Did you know that Multiple Alignment is NP-hard?

Isaac Elias

Royal Institute of Technology
Sweden

Results

- Multiple Alignment with SP-score
- Star Alignment
- Tree Alignment (with given phylogeny)

are NP-hard under all metrics!

Pairwise Alignment

Mutations: substitutions, insertions, and deletions.

Input: Two related strings s_1 and s_2.

Output: The least number of mutations needed to $s_1 \rightarrow s_2$.

$$\begin{array}{rcccccc} s_1 = & \mathsf{a} & \mathsf{a} & \mathsf{g} & \mathsf{a} & \mathsf{c} & \mathsf{t} \\ s_2 = & \mathsf{a} & \mathsf{g} & \mathsf{t} & \mathsf{g} & \mathsf{c} & \mathsf{t} \end{array}$$

Pairwise Alignment

Mutations: substitutions, insertions, and deletions.

Input: Two related strings s_1 and s_2.

Output: The least number of mutations needed to $s_1 \longrightarrow s_2$.

$2 \times l$ matrix A

$$\begin{array}{lccccccc} s_1 = & \mathsf{a} & \mathsf{a} & \mathsf{g} & \mathsf{a} & - & \mathsf{c} & \mathsf{t} \\ s_2 = & - & \mathsf{a} & \mathsf{g} & \mathsf{t} & \mathsf{g} & \mathsf{c} & \mathsf{t} \end{array}$$

$\mathtt{d}_A(s_1, s_2) = 3$ mutations

Metric symbol distance

Unit metric - binary alphabet
(the edit distance)

Σ	**0**	**1**	$-$
0	0	1	1
1	1	0	1
$-$	1	1	0

Metric symbol distance

Unit metric - binary alphabet
(the edit distance)

Σ	**0**	**1**	$-$
0	0	1	1.5
1	1	0	1.5
$-$	1.5	1.5	0

Insertions and deletions occur less frequently!

Metric symbol distance

Unit metric - binary alphabet
(the edit distance)

Σ	**0**	**1**	$-$
0	0	1	1
1	1	0	1
$-$	1	1	0

Insertions and deletions occur less frequently!

General metric - α, β, γ have metric properties
(identity, symmetry, triangle ineq.)

Σ	**0**	**1**	$-$
0	0	α	β
1	α	0	γ
$-$	β	γ	0

Multiple Alignment

When sequence similarity is low pairwise alignment may fail to find similarities.

Multiple Alignment

When sequence similarity is low pairwise alignment may fail to find similarities.

Considering several strings may be helpful.

Input: k strings $s_1 \ldots s_k$.
Output: $k \times l$ matrix A.

a	a	g	a	-	c	t
-	-	g	a	g	c	t
a	c	g	a	g	c	-
a	-	g	t	g	c	t

Multiple Alignment

When sequence similarity is low pairwise alignment may fail to find similarities.

Considering several strings may be helpful.

Input: k strings $s_1 \ldots s_k$.
Output: $k \times l$ matrix A.

a	a	g	a	-	c	t
-	-	g	a	g	c	t
a	c	g	a	g	c	-
a	-	g	t	g	c	t

How do we score columns?

Sum of Pairs score (SP-score)

Let the cost be the sum of costs for all pairs of rows:

$$\sum_{i=1}^{k} \sum_{j=i}^{k} \mathtt{d}_A(s_i, s_j),$$

where $\mathtt{d}_A(s_i, s_j)$ is the pairwise distance between the rows containing strings s_i and s_j.

Earlier NP results

- Wang and Jiang '94 - Non-metric
- Bonizzoni and Vedova '01 - Specific metric
  (we build on their construction)
- Just '01 - Many metrics

Earlier NP results

- Wang and Jiang '94 - Non-metric
- Bonizzoni and Vedova '01 - Specific metric
  (we build on their construction)
- Just '01 - Many metrics

Earlier NP results

- Wang and Jiang '94 - Non-metric
- Bonizzoni and Vedova '01 - Specific metric
  (we build on their construction)
- Just '01 - Many metrics

Earlier NP results

- Wang and Jiang '94 - Non-metric
- Bonizzoni and Vedova '01 - Specific metric
  (we build on their construction)
- Just '01 - Many metrics

In particular it was unknown for the unit metric.

Earlier NP results

- Wang and Jiang '94 - Non-metric
- Bonizzoni and Vedova '01 - Specific metric
  (we build on their construction)
- Just '01 - Many metrics

In particular it was unknown for the unit metric.

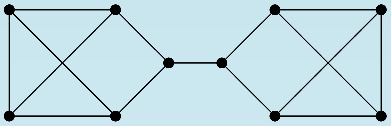
Here: All binary or larger alphabets under all metrics!

Independent R3 Set

Independent set in three regular graphs, i.e. all vertices have degree 3, is NP-hard.

