Genomic distance under gene substitutions

Marília Braga, Raphael Machado,
Leonardo Ribeiro and Jens Stoye

(Inmetro - Brazil / Bielefeld University - Germany)

RECOMB-CG 2011

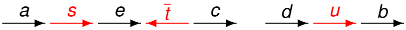
Universität Bielefeld

INMETRO

Overview

Overview

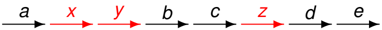
Motivation and Background



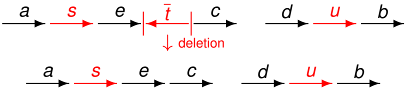
Sorting genomes with unequal contents, but without duplicated markers:

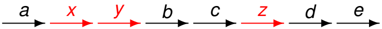
. rearrangements that change number of chromosomes, order and orientation of markers (modify the genome **organization**)

.insertions and deletions (modify the genome **content**)

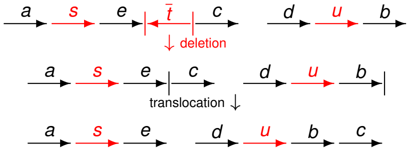


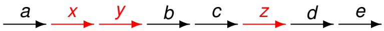
Motivation and Background



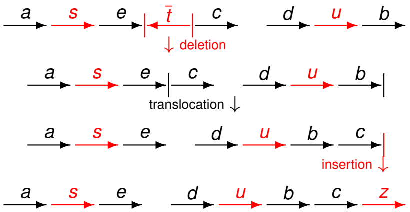


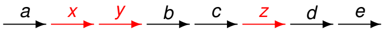
Motivation and Background



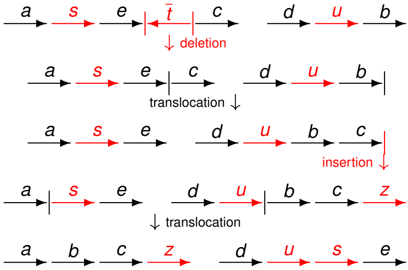


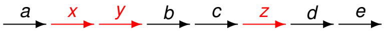
Motivation and Background



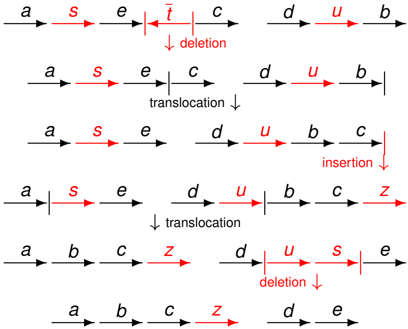


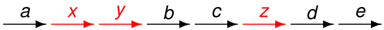
Motivation and Background



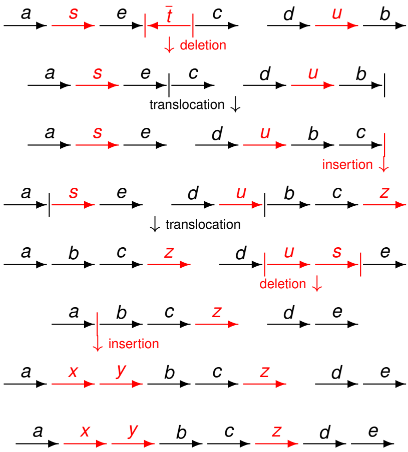


Motivation and Background

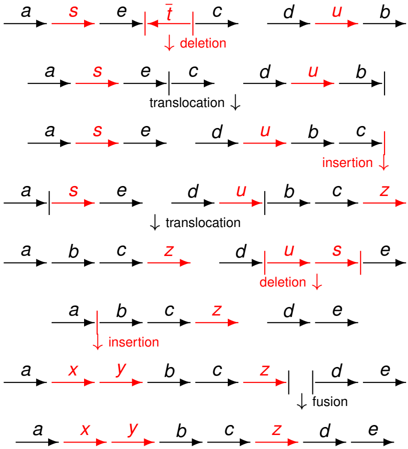




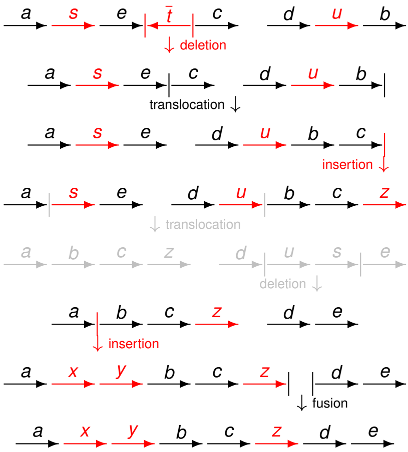
Motivation and Background



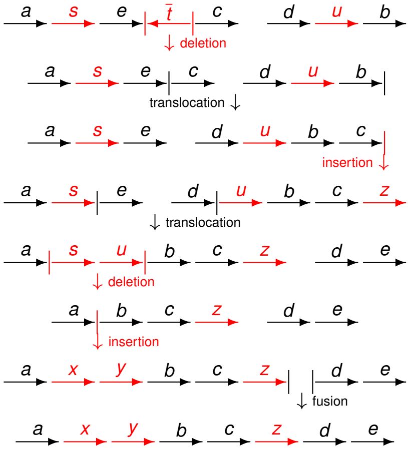
Motivation and Background



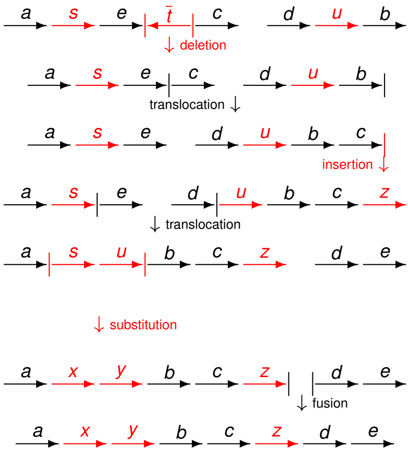
Motivation and Background



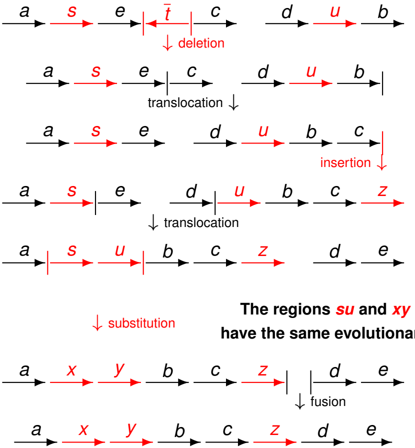
Motivation and Background



Motivation and Background

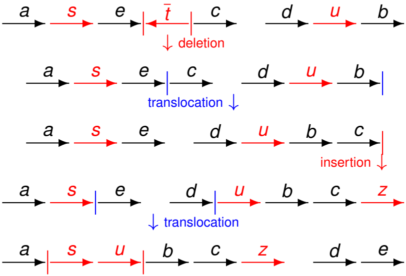


Motivation and Background



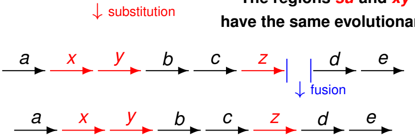
The regions *su* and *xy* could have the same evolutionary origin

Motivation and Background



(translocations and fusions are **double-cut-and-join** operations)

The regions *su* and *xy* could have the same evolutionary origin



Background

The double cut and join model (DCJ) [Yancopoulos *et al.* 2005]:

Background

The double cut and join model (DCJ) [Yancopoulos *et al.* 2005]:

- genomes with the **same content**

Background

The double cut and join model (DCJ) [Yancopoulos *et al.* 2005]:

- genomes with the **same content**
- **cuts** the genome **twice** and **rejoins** loose ends: represents most genome rearrangement operations (inversions, translocations, fusions, fissions. . .)

