Lagrangian approaches for solving PTP

Rumen Andonov and Nicola Yanev

joint work with

Stefan Balev, Philippe Veber and Vincent Poirriez

in the framework of our colaboration with J-F. Gibrat

and A. Marin

Outline of the talk

- Introducing the problem
- PTP is a matching problem
- Mixed Integer Formulation
- Lagrangian approaches for PTP
- Computational results

Protein Folding Problem

SNGIEASLLTDPKDVSGRTVDYIIAGGGLTGLTTAARLTENPNIS
SGSYESDRGPIIEDLNAYGDIFGSSVDHAYETVELATNNQTALIR

Protein Folding Problem

SNGIEASLLTDPKDVSGRTVDYIIAGGGLTGLTTAARLTENPNIS
SGSYESDRGPIIEDLNAYGDIFGSSVDHAYETVELATNNQTALIR

A sequence in a protein data bank

Protein Folding Problem

SNGIEASLLTDPKDVSGRTVDYIIAGGGLTGLTTAARLTENPNIS
SGSYESDRGPIIEDLNAYGDIFGSSVDHAYETVELATNNQTALIR

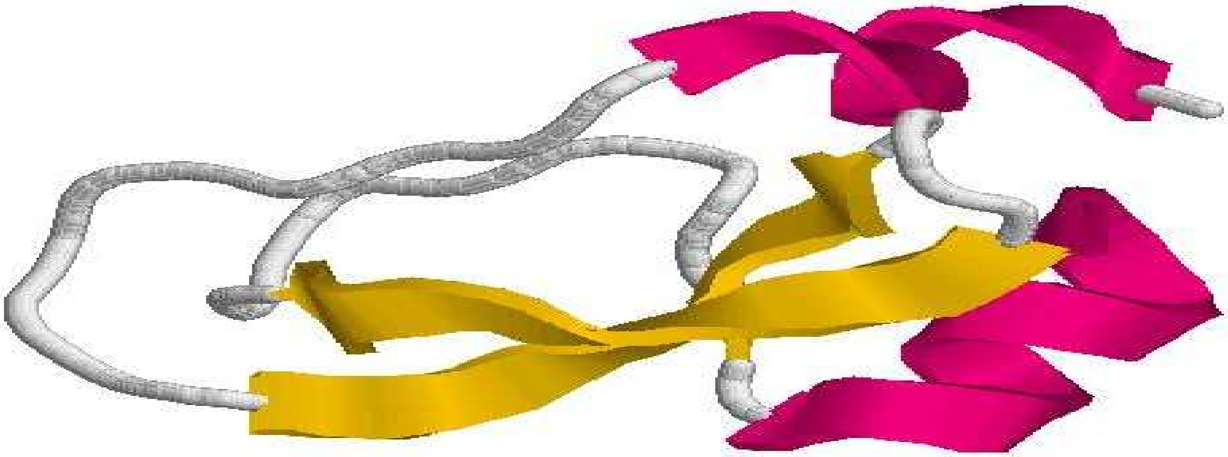
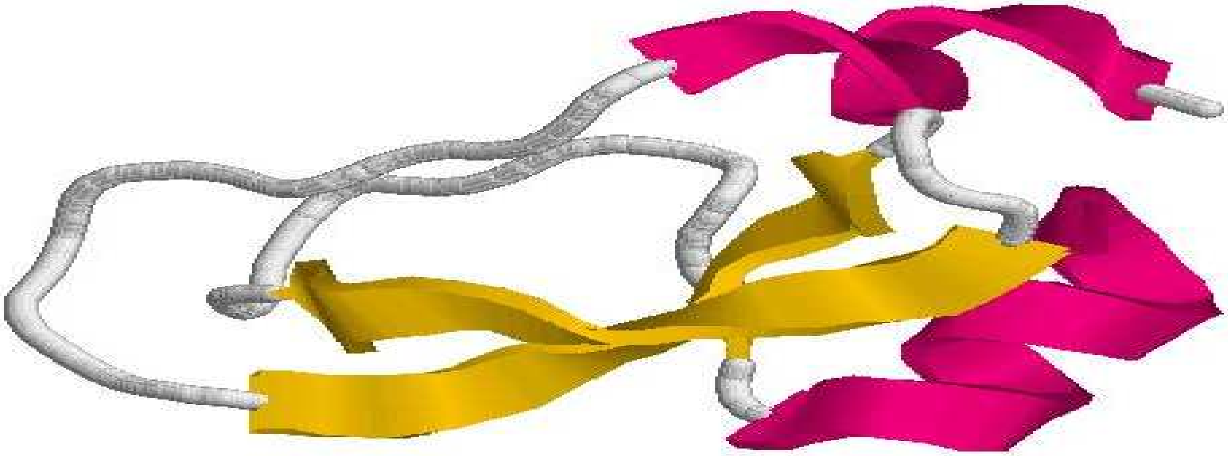


Figure 0: in fact this is its real (3D) shape

Protein Folding Problem

SNGIEASLLTDPKDVSGRTVDYIIAGGGLTGLTTAARLTENPNIS
SGSYESDRGPIIEDLNAYGDIFGSSVDHAYETVELATNNQTALIR

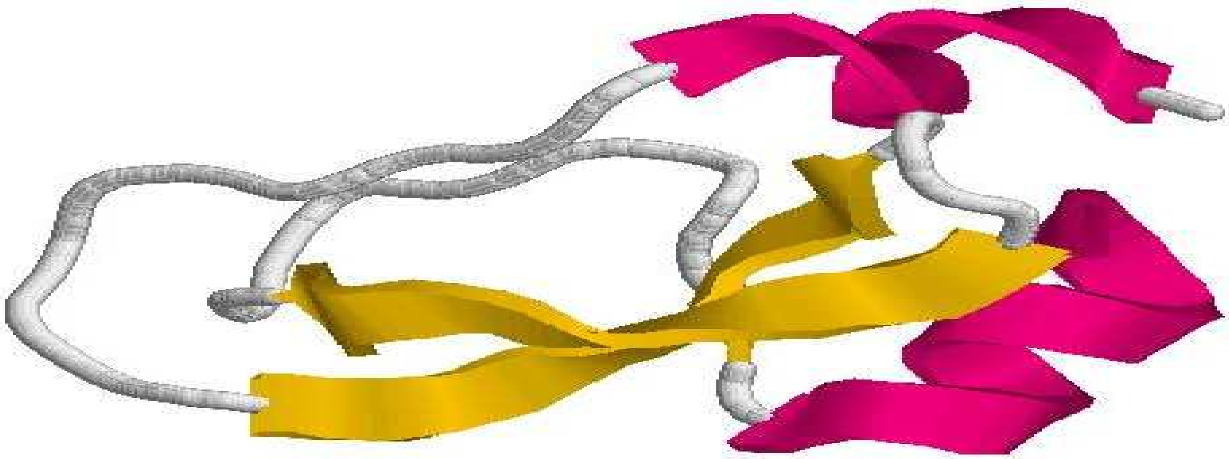


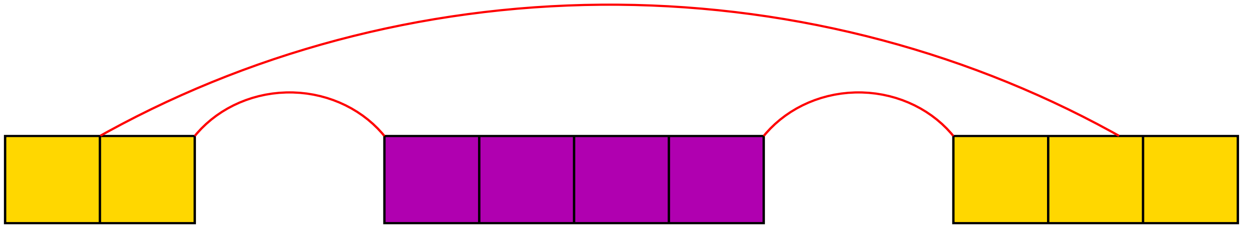
Protein Folding Problem :

- **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





structure template (core)

Figure 0: Generalized contact map graph—describes the interactions between the blocks

Protein Threading Problem

- **Associate to a protein sequence an already known *3D* structure.**

Protein Threading Problem

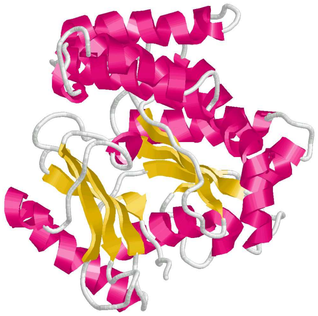
- **Associate to a protein sequence an already known *3D* structure.**

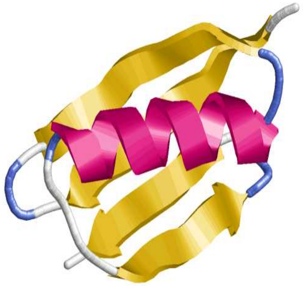
sequence $\overset{?}{\Longrightarrow}$

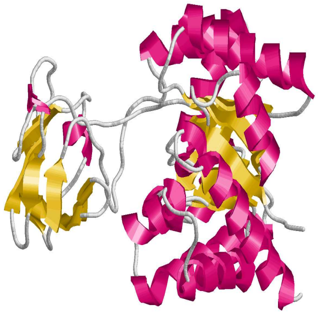
Protein Threading Problem

- **Associate to a protein sequence an already known *3D* structure.**

sequence $\overset{?}{\Longrightarrow}$







. . .