Input: A three regular graph $G = (V, E)$ and a integer c.

Output: "Yes" if there is an independent set $V' \subseteq V$ of size $\geq c$, otherwise "No".

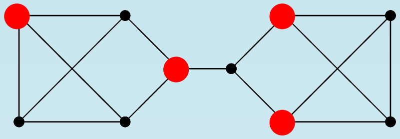


Independent R3 Set

Independent set in three regular graphs, i.e. all vertices have degree 3, is NP-hard.

Input: A three regular graph $G = (V, E)$ and a integer c.

Output: "Yes" if there is an independent set $V' \subseteq V$ of size $\geq c$, otherwise "No".

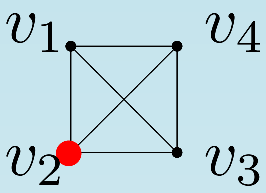


Reduction

Independent R3 Set		**SP-score**
$G = (V, E)$	$\longrightarrow$	Set of strings
c		K
set of size $\geq c$		matrix of cost $\leq K$
V'	$\longleftrightarrow$	A

Reduction

Independent R3 Set		**SP-score**
$G = (V, E)$	$\rightarrow$	Set of strings
c		K
set of size $\geq c$		matrix of cost $\leq K$
V'	$\leftrightarrow$	A



	v_1		**v_2**		v_3		v_4	
	1	0. . . 0. . .	1	0. . . 0. . .	1	0. . . 0. . .	1	
	$\vdots$	$\vdots$ $\vdots$	$\vdots$	$\vdots$ $\vdots$	$\vdots$	$\vdots$ $\vdots$	$\vdots$	
	1	0. . . 0. . .	1	0. . . 0. . .	1	0. . . 0. . .	1	
			1					
			$\vdots$					
			1					
	0	0. . . 10. . .	1	0. . . 0. . .	0	0. . . 0. . .	0	0. . .
0. . .	1	0. . . 0. . .	0	0. . . 10. . .	0	0. . . 0. . .	0	
	0	0. . . 10. . .	0	0. . . 0. . .	0	0. . . 0. . .	1	0. . .
0. . .	0	0. . . 0. . .	1	0. . . 10. . .	0	0. . . 0. . .	0	
0. . .	0	0. . . 0. . .	1	0. . . 0. . .	0	0. . . 10. . .	0	
0. . .	0	0. . . 0. . .	0	0. . . 0. . .	1	0. . . 10. . .	0	

Gadgetering

1. Each vertex is represented by a column (n vertex columns).
2. c vertex columns are picked.
3. Each edge is represented by an edge string.

	v_1		v_2	. . .	v_i	. . .	v_n	
	1	0. . . 0. . .	1	. . .	1	. . .	1	
	⋮	⋮ ⋮	⋮	⋮ ⋮	⋮	. . .	⋮	
	1	0. . . 0. . .	1	. . .	1	. . .	1	
			1		1			
			⋮		⋮			
			1		1			
	0	0. . . 10. . .	1	0. . . 0. . .	0	0. . . 0. . .	0	0. . .
0. . .	1	0. . . 0. . .	0	0. . . 10. . .	0	0. . . 0. . .	0	
	0	0. . . 10. . .	0	0. . . 0. . .	0	0. . . 0. . .	1	0. . .
0. . .	0	0. . . 0. . .	1	0. . . 10. . .	0	0. . . 0. . .	0	
0. . .	0	0. . . 0. . .	1	0. . . 0. . .	0	0. . . 10. . .	0	
0. . .	0	0. . . 0. . .	0	0. . . 0. . .	1	0. . . 10. . .	0	

Template strings → Vertex columns

We add **very** many template strings:

$$T = (10^b)^{n-1}1$$

v_1		v_2	...	v_i	...	v_n
1	0. . . 0. . .	1	...	1	...	1
⋮	⋮ ⋮	⋮	⋮ ⋮	⋮	...	⋮
1	0. . . 0. . .	1	...	1	...	1

Fact: Identical strings are aligned identically in an optimal alignment.

Pick strings $\rightarrow V' \subseteq V$

We add **very** many pick strings: $P = 1^c$.

	v_1		v_2	. . .	v_i	. . .	v_n	
	1	0. . . 0. . .	1	. . .	1	. . .	1	
	⋮	⋮ ⋮	⋮	⋮ ⋮	⋮	. . .	⋮	
	1	0. . . 0. . .	1	. . .	1	. . .	1	
			1		1			
			⋮		⋮			
			1		1			

Fact: c vertex columns are picked since this is the alignment with least missmatches.

