment problem)-mathematical model




| | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | 5 | 8 | 12 | **5** |
| 2 | 7 | **4** | 5 | 4 |
| 3 | **11** | 13 | 12 | 22 |
| 4 | 7 | 5 | **4** | 5 |

$$\sum_i \sum_j c_{ij} x_{ij} \rightarrow \max$$

$$\sum_j x_{ij} = 1, \ i = 1, n$$

$$\sum_i x_{ij} = 1, \ j = 1, n$$

$$x_{ij} \in \{0, 1\}$$

## *about the distance between continuous and integer solutions*

$8x_1 + 11x_2 + 6x_3 + 4x_4$ maximize

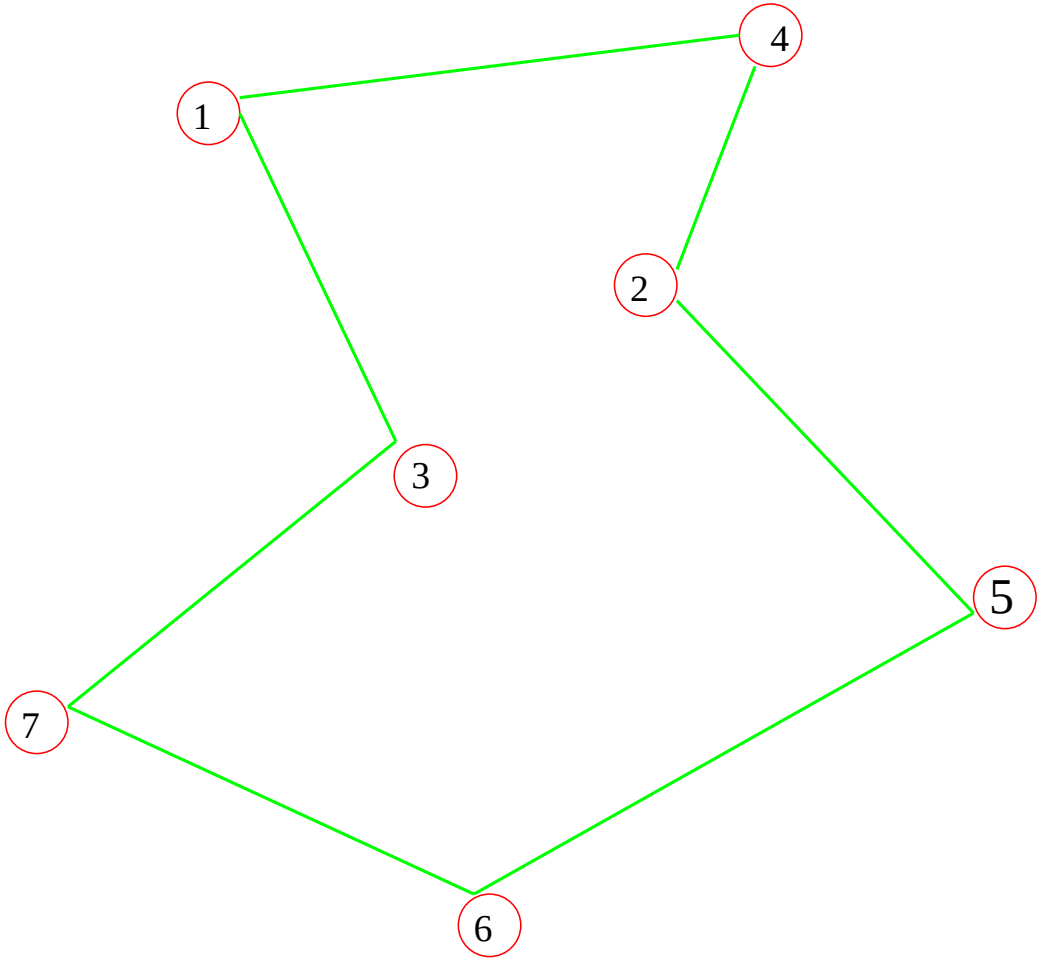
$5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14$

$x \in \{0, 1\}$

$x_1 = 1, x_2 = 1, x_3 = 0.5, x_4 = 0$ $v_{lp} = 22$ (continuous solution)

$x_1 = 0, x_2 = 1, x_3 = 1,\ x_4 = 1$ $v_{ip} = 21$ (integer solution)

# *good vs. bad model*

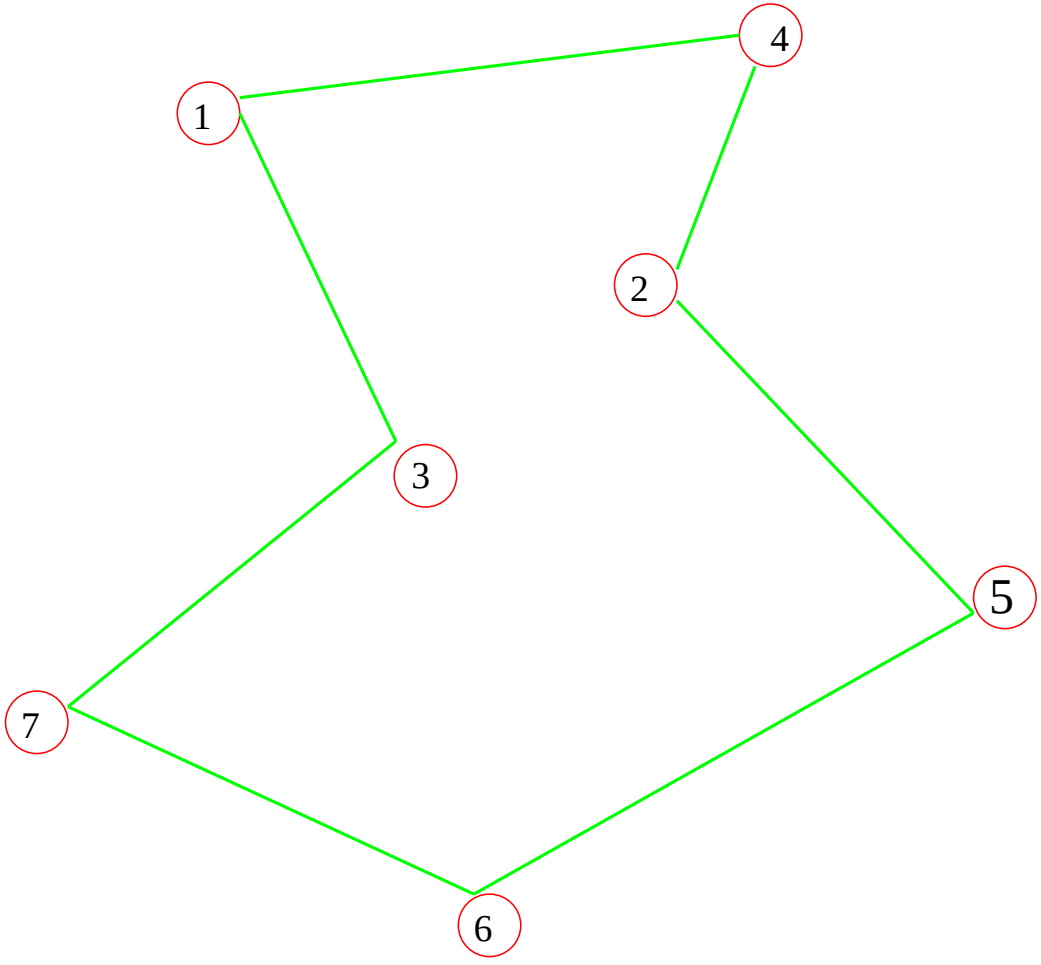



The tour(hamiltonian)

$1-4-2-5-6-7-3-1$ is of length

$C_{14}+C_{42}+C_{25}+C_{56}+C_{67}+C_{73}+C_{31}$

# *good vs. bad model-assignment problem relaxation*



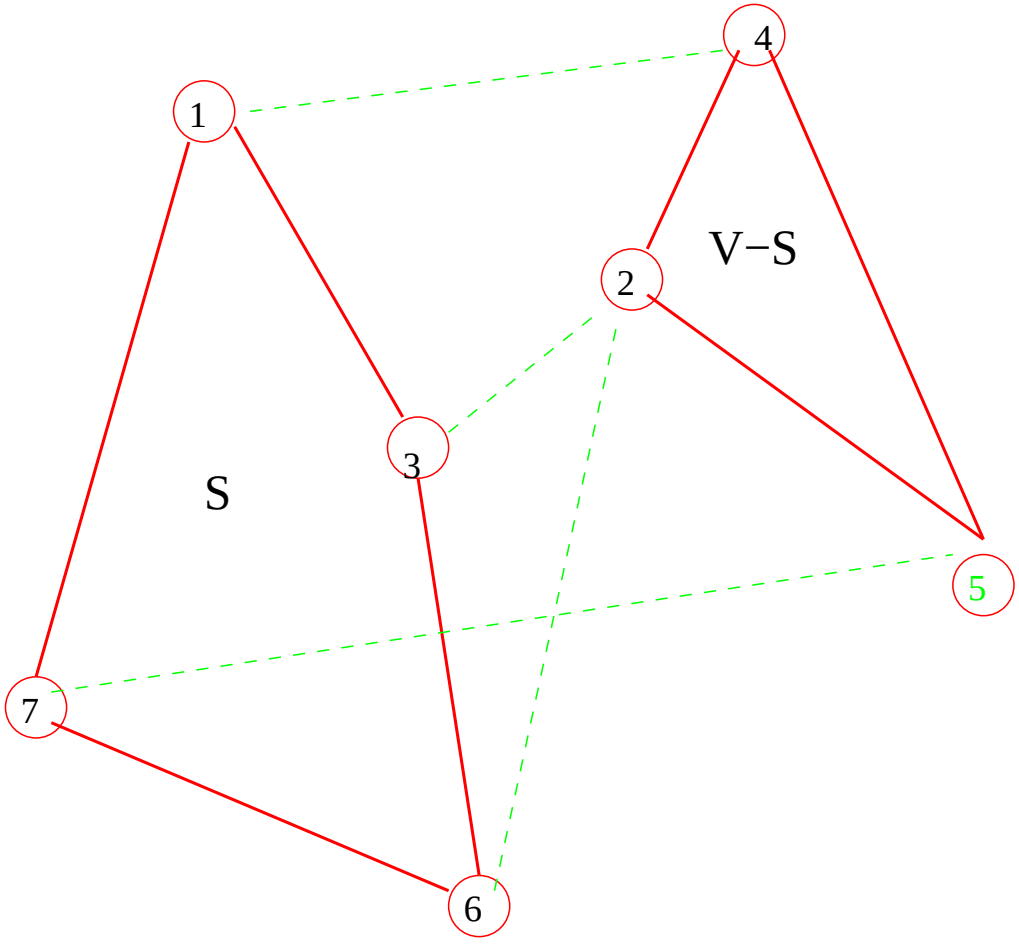

$$x_{ij} = \begin{cases} 1 \text{ from i directly to} \\ 0 \text{ otherwise} \end{cases}$$

$$\sum_j x_{ij} = 1 \; i = 1, n$$

$$\sum_i x_{ij} = 1 \; j = 1, n$$

$$x_{ij} \in \{0, 1\}$$

# *good vs. bad model- break the loops*




$$\sum_j x_{ij} = 1 \; i = 1, n$$

$$\sum_i x_{ij} = 1 \; j = 1, n$$

$$x_{ij} \in \{0, 1\}$$

$$\sum_{i \in S, j \in V-S} x_{ij} \geq 1 \text{ for each S} \in V$$

# *good vs. bad model- break the loops*

For $n = 300$ the number of loop destroyers is
10185179881672430431342228442046890805257341968329681253180702246771906498816683530916986 88

# *good vs. bad model- linearization*

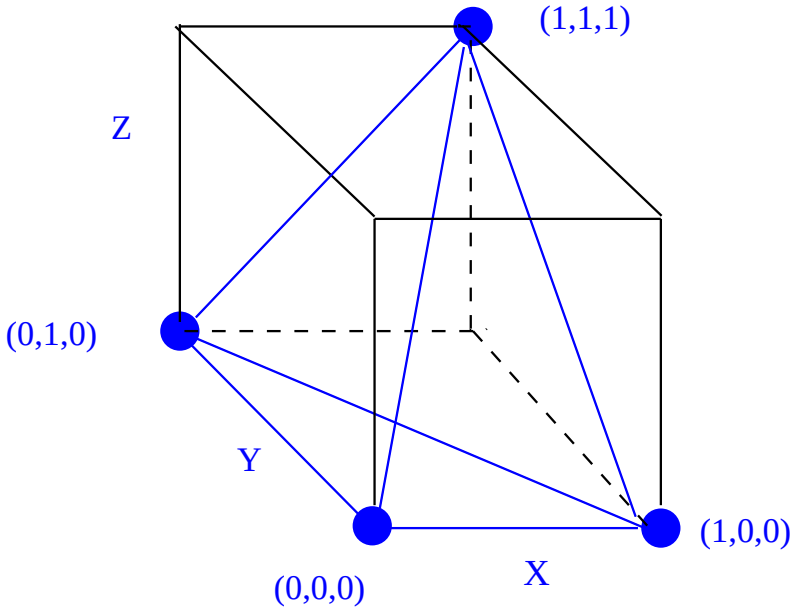
how to linearize quadratic terms $xy$ for $0/1$ variables ?

the trick is to

set $xy = z$ and force $z$ to be equal to $1$ iff $x = 1, y = 1$

$$z \leq x \ , z \leq y$$

$$x + y - z \leq 1$$




- thease are all feasible points for Z = min { X, Y }
  X, Y 0\1 variables

# *Lagrangian Relaxation and Duality*

maximize $13x_1 + 9x_2 + 18x_3 + 5x_4 + 12x_5$

$$4x_1 + 3x_2 + 7x_3 + 2x_4 + 5x_5 \leq 13$$

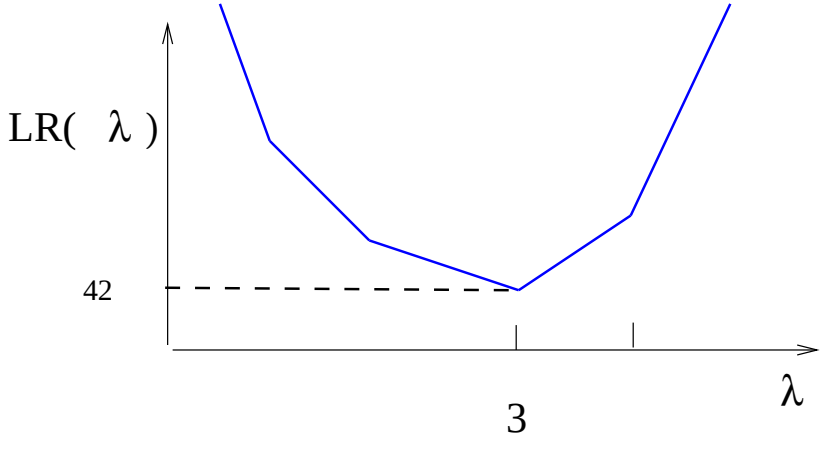
$$x_i, i = 1, \ldots 5 \text{ integer}$$

LP solution $x_1 = 3, x_2 = \frac{1}{3}, V_{LP} = 42$ (if the bounds $x_1 \leq 3, x_2 \leq 4, x_3 \leq 1, x_4 \leq 6, x_5 \leq 2$ are added to the feasible set, otherwise $V_{LP} = 42.5$)

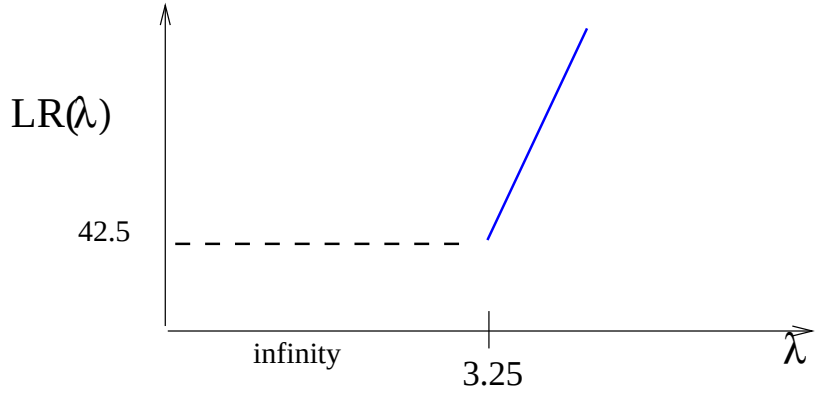
# Lagrangian Relaxation and Duality

$$Z(x,\lambda) = (13-4\lambda)x_1+(9-3\lambda)x_2+(18-7\lambda)x_3+(5-2\lambda)x_4+(12-5\lambda)x_5$$

Lagrangian relaxation: $LR(\lambda) = \max_x Z(x,\lambda)$




BOUNDED CASE




UNBOUNDED CASE

# Lagrangian Relaxation and Duality

$$Z(x,\lambda) = (13-4\lambda)x_1+(9-3\lambda)x_2+(18-7\lambda)x_3+(5-2\lambda)x_4+(12-5\lambda)x_5+13$$

$$\begin{array}{lcccccccc}
LR(0) & = & 3\times 13 & +4\times 9 & +1\times 18 & +6\times 5 & +2\times 12 & & = 147 \\
LR(1) & = & 3\times 9 & +4\times 6 & +1\times 11 & +6\times 3 & +2\times 7 & +13 & = 107 \\
LR(2) & = & 3\times 5 & +4\times 3 & +1\times 4 & +6\times 3 & +2\times 2 & +26 & = 63 \\
LR(3) & = & 3\times 1 & +0 & +0 & +0 & +0 & +39 & = 42 \\
LR(4) & = & & & & & & 52 & = 52
\end{array}$$

$$LR(3+\epsilon) = 42+\epsilon > 42 \text{ and } LR(3-\epsilon) = 42+11\times\epsilon > 42$$

$$\Longrightarrow Z_{LD} = \min_{\lambda\geq 0} LR(\lambda) = LR(3) = 42 = Z_{LP}$$

Theory: $Z_{IP} \leq Z_{LD} \leq Z_{LP}$

# *Recapitulation- Part 1*

- Intro through examples of some integer programming topics :
  - terms: objective function, feasible set,polytopes, optimal solution,relaxation, bounds, gap, lagrangian duality.
- Classical problems like :
  - knapsack, assignment, travelling salesman.
- Mentioning of graphs as a valuable tool for modelling many combinatorial problems.