Motivation and Background

Background

The double cut and join model (DCJ) [Yancopoulos *et al.* 2005]:

- genomes with the **same content**
- **cuts** the genome **twice** and **rejoins** loose ends: represents most genome rearrangement operations (inversions, translocations, fusions, fissions. . .)
- **DCJ distance** can be computed in **linear time** [Bergeron *et al.* 2006]

Motivation and Background

Background

The double cut and join model (DCJ) [Yancopoulos *et al.* 2005]:

- genomes with the **same content**
- **cuts** the genome **twice** and **rejoins** loose ends: represents most genome rearrangement operations (inversions, translocations, fusions, fissions. . .)
- **DCJ distance** can be computed in **linear time** [Bergeron *et al.* 2006]

DCJ with indels [Yancopoulos *et al.* 2008]:

Motivation and Background

Background

The double cut and join model (DCJ) [Yancopoulos *et al.* 2005]:

- genomes with the **same content**
- **cuts** the genome **twice** and **rejoins** loose ends: represents most genome rearrangement operations (inversions, translocations, fusions, fissions. . .)
- **DCJ distance** can be computed in **linear time** [Bergeron *et al.* 2006]

DCJ with indels [Yancopoulos *et al.* 2008]:

- genomes with **unequal contents**

Background

The double cut and join model (DCJ) [Yancopoulos *et al.* 2005]:

- genomes with the **same content**
- **cuts** the genome **twice** and **rejoins** loose ends: represents most genome rearrangement operations (inversions, translocations, fusions, fissions. . .)
- **DCJ distance** can be computed in **linear time** [Bergeron *et al.* 2006]

DCJ with indels [Yancopoulos *et al.* 2008]:

- genomes with **unequal contents**
- allows **DCJ operations**, **insertions** and **deletions** (**indels**)

Background

The double cut and join model (DCJ) [Yancopoulos *et al.* 2005]:

- genomes with the **same content**
- **cuts** the genome **twice** and **rejoins** loose ends: represents most genome rearrangement operations (inversions, translocations, fusions, fissions. . .)
- **DCJ distance** can be computed in **linear time** [Bergeron *et al.* 2006]

DCJ with indels [Yancopoulos *et al.* 2008]:

- genomes with **unequal contents**
- allows **DCJ operations**, **insertions** and **deletions** (**indels**)
- A **block of markers** can be inserted or deleted **at once**

Motivation and Background

Background

The double cut and join model (DCJ) [Yancopoulos *et al.* 2005]:

- genomes with the **same content**
- **cuts** the genome **twice** and **rejoins** loose ends: represents most genome rearrangement operations (inversions, translocations, fusions, fissions. . .)
- **DCJ distance** can be computed in **linear time** [Bergeron *et al.* 2006]

DCJ with indels [Yancopoulos *et al.* 2008]:

- genomes with **unequal contents**
- allows **DCJ operations**, **insertions** and **deletions** (**indels**)
- A **block of markers** can be inserted or deleted **at once**
- **DCJ-indel distance** can be computed in **linear time** [WABI 2010]

Our results

Our results

DCJ with substitutions:

Our results

DCJ with substitutions:

- genomes with **unequal contents**

Our results

DCJ with substitutions:

- genomes with **unequal contents**
- allows **DCJ operations** and **substitutions** (include **indels**):
  more parsimonious model

Our results

DCJ with substitutions:

- genomes with **unequal contents**
- allows **DCJ operations** and **substitutions** (include **indels**):
  more parsimonious model
- DCJ-substitution distance is **upper bounded** by the DCJ-indel distance

Our results

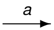
DCJ with substitutions:

- genomes with **unequal contents**
- allows **DCJ operations** and **substitutions** (include **indels**):
  more parsimonious model
- DCJ-substitution distance is **upper bounded** by the DCJ-indel distance
- DCJ-substitution distance can be **exactly computed in linear time**

Overview

Definitions

- *marker:* piece of DNA that has an orientation



Preliminaries

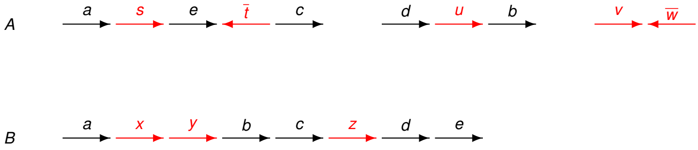
Definitions

- *marker:* piece of DNA that has an orientation
- A *genome* is a set of chromosomes composed of a finite number of markers

Preliminaries

Definitions

- *marker:* piece of DNA that has an orientation
- A *genome* is a set of chromosomes composed of a finite number of markers



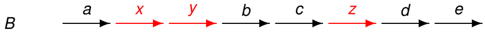
![INMETRO / Universität Bielefeld]

Preliminaries

Definitions

- *marker:* piece of DNA that has an orientation
- A *genome* is a set of chromosomes composed of a finite number of markers
- $\mathcal{G}$*:* set of markers that occur once in A and once in B

$$\mathcal{G} = \{a, b, c, d, e\}$$



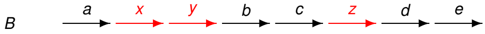
Definitions

- *marker:* piece of DNA that has an orientation
- A *genome* is a set of chromosomes composed of a finite number of markers
- $\mathcal{G}$: set of markers that occur once in A and once in B
- $\mathcal{A}$ and $\mathcal{B}$: sets of *unique markers* of A resp. B