Edge strings

Property: If there is an edge (v_i, v_j) then both v_i and v_j can not be part of the independent set.

$$E_{ij} = 0^s(00^b)^{i-1}10^{b-s}(00^b)^{j-i-1}10^b(00^b)^{n-j-1}00^s$$

		v_1		v_i	$\dots \quad \dots$	v_j	$\dots$	v_n	
		1	0. . . 0. . .	1	$\dots \quad \dots$	1	$\dots$	1	
		$\vdots$	$\vdots \quad \vdots$	$\vdots$	$\dots \quad \dots$	$\vdots$	$\dots$	$\vdots$	
		1	0. . . 0. . .	1	$\dots \quad \dots$	1	$\dots$	1	
good	0^s	0	0. . . 0. . .	1	0. . . $1\ 0^{s-1}$	0	0. . . 0. . .	0	
good		0	0. . . 0. . .	0	$0^{s-1}\ 1$ 0. . .	1	0. . . 0. . .	0	0^s
bad	0^s	0	0. . . 0. . .	1	$-^s$ 0. . .	1	0. . . 0. . .	0	0^s

Independent set $\geq c \Leftrightarrow$ Alignment $\leq K$

$$K = (n-c)\gamma b^2 + b(n-1)\beta b^2 + (s\beta + n\alpha)bm + \Big(c\alpha + (s + b(n - 1) + n - c - 2)\beta + 2\gamma\Big)bm - 3cb(\alpha + \gamma - \beta) + (s\beta + 2\alpha)m^2$$

Independent set $\geq c \Leftrightarrow$ Alignment $\leq K$

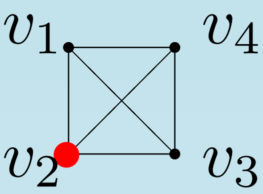
1. Remember three regular graph. So there are atmost three ones in each vertex column.

2. If a column with only two ones is picked then there is a missmatch extra for each pick string ($\Rightarrow$ score $> K$).

Example

1. Atmost three ones in each vertex column.

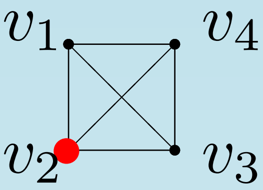
2. Pick strings pick columns with most ones.



		v_1		v_2		v_3		v_4	
		1 ⋮ 1	0. . . 0. . . ⋮ ⋮ 0. . . 0. . .	1 ⋮ 1	0. . . 0. . . ⋮ ⋮ 0. . . 0. . .	1 ⋮ 1	0. . . 0. . . ⋮ ⋮ 0. . . 0. . .	1 ⋮ 1	
				1 ⋮ 1					
E_{12}		**0**	**0. . . 10. . .**	**1**	**0. . . 0. . .**	**0**	**0. . . 0. . .**	**0**	**0. . .**
E_{13}	0. . .	1	0. . . 0. . .	0	0. . . 10. . .	0	0. . . 0. . .	0	
E_{14}		0	0. . . 10. . .	0	0. . . 0. . .	0	0. . . 0. . .	1	0. . .
E_{23}	0. . .	0	0. . . 0. . .	1	0. . . 10. . .	0	0. . . 0. . .	0	
E_{24}	**0. . .**	**0**	**0. . . 0. . .**	**1**	**0. . . 0. . .**	**0**	**0. . . 0. . .**	**0**	
E_{34}	0. . .	0	0. . . 0. . .	0	0. . . 0. . .	1	0. . . 10. . .	0	

Example

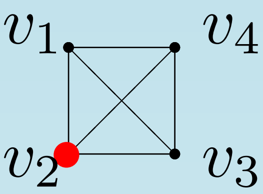
1. Atmost three ones in each vertex column.
2. Pick strings pick columns with most ones.



		v_1		v_2		v_3		v_4	
		1	0. . . 0. . .	1	0. . . 0. . .	1	0. . . 0. . .	1	
		⋮	⋮ ⋮	⋮	⋮ ⋮	⋮	⋮ ⋮	⋮	
		1	0. . . 0. . .	1	0. . . 0. . .	1	0. . . 0. . .	1	
				1					
				⋮					
				1					
E_{12}	**0. . .**	**1**	**0. . . 10. . .**	**0**	**0. . . 0. . .**	**0**	**0. . . 0. . .**	**0**	
E_{13}	0. . .	1	0. . . 0. . .	0	0. . . 10. . .	0	0. . . 0. . .	0	
E_{14}		0	0. . . 10. . .	0	0. . . 0. . .	0	0. . . 0. . .	1	0. . .
E_{23}	0. . .	0	0. . . 0. . .	1	0. . . 10. . .	0	0. . . 0. . .	0	
E_{24}	**0. . .**	**0**	**0. . . 0. . .**	**1**	**0. . . 0. . .**	**0**	**0. . . 0. . .**	**0**	
E_{34}	0. . .	0	0. . . 0. . .	0	0. . . 0. . .	1	0. . . 10. . .	0	