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

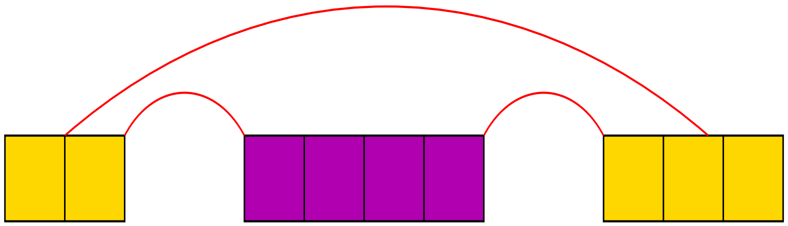
- choosing and objective function (score function) to evaluate any alignment of a sequence to a structural template

- finding the best alignment of the query sequence to each core in the library—NP-hard problem. (need of good combinatorial optimization alg.)

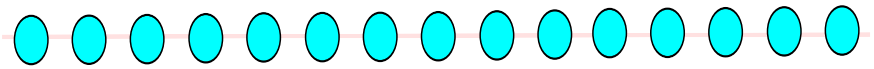
- choosing the most appropriate core based on normalized scores of the optimal alignments (requires good statistical model and the power of distributed computing)

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 $l_1 = 2, l_2 = 4, l_3 = 3$;

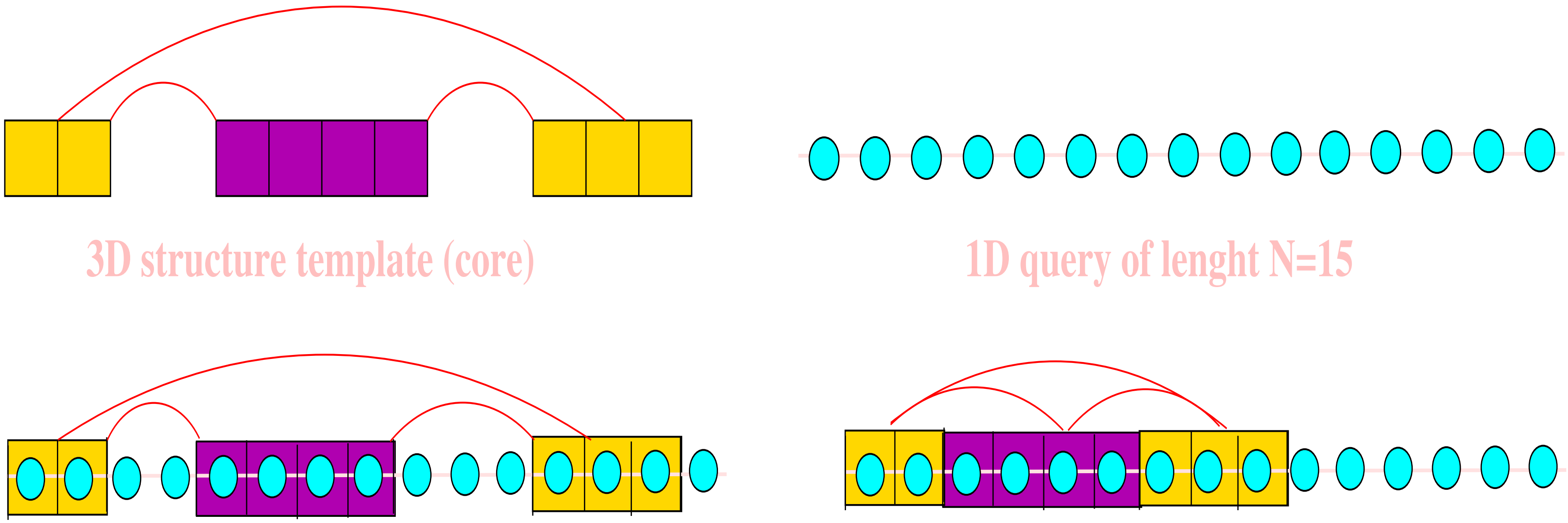
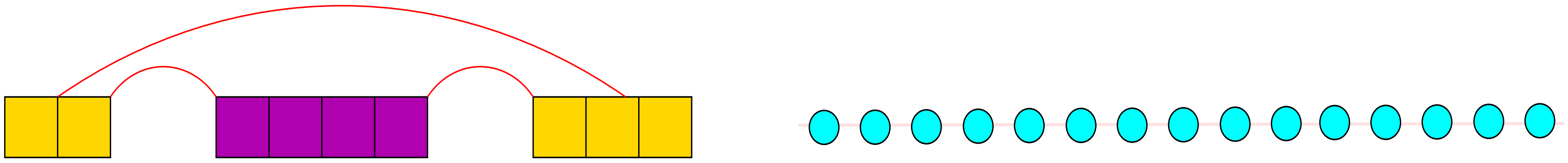


Figure 0: two possible alignments.

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.

Absolute and relative positions

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



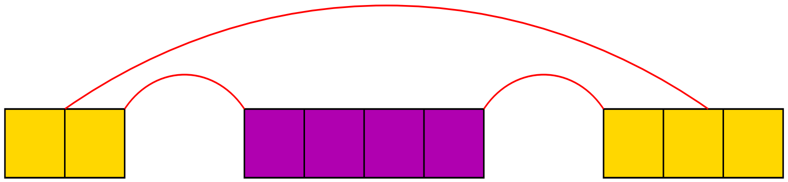
3D structure template (core)

1D query of lenght N=15

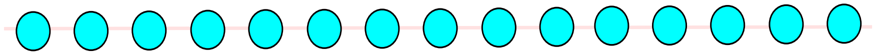
abs. position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
rel. pos. block 1	1	2	3	4	5	6	7								
rel. pos. block 2			1	2	3	4	5	6	7						
rel. pos. block 3							1	2	3	4	5	6	7		

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	15
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 $|T| = \binom{n-1+m}{m} = \frac{(n-1+m)!}{m!(n-1)!}$.

Size of the solution space

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

Size of the solution space

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

query	core	size		space
name	name	segm.	pos.	size
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

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

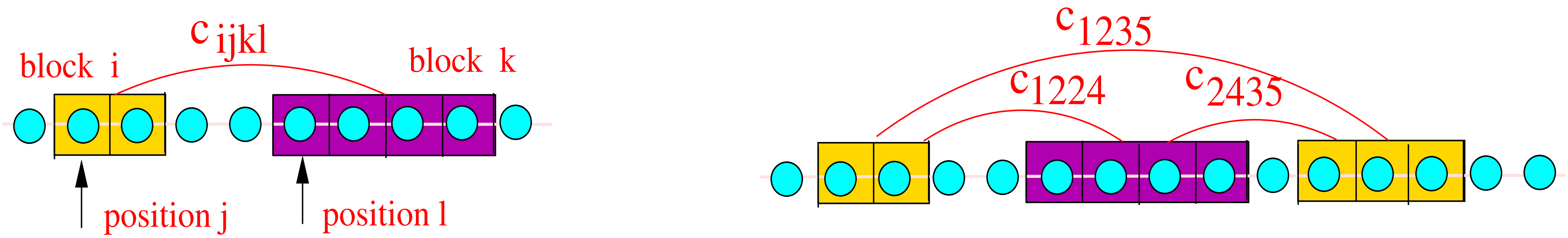


Figure 1: Here are all interactions..

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

The score function is supposed to be

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

Score function

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

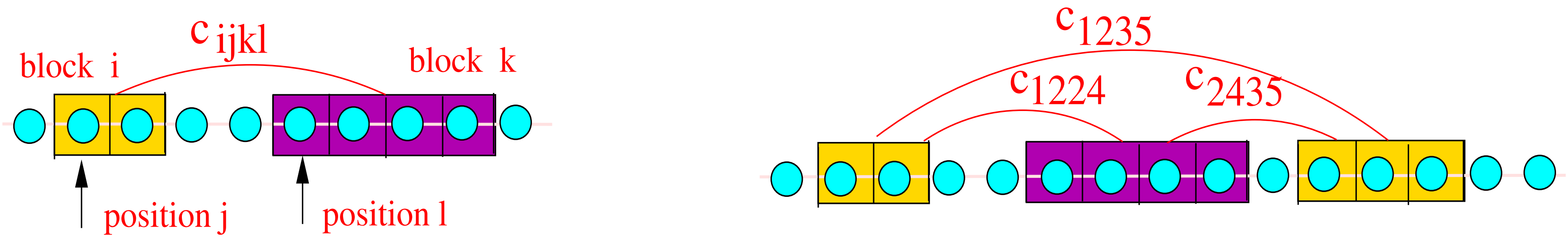


Figure 1: Here are all interactions..

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

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

$$\varphi(\pi) = \sum_{(i,k)\in E} c_{i\pi_i k\pi_k}$$

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 (Fold Recognition-Oriented Search Tool)?

- A database of known 3D templates
  - $\sim$ *1200*

FROST (Fold Recognition-Oriented Search Tool) ?

- A database of known 3D templates
- A series of filters

FROST (Fold Recognition-Oriented Search Tool) ?

- A database of known 3D templates
- A series of filters
  - one filter ↔ a specific fitness function (ff)

FROST (Fold Recognition-Oriented Search Tool) ?

- A database of known 3D templates
- A series of filters
  - one filter ↔ a specific fitness function (ff)
  - Currently two filters:
    - Local interaction filter **(Ali1D)** $ff : O(n^2)$
      dynamic programming

FROST (Fold Recognition-Oriented Search Tool) ?

- A database of known 3D templates
- A series of filters
  - one filter ↔ a specific fitness function (ff)
  - Currently two filters:
    - Local interaction filter **(Ali1D)** $ff : O(n^2)$
      dynamic programming
    - Space interaction filter **(Ali3D)** $ff : NP$-complete
      MIP models solved using Lagrangian relaxation.

FROST (Fold Recognition-Oriented Search Tool) ?