# *Protein Folding Problem*

SNGIEASLLTDPKDVSGRTVDYIIAGGGLTGLTTAARLTENPNIS

SGSYESDRGPIIEDLNAYGDIFGSSVDHAYETVELATNNQTALIR

# *Protein Folding Problem*

SNGIEASLLTDPKDVSGRTVDYIIAGGGLTGLTTAARLTENPNIS

SGSYESDRGPIIEDLNAYGDIFGSSVDHAYETVELATNNQTALIR

A sequence in a protein data bank

# *Protein Folding Problem*

SNGIEASLLTDPKDVSGRTVDYIIAGGGLTGLTTAARLTENPNIS
SGSYESDRGPIIEDLNAYGDIFGSSVDHAYETVELATNNQTALIR

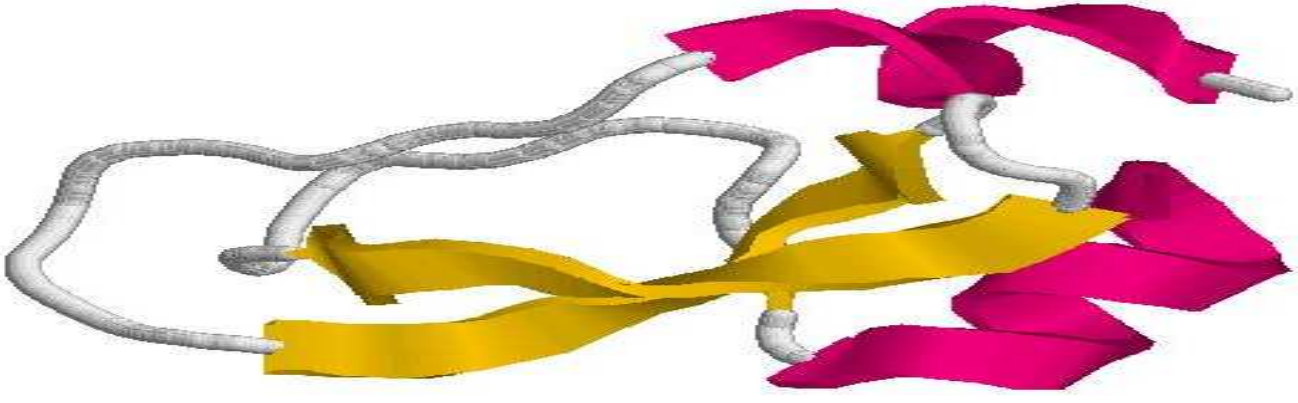
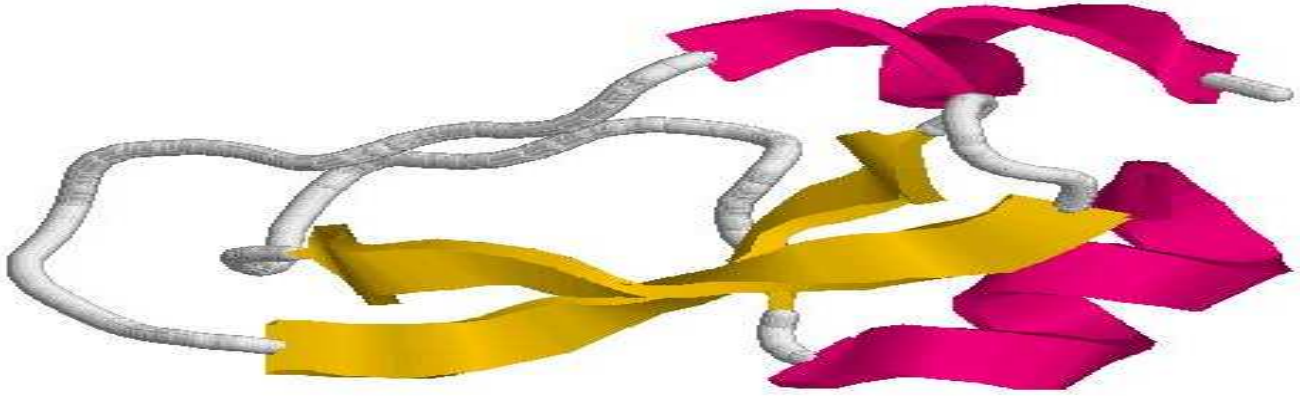


Figure 1: in fact this is its real (3D) shape

# *Protein Folding Problem*

SNGIEASLLTDPKDVSGRTVDYIIAGGGLTGLTTAARLTENPNIS
SGSYESDRGPIIEDLNAYGDIFGSSVDHAYETVELATNNQTALIR



Protein Folding Problem :

- **Input**: $a_1, a_2, \ldots, a_N$—a sequence over the 20-letter amino acid alphabet
- **Output**: $(x_i, y_j, z_j), j = 1, \ldots, N$—the coordinates of $a_j$

# *Protein Folding Problem*

SNGIEASLLTDPKDVSGRTVDYIIAGGGLTGLTTAARLTENPNIS

SGSYESDRGPIIEDLNAYGDIFGSSVDHAYETVELATNNQTALIR

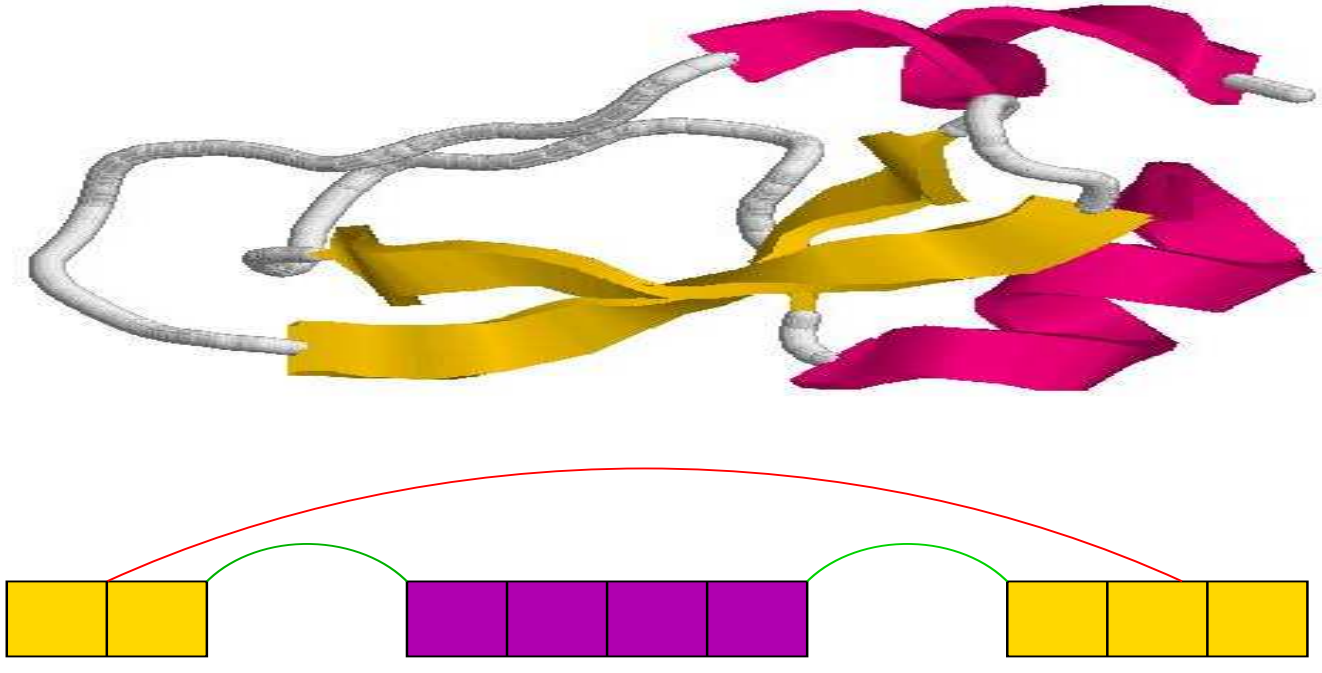



Figure 2: Generalized contact map graph—describes the interactions between the blocks

# *3D structure determination methods*

- Experimental *(in vitro)*: *X*-ray crystallography, NMR. Slow and expensive. Require knowledge of the proteins structural domains.
- Computational *(in silico)*
  - Direct approach: Seeks to minimize the free energy using classical mechanics models. Computationally very expensive—BLUE GENE supercomputer
  - Sequence alignment methods: BLAST, FASTA, PSI-BLAST. Cannot compare remote homologs.
  - Fold recognition methods
    - Protein Threading (this talk)

# *Protein Threading—basic assumptions*

- the sequence (1D structure) determines the 3D structure
- homologous proteins have similar structure (and function)
- homologous proteins have conserved structural cores and variable loop regions
- Postulate: there between 1000 and 2000 different protein structural families (library of 3D structures/cores)

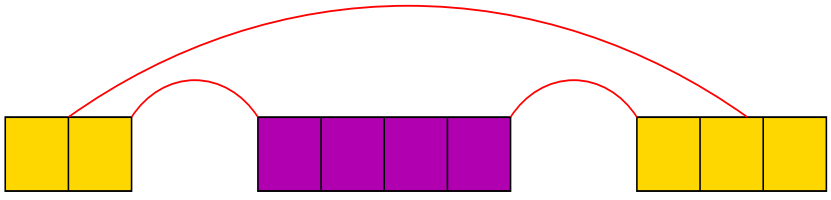
# *Protein Threading—main steps*

- constructing a library of core folds (structural templates) —see the 3D catalogue
- choosing and objective function (score function) to evaluate any alignment of a sequence to a structural template
- fi nding the best alignment of the query sequence to each core in the library—NP-hard problem. (need of good combinatorial optimization alg.)
- choosing the most appropriate core based on normalized scores of the optimal alignments (requires good statistical model and the power of distributed computing)

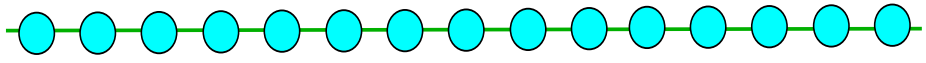
# Building graph theoretical formalism

# Query-to-structure alignment

$m = 3$ segments of lengths $l_1 = 2, l_2 = 4, l_3 = 3$ ;



3D structure template (core)



1D query of lenght N=15

# Query-to-structure alignment

$m = 3$ segments of lengths $l_1 = 2, l_2 = 4, l_3 = 3$ ;

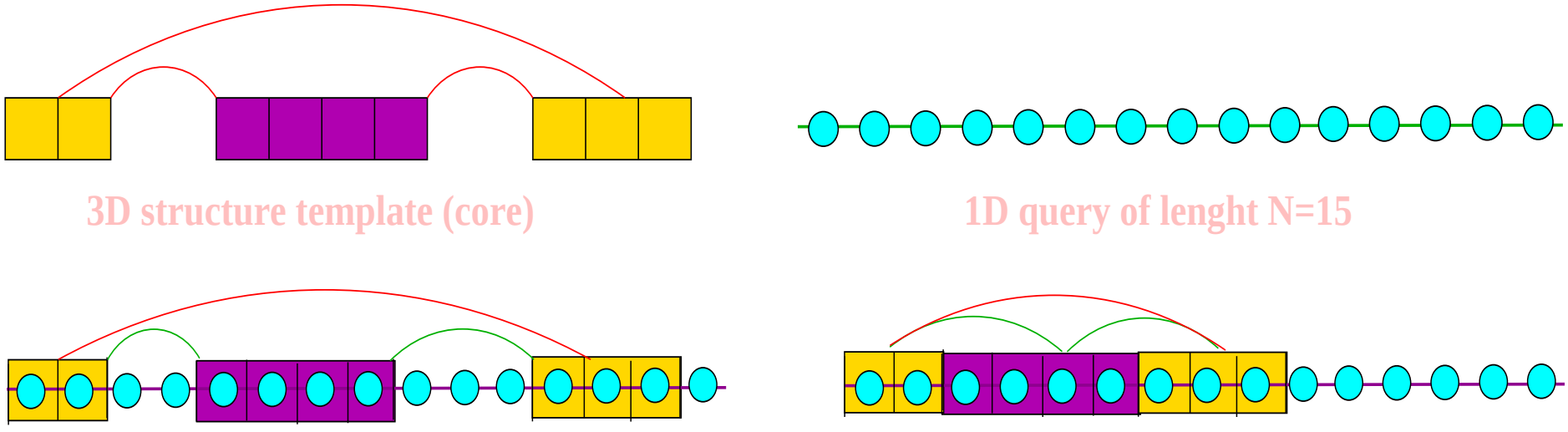
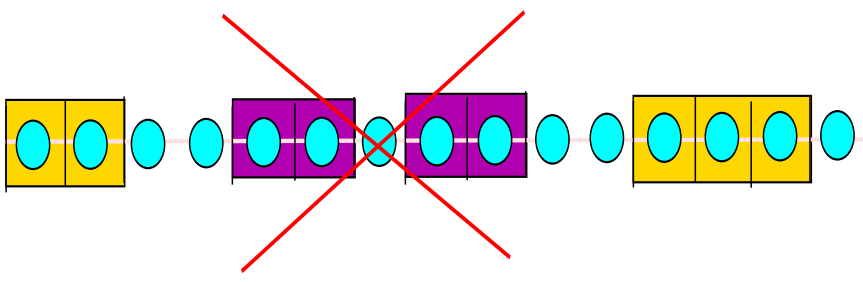
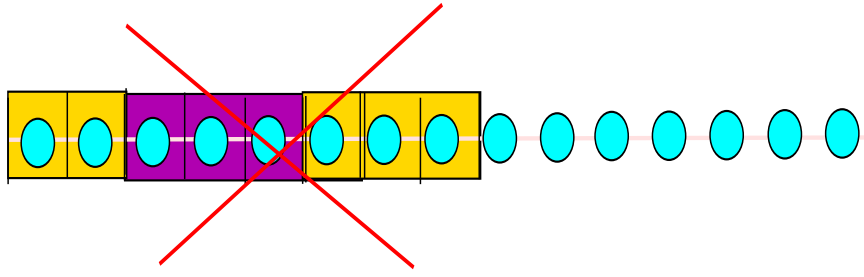



Figure 3: two possible alignments.

*Alignment (threading)*: covering the elements of query by the template blocks/segments. A threading is completely determined by the starting positions of the blocks. To any threading is associated a score.

# *Query-to-structure alignment: “classical” threading rules*

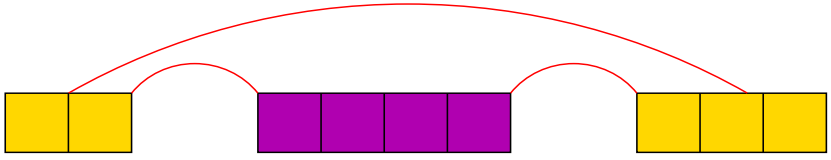




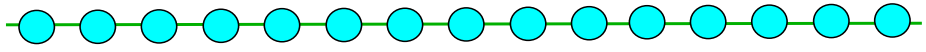
- blocks preserve their order
- block do not overlap
- no gaps in the blocks
- blocks are of fi xed lenght

# Absolute and relative positions

$m = 3$ segments of lengths: $l_1 = 2, l_2 = 4, l_3 = 3$ ;



3D structure template (core)

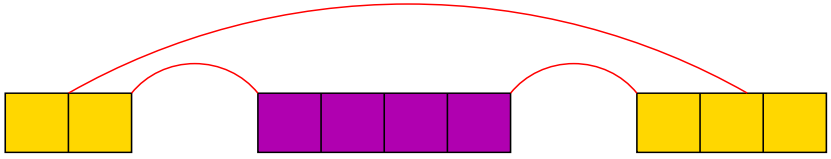


1D query of lenght N=15

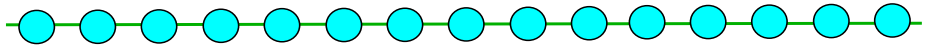
| abs. position | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| rel. pos. block 1 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | | | | | | | |
| rel. pos. block 2 | | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | | | | | |
| rel. pos. block 3 | | | | | | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |

# Absolute and relative positions

$m = 3$ segments of lengths: $l_1 = 2, l_2 = 4, l_3 = 3$ ;



3D structure template (core)



1D query of lenght N=15

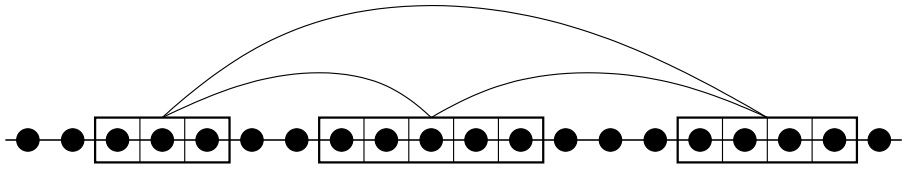
| abs. position | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| rel. pos. block 1 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | | | | | | | |
| rel. pos. block 2 | | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | | | | | |
| rel. pos. block 3 | | | | | | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |

$n = N + 1 - \sum_{i=1}^{m} l_i$ is the degree of freedom for each block;

$n = 7$ for the considered example

Number of possible threadings $|T| = \binom{n-1+m}{m} = \frac{(n-1+m)!}{m!(n-1)!}$.

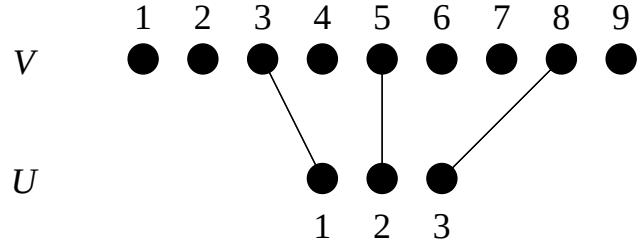
# PTP is a matching problem



(a)

| abs. position | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| rel. position block 1 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | | | | | | | | | | | |
| rel. position block 2 | | | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | | | | | | | | |
| rel. position block 3 | | | | | | | | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | | | |

(b)




(c)

Figure 4: (a) Example of alignment of query sequence of length 20 and template containing 3 segments of lengths 3, 5 and 4. (b) Correspondence between absolute and relative block positions. (c) A matching corresponding to the alignment of (a).

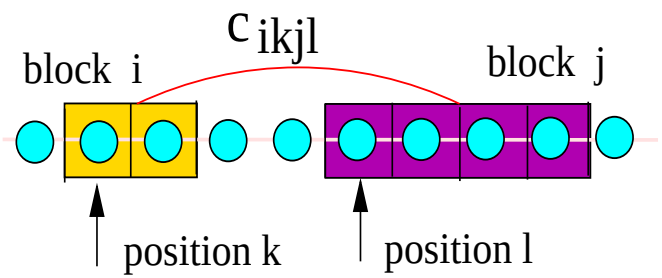
# *Size of the solution space*

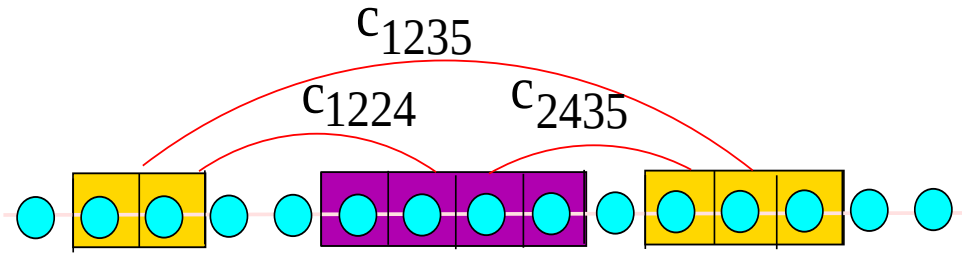
Number of possible threadings $|T| = \binom{n-1+m}{m} = \frac{(n-1+m)!}{m!(n-1)!}$.

| query name | core name | size | | space size |
|---|---|---|---|---|
| | | segm. | pos. | |
| 2cyp_0 | 2cyp_0 | 15 | 98 | 1.5e+18 |
| 1coy_0 | 1gal_0 | 36 | 81 | 1.3e+30 |
| 3mina0 | 4kbpa0 | 23 | 189 | 3.2e+30 |
| 3minb0 | 1gpl_0 | 23 | 215 | 5.3e+31 |
| 1gal_0 | 1ad3a0 | 31 | 212 | 1.3e+39 |
| 1coy_0 | 1fcba0 | 34 | 190 | 1.7e+40 |
| 1kit_0 | 1reqa0 | 41 | 194 | 9.9e+45 |

# Score function: pairwise interactions

$c_{ijkl}, 1 \leq j \leq l \leq n$—score of putting block $i$ on position $k$ and block $j$ on position $l$




The above alignment corresponds to threading (2,4,5) with cost $\varphi(2,4,5) = s_{12} + s_{24} + s_{35} + c_{1224} + c_{2435} + c_{1235}$.