$\mathcal{A} = \{s, t, u, v, w\}$

$\mathcal{G} = \{a, b, c, d, e\}$

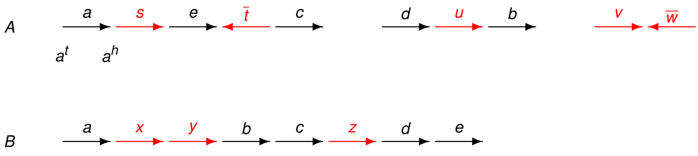
$\mathcal{B} = \{x, y, z\}$



Preliminaries

Definitions

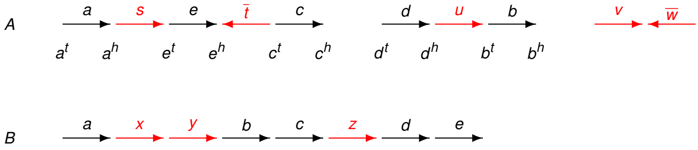
- *marker:* piece of DNA that has an orientation
- A *genome* is a set of chromosomes composed of a finite number of markers
- $\mathcal{G}$: set of markers that occur once in A and once in B
- $\mathcal{A}$ and $\mathcal{B}$: sets of *unique markers* of A resp. B
- a marker $g \in \mathcal{G}$ has two extremities: *head* (g^h) and *tail* (g^t)



Preliminaries

Definitions

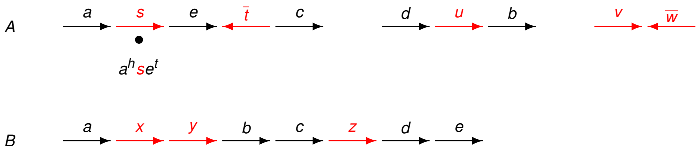
- *marker:* piece of DNA that has an orientation
- A *genome* is a set of chromosomes composed of a finite number of markers
- $\mathcal{G}$*:* set of markers that occur once in A and once in B
- $\mathcal{A}$ and $\mathcal{B}$: sets of *unique markers* of A resp. B
- a marker $g \in \mathcal{G}$ has two extremities: *head* (g^h) and *tail* (g^t)



Preliminaries

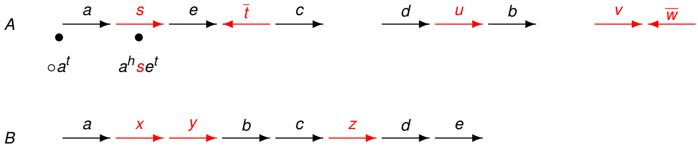
Definitions

- *marker:* piece of DNA that has an orientation
- A *genome* is a set of chromosomes composed of a finite number of markers
- $\mathcal{G}$: set of markers that occur once in A and once in B
- $\mathcal{A}$ and $\mathcal{B}$: sets of *unique markers* of A resp. B
- a marker $g \in \mathcal{G}$ has two extremities: *head* (g^h) and *tail* (g^t)
- *Adjacency:* occurs between extremities of “consecutive" markers from $\mathcal{G}$ (and/or *telomeres*) with a *label* from $\mathcal{A}$ or $\mathcal{B}$



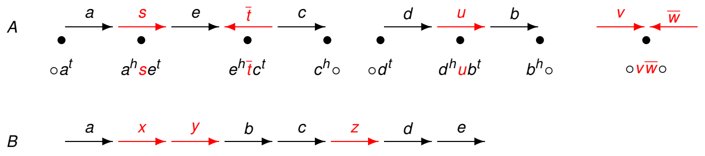
Definitions

- *marker:* piece of DNA that has an orientation
- A *genome* is a set of chromosomes composed of a finite number of markers
- $\mathcal{G}$: set of markers that occur once in A and once in B
- $\mathcal{A}$ and $\mathcal{B}$: sets of *unique markers* of A resp. B
- a marker $g \in \mathcal{G}$ has two extremities: *head* (g^h) and *tail* (g^t)
- *Adjacency:* occurs between extremities of “consecutive" markers from $\mathcal{G}$ (and/or *telomeres*) with a *label* from $\mathcal{A}$ or $\mathcal{B}$

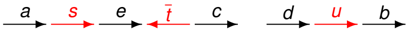


Definitions

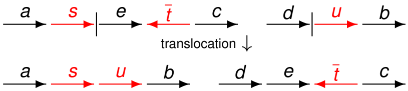
- *marker:* piece of DNA that has an orientation
- A *genome* is a set of chromosomes composed of a finite number of markers
- $\mathcal{G}$: set of markers that occur once in A and once in B
- $\mathcal{A}$ and $\mathcal{B}$: sets of *unique markers* of A resp. B
- a marker $g \in \mathcal{G}$ has two extremities: *head* (g^h) and *tail* (g^t)
- *Adjacency:* occurs between extremities of “consecutive" markers from $\mathcal{G}$ (and/or *telomeres*) with a *label* from $\mathcal{A}$ or $\mathcal{B}$



DCJ operation

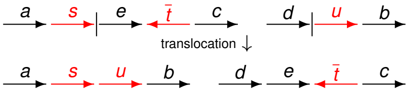


DCJ operation



DCJ: $a^h s | e^t \ , \ d^h | u b^t \ \rightarrow \ a^h s | u b^t \ , \ d^h | e^t$

DCJ operation



DCJ: $a^h s | e^t \ , \ d^h | u b^t \ \rightarrow \ a^h s | u b^t \ , \ d^h | e^t$

(a DCJ operation rearranges two adjacencies)

Adjacency graph and DCJ-distance

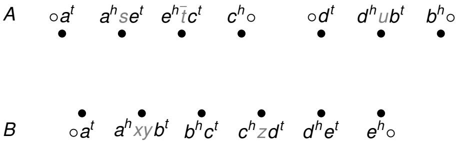
[Bergeron *et al.* 2006]

A $\circ a^t$ $a^h s e^t$ $e^h \bar{t} c^t$ $c^h \circ$ $\circ d^t$ $d^h u b^t$ $b^h \circ$

B $\circ a^t$ $a^h xy b^t$ $b^h c^t$ $c^h z d^t$ $d^h e^t$ $e^h \circ$

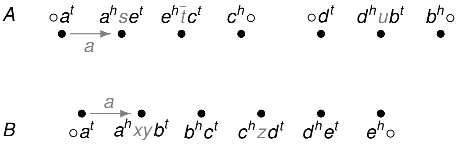
Adjacency graph and DCJ-distance

[Bergeron *et al.* 2006]



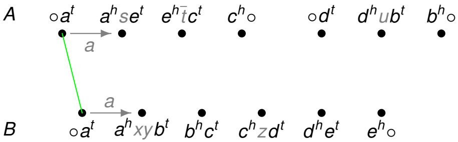
Adjacency graph and DCJ-distance

[Bergeron *et al.* 2006]



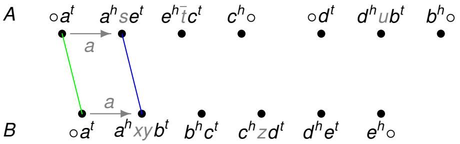
Adjacency graph and DCJ-distance

[Bergeron *et al.* 2006]



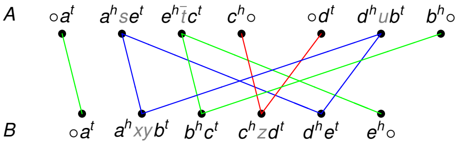
Adjacency graph and DCJ-distance

[Bergeron *et al.* 2006]