- A database of known 3D templates
- A series of filters {Ali1D: $O(n^2)$; Ali3D: NP}

The heaviest phase: distribution computing

- Procedure

  for each template and each filter

  - *Align non homologous sequences.*
  - *5 distributions using about 200 sequences.*

The heaviest phase: distribution computing

- Procedure

- Complexity

  1200 templates

  $O(n^2)$

  NP-complete

  - *1,200,000 alignments* `Ali1D` *and* `Ali3D`

The heaviest phase: distribution computing

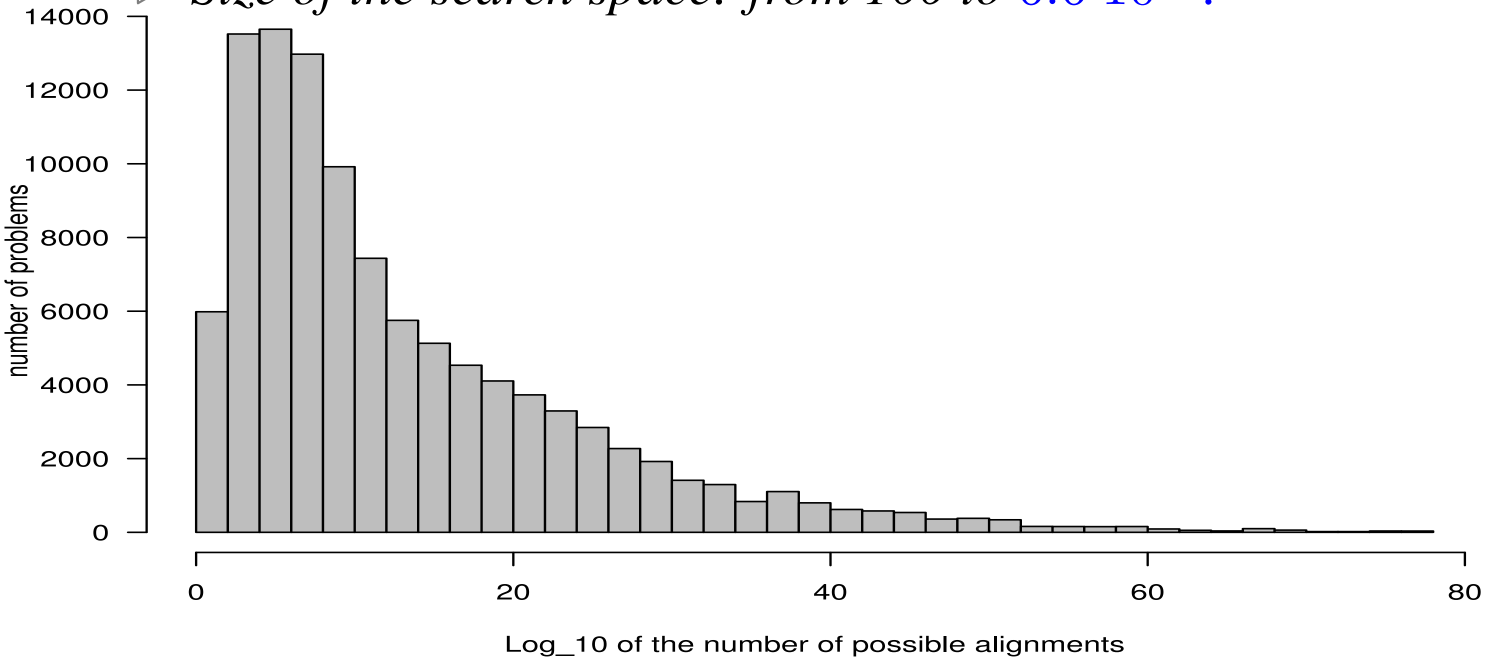
- Procedure
- Complexity

  1200 templates

  - *1,200,000 alignments* `Ali1D` *and* `Ali3D`
  - *Size of the search space: from 100 to* $6.6\ 10^{77}$.

$O(n^2)$

NP*-complete*

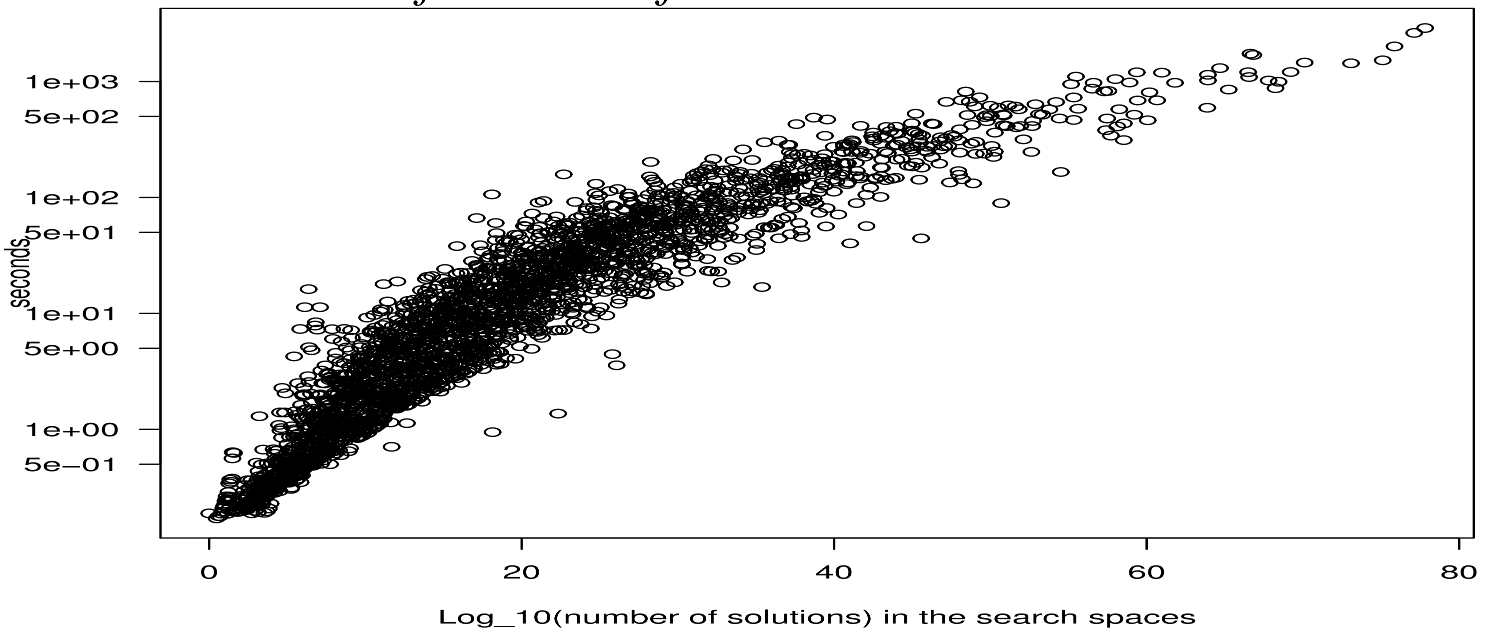


The heaviest phase: distribution computing

- Procedure

- Complexity
  1200 templates
  - *1,200,000 alignments* `Ali1D` *and* `Ali3D`
  - *CPU time for Ali3D: from 0 second to* 800 *seconds.*

NP-complete



The heaviest phase: distribution computing

- Procedure

- Complexity

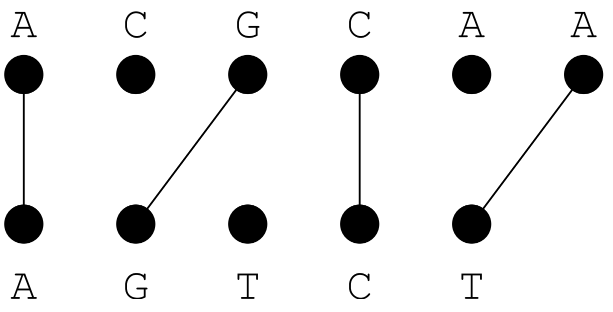
  1200 templates

  - *1,200,000 alignments* `Ali1D` *and* `Ali3D`
  - *about 40 days on a 2.4 GHz computer!*

$\mathcal{NP}$-complete

Our current vision of PTP

Matching and sequence alignment

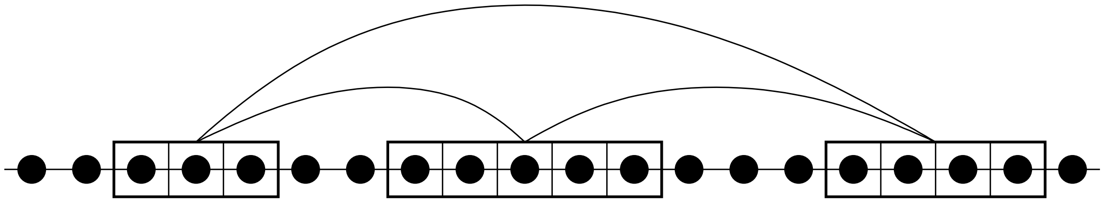


ACG_CAA
A_GTC_T

Figure 2: Matching interpretation of sequence alignment problem

Alignment problems are special cases of matching in bipartite graphs. Because of the intrinsic order on the graph vertices, feasible alignments are 1-matchings *without crossing* edges.

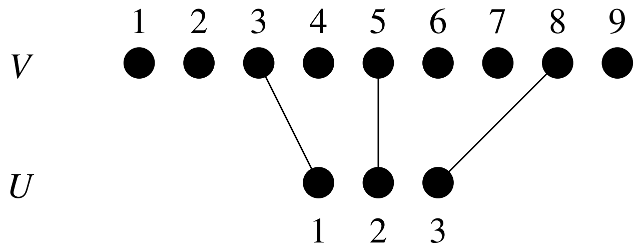
PTP is a matching problem



(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 3: (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).

Related work

- Lathrop_Smith's branch&bound,(J.Mol.Biol., 1996);
- Xu_Xu_Uberbacher's divide&conquer (J. Comp. Biol., 1998).
- J. Xu, M. Li, G. Lin, D. Kim and Y. Xu, Protein threading by linear programming, PSB, January, 2003, (JBCB, March 2003)
- N. Yanev, R. Andonov, Parallel Divide&Conquer Approach for Protein Threading Problem, HiCOMB'03, April, 2003, Nice
- Andonov, Balev, Yanev, Protein Threading Problem: From Mathematical Models to Parallel Implementations,INFORMS Journal on Computing, Eds. Greenberg, Gusfield, Xu (in print)
- A. Marin, J.Pothier, K. Zimmermann, J-F. Gibrat, FROST: A Filter Based Recognition Method, Proteins: Struct. Funct. Genet. 2002

Our approach : network flow model

Which is the shortest path from S to T ?