The score function is supposed to be

- additive
- can be computed considering no more than two blocks at a time

# Protein threading problem

$$\min\{\varphi(\pi)|\pi \in T\}$$

where

$$\varphi(\pi) = \sum_{i=1}^{m} s_{i\pi_i} + \sum_{(i,k)\in E} c_{i\pi_i k\pi_k}$$

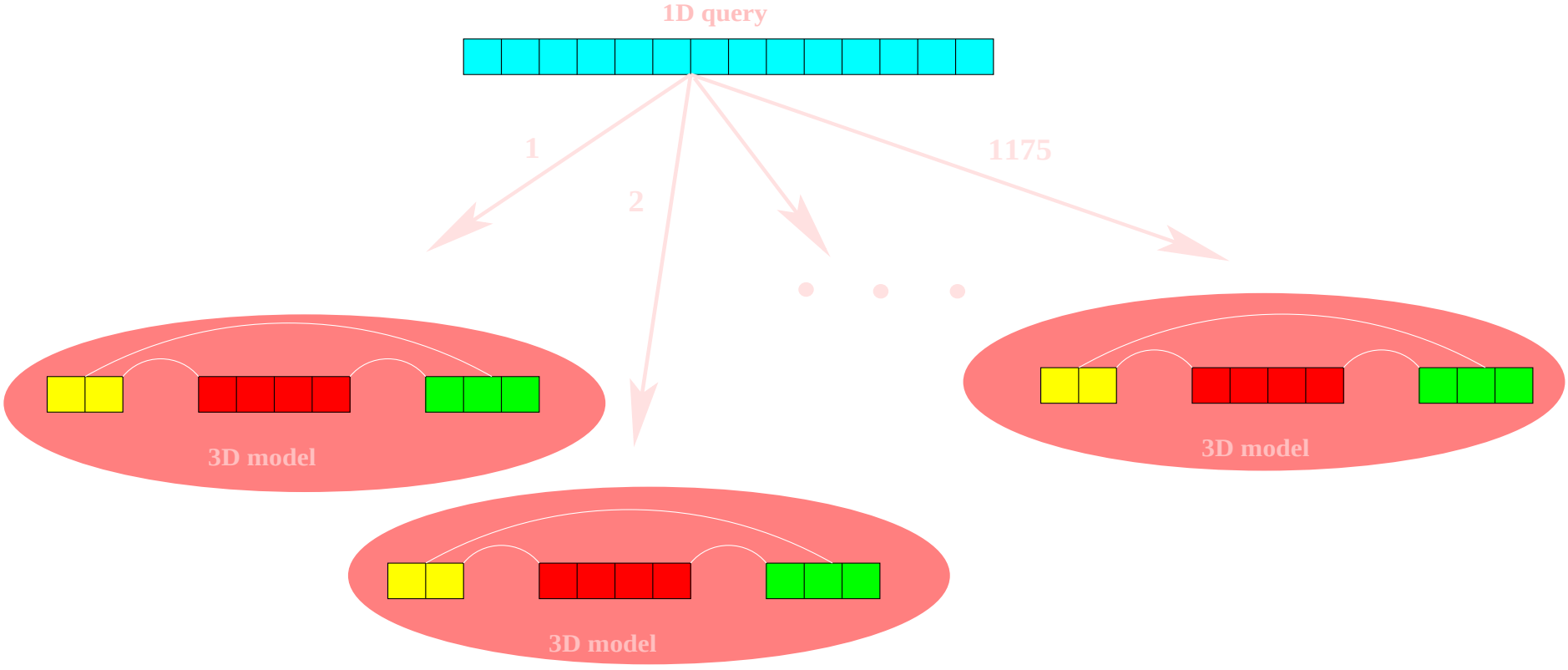
and T is the set of threadings

$$T = \{(\pi_1, \ldots, \pi_m) \mid 1 \leq \pi_1 \leq \ldots, \pi_m \leq n\}$$

The problem is proven to be NP_hard (Lathrop,94) and MAX-SNP-hard (Akutsu&Miyano,99).

# *FROST : requires score normalization*

1175 classes are know today. We need to classify the query in one of these classes. Huge computations convenient to gridifi cation.



# *Related work*

- A. Marin, J.Pothier, K. Zimmermann, J-F. Gibrat, FROST: A Filter Based Recognition Method, Proteins: Struct. Funct. Genet. 2002
- Lathrop&Smith's branch&bound,(J.Mol.Biol., 1996);
- Akutsu&Miyano, On the approximation of protein threading, TCS, (1999)
- J. Xu, M. Li, G. Lin, D. Kim and Y. Xu, Protein threading by linear programming, PSB, January, 2003, (JBCB, March 2003)
- Yanev&Andonov, Parallel Divide&Conquer Approach for PTP, HiCOMB'03, April, 2003, Nice (CCPE, 16:1-14, 2004)
- Andonov, Balev, Yanev, Protein Threading Problem: From Mathematical Models to Parallel Implementations,INFORMS J.on Computing, 2004
- S. Balev, Solving the PTP by Lagrangian Relaxation, WABI 2004
- Veber&Yanev&Andonov&Poirriez, Optimal PTP by cost splitting, WABI 2005

# *Network flow model*

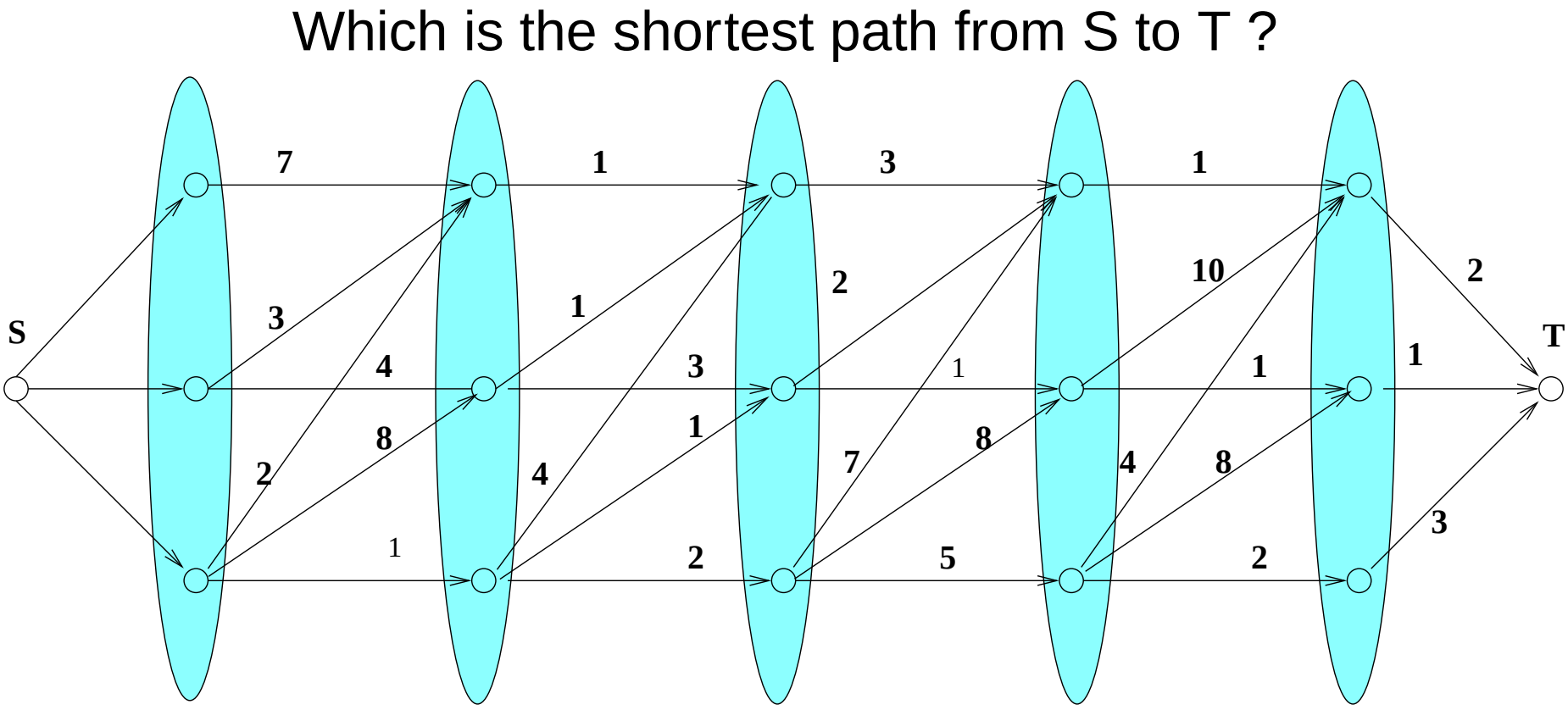
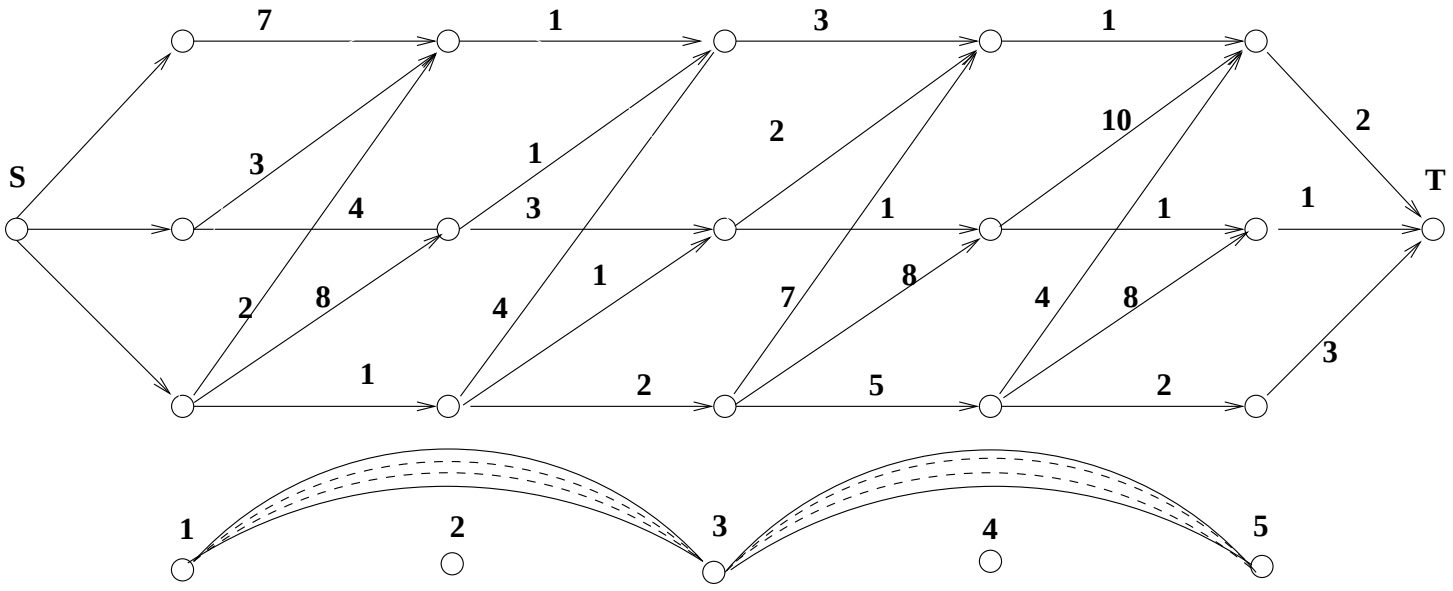



Figure 5: Five segments and their local interactions. The degree of freedom is three.

# Network flow model

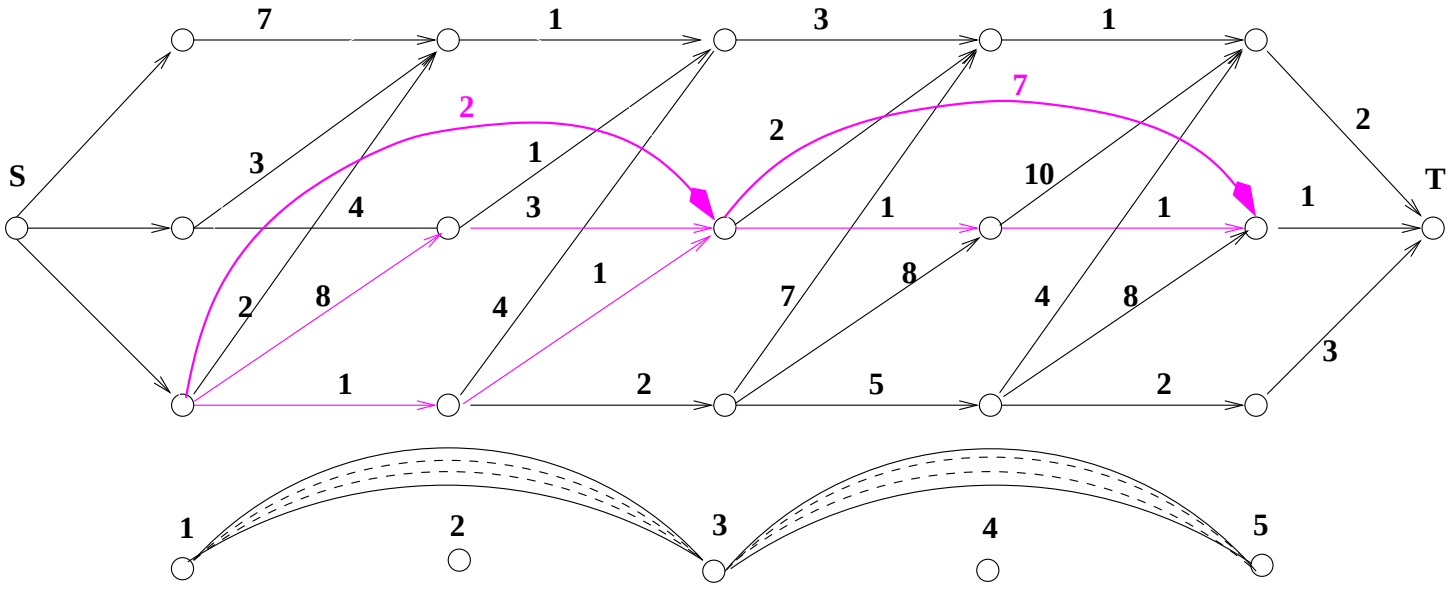



NON−LOCAL COSTS

| (1 3) | | (3 5) | |
|---|---|---|---|
| ( 1 1 ) (3 1) | 4 | (3 1) (5 1) | 1 |
| ( 1 1 ) (3 2) | 2 | ( 3 1 ) (5 2) | 4 |
| ( 1 1 ) (3 3) | 7 | (3 1) (5 3) | 2 |
| ( 1 2 ) (3 2) | 3 | (3 2) (5 2) | 7 |
| ( 1 2 ) (3 3) | 8 | (3 2) (5 3) | 5 |
| ( 1 3 ) (3 3) | 5 | (3 3) (5 3) | 2 |

Figure 6: Here are all interactions. The non-local interactions make the problem NP-complete.

# Network flow model



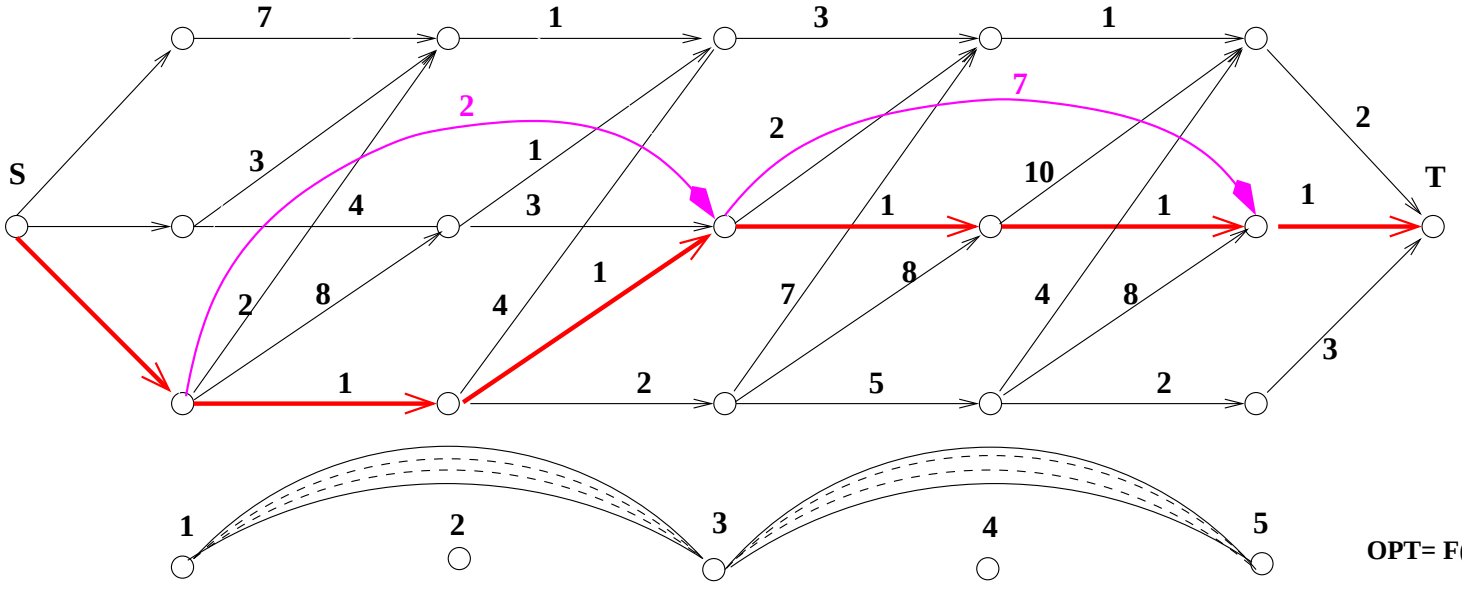

NON−LOCAL COSTS

| (1 3) | | (3 5) | |
|---|---|---|---|
| ( 1 1 ) (3 1) | 4 | (3 1) (5 1) | 1 |
| ( 1 1 ) (3 2) | 2 | ( 3 1 ) (5 2) | 4 |
| ( 1 1 ) (3 3) | 7 | (3 1) (5 3) | 2 |
| ( 1 2 ) (3 2) | 3 | ( 3 2 ) (5 2) | 7 |
| ( 1 2 ) (3 3) | 8 | (3 2) (5 3) | 5 |
| ( 1 3 ) (3 3) | 5 | (3 3) (5 3) | 2 |

Figure 7: Impact of the non-local interactions. A path from S to T activates complementary edes corresponding to the remote link. We call it *augmented path*

# Network flow model

Protein threading problem: fi nd the augmented path of minimal lenght.