Example

1. Atmost three ones in each vertex column.
2. Pick strings pick columns with most ones.

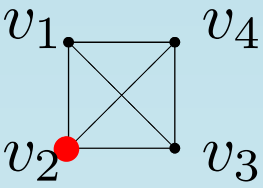


		v_1		v_2		v_3		v_4	
		1	0. . . 0. . .	1	0. . . 0. . .	1	0. . . 0. . .	1	
		⋮	⋮ ⋮	⋮	⋮ ⋮	⋮	⋮ ⋮	⋮	
		1	0. . . 0. . .	1	0. . . 0. . .	1	0. . . 0. . .	1	
				1					
				⋮					
				1					
E_{12}	**0. . .**	**1**	**0. . . 10. . .**	**0**	**0. . . 0. . .**	**0**	**0. . . 0. . .**	**0**	
E_{13}	0. . .	1	0. . . 0. . .	0	0. . . 10. . .	0	0. . . 0. . .	0	
E_{14}		0	0. . . 10. . .	0	0. . . 0. . .	0	0. . . 0. . .	1	0. . .
E_{23}	0. . .	0	0. . . 0. . .	1	0. . . 10. . .	0	0. . . 0. . .	0	
E_{24}		**0**	**0. . . 0. . .**	**0**	**0. . . 0. . .**	**0**	**0. . . 0. . .**	**1**	**0. . .**
E_{34}	0. . .	0	0. . . 0. . .	0	0. . . 0. . .	1	0. . . 10. . .	0	

Example

1. Atmost three ones in each vertex column.

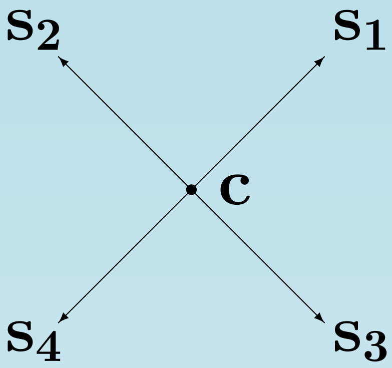
2. Pick strings pick columns with most ones.



		v_1		v_2		v_3		v_4	
		1 ⋮ 1	0. . . 0. . . ⋮ ⋮ 0. . . 0. . .	1 ⋮ 1	0. . . 0. . . ⋮ ⋮ 0. . . 0. . .	1 ⋮ 1	0. . . 0. . . ⋮ ⋮ 0. . . 0. . .	1 ⋮ 1	
		1 ⋮ 1							
E_{12}	**0. . .**	**1**	**0. . . 10. . .**	**0**	**0. . . 0. . .**	**0**	**0. . . 0. . .**	**0**	
E_{13}	0. . .	1	0. . . 0. . .	0	0. . . 10. . .	0	0. . . 0. . .	0	
E_{14}		0	0. . . 10. . .	0	0. . . 0. . .	0	0. . . 0. . .	1	0. . .
E_{23}	0. . .	0	0. . . 0. . .	1	0. . . 10. . .	0	0. . . 0. . .	0	
E_{24}		**0**	**0. . . 0. . .**	**0**	**0. . . 0. . .**	**0**	**0. . . 0. . .**	**1**	**0. . .**
E_{34}	0. . .	0	0. . . 0. . .	0	0. . . 0. . .	1	0. . . 10. . .	0	

Star Alignment

SP-score not a good model of evolution.

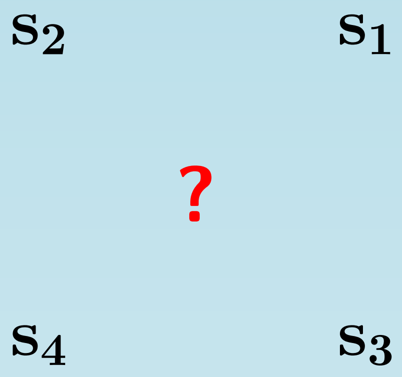


Star Alignment

SP-score not a good model of evolution.

Input: k strings $s_1 \ldots s_k$
Output: string c minimizing

$$\sum_i \mathtt{d}(c, s_i)$$

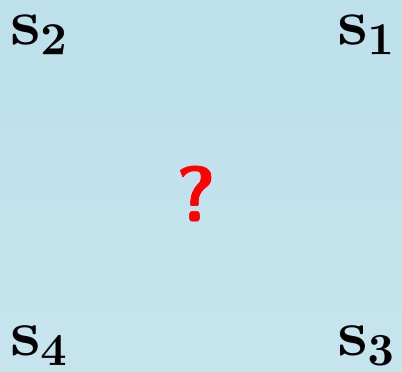


Star Alignment

SP-score not a good model of evolution.

Input: k strings $s_1 \ldots s_k$
Output: string c minimizing

$$\sum_i \mathtt{d}(c, s_i)$$



Symbol distance metric $\Longrightarrow$ Pairwise alignment distance metric

Star Alignment is a special case of **Steiner Star** in a metric space.