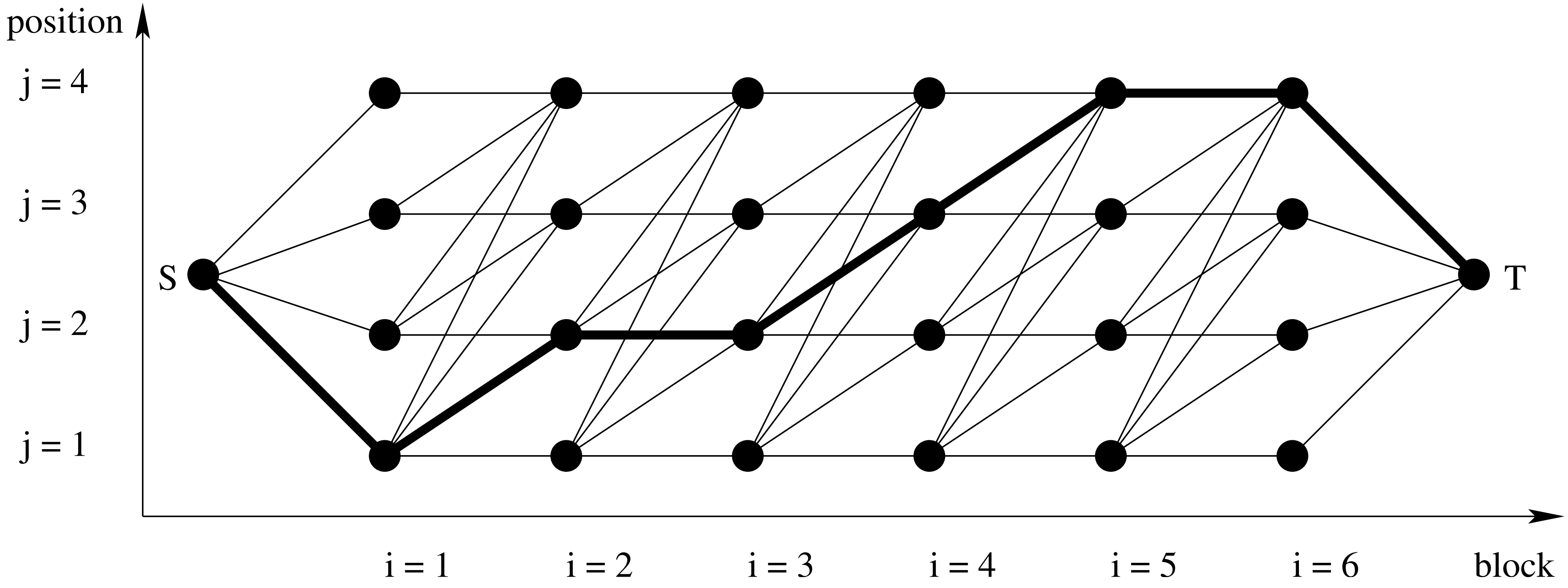
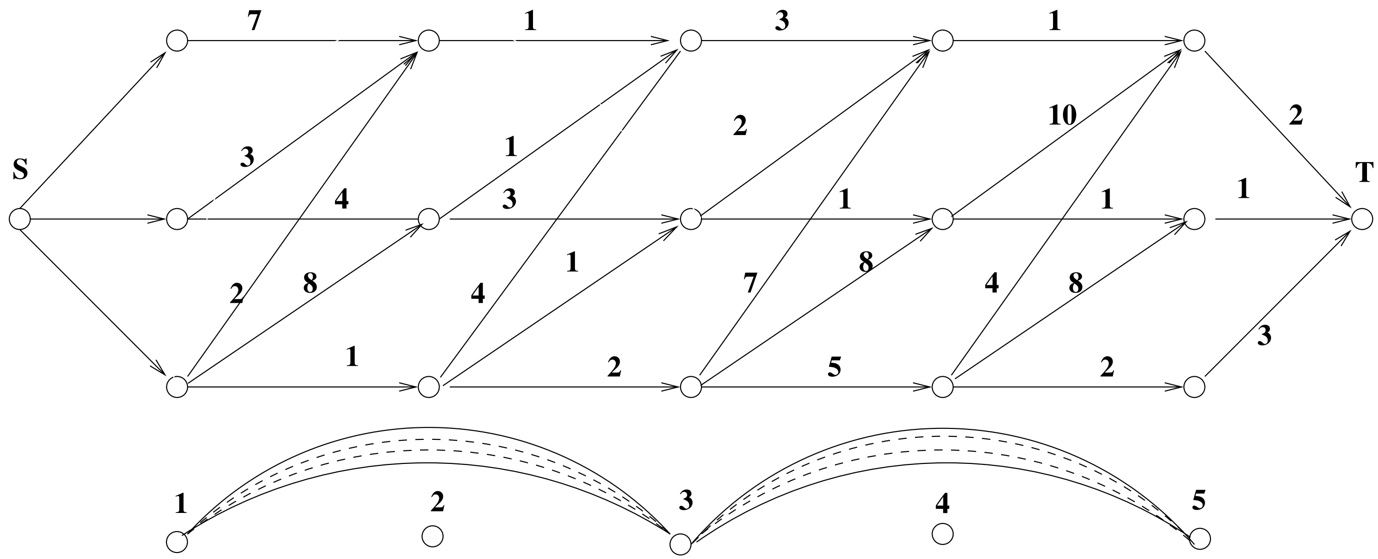


Figure 4: In the *alignment graph* $G = (U \times V, E)$ each vertex corresponds to an edge of the matching graph. The path in thick lines corresponds to the threading in which the positions of the blocks are 1,2,2,3,4,4. Non-decreasing paths express non-crossing matchings. There is an one-to-one correspondence between the set of feasible threadings (or matchings) and the set of S-T paths in G.

Our approach : network flow model

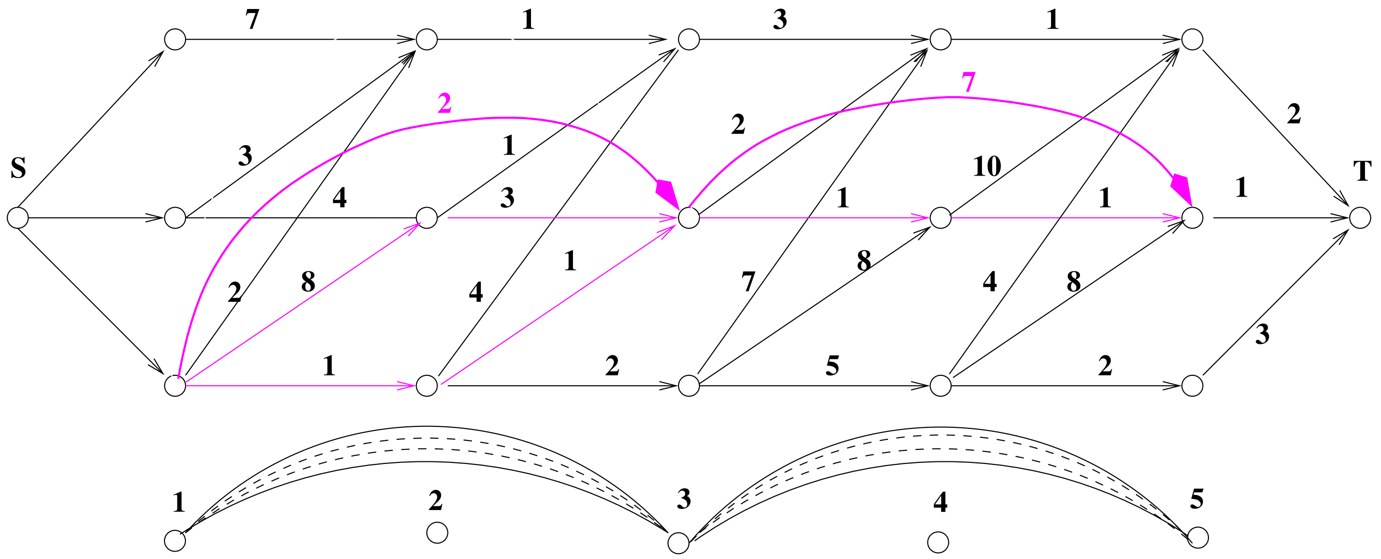


NON–LOCAL COSTS

(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 4: Here are all interactions. The non-local interactions make the problem NP-complete.