NON−LOCAL COSTS

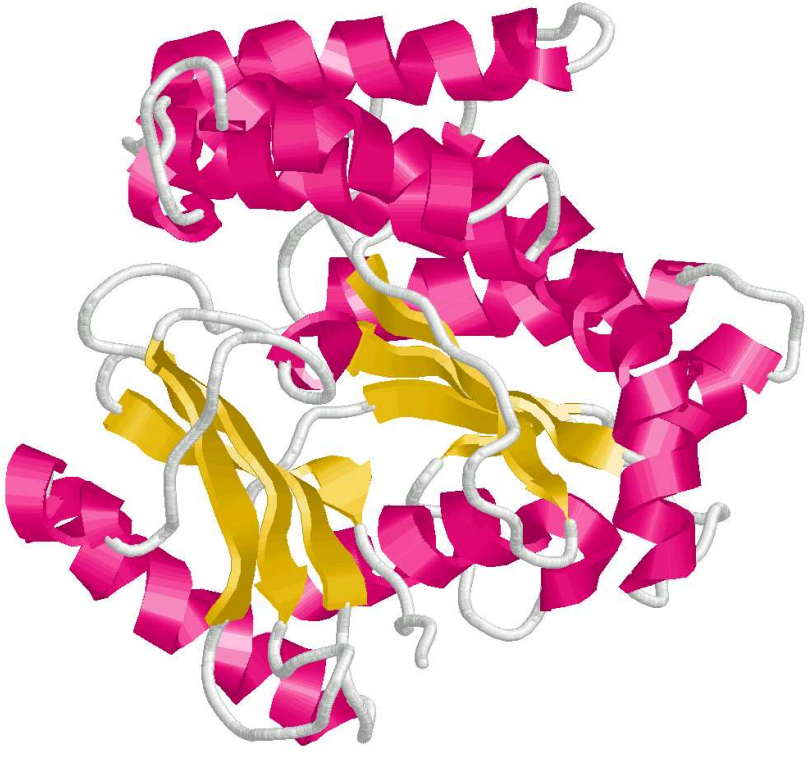
| (1 3) | | (3 5) | |
|---|---|---|---|
| ( 1 1 ) (3 1) | 4 | (3 1) (5 1) | 1 |
| ( 1 1 ) (3 2) | 2 | ( 3 1 ) (5 2) | 4 |
| ( 1 1 ) (3 3) | 7 | (3 1) (5 3) | 2 |
| ( 1 2 ) (3 2) | 3 | ( 3 2 ) (5 2) | 7 |
| ( 1 2 ) (3 3) | 8 | (3 2) (5 3) | 5 |
| ( 1 3 ) (3 3) | 5 | (3 3) (5 3) | 2 |

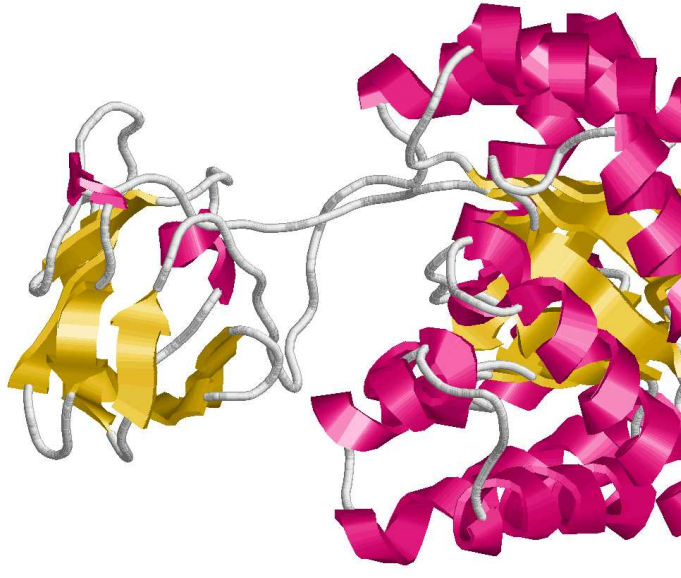
OPT= F((1,1) (2,1) (3,2) (4,2) (5,2))=14.0

Figure 8: The red path corresponds to the threading (1,1,2,2,2).

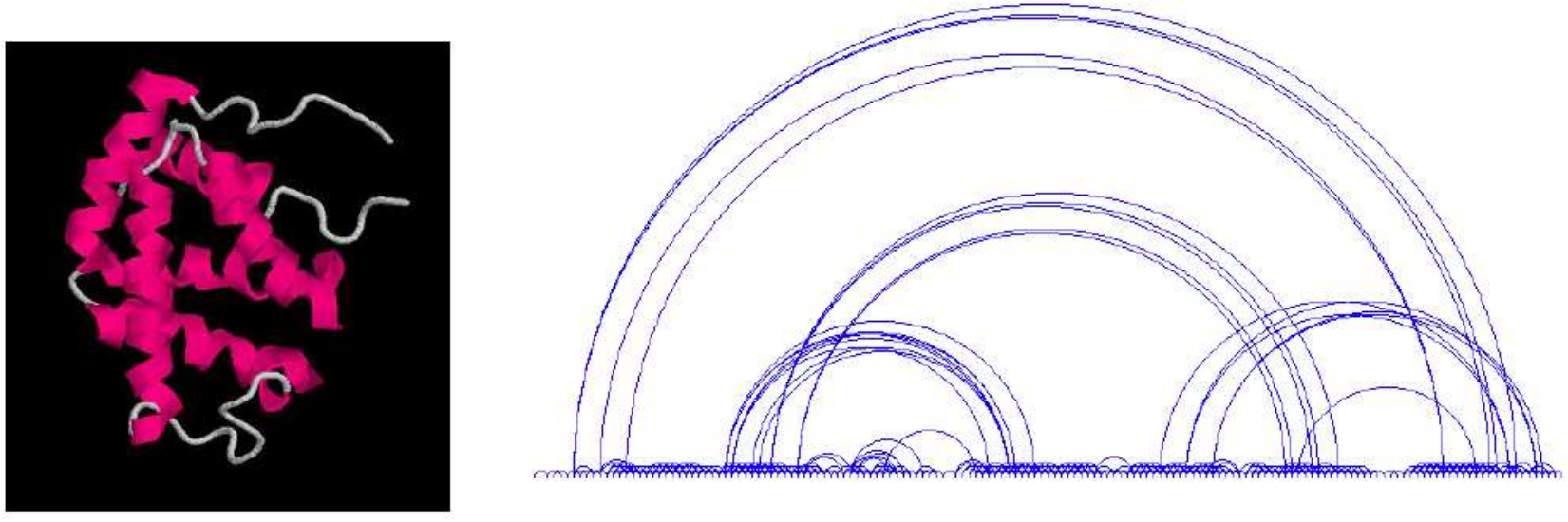
# *Comparison of proteins 3D structures*

How to compare these two structures???





# Contact Map Overlap I



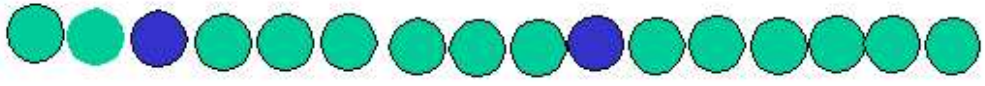
(a) (b)

Fig. 4. Native structure (a) for protein 1hlm taken from the PDB [2] and its contact map (b).

Attention: in the contact map graph any node is an AA

# *Contact Map Overlap (Cont.)*

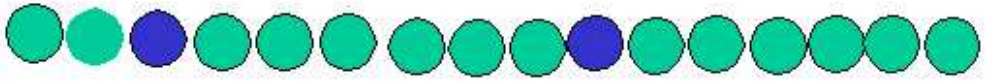
CONTACT MAPS



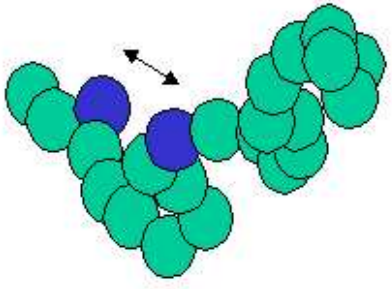
**Unfolded protein**

# *Contact Map Overlap (Cont.)*

CONTACT MAPS

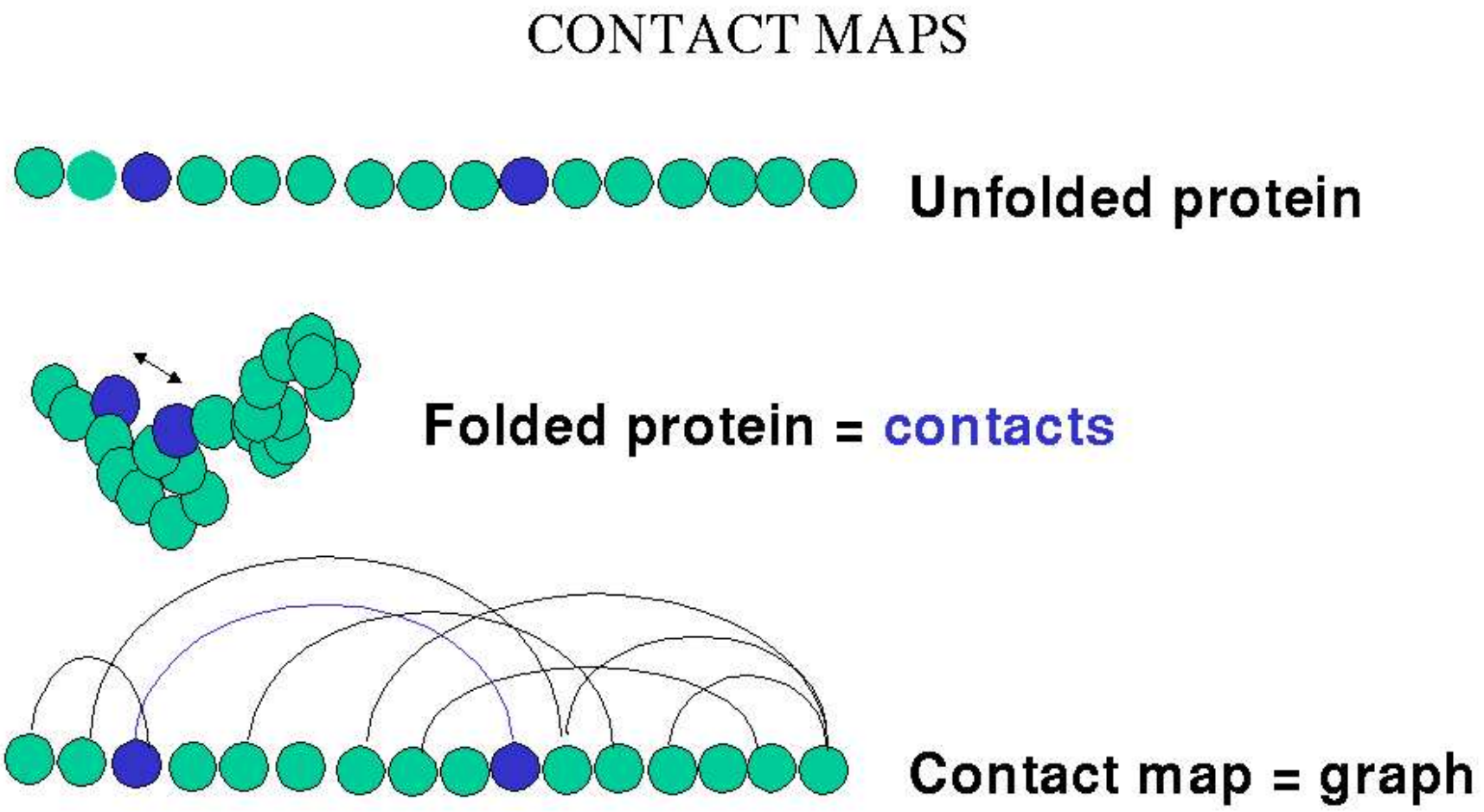


**Unfolded protein**



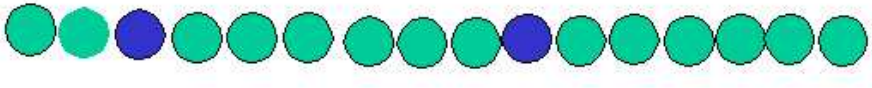
**Folded protein = contacts**

# *Contact Map Overlap (Cont.)*

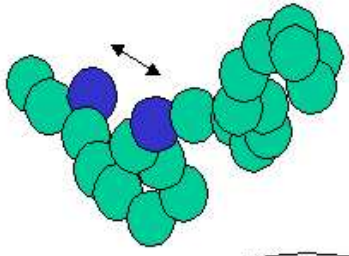


# *Contact Map Overlap (Cont.)*

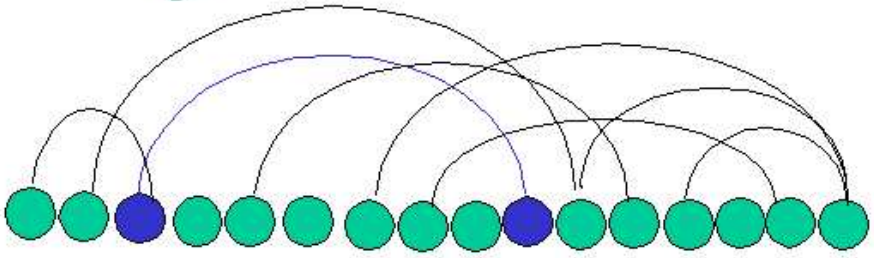
CONTACT MAPS



**Unfolded protein**



**Folded protein = contacts**

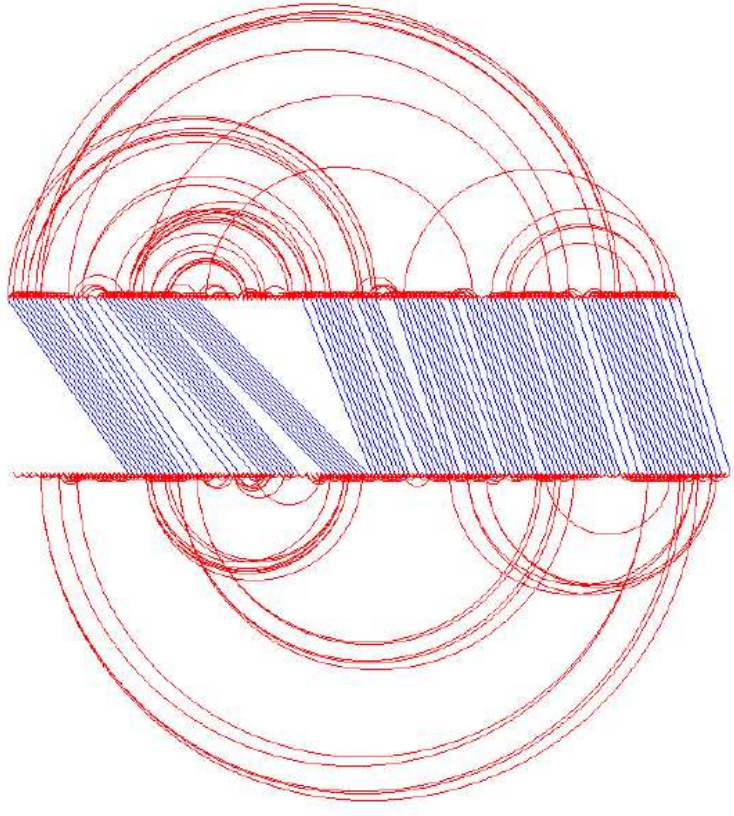


**Contact map = graph**

**OBJECTIVE:** align 3d folds of proteins

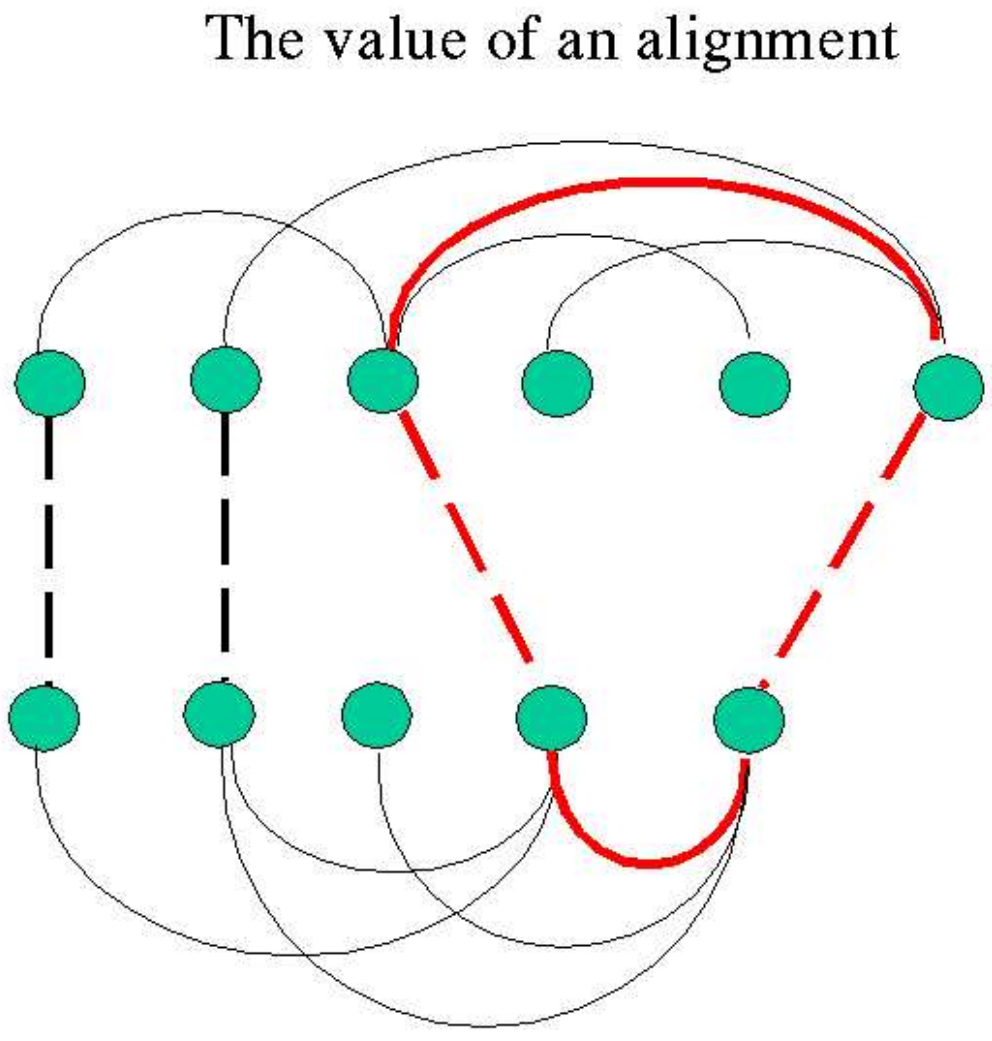
= align contact maps

# *Contact Map Overlap (Cont.)*



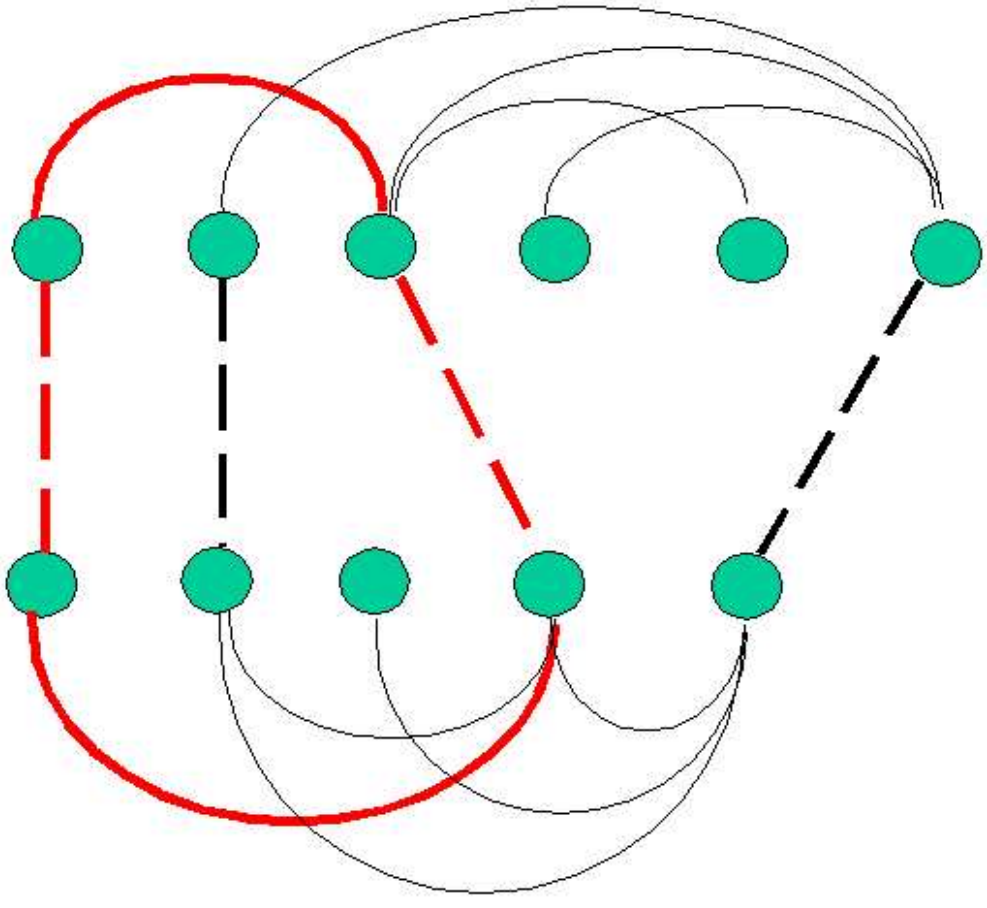
Contact map overlap problem is a kind of matching problem