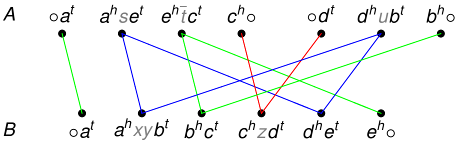
Adjacency graph and DCJ-distance

[Bergeron *et al.* 2006]



Adjacency graph and DCJ-distance

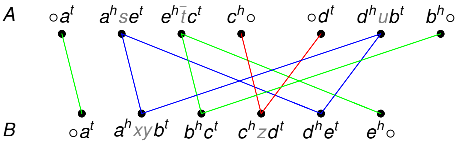
[Bergeron *et al.* 2006]



Components of AG(A, B): 1 cycle, 2 *AB*-paths and 1 *AA*-path

Adjacency graph and DCJ-distance

[Bergeron *et al.* 2006]



Components of AG(A, B): 1 cycle, 2 AB-paths and 1 AA-path

Minimum number of DCJs necessary to sort A into B (ignoring labels):

$$d_{\text{DCJ}}(A, B) = |\mathcal{G}| - (c + \frac{b}{2}),$$

where $c = \#$ cycles, $b = \#$ AB-paths

Types of DCJ operation

c: number of cycles in AG(A, B)

b: number of AB-paths in AG(A, B)

DCJ	effect on AG(A, B)	weight
optimal	increase c or b	0
neutral	c and b unchanged	+1
counter-optimal	decrease c or b	+2

Types of DCJ operation

c: number of cycles in AG(A, B)

b: number of AB-paths in AG(A, B)

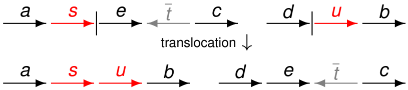
DCJ	effect on AG(A, B)	weight
optimal	increase c or b	0
neutral	c and b unchanged	+1
counter-optimal	decrease c or b	+2

An **optimal DCJ** either **creates** one cycle or two AB-paths.

Overview

Using DCJs to save substitutions

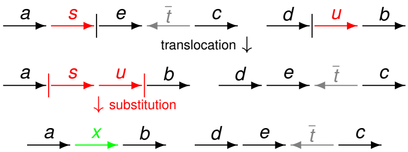
DCJs x Substitutions



DCJ: $a^h s|e^t \,,\; d^h|ub^t \;\to\; a^h s|ub^t \,,\; d^h|e^t$

Using DCJs to save substitutions

DCJs x Substitutions

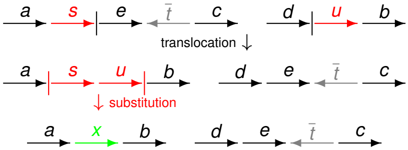


DCJ: $a^h s|e^t \; , \; d^h|ub^t \;\rightarrow\; a^h s|ub^t \; , \; d^h|e^t$

Substitution: $a^h|su|b^t \;\rightarrow\; a^h|x|b^t$

Using DCJs to save substitutions

DCJs x Substitutions



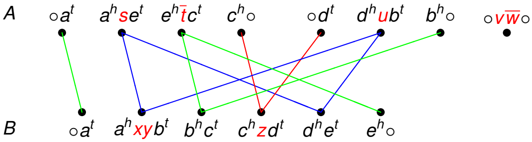
DCJ: $a^h s | e^t \,,\, d^h | u b^t \;\rightarrow\; a^h s | u b^t \,,\, d^h | e^t$

Substitution: $a^h | s u | b^t \;\rightarrow\; a^h | x | b^t$

(a substitution affects the label of a single adjacency)

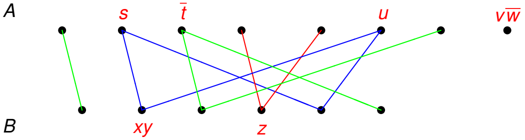
Using DCJs to save substitutions

Representing a single component in $AG(A, B)$



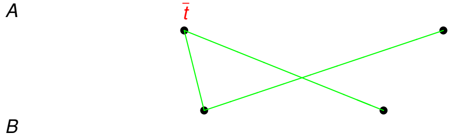
Using DCJs to save substitutions

Representing a single component in $AG(A, B)$



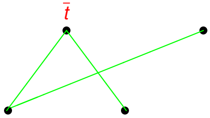
Using DCJs to save substitutions

Representing a single component in $\text{AG}(A, B)$



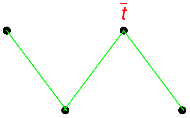
Using DCJs to save substitutions

Representing a single component in $AG(A, B)$



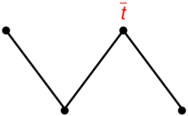
Using DCJs to save substitutions

Representing a single component in $AG(A, B)$



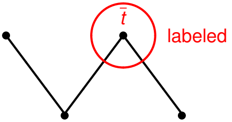
Using DCJs to save substitutions

Representing a single component in $AG(A, B)$



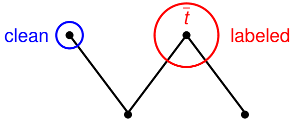
Using DCJs to save substitutions

Representing a single component in $AG(A, B)$



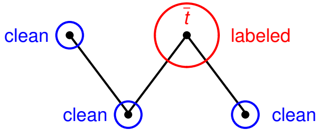
Using DCJs to save substitutions

Representing a single component in $AG(A, B)$



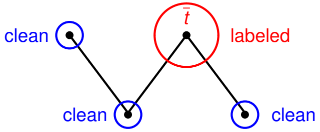
Using DCJs to save substitutions

Representing a single component in $AG(A, B)$



Using DCJs to save substitutions

Representing a single component in AG(A, B)

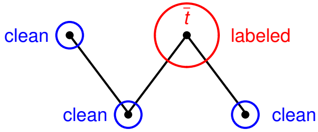


Ignoring substitutions (labels), it is possible to
sort a component individually with **optimal DCJs only**:

$d_{\text{DCJ}}(C) =$ number of optimal DCJs required to sort the component C

Using DCJs to save substitutions

Representing a single component in AG(A, B)



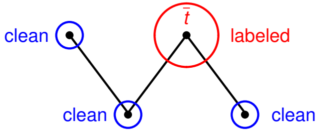
Ignoring substitutions (labels), it is possible to
sort a component individually with **optimal DCJs only**:

$d_{\text{DCJ}}(C)$ = number of optimal DCJs required to sort the component C

$$\sum_{C \in \text{AG}(A,B)} d_{\text{DCJ}}(C) = d_{\text{DCJ}}(A, B)$$

Using DCJs to save substitutions

Representing a single component in $\mathrm{AG}(A, B)$



Ignoring substitutions (labels), it is possible to
sort a component individually with **optimal DCJs only**:

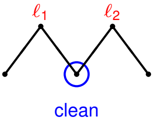
$d_{\mathrm{DCJ}}(C) =$ number of optimal DCJs required to sort the component C

$$\sum_{C \in \mathrm{AG}(A,B)} d_{\mathrm{DCJ}}(C) = d_{\mathrm{DCJ}}(A, B) \quad \left(= |\mathcal{G}| - (c + \frac{b}{2}) \right)$$

Using DCJs to save substitutions

Sorting the components of AG(A, B) individually

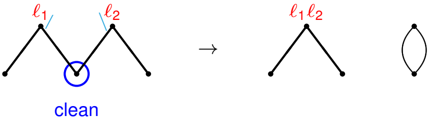
An optimal DCJ *accumulates* labels in a single vertex:



Using DCJs to save substitutions

Sorting the components of $AG(A, B)$ individually

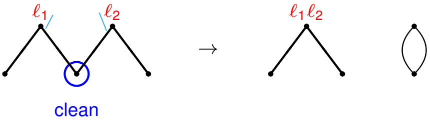
An optimal DCJ *accumulates* labels in a single vertex:



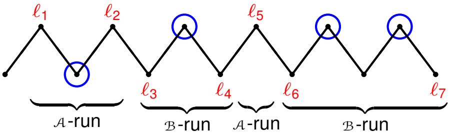
Using DCJs to save substitutions

Sorting the components of AG(A, B) individually

An optimal DCJ *accumulates* labels in a single vertex:



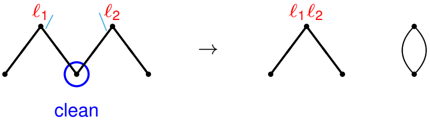
Runs of a component C:



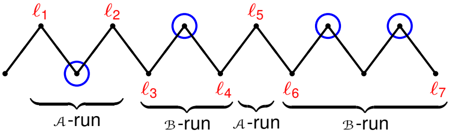
Using DCJs to save substitutions

Sorting the components of AG(A, B) individually

An optimal DCJ *accumulates* labels in a single vertex:



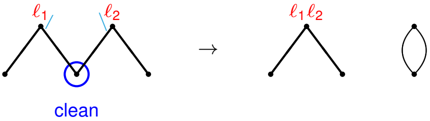
Runs of a component C:
$\Lambda(C) = 4$



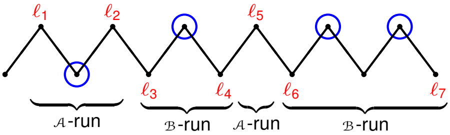
Using DCJs to save substitutions

Sorting the components of AG(A, B) individually

An optimal DCJ *accumulates* labels in a single vertex:



Runs of a component C:
$\Lambda(C) = 4$

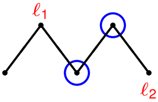


Each *run* can be entirely *accumulated* in the label of a single adjacency with optimal DCJs.

Using DCJs to save substitutions

Sorting the components of $\mathrm{AG}(A, B)$ individually

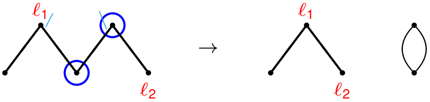
Optimal DCJs
eliminate gaps
between runs:



Using DCJs to save substitutions

Sorting the components of AG(A, B) individually

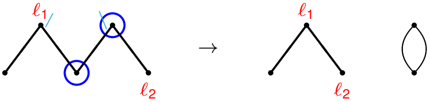
Optimal DCJs *eliminate gaps* between runs:



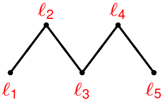
Using DCJs to save substitutions

Sorting the components of AG(A, B) individually

Optimal DCJs *eliminate gaps* between runs:



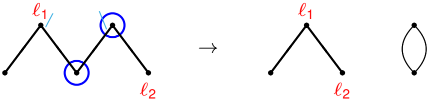
A DCJ can *merge* at most two $\mathcal{A}$-runs and two $\mathcal{B}$-runs:



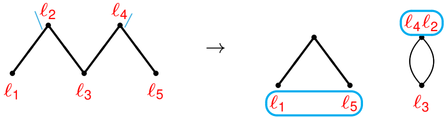
Λ : 5 *runs*

Sorting the components of AG(A, B) individually

Optimal DCJs
eliminate gaps
between runs:



A DCJ can *merge*
at most two $\mathcal{A}$-runs
and two $\mathcal{B}$-runs:



Λ : 5 *runs* 1 + 2 *runs* ($\Delta\Lambda = -2$)

Using DCJs to save substitutions

Sorting the components of AG(A, B) individually

Using DCJs to save substitutions

Sorting the components of $AG(A, B)$ individually

$d_{\text{DCJ}}(C) =$ number of optimal DCJs required to sort a component C

Using DCJs to save substitutions

Sorting the components of $\mathrm{AG}(A, B)$ individually

$d_{\mathrm{DCJ}}(C) =$ number of optimal DCJs required to sort a component C

Substitution-potential of a component C

Minimum number of substitutions (pairs of consecutive runs) obtained by sorting C with $d_{\mathrm{DCJ}}(C)$ **optimal** DCJs:

$$\sigma(C) = \left\lceil \frac{\Lambda(C) + 1}{4} \right\rceil \qquad (\textit{for } \Lambda(C) \geq 1)$$

Using DCJs to save substitutions

Sorting the components of $\text{AG}(A, B)$ individually

$d_{\text{DCJ}}(C)$ = number of optimal DCJs required to sort a component C

Substitution-potential of a component C

Minimum number of substitutions (pairs of consecutive runs) obtained by sorting C with $d_{\text{DCJ}}(C)$ **optimal** DCJs:

$$\sigma(C) = \left\lceil \frac{\Lambda(C) + 1}{4} \right\rceil \qquad (\textit{for } \Lambda(C) \geq 1)$$

$\Lambda(C)$	$\sigma(C)$
0	0
1	1
2	1
3	1
4	2
5	2
6	2
7	2
⋮	⋮

Using DCJs to save substitutions

Sorting the components of $\text{AG}(A, B)$ individually

$d_{\text{DCJ}}(C)$ = number of optimal DCJs required to sort a component C

$\Lambda(C)$	$\sigma(C)$
0	0
1	1
2	1
3	1
4	2
5	2
6	2
7	2
⋮	⋮

Substitution-potential of a component C

Minimum number of substitutions (pairs of consecutive runs) obtained by sorting C with $d_{\text{DCJ}}(C)$ **optimal** DCJs:

$$\sigma(C) = \left\lceil \frac{\Lambda(C) + 1}{4} \right\rceil \qquad (\textit{for } \Lambda(C) \geq 1)$$

A component C can be sorted with $\boldsymbol{d_{\text{DCJ}}(C)}$ **optimal DCJs** $\boldsymbol{+\sigma(C)}$ **substitutions**.