Our approach : network flow model



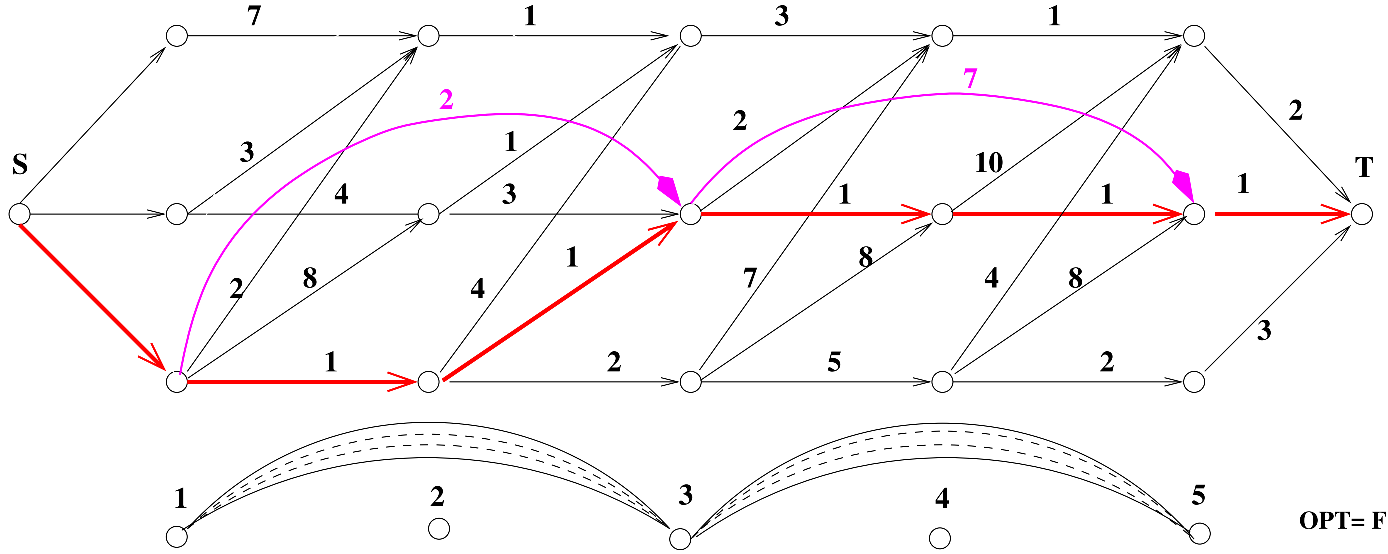
NON–LOCAL COSTS

(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 4: 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*

Our approach : network flow model

Protein threading problem: find 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 4: The red path corresponds to the threading (1,1,2,2,2).

Integer programming models

Notations

- $L = \{(i, k) \mid i < k$ and blocks i and k interact$\}$ (inter-blocks interactions)
- c_{ij} : score of block i on position j
- d_{ijkl}, $(i, k) \in L$: score of interactions between blocks i and k when block i is on position j and block k is on position l.
- y_{ij} is one if block i is on position j and zero otherwise.
- z_{ijkl}, $(i, k) \in L$ is one if block i is on position j while block k is placed on position l; and zero otherwise.
- A_{ik} : node-arc incidence matrix for the subgraph spanned by the layers i and k, $(i, k) \in L$.

Space Y of feasible threadings

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

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

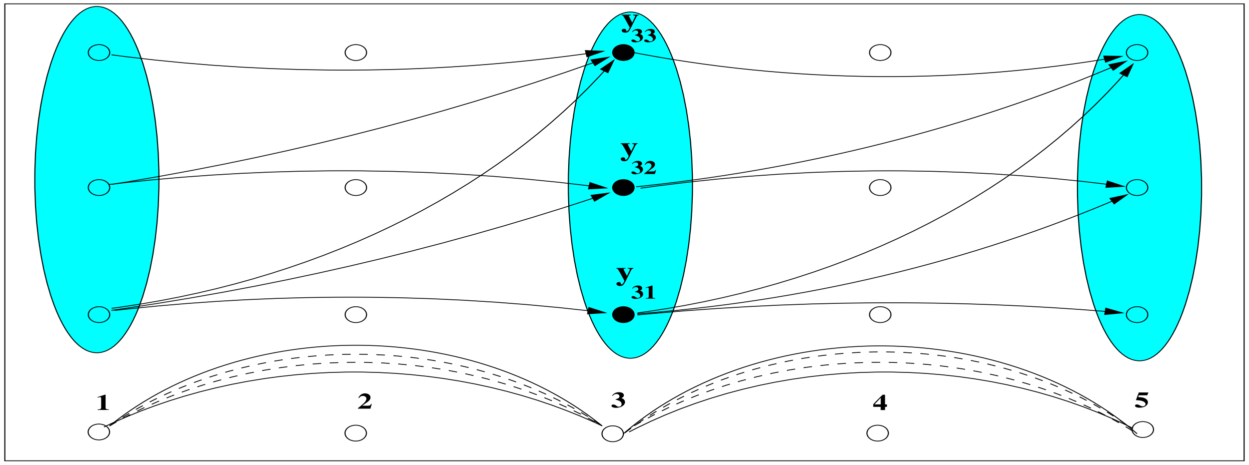
$$y_{ik} \in \{0, 1\} \qquad i = 1, m,\ k = 1, n \tag{3}$$

(18) $y_{ik} = 1 \Leftrightarrow$ block i is on position k

(16) block i is on exactly one position

(17) if block $i + 1$ is on positions l, then block i is before position l

Introducing z variables to Y



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

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

Integer programming formulation : MYZ

$$\sum_{i=1}^{m}\sum_{k=1}^{n} c_{ik}y_{ik} + \sum_{(i,l)\in L} d_{il}z_{il} \Rightarrow \min \tag{4}$$

$$y_{ik} = \sum_{l=k}^{n} z_{ikjl} \quad (i,j) \in L,\ k = 1,n \tag{5}$$

$$y_{jl} = \sum_{k=1}^{l} z_{ikjl} \quad (i,j) \in L,\ l = 1,n \tag{6}$$

$$y \in Y \tag{7}$$

$$z_{il} \geq 0 \quad (i,l) \in L \tag{8}$$

Integer programming formulation II : MYZ

Now the protein threading problem $PTP(L)$ is defined as:

$$z_{IP}^{L} = v(PTP(L)) = \min\{\sum_{i=1}^{m} c_i y_i + \sum_{(i,k)\in L} d_{ik} z_{ik}\} \tag{9}$$

$$\text{subject to: } y = (y_1, \ldots, y_m) \in Y, \tag{10}$$

$$y_i = A_i^k z_{ik}, \;\; y_k = A_k^i z_{ik} \qquad (i,k) \in L \tag{11}$$

$$z_{ik} \in B^{\frac{n(n+1)}{2}} \qquad (i,k) \in L \tag{12}$$

where Y is defined by (16)-(18).

Properties of the polytope Y

$$\sum_{k=1}^{n} y_{ik} = 1 \qquad i = 1, m \tag{13}$$

$$\sum_{l=1}^{j} y_{il} - \sum_{l=1}^{j} y_{i+1,l} \geq 0 \qquad i = 1, \ldots, m-1,\ j = 1, \ldots, n-1 \tag{14}$$

$$y_{ik} \in \{0, 1\} \qquad i = 1, m,\ k = 1, n \tag{15}$$

(18) $y_{ik} = 1 \Leftrightarrow$ block i is on position k

(16) block i is on exactly one position

(17) 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.*

Properties of the polytope (Y, Z)

$$\sum_{k=1}^{n} y_{ik} = 1 \qquad i = 1, m \tag{16}$$

$$\sum_{l=1}^{j} y_{il} - \sum_{l=1}^{j} y_{i+1,l} \geq 0 \qquad i = 1, \ldots, m-1,\ j = 1, \ldots, n-1 \tag{17}$$

$$y_{ik} \in \{0, 1\} \qquad i = 1, m,\ k = 1, n \tag{18}$$

$$y_{ik} = \sum_{l=k}^{n} z_{ikjl} \quad (i, j) \in L,\ k = 1, n \tag{19}$$

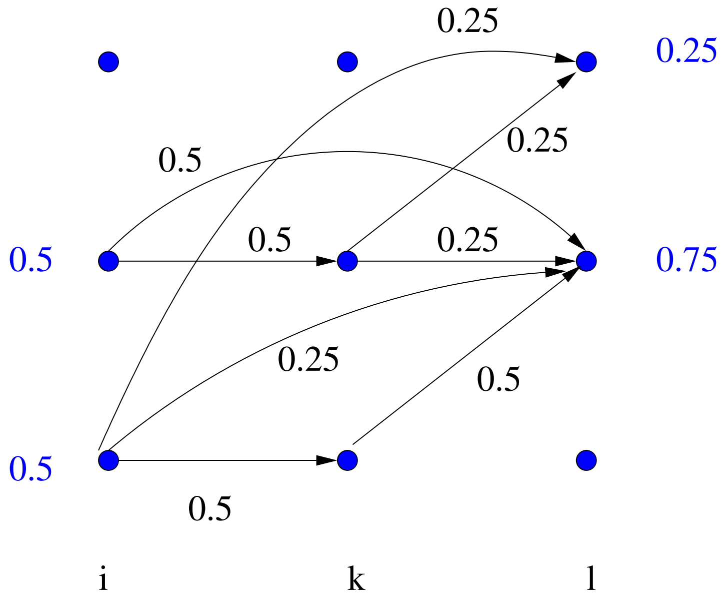
$$y_{jl} = \sum_{k=1}^{l} z_{ikjl} \quad (i, j) \in L,\ l = 1, n \tag{20}$$

$$y \in Y, \qquad z_{il} \geq 0 \quad (i, l) \in L \tag{21}$$

Proposition 2 *If the set L contains only local links then the polytope (Y, Z) is integral, i.e. it has only integer-valued vertices.*

Properties of the polytope (Y, Z)

If the contact graph has a cycle (i.e. the set L contains at least three links $(i, k), (i, l), (k, l)$ then the polytope (Y, Z) contains at least one non-integer valued vertex.



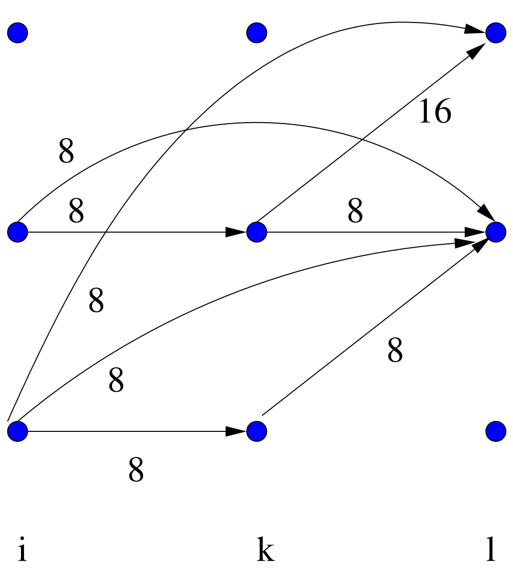


Figure 5: Left: The non-zero components of the point $(\bar{y}, \bar{z})$ are given by the weights on the edges. Right: weights represent the coefficients of the objectif function f. Maximal integer value of f is 24, while $f(\bar{y}, \bar{z}) = 26$.