# *Contact Map Overlap (Cont.)*

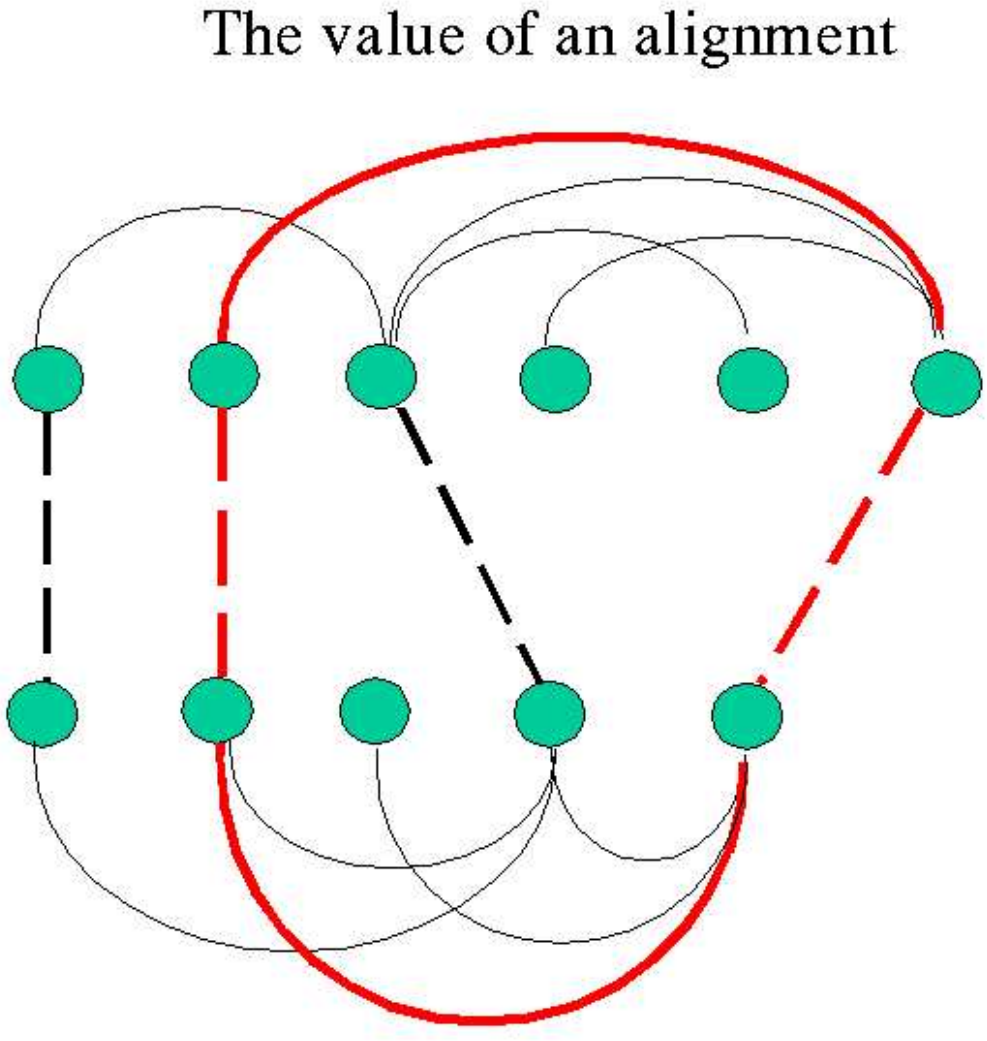


# *Contact Map Overlap (Cont.)*

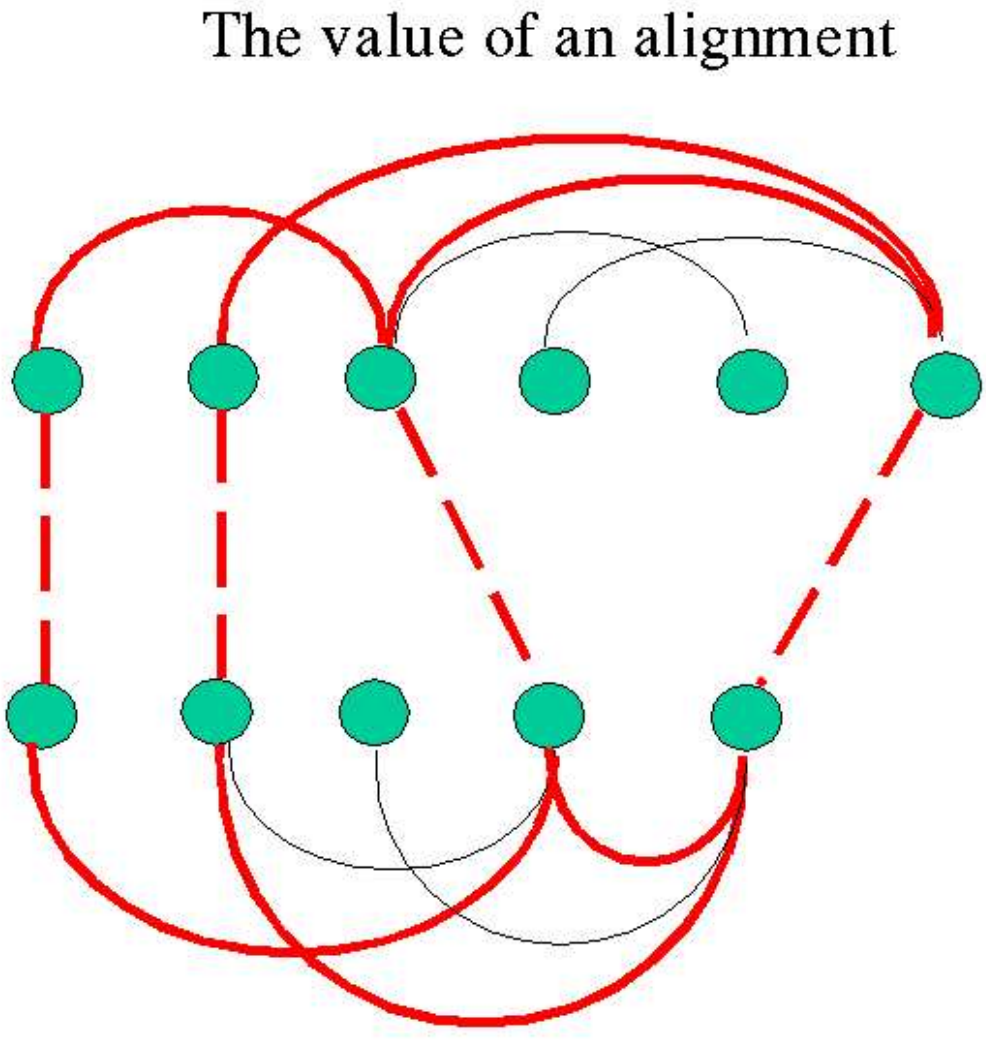
The value of an alignment



# *Contact Map Overlap (Cont.)*

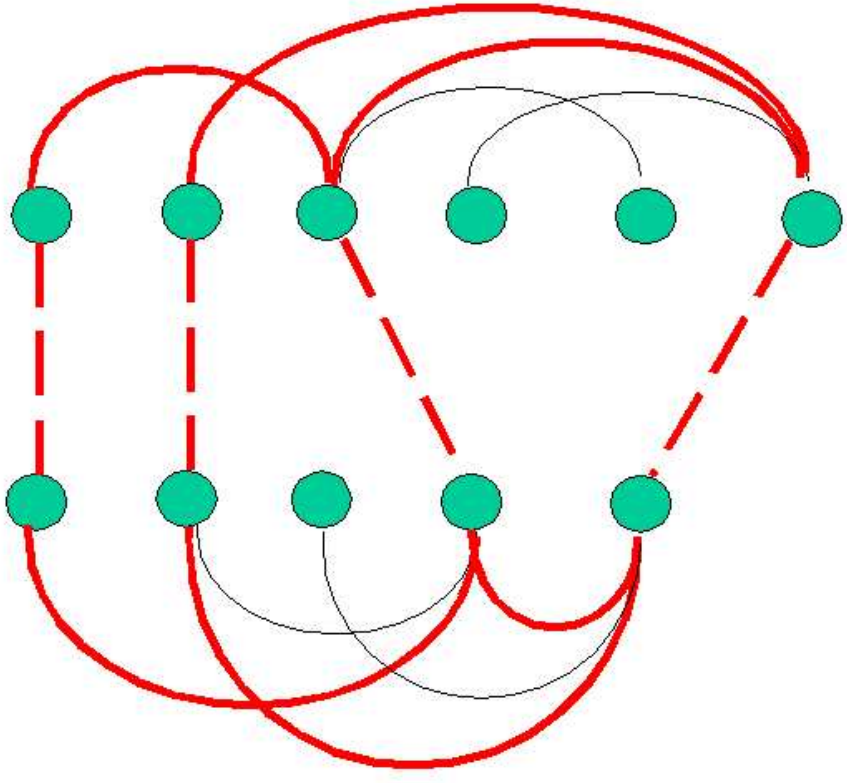


# *Contact Map Overlap (Cont.)*



# *Contact Map Overlap (Cont.)*

The value of an alignment

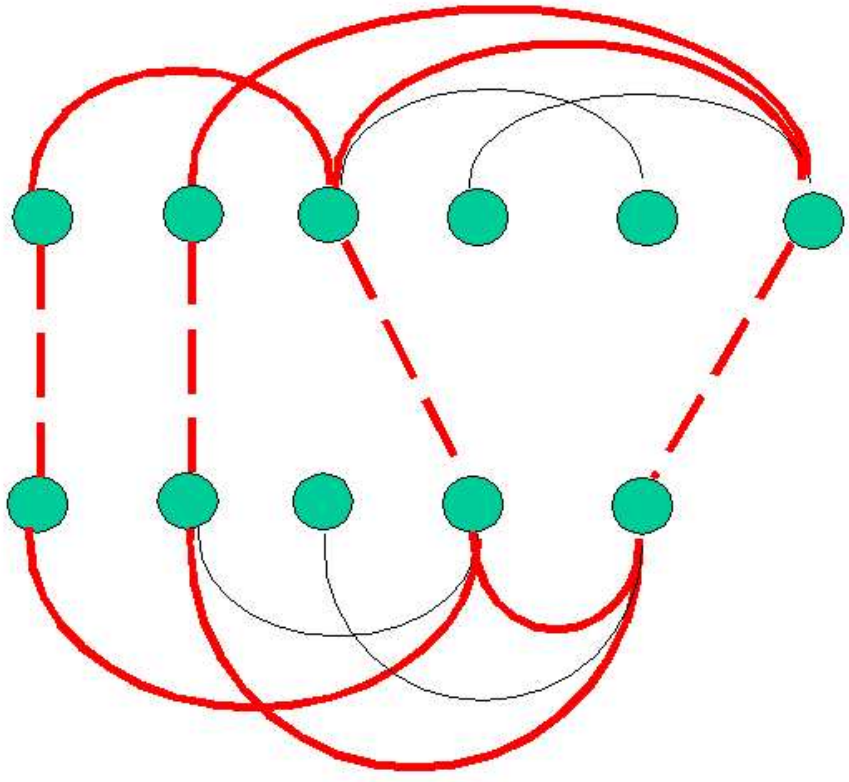


**Value = 3**

**We want to maximize the value**

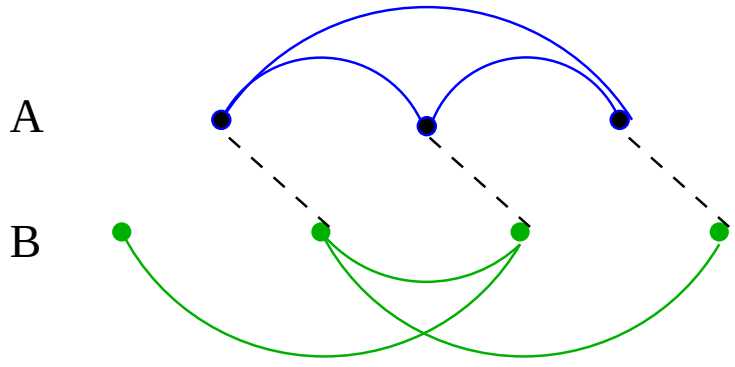
# *Contact Map Overlap (Cont.)*

The value of an alignment



**NP-Hard (Goldman, Istrail, Papadimitriou, 1999)**

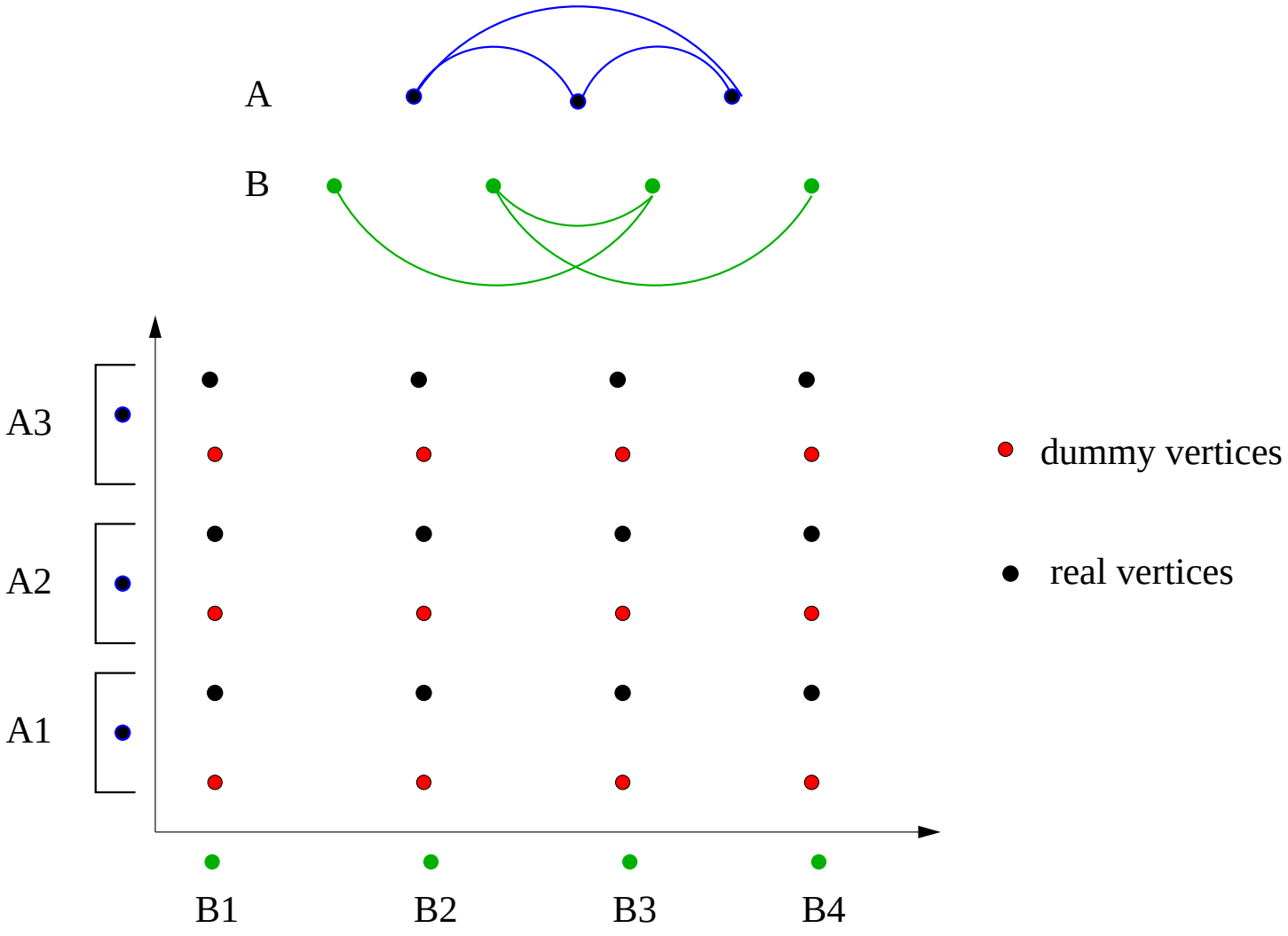
# *Contact Map Overlap (Cont.)*




An alignment of A to B

Contact map overlap is again a matching ptoblem.