Earlier results - Star Alignment

- Wang and Jiang '94 - Non-metric APX-complete
- Li, Ma, Wang '99 - Constant number gaps, NP-hard and PTAS
- Sim and Park '01 - Specific metric NP-hard

Earlier results - Star Alignment

- Wang and Jiang '94 - Non-metric APX-complete
- Li, Ma, Wang '99 - Constant number gaps, NP-hard and PTAS
- Sim and Park '01 - Specific metric NP-hard

Earlier results - Star Alignment

- Wang and Jiang '94 - Non-metric APX-complete
- Li, Ma, Wang '99 - Constant number gaps, NP-hard and PTAS
- Sim and Park '01 - Specific metric NP-hard

Earlier results - Star Alignment

- Wang and Jiang '94 - Non-metric APX-complete
- Li, Ma, Wang '99 - Constant number gaps, NP-hard and PTAS
- Sim and Park '01 - Specific metric NP-hard

Here: All binary or larger alphabets under all metrics!

Reduction

Vertex Cover		**Star Alignment**
$G = (V, E)$	$\longrightarrow$	Set of strings
minimum cover		minimum string
V'	$\leftrightarrow$	$c = DDCDD$

Reduction

Vertex Cover		**Star Alignment**
$G = (V, E)$	$\longrightarrow$	Set of strings
minimum cover		minimum string
V'	$\longleftrightarrow$	$c = DD\mathbf{C}DD$

$$\mathbf{C} \quad = \quad \begin{array}{c|c|c|c|c|c|c|c|c} & \mathbf{v_1} & & \mathbf{v_2} & & \mathbf{v_3} & \dots & \mathbf{v_n} & \\ \hline \dots & ? & \dots & ? & \dots & ? & \dots & ? & \dots \\ \hline \end{array}$$

Reduction

Vertex Cover		**Star Alignment**
$G = (V, E)$	$\longrightarrow$	Set of strings
minimum cover		minimum string
V'	$\longleftrightarrow$	$c = DD\mathbf{C}DD$

$$\mathbf{C_V} = \begin{array}{c|c|c|c|c|c|c|c|c} & \mathbf{v_1} & & \mathbf{v_2} & & \mathbf{v_3} & \dots & \mathbf{v_n} & \\ \hline \dots & \mathbf{1} & \dots & \mathbf{1} & \dots & \mathbf{1} & \dots & \mathbf{1} & \dots \\ \hline \end{array}$$

C_V - Only **1's**.

Reduction

Vertex Cover		**Star Alignment**
$G = (V, E)$	$\longrightarrow$	Set of strings
minimum cover		minimum string
V'	$\longleftrightarrow$	$c = DD\mathbf{C}DD$

$\mathbf{C}_\emptyset$ =

	$\mathbf{v_1}$		$\mathbf{v_2}$		$\mathbf{v_3}$	$\ldots$	$\mathbf{v_n}$	
$\ldots$	**0**	$\ldots$	**0**	$\ldots$	**0**	$\ldots$	**0**	$\ldots$

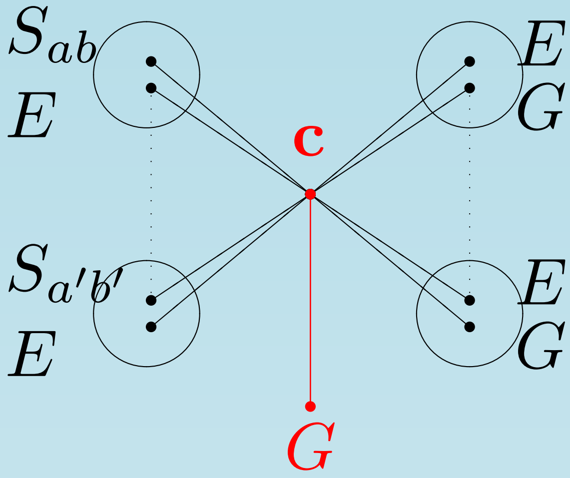
C_V - Only **1's**. $C_\emptyset$ - Only **0's**.

Construction Idea

$(E, G) \quad \longrightarrow \quad c = DDCDD$

$(E, S_{ab}) \quad \longrightarrow \quad c =$ vertex cover

$G \quad \longrightarrow$ minimum cover



String minimizing $\sum_i \mathtt{d}(c, s_i) \leftrightarrow$ minimum cover

Canonical Structure

$$E = DDC_V DD \qquad G = DDC_\emptyset DD$$

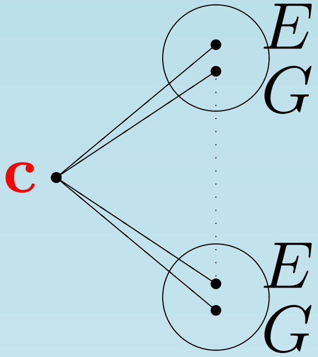
Differ only in the vertex positions.