Lagrangian relaxation and duality

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

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

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

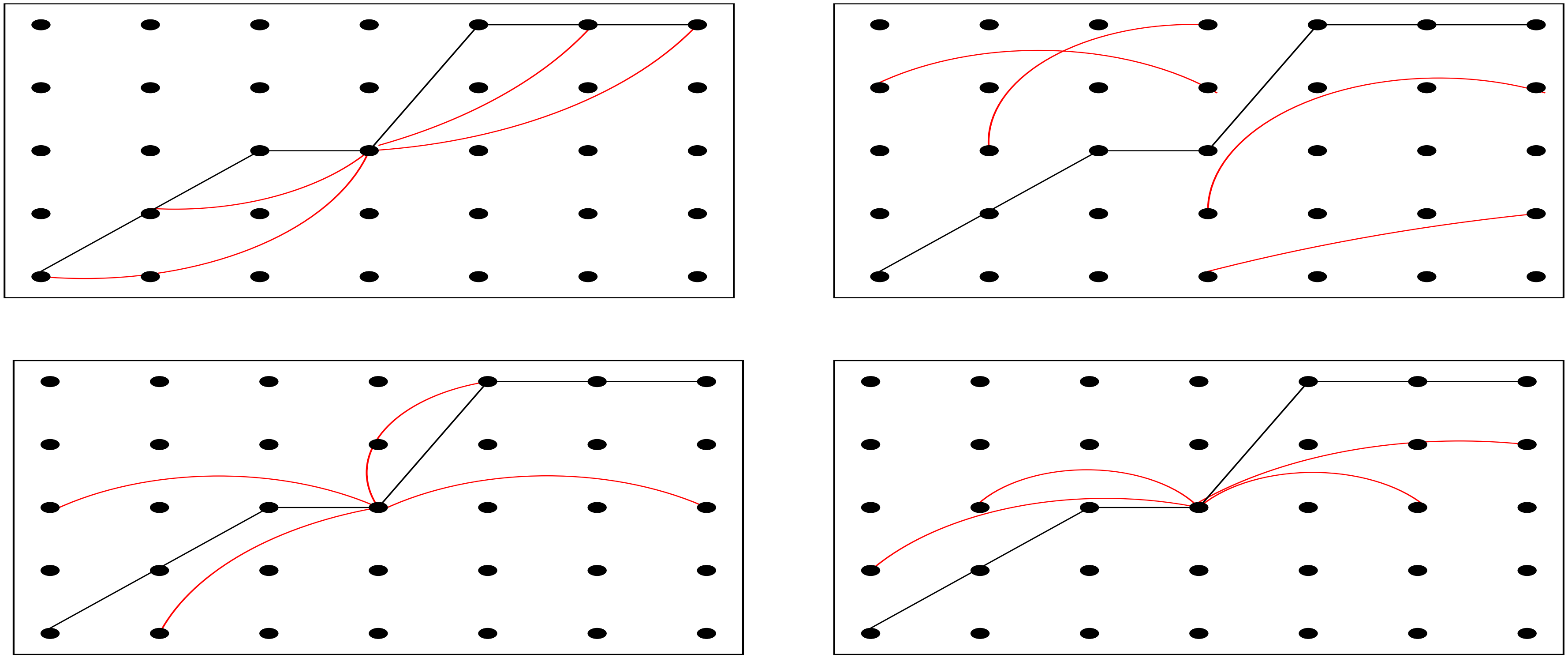
- LR is also an IP problem, but easier to solve than IP
- LR is relaxation of IP for *any* λ (i.e. $Z_{LR}(\lambda) \leq Z_{IP}$)

Lagrangian dual: $Z_{LD}(\lambda) = \text{max}_{\lambda} Z_{LR}(\lambda)$

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

LR for protein threading

the “complicated” constraints are those connecting y and z varialbles



a) original problem

b) all connecting contraints are relaxed

c) the constraints corresponding to one of the ends of each link are relaxed

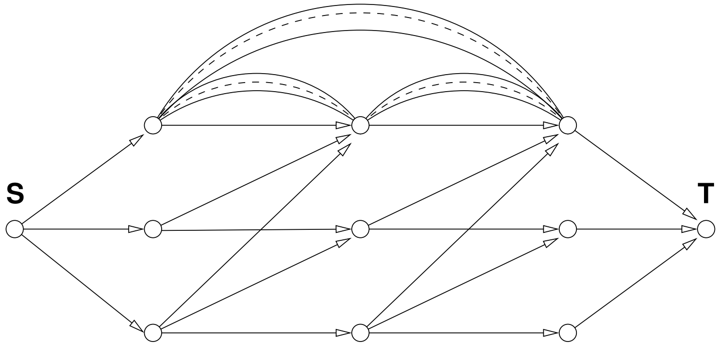
d) like c) but the order on th free ends of the links is imposed

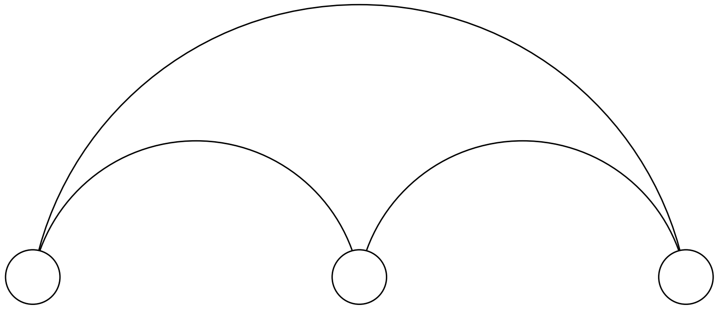
Solving protein threading problem

- LR is solved using dynamic programming similar to the one proposed by Lathrop&Smith. Complexity $O((m+r)n^2)$, where r is the number of remote links between blocks. In the worst case this is $O(m^2n^2)$, but for real-life instances it is $O(mn^2)$
- LD is computed using subgradient optimization limited to 500 iterations
- protein threading problem is solve by B&B using LD bounds

Zoology of PTP

- Reminder: L is the inter-block interaction graph

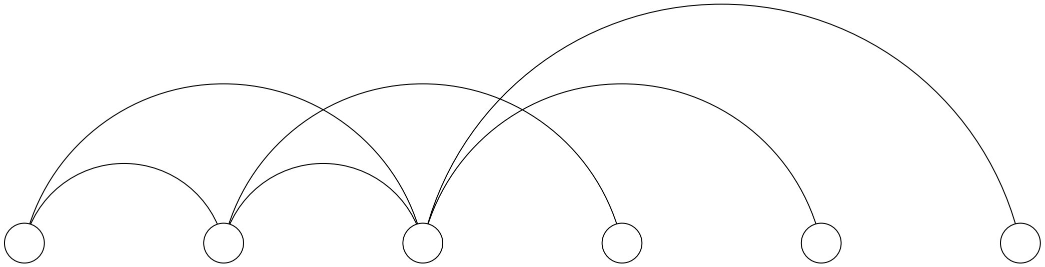




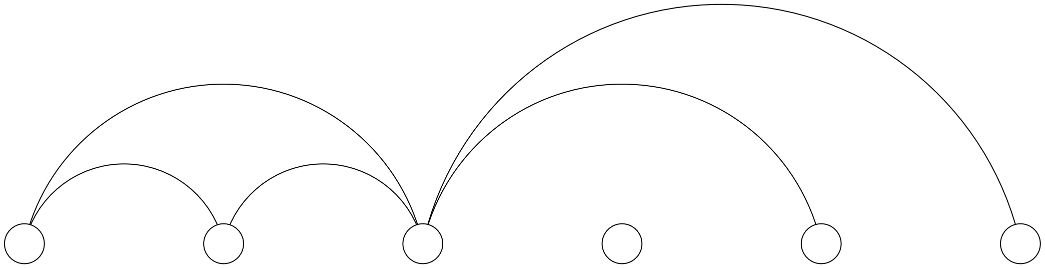
- *complexity* of PTP strongly depends on the *topology* of L
  - $L = \emptyset \longrightarrow$ PTP polynomially solvable
  - L dense $\longrightarrow$ PTP NP-hard
- What about intermediate cases ?