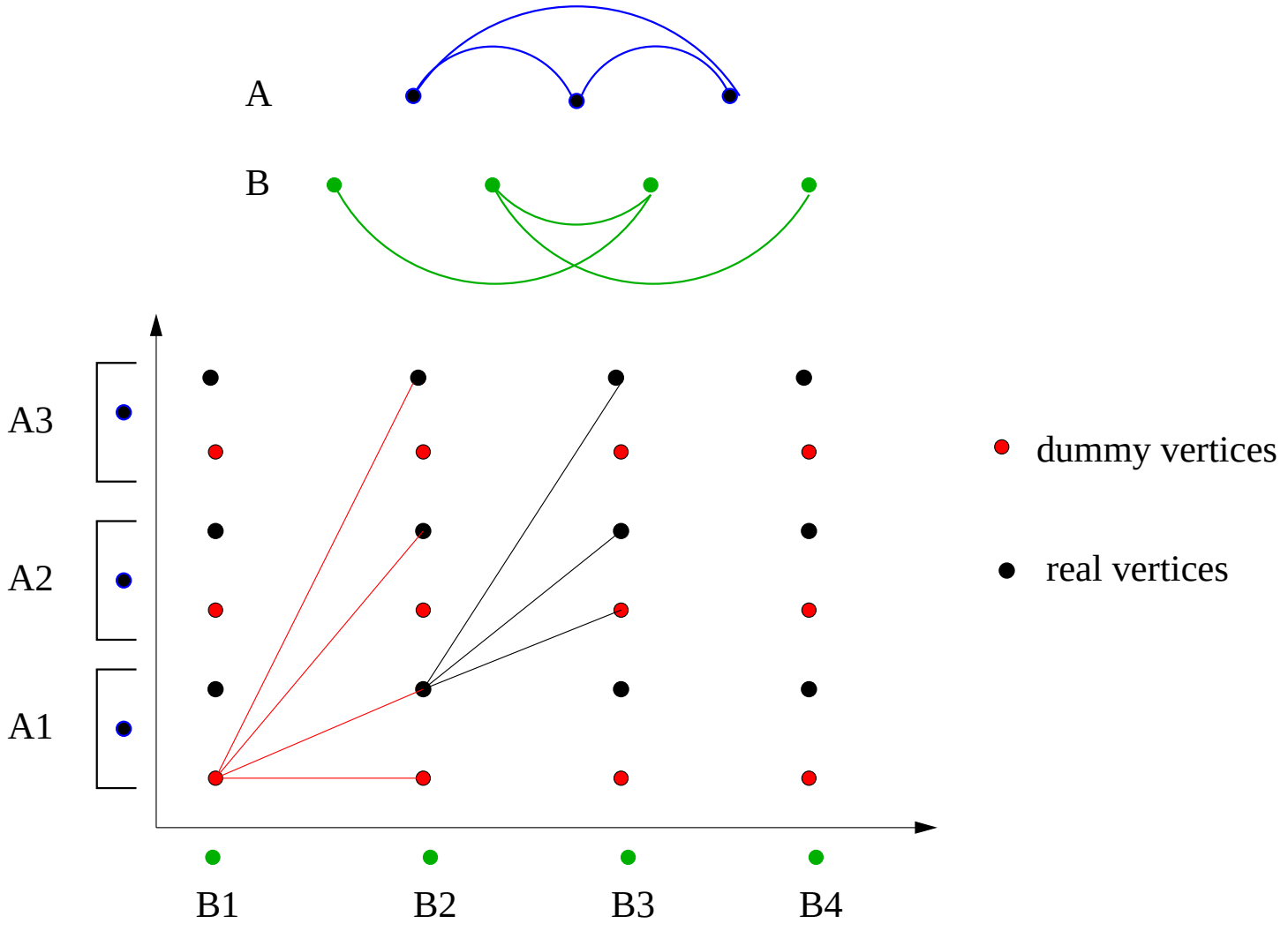
# Network flow for Contact Map Overlap



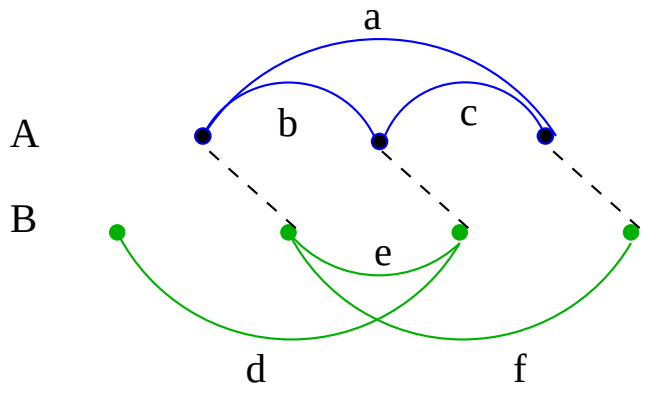

Network flow graph. Real vertices model possible alignments.

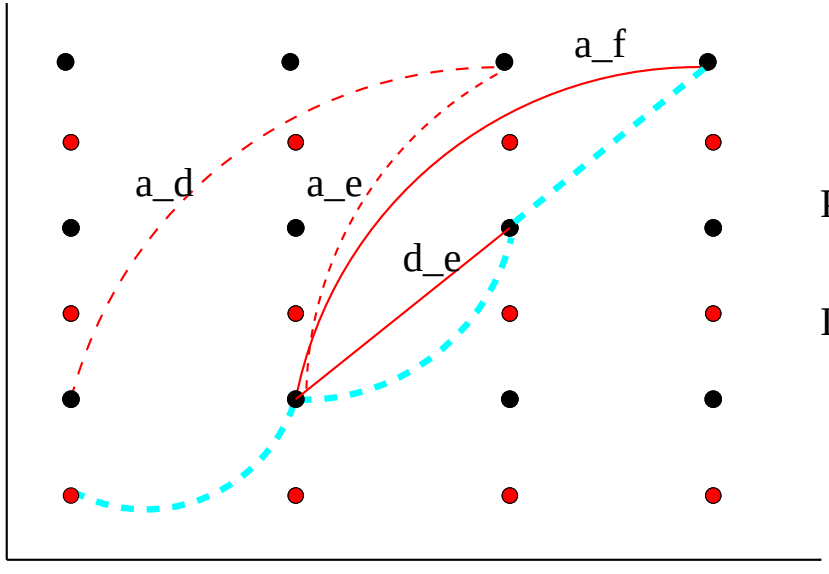
Dummy vertices model ommissions.

# Network flow for Contact Map Overlap (Cont.)



# Contact Map Optimization Problem



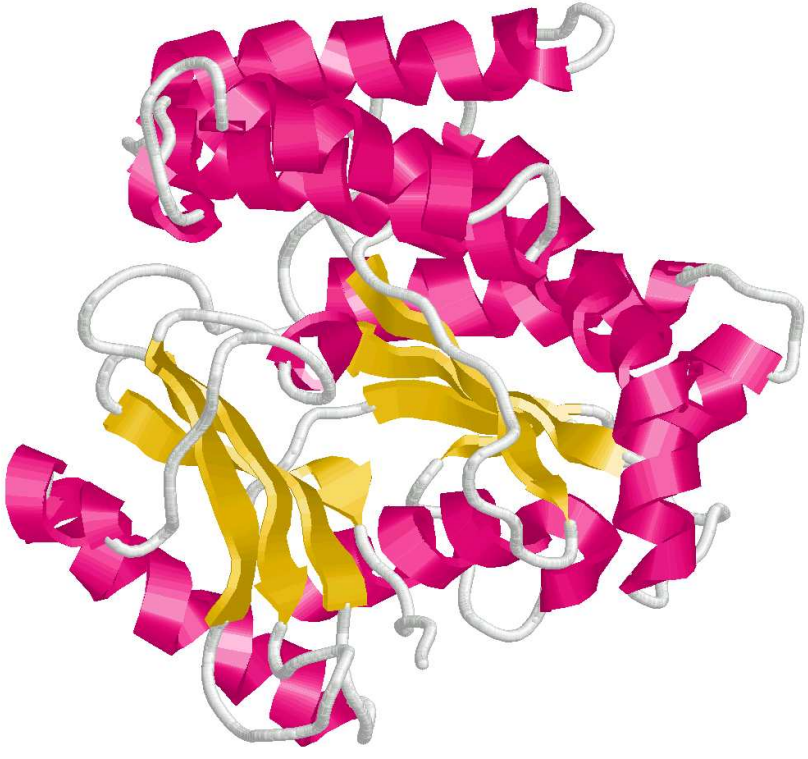

Path activates arcs a_f and d_e.

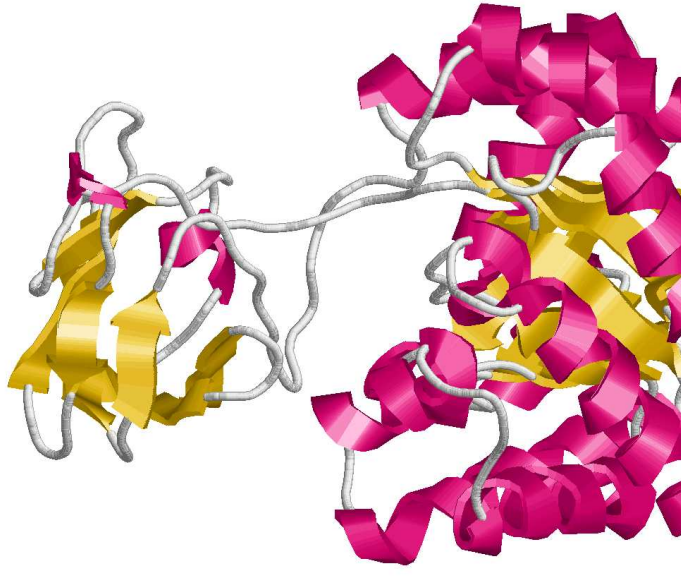
It corresponds to the above given alignment of A to B.

Find the path in the network flow graph activating maximum number of arcs.

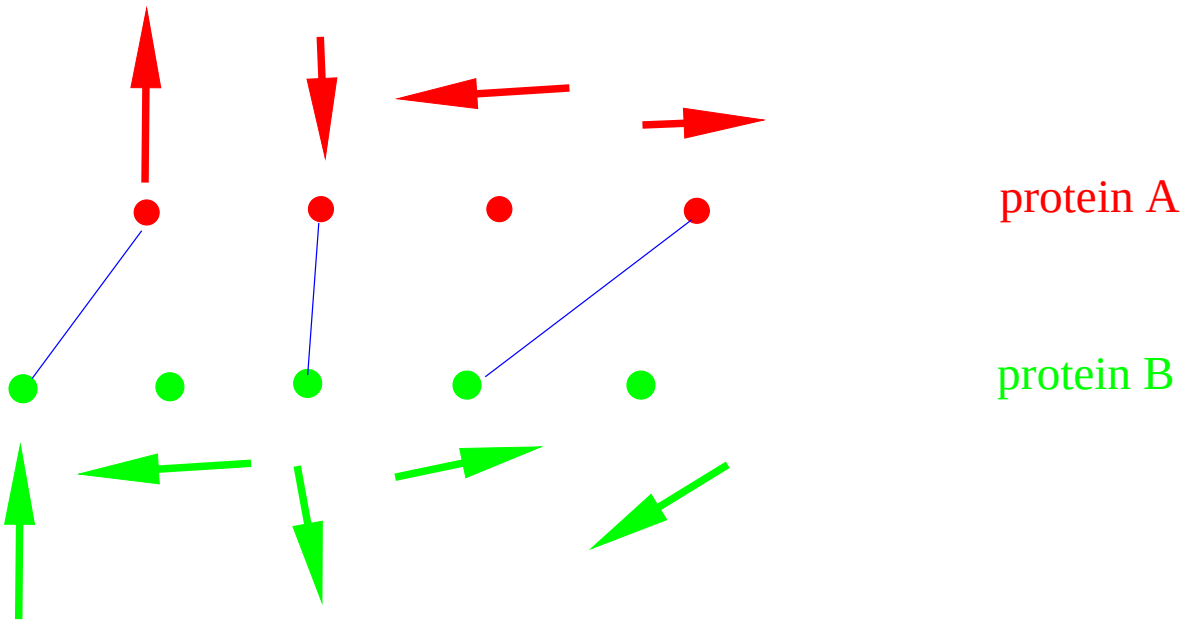
# *VAST approach for proteins comparison*

Again: how to compare these two 3D structures???





# *Vector Alignment Search Tool (Cont.)*



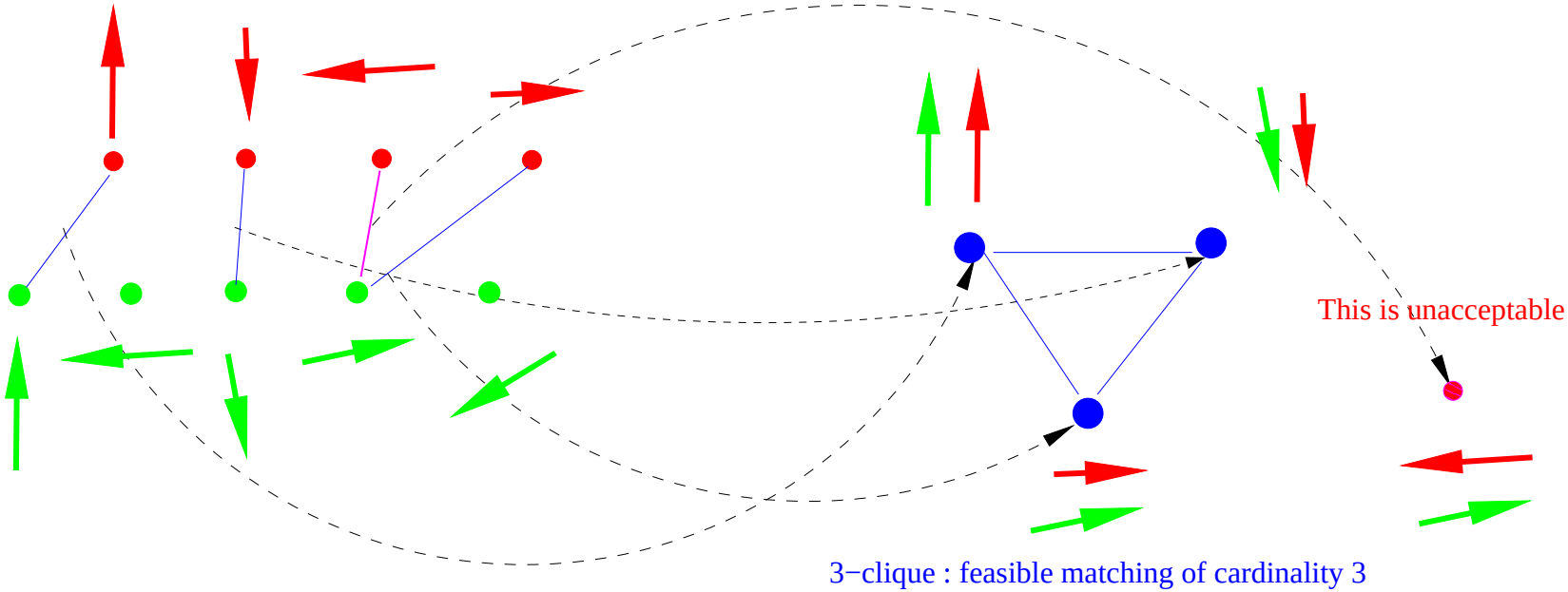

Attention: in this approach any node is a secondary structure.

Advantage : reduction of the solutions space size!

Gibrat&Madej&Bryant, Surprising similarities in structure comparison. Curr. Opin. Struct. Biol., 1996, 6(3):377-385

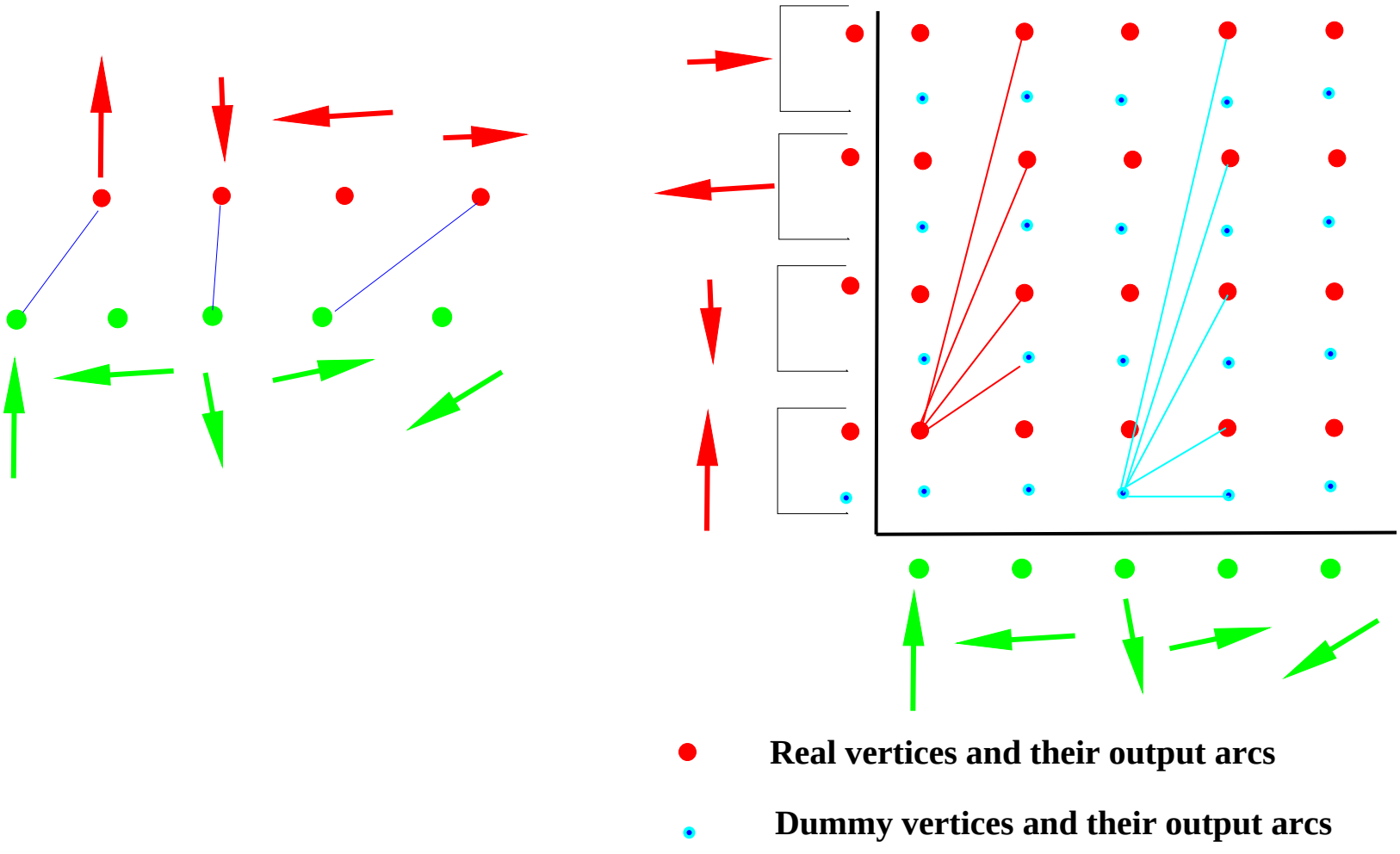
`http:www.ncbi.nlm.hih.gov/Structure/VAST/vast.sh`

# *VAST: from matching to maximum clique*



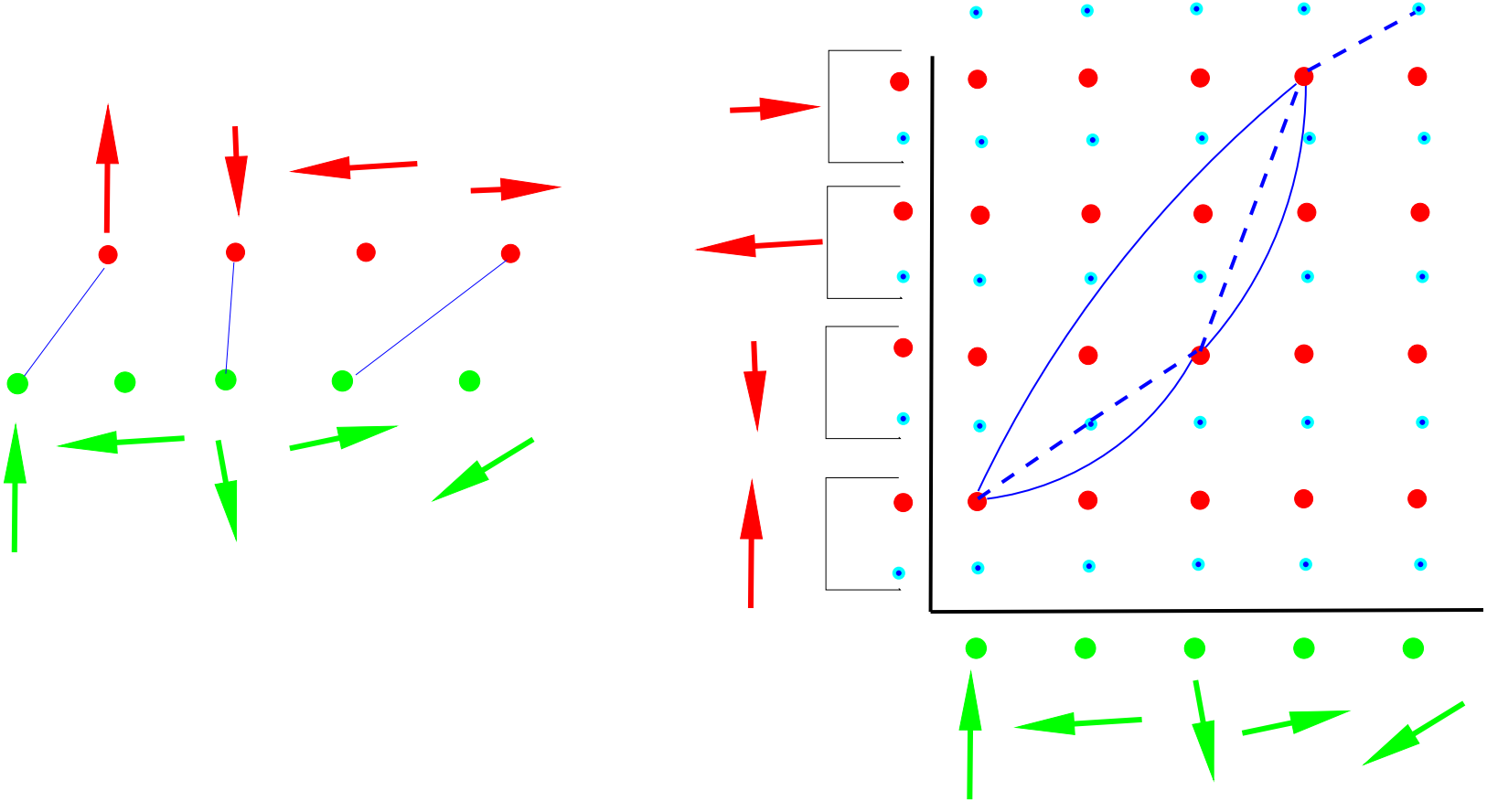

Find a translation and rotation superimposing one couple of vectors to another one. RMSD (Root mean square deviation) is afterwards used to measure the similarity between these couples of vectors. Similar couples are connected by arcs.