Canonical Structure

$$E = DDC_V DD \qquad\qquad G = DDC_\emptyset DD$$

Differ only in the vertex positions.

Many such pairs will vote for the canonical structure.



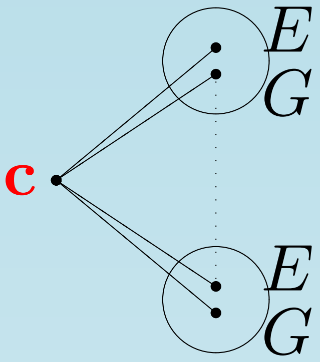
Canonical Structure

$$E = DDC_V DD \qquad\qquad G = DDC_\emptyset DD$$

Differ only in the vertex positions.

Many such pairs will vote for the canonical structure.

Vote even in the vertex positions!



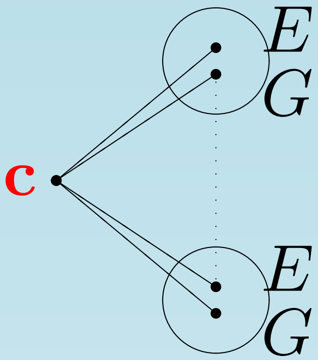
Canonical Structure

$$E = DDC_V DD \qquad G = DDC_\emptyset DD$$

Differ only in the vertex positions.

Many such pairs will vote for the canonical structure.

Vote even in the vertex positions!



$$\Rightarrow \mathbf{c} = \mathbf{DDCDD}$$

String → Cover

Edge (v_a, v_b) add two strings

$$E = DDC_V DD \qquad S_{ab} = C_a D C_b$$

$$\begin{array}{cc|c|c|c|c|c|c|c|c} & & \mathbf{v_1} & \dots & \mathbf{v_a} & \dots & \mathbf{v_b} & \dots & \mathbf{v_n} & \\ \hline \mathbf{C_a} = & \dots & \mathbf{0} & \dots & \mathbf{1} & \dots & \mathbf{0} & \dots & \mathbf{0} & \dots \\ \hline \end{array}$$

String $\longrightarrow$ Cover

Edge (v_a, v_b) add two strings

$$E = DDC_VDD \qquad\qquad S_{ab} = C_aDC_b$$

$$\mathbf{C_b} = \begin{array}{c|c|c|c|c|c|c|c|c} & \mathbf{v_1} & \dots & \mathbf{v_a} & \dots & \mathbf{v_b} & \dots & \mathbf{v_n} & \\ \hline \dots & \mathbf{0} & \dots & \mathbf{0} & \dots & \mathbf{1} & \dots & \mathbf{0} & \dots \\ \hline \end{array}$$

String $\rightarrow$ Cover

Edge (v_a, v_b) add two strings

$$E = DDC_V DD \qquad S_{ab} = C_a D C_b$$

E votes **1** in all vertex positions.

String $\longrightarrow$ Cover

Edge (v_a, v_b) add two strings

$$E = DDC_VDD \qquad\qquad S_{ab} = C_aDC_b$$

E votes **1** in all vertex positions.

S_{ab} votes **0** in all except vertex position a or b.

$$\begin{array}{cc|c|cc} D & D & C & D & D \\ C_a & D & C_b & & \\ & & C_a & D & C_b \end{array}$$

String $\longrightarrow$ Cover

Edge (v_a, v_b) add two strings

$$E = DDC_VDD \qquad S_{ab} = C_aDC_b$$

E votes **1** in all vertex positions.

S_{ab} votes **0** in all except vertex position a or b.

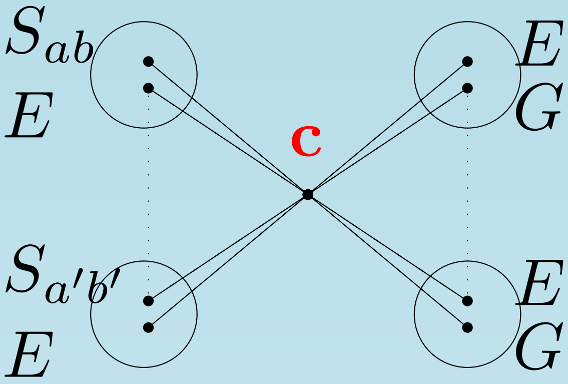
$$\begin{array}{cc|c|cc} D & D & C & D & D \\ C_a & D & C_b & & \\ & & C_a & D & C_b \end{array}$$

Either vertex v_a or vertex v_b is part of the cover.