SP#1: L contains no crossing edges

Crossing edges:

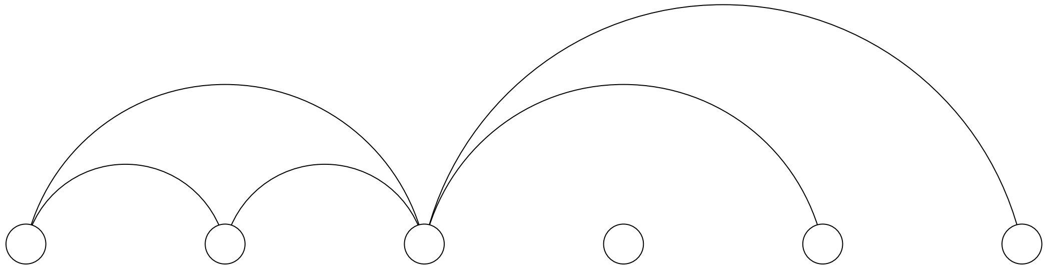


Non-Crossing edges:



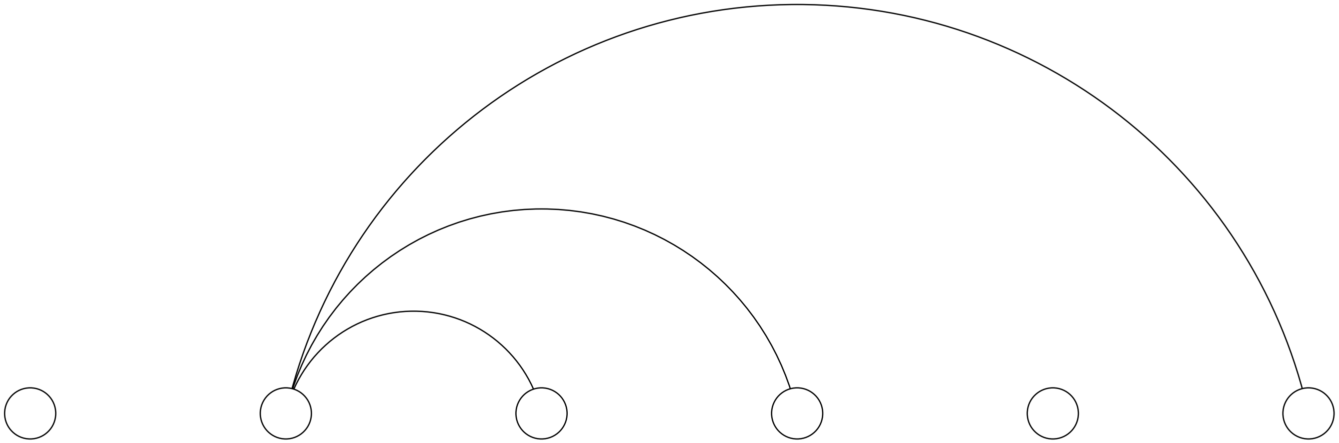
SP#1, continued

- 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

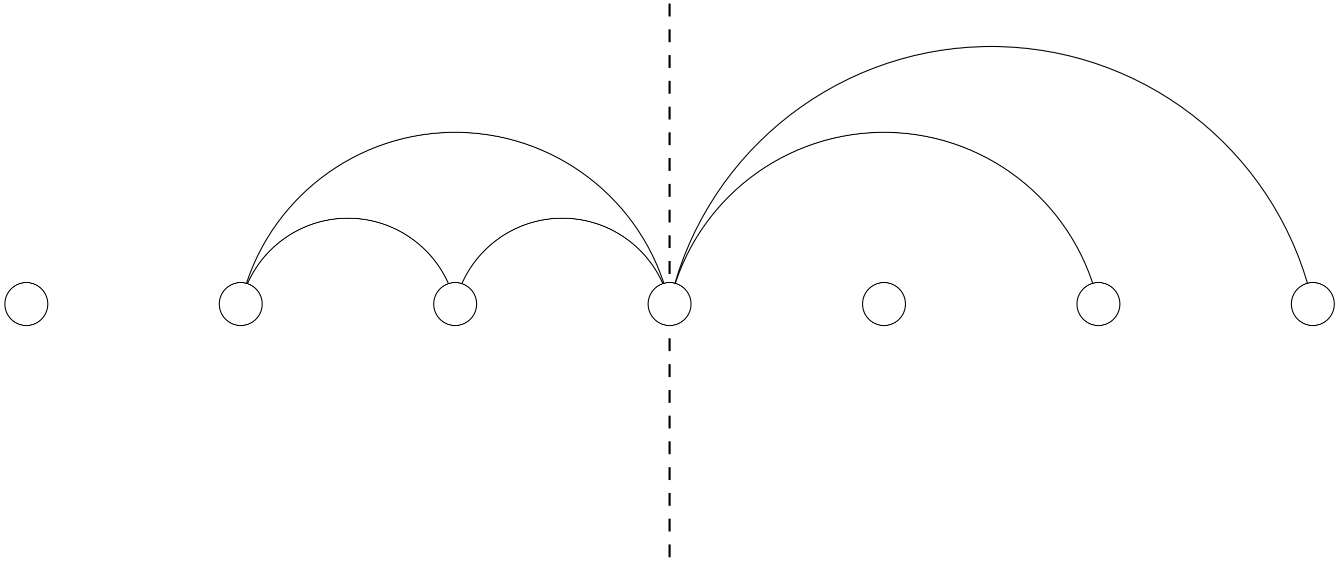
- Star: common left/right end for all links



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

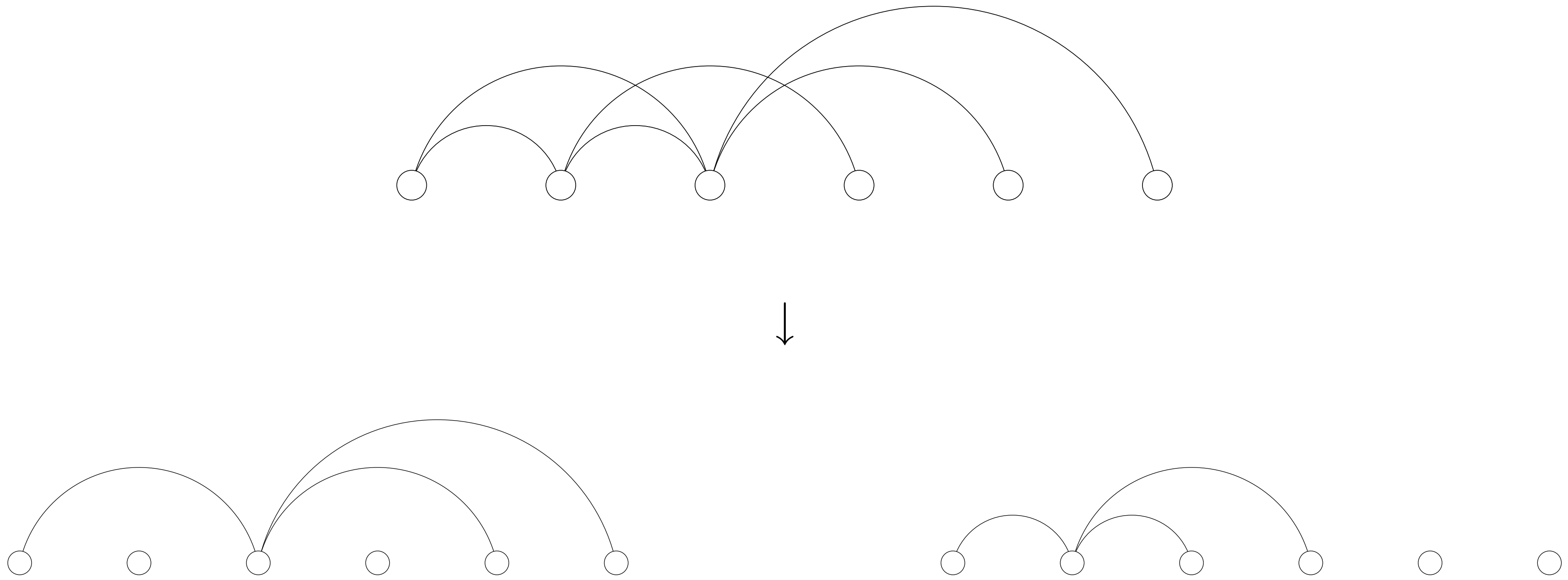
SP#3: sequence of independant subproblems

- partition s.t. no link is cut

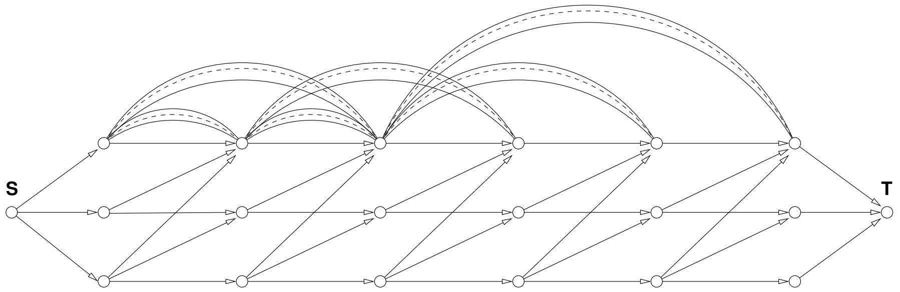


- let $r =$ number of independant subproblems
- $O(rn^2)$ complexity after having solved each subproblem

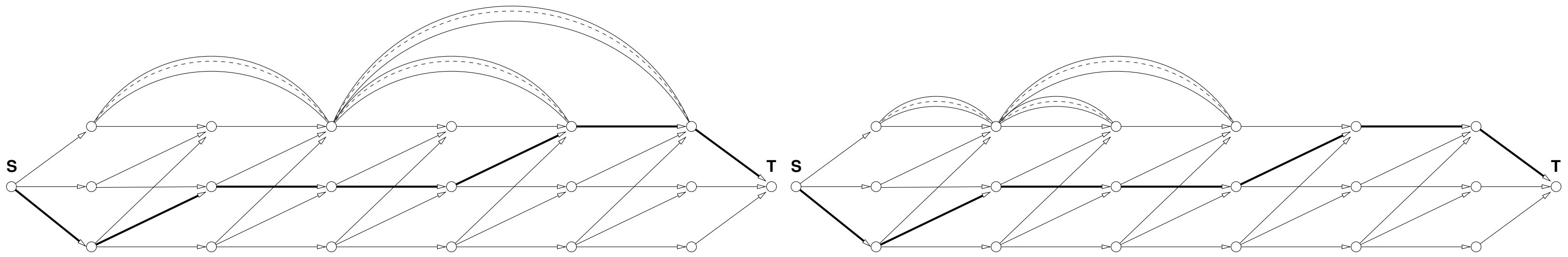
From graph decomposition ...



...to cost-splitting technique



↓



- solve independently and enforce identical solutions

Cost-splitting Lagrangian relaxation

$$z_{IP} = \min cx^1 : \quad A^1x^1 \leq b^1, \quad A^2x^2 \leq b^2$$

$$x^1 - x^2 = 0 \;\; x^1 \in Z^n_+, \quad x^2 \in Z^n_+$$

Taking $x^1 - x^2 = 0$ as the complicated constraint, we obtain the LD

$$z_{IP} = \max_u \{\min c^1x^1 + \min c^2x^2\} \tag{22}$$

$$\text{subject to: } A^1x^1 \leq b^1, \quad A^2x^2 \leq b^2, \tag{23}$$

$$x^1 \in Z^n_+, \quad x^2 \in Z^n_+, \tag{24}$$

where $u = c^2, c^1 = c - u$.