Using DCJs to save substitutions

Sorting the components of $\text{AG}(A, B)$ individually

$d_{\text{DCJ}}(C) =$ number of optimal DCJs required to sort a component C

Substitution-potential of a component C

Minimum number of substitutions (pairs of consecutive runs) obtained by sorting C with $d_{\text{DCJ}}(C)$ **optimal** DCJs:

$$\sigma(C) = \left\lceil \frac{\Lambda(C) + 1}{4} \right\rceil \qquad (\textit{for } \Lambda(C) \geq 1)$$

$\Lambda(C)$	$\sigma(C)$
0	0
1	1
2	1
3	1
4	2
5	2
6	2
7	2
⋮	⋮

A component C can be sorted with $\boldsymbol{d_{\text{DCJ}}(C)}$ **optimal DCJs** $\boldsymbol{+\sigma(C)}$ **substitutions**.

(We cannot do better with neutral or counter-optimal DCJs.)

Using DCJs to save substitutions

Sorting the components of $\text{AG}(A, B)$ individually

(Proving the potential formula by induction...)

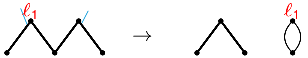
$$T(i) = \left\lceil \frac{i+1}{4} \right\rceil \quad (\textit{for} \;\; i \geq 1)$$

i	$T(i)$
0	0

Sorting the components of AG(A, B) individually

(Proving the potential formula by induction. . .)

$$T(i) = \left\lceil \frac{i+1}{4} \right\rceil \quad (\textit{for} \;\; i \geq 1)$$



$i = 1$ 0 1 ($\Delta\Lambda = 0$)

$T(1) = 1$

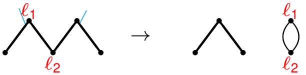
i	$T(i)$
0	0
1	1

Using DCJs to save substitutions

Sorting the components of $\mathrm{AG}(A, B)$ individually

(Proving the potential formula by induction...)

$$T(i) = \left\lceil \frac{i+1}{4} \right\rceil \quad (\textit{for} \;\; i \geq 1)$$



$i = 2$ → 0, 2 $(\Delta\Lambda = 0)$

$T(2) = 1$

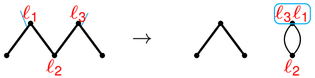
i	$T(i)$
0	0
1	1
2	1

Using DCJs to save substitutions

Sorting the components of $\mathrm{AG}(A, B)$ individually

(Proving the potential formula by induction. . .)

$$T(i) = \left\lceil \frac{i+1}{4} \right\rceil \quad (for \;\; i \geq 1)$$



$i = 3$ 0   2 $(\Delta\Lambda = -1)$

$T(3) = T(2) = 1$

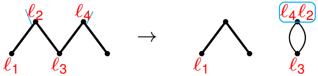
i	$T(i)$
0	0
1	1
2	1
3	1

Using DCJs to save substitutions

Sorting the components of AG(A, B) individually

(Proving the potential formula by induction. . .)

$$T(i) = \left\lceil \frac{i+1}{4} \right\rceil \quad (for \;\; i \geq 1)$$



$i = 4$ 1   2 $(\Delta\Lambda = -1)$

$T(4) = T(1) + T(2) = 1 + 1 = 2$

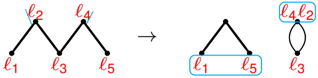
i	$T(i)$
0	0
1	1
2	1
3	1
4	2

Using DCJs to save substitutions

Sorting the components of AG(A, B) individually

(Proving the potential formula by induction...)

$$T(i) = \left\lceil \frac{i+1}{4} \right\rceil \quad (\textit{for} \;\; i \geq 1)$$



$i \geq 5$ $\quad$ $i-4$ $\quad$ 2 $\quad$ $(\Delta\Lambda = -2)$

$$T(i) = T(i-4) + T(2) = \lceil \tfrac{(i-4)+1}{4} \rceil + 1 = \lceil \tfrac{i-4+1+4}{4} \rceil = \lceil \tfrac{i+1}{4} \rceil$$

i	$T(i)$
0	0
1	1
2	1
3	1
4	2
5	2
6	2
7	2
⋮	⋮
i	$\lceil (i+1)/4 \rceil$

Using DCJs to save substitutions

Sorting the components of $\mathrm{AG}(A, B)$ individually

A component C can be sorted with $\boldsymbol{d_{\mathrm{DCJ}}(C) + \sigma(C)}$ operations

Using DCJs to save substitutions

Sorting the components of AG(A, B) individually

A component C can be sorted with $\boldsymbol{d_{\mathrm{DCJ}}(C) + \sigma(C)}$ operations

An **upper bound** to the **DCJ-substitution distance** is given by:

$$d_{\mathrm{DCJ}}^{sb}(A, B) \leq \sum_{C \in \mathrm{AG}(A,B)} (d_{\mathrm{DCJ}}(C) + \sigma(C))$$

INMETRO Universität Bielefeld

Using DCJs to save substitutions

Sorting the components of $\mathrm{AG}(A, B)$ individually

A component C can be sorted with $\boldsymbol{d_{\mathrm{DCJ}}(C) + \sigma(C)}$ operations

An **upper bound** to the **DCJ-substitution distance** is given by:

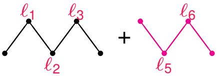
$$d_{\mathrm{DCJ}}^{sb}(A, B) \leq \sum_{C \in \mathrm{AG}(A,B)} (d_{\mathrm{DCJ}}(C) + \sigma(C))$$

$$\downarrow$$

$$\boldsymbol{d_{\mathrm{DCJ}}^{sb}(A, B) \leq d_{\mathrm{DCJ}}(A, B) + \sum_{C \in \mathrm{AG}(A,B)} \sigma(C)}$$

Using DCJs to save substitutions

Recombinations of two paths

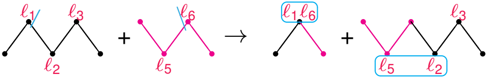


Λ:	3	+	2	
σ:	1	+	1	

$\Lambda(C)$	$\sigma(C)$
0	0
1	1
2	1
3	1
4	2
5	2
6	2
7	2
$\vdots$	$\vdots$

Using DCJs to save substitutions

Recombinations of two paths



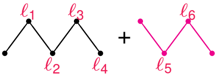
Λ: 3 + 2 → 1 + 2 ($\Delta\Lambda = -2$)

σ: 1 + 1 → 1 + 1 ($\Delta\sigma = 0$)

$\Lambda(C)$	$\sigma(C)$
0	0
1	1
2	1
3	1
4	2
5	2
6	2
7	2
⋮	⋮

Using DCJs to save substitutions

Recombinations of two paths



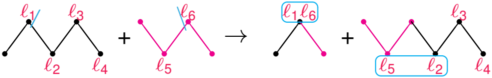
Λ: 4 + 2

σ: 2 + 1

$\Lambda(C)$	$\sigma(C)$
0	0
1	1
2	1
3	1
4	2
5	2
6	2
7	2
⋮	⋮

Using DCJs to save substitutions

Recombinations of two paths

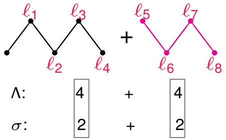


Λ: 4 + 2 → 1 + 3 ($\Delta\Lambda = -2$)