Minimal String $\leftrightarrow$ Minimum Cover

$(E, G) \quad \rightarrow \quad c = DDCDD$

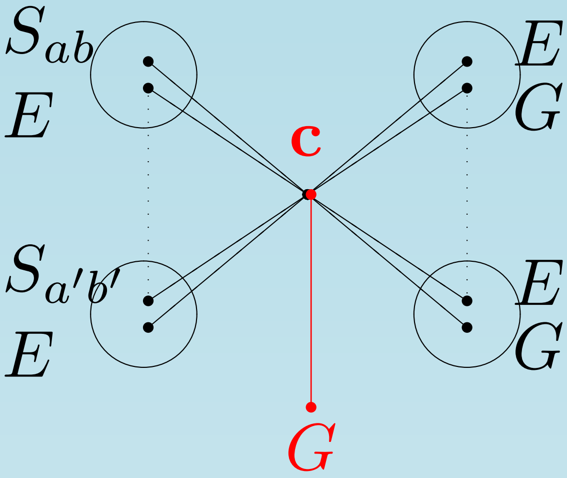
$(E, S_{ab}) \quad \rightarrow \quad c =$ vertex cover



Minimal String ↔ Minimum Cover

$(E, G) \quad \rightarrow \quad c = DDCDD$

$(E, S_{ab}) \quad \rightarrow \quad c =$ vertex cover

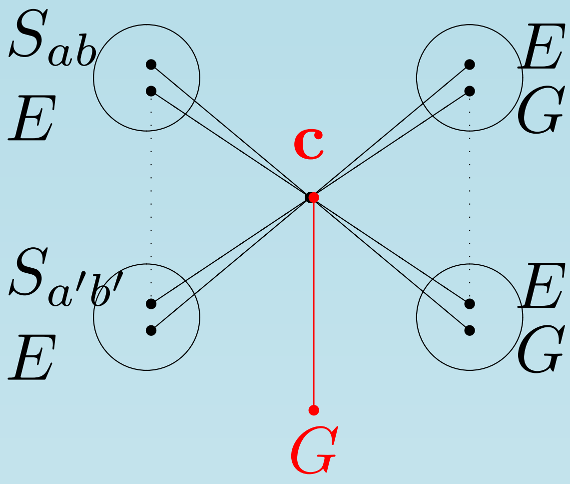


$G = DDC_{\emptyset}DD$ penalizes each **1** in vertex positions.

Minimal String $\leftrightarrow$ Minimum Cover

$$(E, G) \quad \rightarrow \quad c = DDCDD$$

$$(E, S_{ab}) \quad \rightarrow \quad c = \text{vertex cover}$$

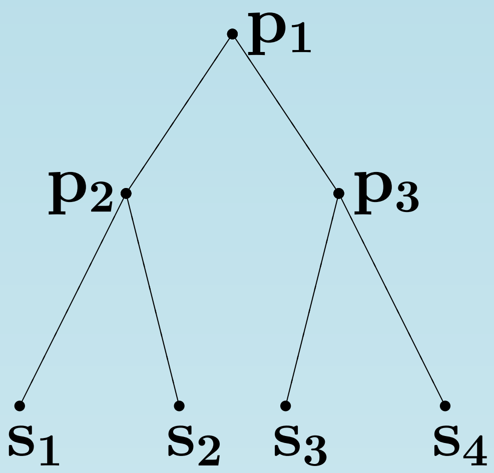


$G = DDC_{\emptyset}DD$ penalizes each **1** in vertex positions.

String minimizing $\sum_i \mathtt{d}(c, s_i) \leftrightarrow$ minimum cover

Tree Alignment (given phylogeny)

Star Alignment not a good model of evolution.

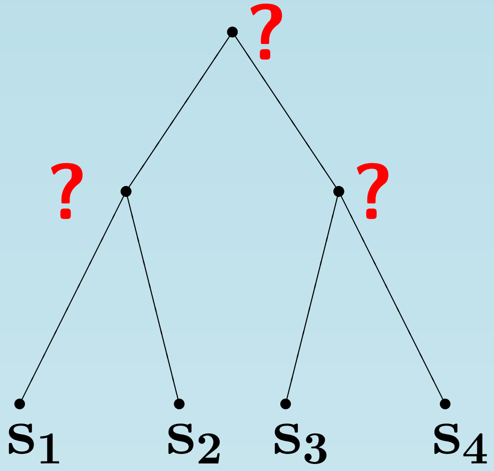


Tree Alignment (given phylogeny)

Star Alignment not a good model of evolution.