# *Network flow for VAST approach*




Possible arcs in the network graph. Dummy vertices allow modeling omissions.

# *Network flow for VAST approach (cont.)*

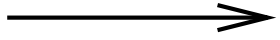



Optimal matching is equivalent to fi nding maximum edge weighted clique in an appropriate graph

# *Toy example*

A T C G

C A T G

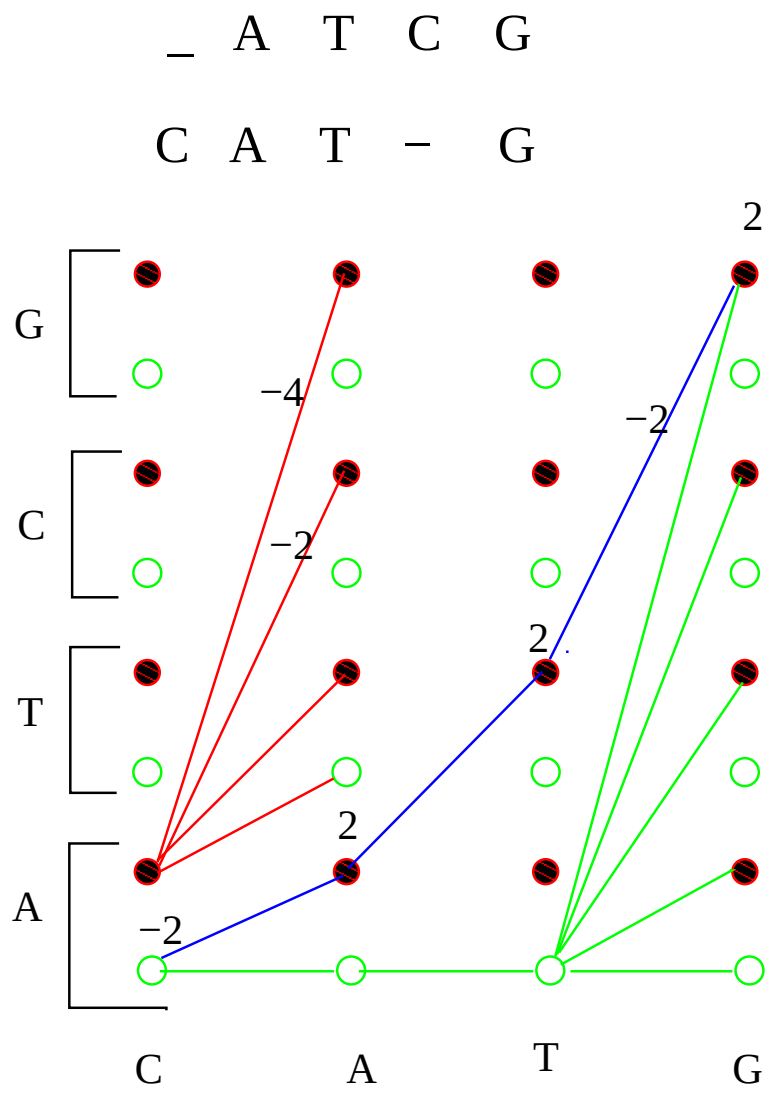


_ A T C G

C A T − G

Draw the network flow graph allowing to obtain this alignment.

# Toy example : solution



# Integer programming models

# *Network flow formulation: notations*

Interactions : $L \subseteq \{(i,j) \mid 1 \leq i < j \leq m\}$ : all

$G(V,E)$–digraph with $V = \{(i,k) \mid i = 1,m;\ k = 1,n\}$ ; where

$$E = \{((i,k),(j,l)) \mid (i,j) \in L,\ 1 \leq k \leq l \leq n\}$$

Variables: $z_e$, $e \in E$, and $y_v$, $v \in V$.

# *Properties of the set of feasible threadings* $Y$

$$\sum_{k=1}^{n} y_{ik} = 1 \qquad i = 1, m \tag{1}$$

$$\sum_{l=1}^{j} y_{il} - \sum_{l=1}^{j} y_{i+1,l} \geq 0 \qquad i = 1, \ldots, m-1,\ j = 1, \ldots, n-1 \tag{2}$$

$$y_{ik} \in \{0, 1\} \qquad i = 1, m,\ k = 1, n \tag{3}$$

(3) $y_{ik} = 1 \Leftrightarrow$ block $i$ is on position $k$
(1) block $i$ is on exactly one position
(2) if block $i + 1$ is on positions $l$, then block $i$ is before position $l$

**Proposition 1** *The polytope* $Y$ *is integral, i.e. it has only integer-valued vertices.*

# A quadratic model

$$\sum_{i=1}^{m}\sum_{k=1}^{n} s_{ik}y_{ik} + \sum_{(i,j)\in E} c_{ikjl}y_{ik}y_{jl} \Rightarrow \min \tag{4}$$

$$y \in Y \tag{5}$$

# Linearizing the model

$$\sum_{i=1}^{m}\sum_{k=1}^{n} s_{ik}y_{ik} + \sum_{(i,j)\in E} c_{ikjl}z_{ikjl} \Rightarrow \text{min} \quad (6)$$

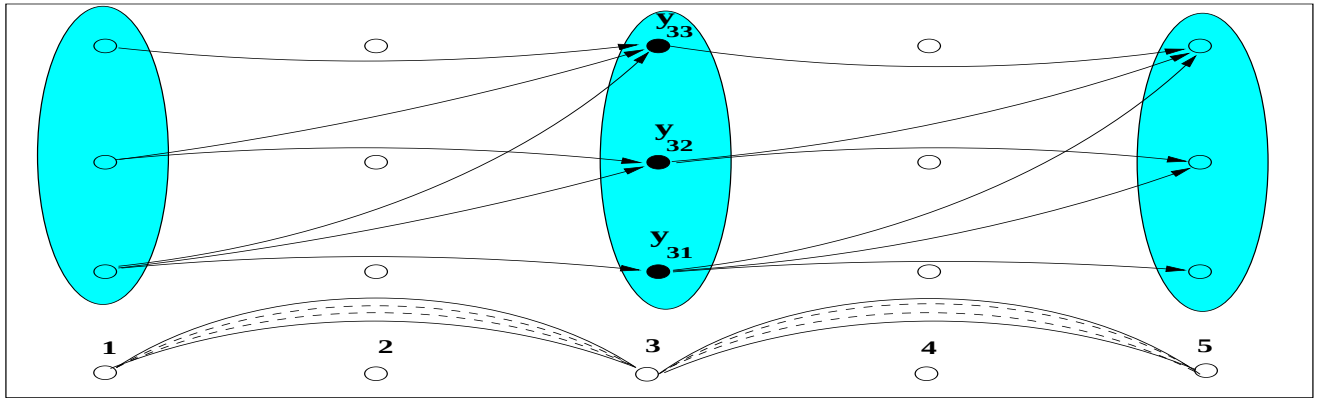
$$y \in Y \quad (7)$$

$$z_{ikjl} \leq y_{ik} \quad (8)$$

$$z_{ikjl} \leq y_{jl} \quad (9)$$

$$y_{ik} + y_{jl} - z_{ikjl} \leq 1 \quad (10)$$

# Strengthening the model




$y_{ij}$ are binary : the corresponding $z_{ikjl}$ are relaxed.

$$
\begin{array}{rcrcrcrcl}
y_{31} & + & y_{32} & + & y_{33} & = & 1 & & \text{as defined in } Y \\
z_{1133} & + & z_{1233} & + & z_{1333} & = & y_{33} & & \Gamma^{-1}(y_{33}) \\
 & & z_{1132} & + & z_{1232} & = & y_{32} & & \Gamma^{-1}(y_{32}) \\
 & & & & z_{1131} & = & y_{31} & & \Gamma^{-1}(y_{32}) \\
 & & & & y_{33} & = & z_{3353} & & \Gamma(y_{32}) \\
 & & & & y_{32} & = & z_{3253} & + z_{3252} & \Gamma(y_{32}) \\
 & & & & y_{31} & = & z_{3153} & + z_{3152} + z_{3151} & \Gamma(y_{31})
\end{array}
$$

# Strengthening the model (cont.)

$$\sum_{i=1}^{m}\sum_{k=1}^{n} s_{ik}y_{ik} + \sum_{e\in E} c_e z_e \Rightarrow \min \tag{11}$$

$$y_{ik} = \sum_{l=k}^{n} z_{ikjl} \quad (i,j) \in L,\ k = 1,n \tag{12}$$

$$y_{jl} = \sum_{k=1}^{l} z_{ikjl} \quad (i,j) \in L,\ l = 1,n \tag{13}$$

$$y \in Y \tag{14}$$

$$z_e \geq 0 \quad e \in E \tag{15}$$

# *Is the protein threading in P?*

Observation : 1 200 000 alignements computed (all FROST data bank);
*only 5%* of the instances the LP relaxation is not integer;
Statistics: $1\times11$ nodes, $2\times10$ nodes, $1\times9$ nodes,
$5\times8$ nodes, $3\times7$ nodes, $3\times6$ nodes,
Majority: 2 nodes - in which cases the value of the solution is 0.5

The subset of real-life PTP is polynomially solvable!

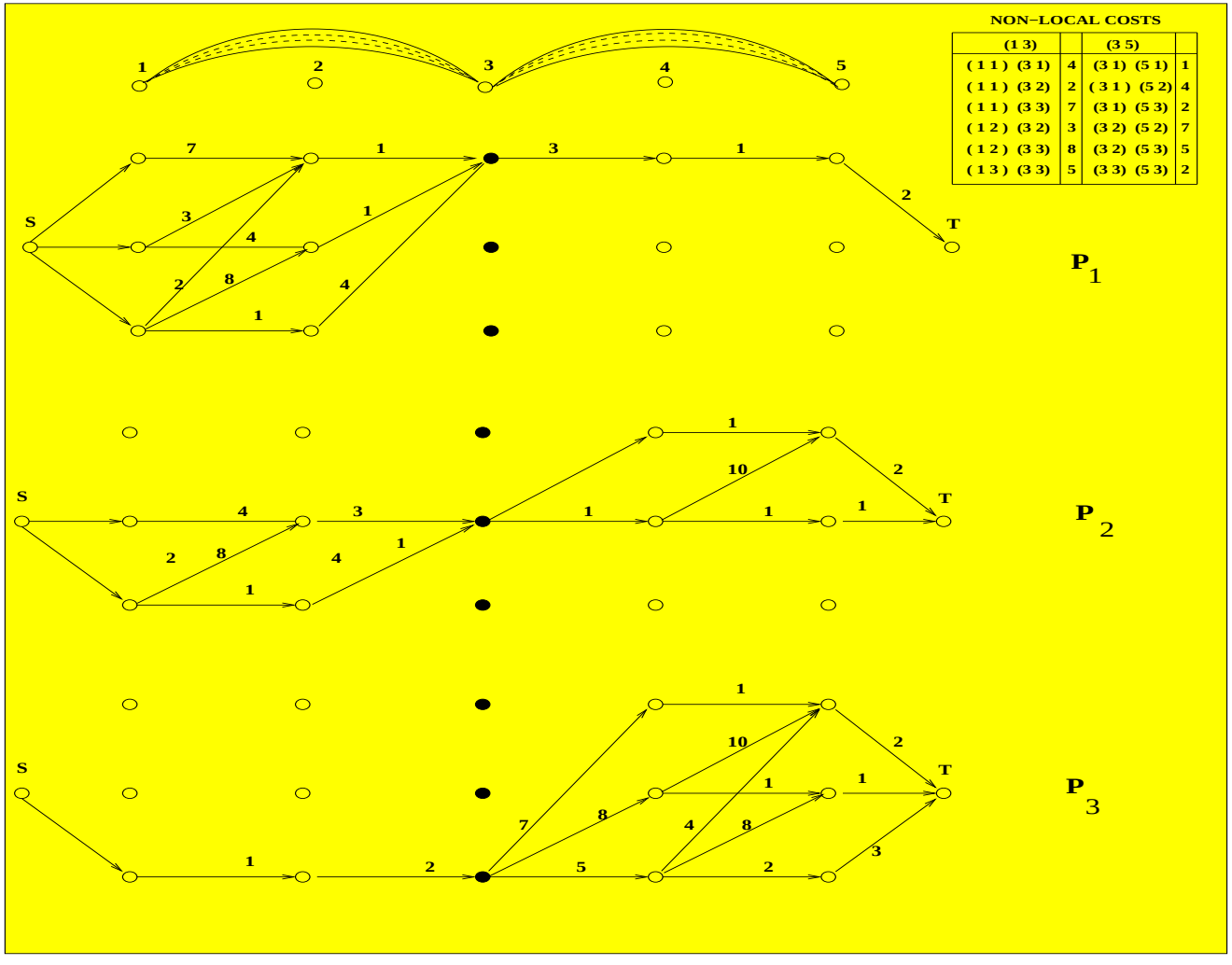
Validated when using the FROST score function.

This is not true when using randomly generated score function.

## *Can we do better?*

Yes, using divide and conquer startegy!

# Split and conquer strategy




The main problem is decomposed into three subproblems.

# *Lagrangian relaxation and duality*

**Idea**: drop part of the constraints in order to make the relaxed problem easier to solve; introduce penalties for violating them in the objective function.

**IP problem:**

$$\begin{aligned} Z_{IP} &= \min cx \\ s.t. \quad & x \in X \text{—“easy” constraints} \\ & Ax = b \text{—“complicated” constraints} \end{aligned}$$

**Lagrangian relaxation**: $Z_{LR}(\lambda) = \min \{cx + \lambda(b - Ax) | x \in X\}$

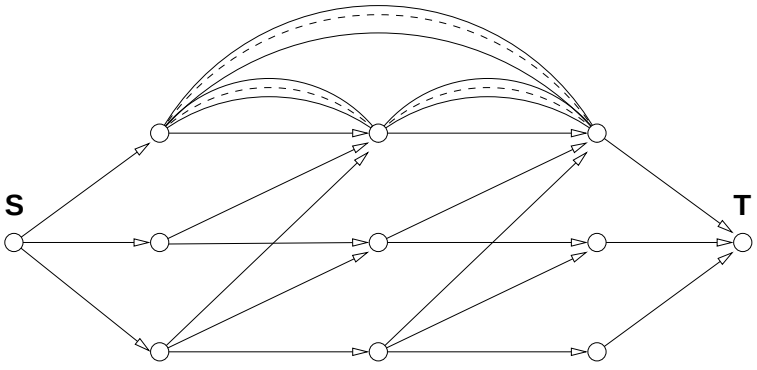
- LR is also an IP problem, but easier to solve than IP
- LR is relaxation of IP for *any* $\lambda$ (i.e. $Z_{LR}(\lambda) \leq Z_{IP}$)

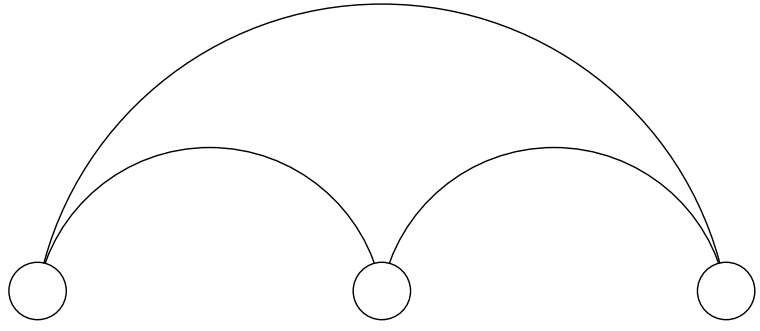
**Lagrangian dual**: $Z_{LD} = \max_{\lambda} Z_{LR}(\lambda)$

- LD is better than LP: $Z_{LP} \leq Z_{LD} \leq Z_{IP}$

# Topology of PTP

- Reminder: $L$ is the inter-block interactions graph

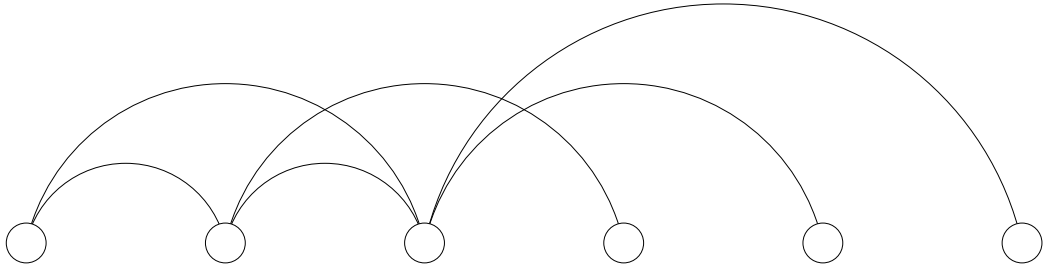


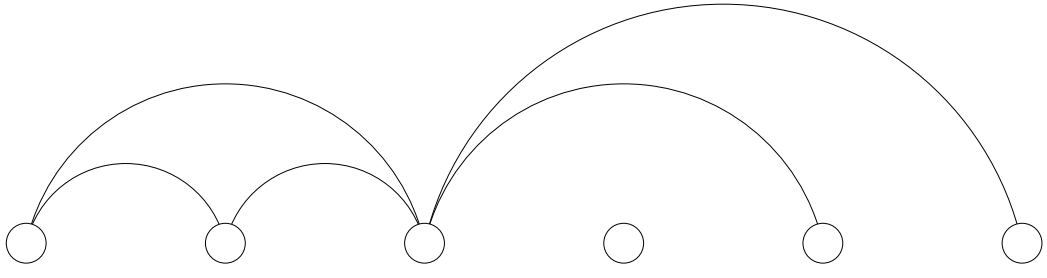
- *complexity* of PTP strongly depends on the *topology* of $L$
  - $L = \emptyset$ or contains only local links $\longrightarrow$ PTP is polynomially solvable
  - $L$ dense $\longrightarrow$ PTP is NP-hard
- What about intermediate cases ?