σ: 2 + 1 → 1 + 1 ($\Delta\sigma = -1$)

$\Lambda(C)$	$\sigma(C)$
0	0
1	1
2	1
3	1
4	2
5	2
6	2
7	2
⋮	⋮

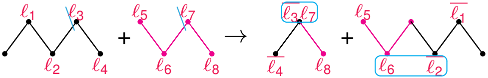
Recombinations of two paths



$\Lambda(C)$	$\sigma(C)$	
0	0	
1	1	
2	1	
3	1	
4	2	(x2)
5	2	
6	2	
7	2	
⋮	⋮	

Using DCJs to save substitutions

Recombinations of two paths



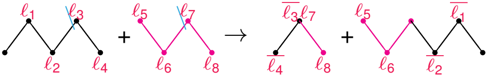
Λ: 4 + 4 → 3 + 3 ($\Delta\Lambda = -2$)

σ: 2 + 2 → 1 + 1 ($\Delta\sigma = -2$)

$\Lambda(C)$	$\sigma(C)$	
0	0	
1	1	
2	1	
3	1	(x2)
4	2	(x2)
5	2	
6	2	
7	2	
⋮	⋮	

Using DCJs to save substitutions

Recombinations of two paths



Λ: 4 + 4 → 3 + 3 ($\Delta\Lambda = -2$)

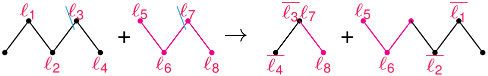
σ: 2 + 2 → 1 + 1 ($\Delta\sigma = -2$)

$$\boldsymbol{\Delta\sigma \geq -2}$$

$\Lambda(C)$	$\sigma(C)$	
0	0	
1	1	
2	1	
3	1	(x2)
4	2	(x2)
5	2	
6	2	
7	2	
⋮	⋮	

Using DCJs to save substitutions

Recombinations of two paths



Λ: 4 + 4 → 3 + 3 ($\Delta\Lambda = -2$)

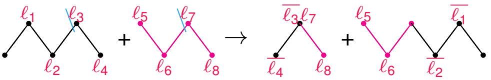
σ: 2 + 2 → 1 + 1 ($\Delta\sigma = -2$)

$$\boldsymbol{\Delta\sigma \geq -2}$$

$\Lambda(C)$	$\sigma(C)$
0	0
1	1
2	1
3	1
4	2
5	2
6	2
7	2
⋮	⋮
$4i$	$i+1$
$4i+1$	$i+1$
$4i+2$	$i+1$
$4i+3$	$i+1$

Using DCJs to save substitutions

Recombinations of two paths

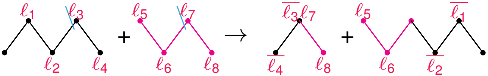


σ: 2 + 2 → 1 + 1 ($\Delta\sigma = -2$)

DCJ	Δd_{DCJ}
optimal	0
neutral	$+1$
counter-opt	$+2$

Change in the DCJ-substitution distance: $\Delta d = \Delta d_{\text{DCJ}} + \Delta\sigma$

Recombinations of two paths



$BB \quad + \quad AB \quad \longrightarrow \quad BB \quad + \quad AB \quad (\Delta d_{\mathrm{DCJ}} = +1)$

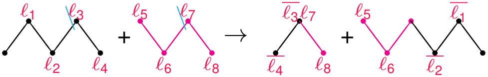
$\sigma: \quad 2 \quad + \quad 2 \quad \longrightarrow \quad 1 \quad + \quad 1 \quad (\Delta\sigma = -2)$

DCJ	Δd_{DCJ}
optimal	0
neutral	$+1$
counter-opt	$+2$

Change in the DCJ-substitution distance: $\Delta d = \Delta d_{\mathrm{DCJ}} + \Delta\sigma$

Using DCJs to save substitutions

Recombinations of two paths



BB + AB $\longrightarrow$ BB + AB $(\Delta d_{\text{DCJ}} = +1)$

σ: 2 + 2 $\longrightarrow$ 1 + 1 $(\Delta\sigma = -2)$

$$\boldsymbol{\Delta d = +1 - 2 = -1}$$

DCJ	Δd_{DCJ}
optimal	0
neutral	$+1$
counter-opt	$+2$

Change in the DCJ-substitution distance: $\Delta d = \Delta d_{\text{DCJ}} + \Delta\sigma$

Using DCJs to save substitutions

The DCJ-substitution distance

Possible ways to achieve $\Delta d \leq -1$:

	Δd_{DCJ}		$\Delta\sigma$		Δd
optimal	0		-2	$=$	-2
optimal	0		-1	$=$	-1
neutral	$+1$		-2	$=$	-1

The DCJ-substitution distance

Possible ways to achieve $\Delta d \leq -1$:

	Δd_{DCJ}	$\Delta \sigma$		Δd
optimal	0	-2	$=$	-2
optimal	0	-1	$=$	-1
neutral	$+1$	-2	$=$	-1

Considering all labeled paths of AG(A, B), find a shortest sequence of recombinations $\boldsymbol{S}$

such that the **weight** $\boldsymbol{w(S)} = \sum_{\rho \in S} \Delta d(\rho)$ is minimum: **can be solved in linear time**

Using DCJs to save substitutions

The DCJ-substitution distance

Possible ways to achieve $\Delta d \leq -1$:

	Δd_{DCJ}	$\Delta\sigma$		Δd
optimal	0	-2	$=$	-2
optimal	0	-1	$=$	-1
neutral	$+1$	-2	$=$	-1

Considering all labeled paths of AG(A, B), find a shortest sequence of recombinations $\boldsymbol{S}$

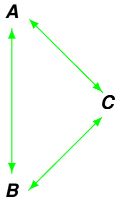
such that the **weight $\boldsymbol{w(S)}$** $= \sum_{\rho \in S} \Delta d(\rho)$ is minimum: **can be solved in linear time**

DCJ-substitution distance formula

$$d^{sb}_{\text{DCJ}}(A,B) = d_{\text{DCJ}}(A,B) + \sum_{C \in \text{AG}(A,B)} \sigma(C) + w(S)$$

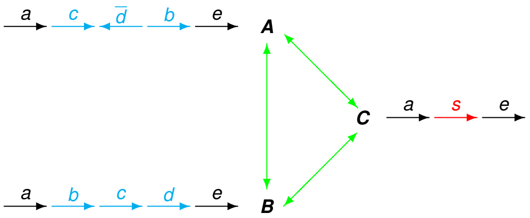
Establishing the triangular inequality

Triangular inequality:



Establishing the triangular inequality

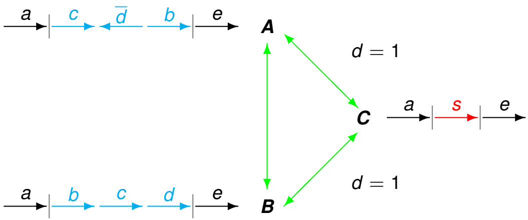
Triangular inequality:



Using DCJs to save substitutions

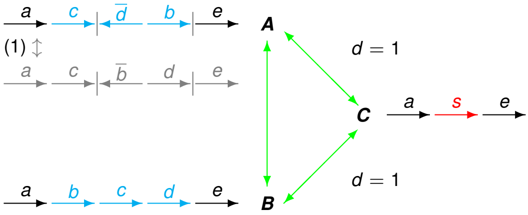
Establishing the triangular inequality

Triangular inequality:



Establishing the triangular inequality

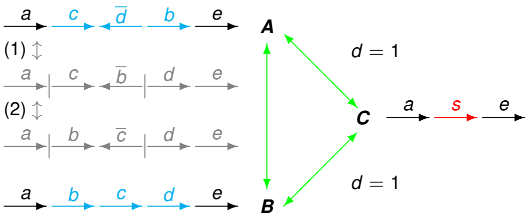
Triangular inequality:



Using DCJs to save substitutions

Establishing the triangular inequality

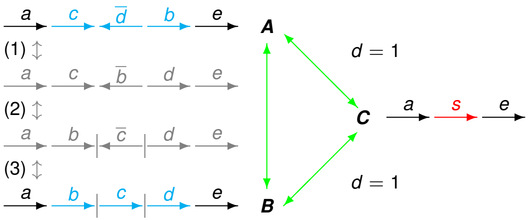
Triangular inequality:



Using DCJs to save substitutions

Establishing the triangular inequality

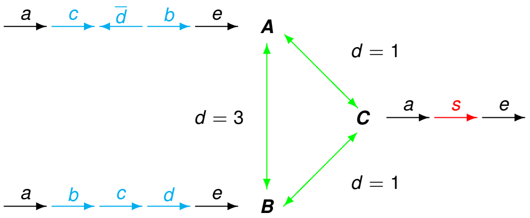
Triangular inequality:



Using DCJs to save substitutions

Establishing the triangular inequality

Triangular inequality:

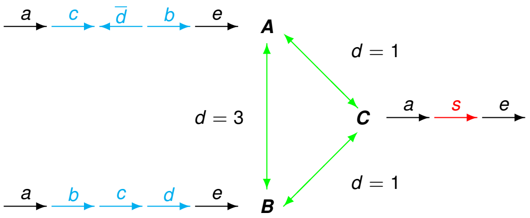


Using DCJs to save substitutions

Establishing the triangular inequality

Triangular inequality: $d(A, B) \leq d(A, C) + d(B, C)$ (!)

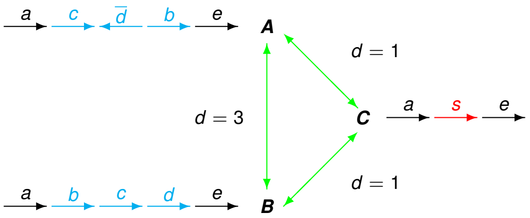
$3 > 1 + 1$ (!)



Establishing the triangular inequality

Triangular inequality: $d(A, B) \leq d(A, C) + d(B, C)$ (!)

- **Correction can be done** *a posteriori* [presented yesterday by Jens Stoye]



Overview

1. Motivation and Background
2. Preliminaries
   - Definitions
   - Adjacency graph and DCJ-distance
3. Using DCJs to save substitutions
   - Substitution potential and distance upper bound
   - Recombinations and the DCJ-substitution distance
4. **Conclusions and Future Work**

Conclusions and Future Work

Conclusions

Genomic distance under DCJs and substitutions:

Conclusions and Future Work

Conclusions

Genomic distance under DCJs and substitutions:

- Genomes with unequal contents but free of duplicated markers

Conclusions and Future Work

Conclusions

Genomic distance under DCJs and substitutions:

- Genomes with unequal contents but free of duplicated markers
- Substitutions include indels

Conclusions and Future Work

Conclusions

Genomic distance under DCJs and substitutions:

- Genomes with unequal contents but free of duplicated markers
- Substitutions include indels
- Distance can be computed in linear time

Conclusions and Future Work

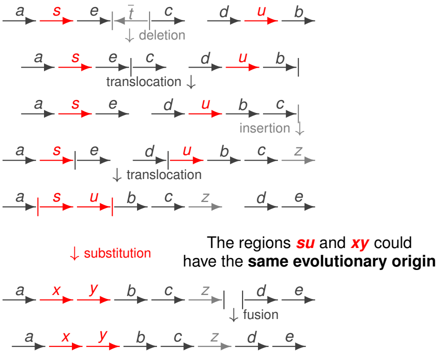
Conclusions

Genomic distance under DCJs and substitutions:

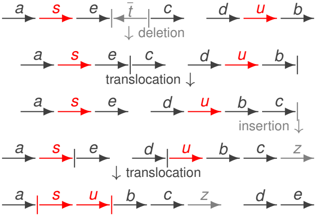
- Genomes with unequal contents but free of duplicated markers
- Substitutions include indels
- Distance can be computed in linear time
- Triangular inequality can be established *a posteriori*

Conclusions and Future Work

Future work

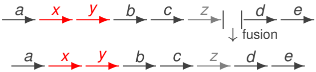


Future work



↓ substitution

The regions ***su*** and ***xy*** could be **homologous**?
(Do they correspond to an **unannotated orthology**?)



Conclusions and Future Work

Future work

Future work

- Explore the solution space of sorting genomes with DCJ operations and substitutions

Conclusions and Future Work

Future work

- Explore the solution space of sorting genomes with DCJ operations and substitutions
- Develop a method to **refine orthology assignments**

Conclusions and Future Work

Future work

- Explore the solution space of sorting genomes with DCJ operations and substitutions
- Develop a method to **refine orthology assignments**
- Extend the model to handle duplicated markers

Thank you for your attention!

References

- A. Bergeron, J. Mixtacki and J. Stoye. A unifying view of genome rearrangements. In Proceedings of *WABI*, 2006.
- M. D. V. Braga, L. C. Ribeiro, R. Machado and J. Stoye. On the weight of indels in genomic distances. *RECOMB-CG*, 2011.
- M. D. V. Braga and J. Stoye. The solution space of sorting by DCJ. *Journal of Computational Biology*, 2010.
- M. D. V. Braga, E. Willing and J. Stoye. Double cut and join with insertions and deletions. *Journal of Computational Biology*, 2011.
- S. Yancopoulos, O. Attie and R. Friedberg. Efficient sorting of genomic permutations by translocation, inversion and block interchange. *Bioinformatics*, 2005.
- S. Yancopoulos and R. Friedberg. DCJ path formulation for genome transformations which include insertions, deletions, and duplications. *Journal of Computational Biology*, 2009.