Input: k strings $s_1 \ldots s_k$
phylogeny T
Output: strings $p_1 \ldots p_{k-1}$

$$\min \sum_{(a,b) \in \mathtt{E}(T)} \mathtt{d}(a, b)$$

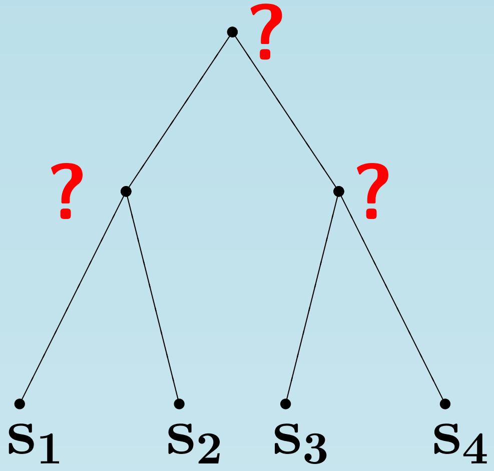


Tree Alignment (given phylogeny)

Star Alignment not a good model of evolution.

Input: k strings $s_1 \ldots s_k$
phylogeny T
Output: strings $p_1 \ldots p_{k-1}$

$$\min \sum_{(a,b)\in \mathtt{E}(T)} \mathtt{d}(a,b)$$

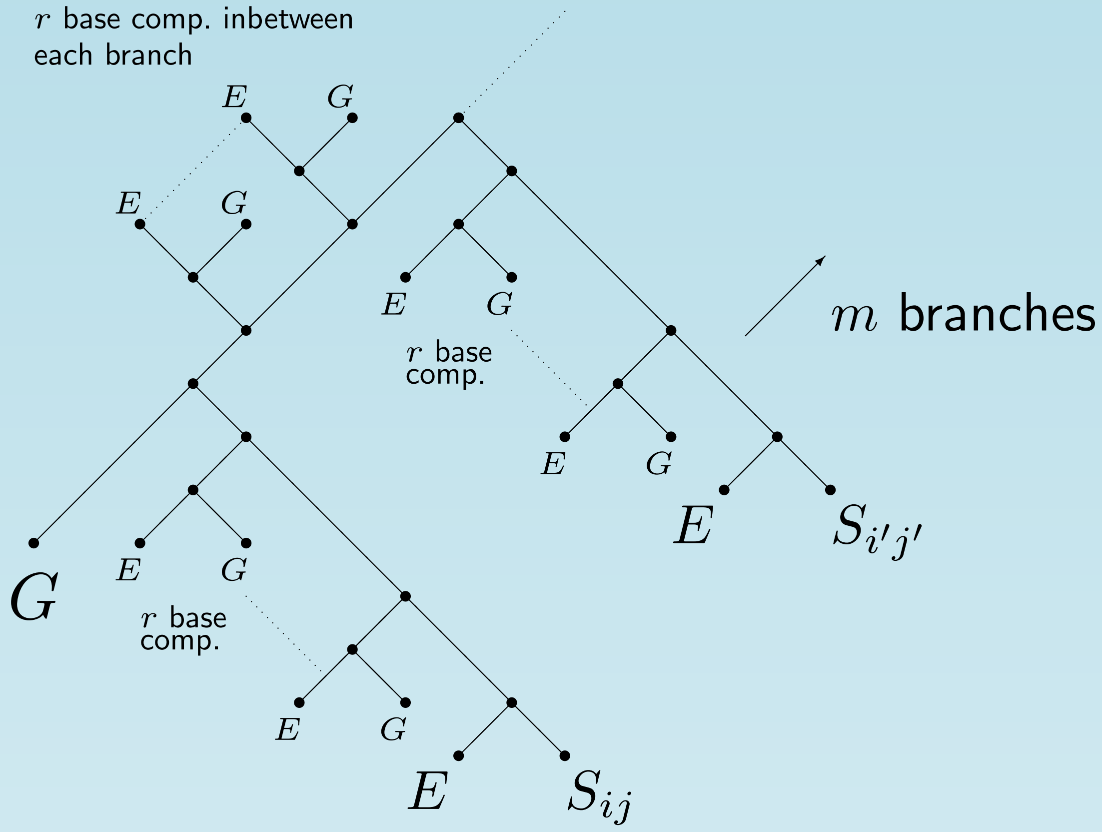


Symbol distance metric $\Longrightarrow$ Pairwise alignment distance metric

Tree Alignment is a special case of **Steiner Tree** in a metric space.

Construction Idea

Same idea as for STAR ALIGNMENT.



Overview and Open problems

Problem	**Here**	**Approx**
SP-score	all metrics	2-approx [GPBL]
Star Alignment	all metrics	2-approx
Tree Alignment (given phylogeny)	all metrics	PTAS [WJGL]

Overview and Open problems

Problem	**Here**	**Approx**
SP-score	all metrics	2-approx [GPBL]
Star Alignment	all metrics	2-approx
Tree Alignment (given phylogeny)	all metrics	PTAS [WJGL]
Consensus Patterns	NP-hard	PTAS [LMW]
Substring Parsimony	NP-hard	PTAS [$\approx$ WJGL]

Acknowledgments

My advisor
Prof. Jens Lagergren

Prof. Benny for hosting me

Thanks!