# *SP#1: $L$ contains no crossing edges*

Crossing edges:

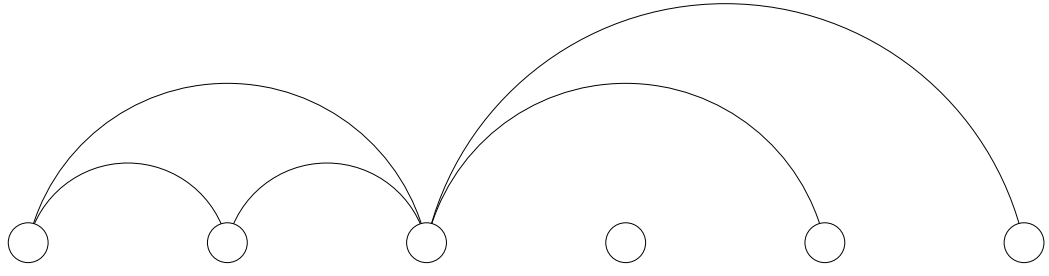


Non-Crossing edges:



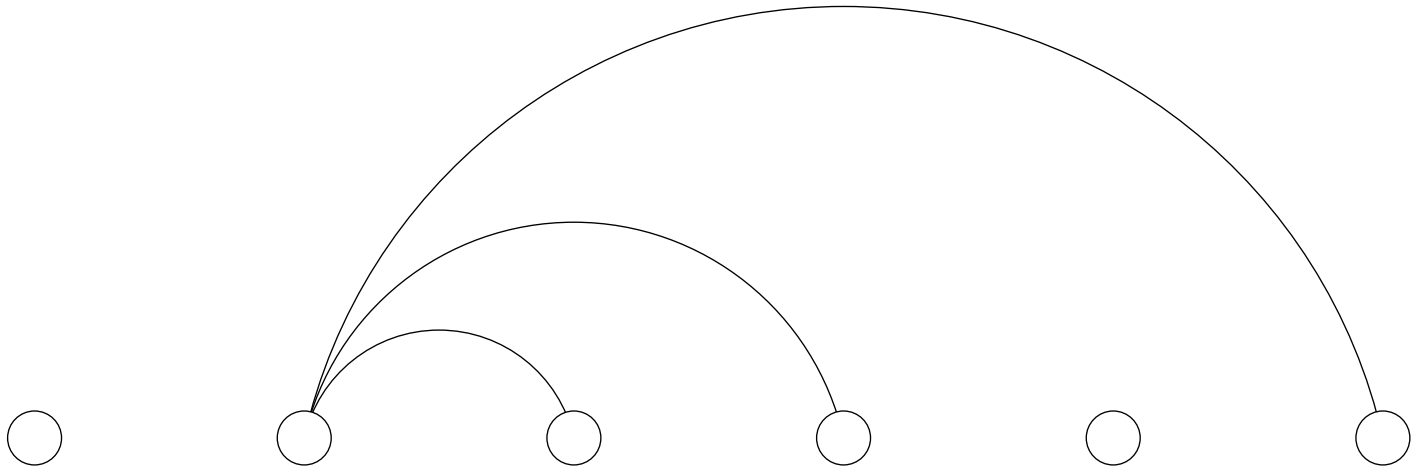
# *SP#1, $L$ contains no crossing edges*

- $l =$ number of links in $L$
- $n =$ number of vertices in a layer
- SP#1 can be solved using a DP approach, with complexity $O(ln^3)$.



# SP#2: $L$ is a star

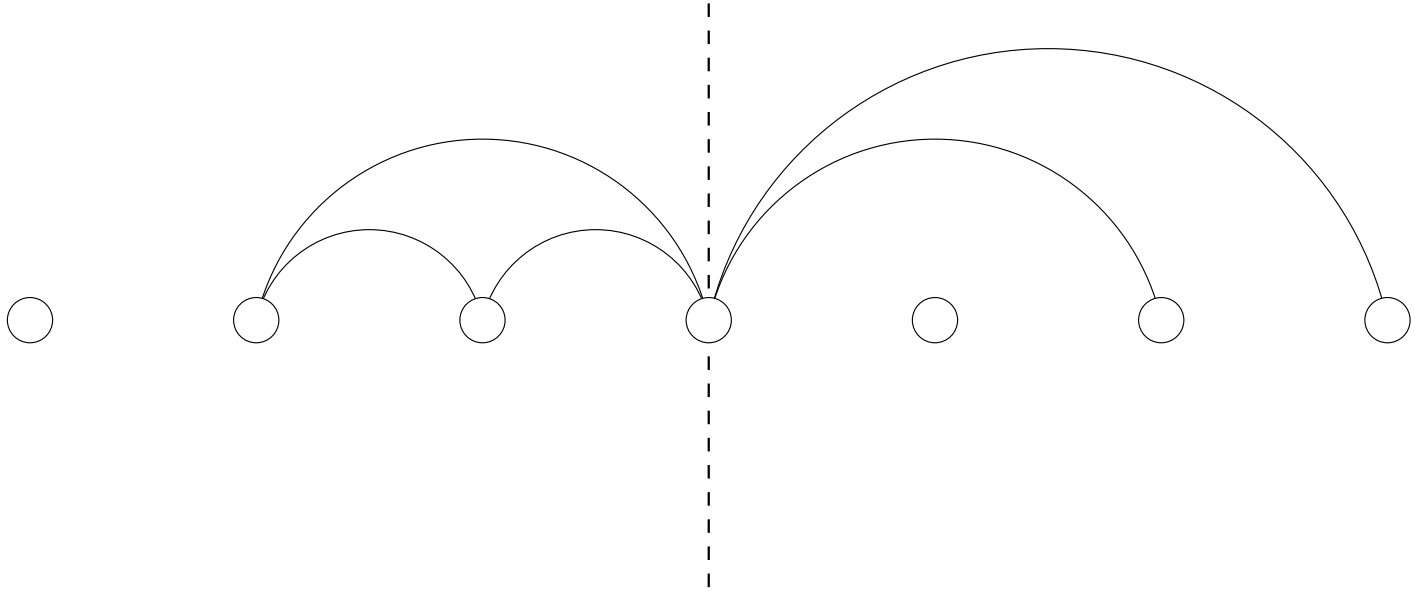
- Star: common left/right end for all links



- $O(ln^2)$ complexity using DP programming

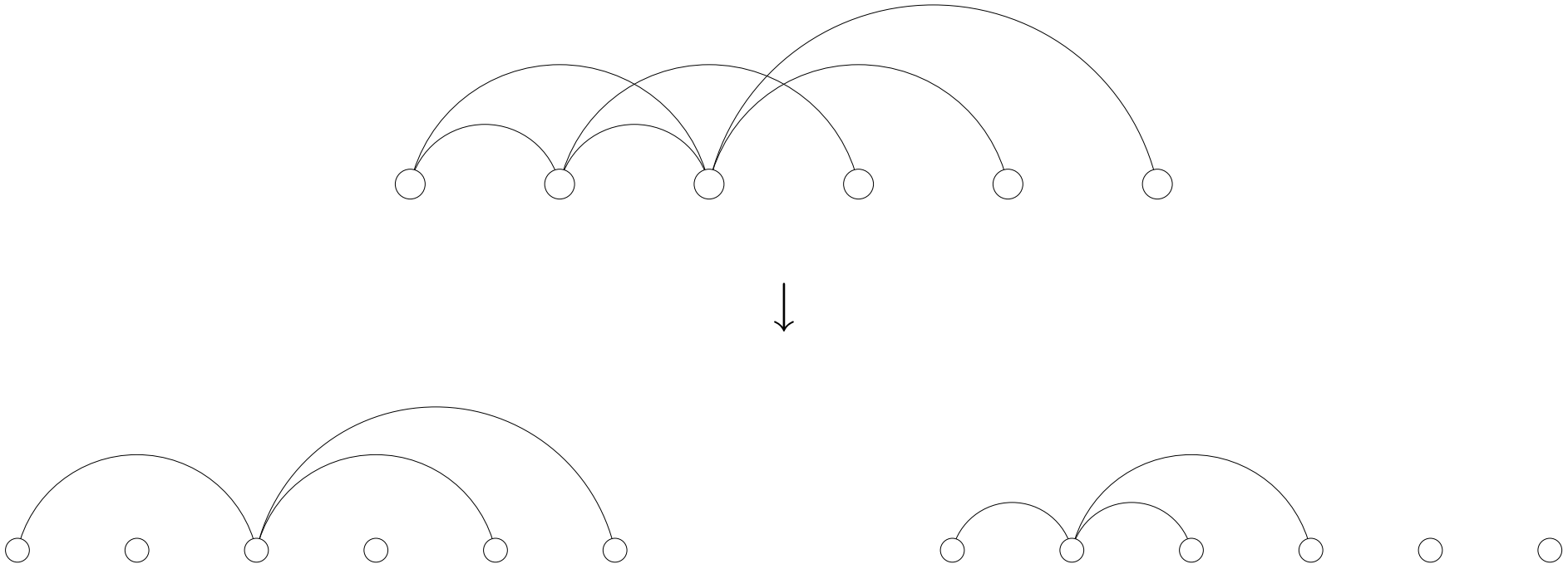
# *SP#3: sequence of independent subproblems*

- partition s.t. no link is cut

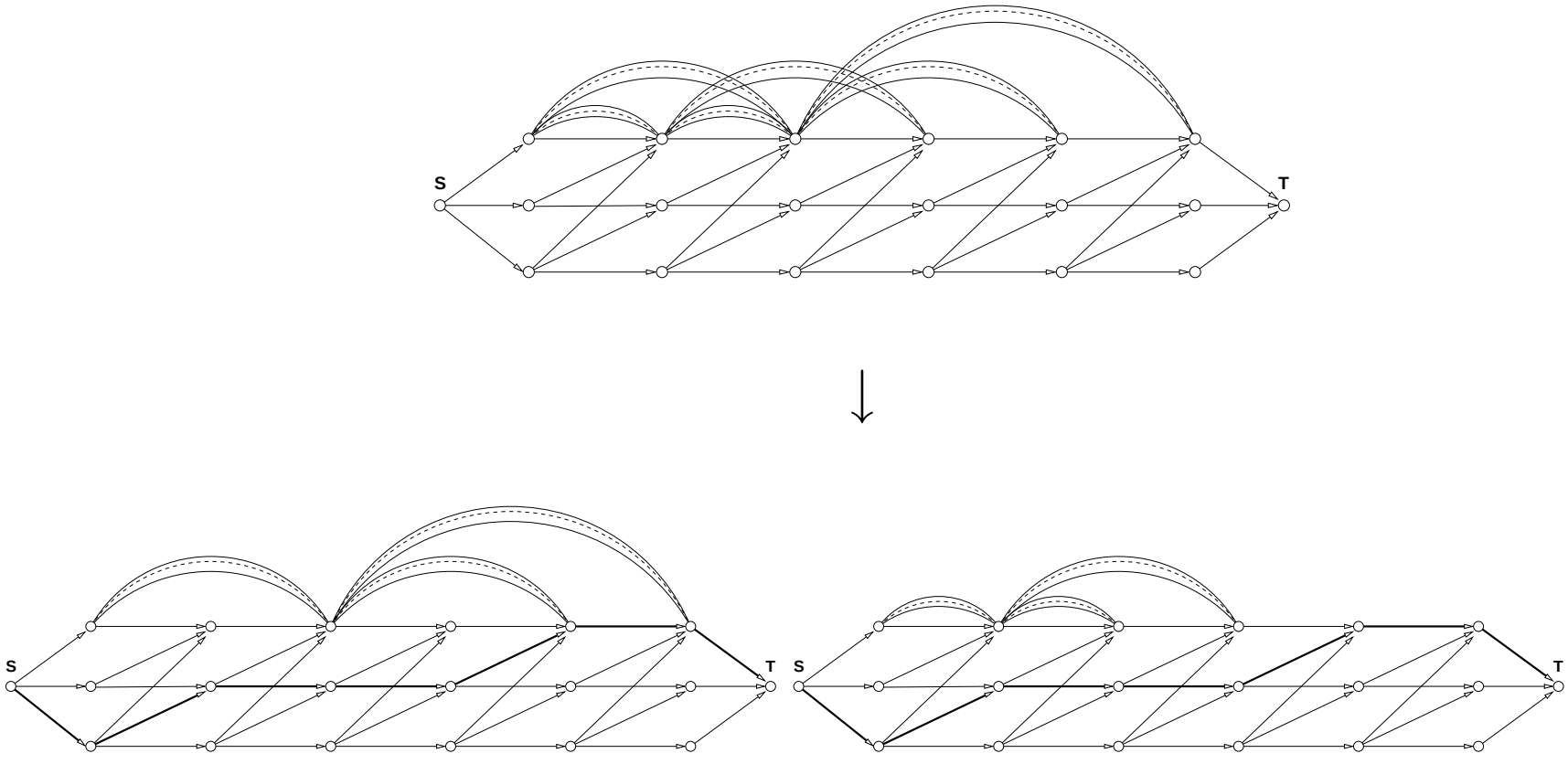


- let $r =$ number of independent subproblems
- $O(rn^2)$ complexity after having solved each subproblem

# *From graph decomposition . . .*



# . . . to cost-splitting technique




- solve independently and enforce identical solutions

# *Optimization*

- equality constraint between sub-problems is the hard one
- Practical resolution:
  - Lagrangian relaxation
  - Maximization of the dual using its sub-gradient
  - In theory, only gives a lower bound on the objective
  - Branch and Bound for exact resolution
  - In practice, the solution is obtained at the root

# Cost-splitting Lagrangian relaxation

$L = L^1 \bigcup L^2 ... \bigcup L^t$ where each $L^s$ induces an easy solvable $PTP(L^s)$,

$$v_{ip}^L = \min \left\{ \sum_{s=1}^{t} ( \sum_{i=1}^{m} d_i^s y_i^s + \sum_{(i,k) \in L^s} c_{ik} z_{ik} ) \right\} \tag{16}$$

subject to:

$$y_i^1 = y_i^s, \qquad s = 2, t \tag{17}$$

$$y^s = (y_1^s, .. y_m^s) \in Y, \qquad s = 1, \ldots, t \tag{18}$$

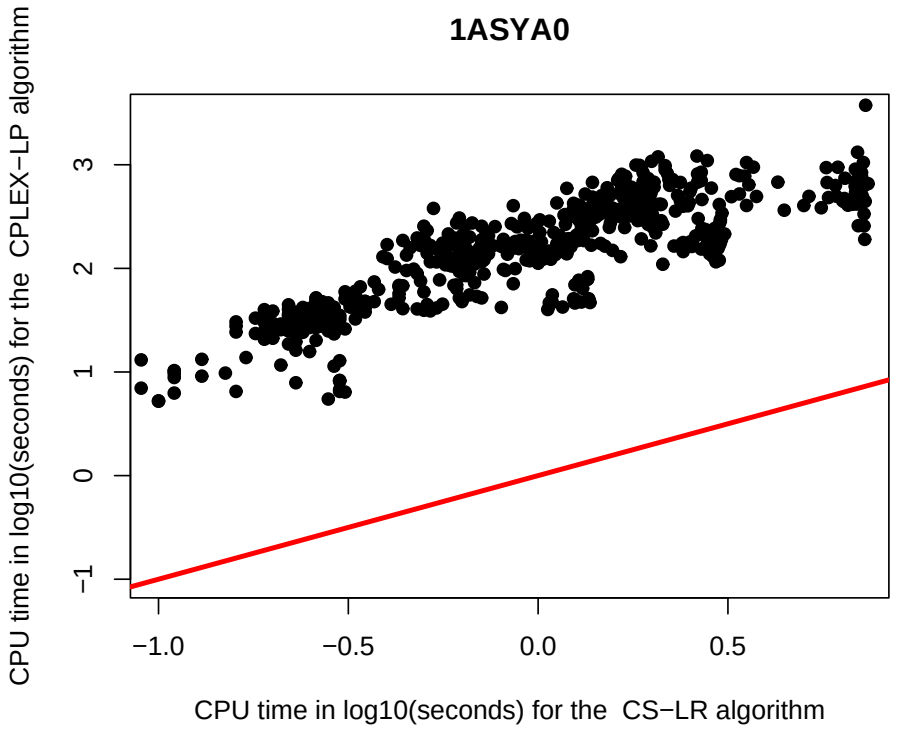
$$y_i^s = A_i z_{ik}, \;\; y_k^s = A_k z_{ik} \qquad s = 1, \ldots, t \qquad (i,k) \in L^s \tag{19}$$

$$z_{ik} \in B^{\frac{n(n+1)}{2}} \qquad s = 1, \ldots, t \qquad (i,k) \in L^s \tag{20}$$

$$v_{csd} = \max_{\lambda} \min_{y} \sum_{s=1}^{t} ( \sum_{i=1}^{m} d_i^s(\lambda) y_i^s + \sum_{(i,k) \in L^s} c_{ik} z_{ik} ) = \max_{\lambda} \sum_{s=1}^{t} v_{ip}^{L^s}(\lambda) \tag{21}$$

subject to (18), (19) and (20).

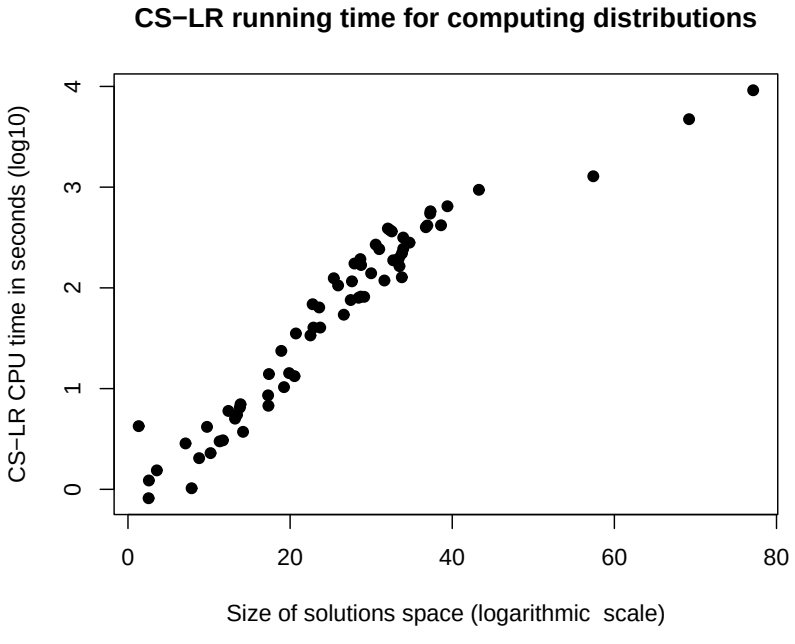
# *Cost-Splitting LR (CS-LR) versus LP*




Computing 962 threading instances associated to the template 1ASYA0. The linear curve in the plot is the line $y = x$. We observe a signifi cant performance gap between the algorithms. CS-LR is **from** $100$ **to** $250$ **times faster** than LP relaxation.

Figure 9: Cost-Splitting Lagrangian Relaxation versus Linear Programming Relaxation

# *More experimental results*




Each point in this plot corresponds to the total time required by CS-LR algorithm to compute one distribution determined by approximately 200 alignments of the same size. About 60 distributions have been computed which needed solving about 12000 alignments totally. The size of the biggest instance is $O(10^{77})$.

# *Conclusions*

- MIP formulations are very convenient for PTP;
- complete integration in FROST and its application on the GRID;
- the commercial package CPLEX of ILOG is avoided using a dedicated software for PTP (based on LR).
- computational results and comparisons between exact and approximated methods are provided. Huge real-life instances have been solved.
- Finding the exact global minimum in the optimal threading permitted FROST sensibility and quality of prediction to be improved (+7% and +5% respectively)

Merci!