Optimization

- equality constraint between different copies is the hard one
- Practical resolution:
  - Lagrangian relaxation
  - Maximisation of the dual using its subgradient
  - 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 LR (CS-LR) versus LP

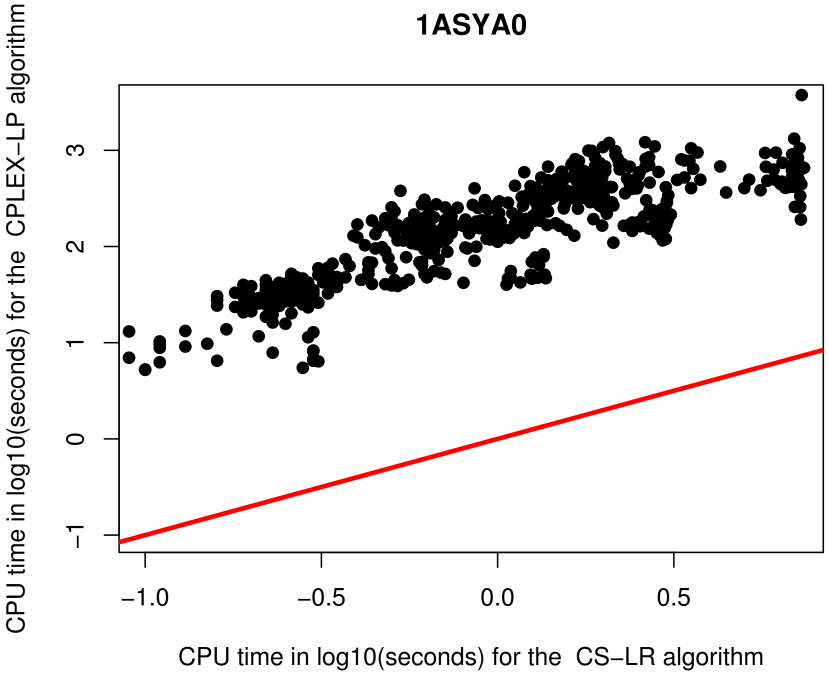


Figure 6: The linear curve in the plot is the line $y = x$. We observe a significant performance gap between the algorithms. CS-LR is **from** 100 **to** 250 **times faster** than LP relaxation.

CS–LR versus SB–LR

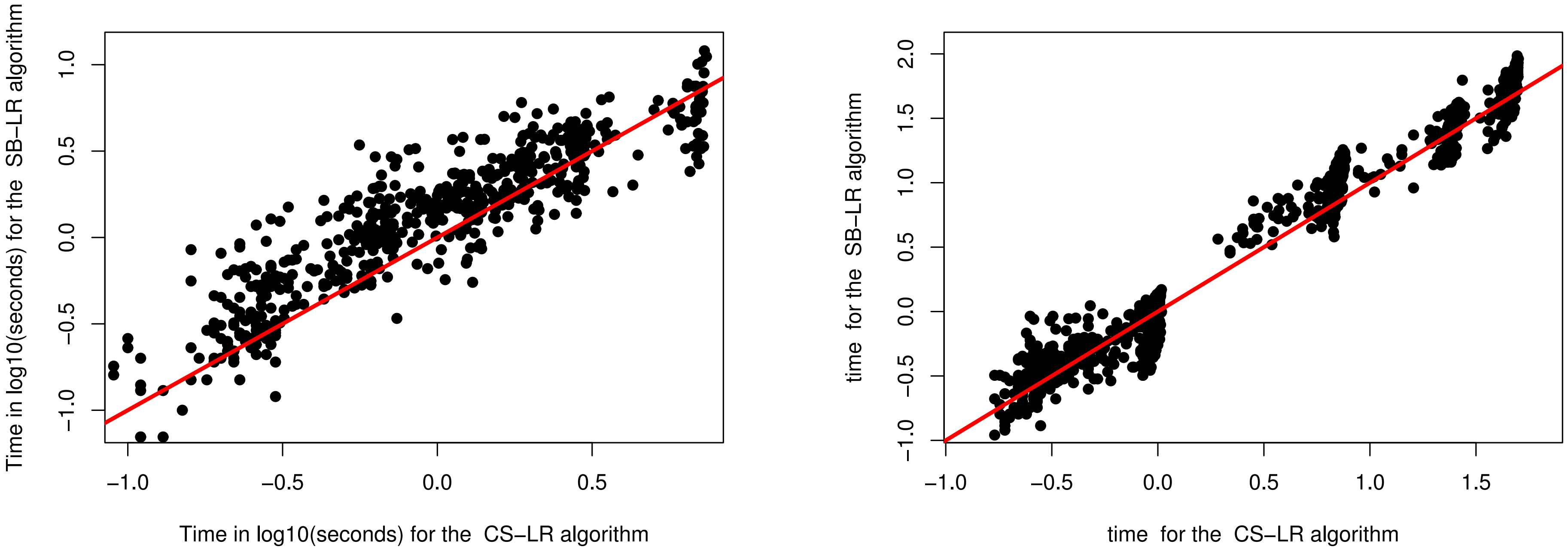
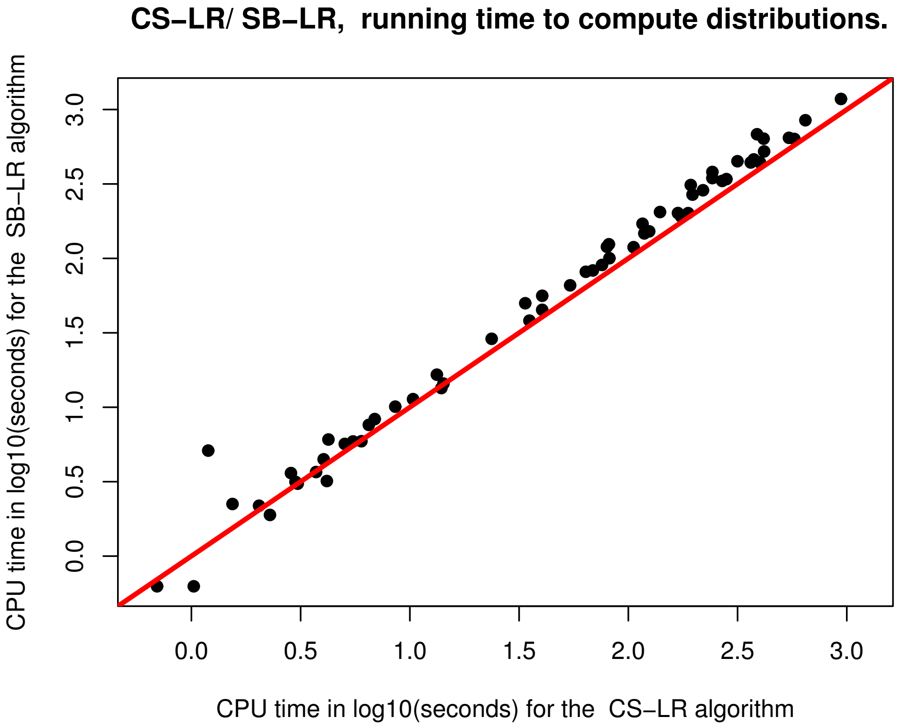


Figure 7: **Left:** The template 1ASYA has been threaded with 962 sequences. **Right:** 1ALO_0 is one of the templates yielding the biggest problem instances when aligned with the 704 sequences associated to it in the database. Although CS-LR is often faster than SB-LR, in general the performance of both algorithms is very close.

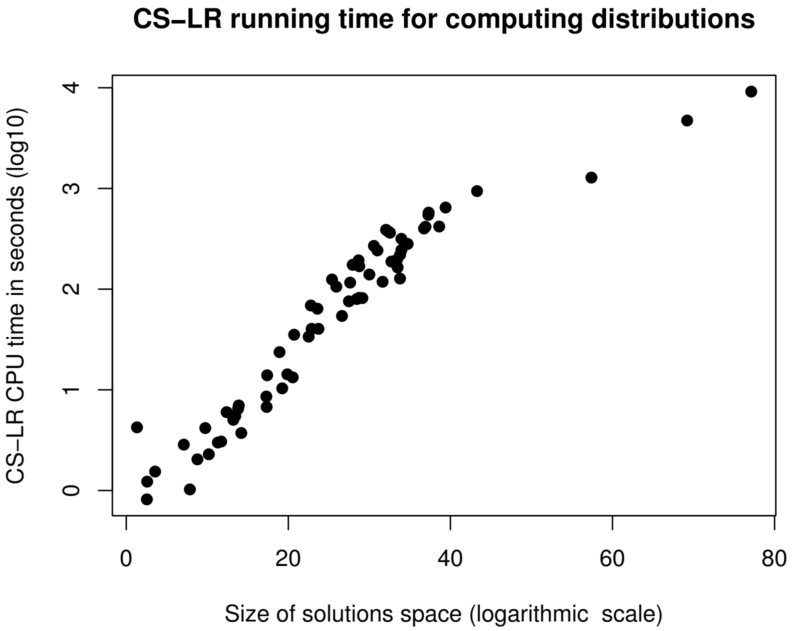
CS–LR versus SB–LR (suite)



Each point corresponds to the total time needed to compute one distribution determined by approximately 200 alignments of the same size. 61 distributions have been computed which needed solving totally 12125 alignments. The linear curve in the plot is the line $y = x$. CS-LR is consistently faster than SB-LR algorithm.

Figure 8: Recapitulation plot concerning 12125 alignments.

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 $O(10^{77})$.

Figure 9: Evolution in time as a function of the solutions space size.