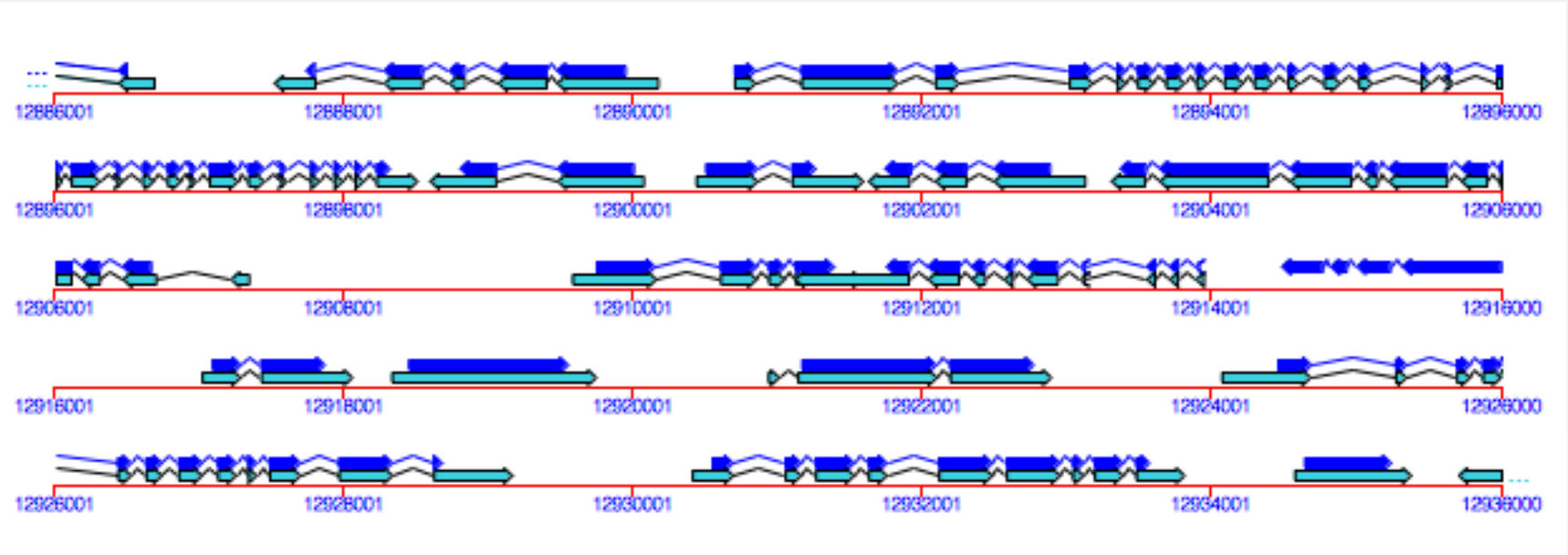
A Pseudo-Boolean programming approach to compute differences and similarities between two genomes

Annelyse Thévenin

BioInfo Lab, Laboratoire de Recherche en Informatique (LRI),
Université Paris-Sud 11, France

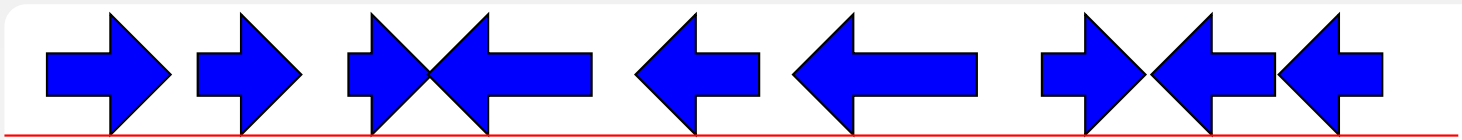
The genome consists of genes



FLAGdb - *Arabidopsis thaliana*

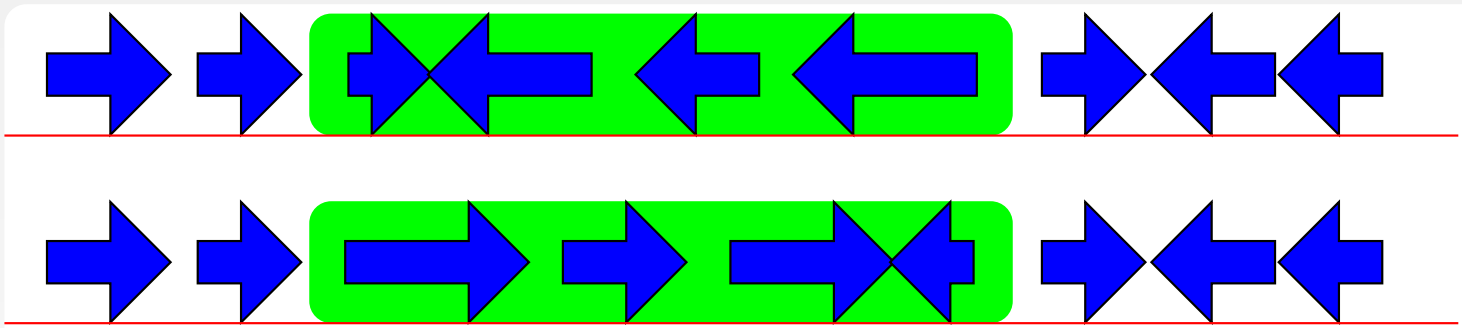
Evolution of a genome

Ancestral genome



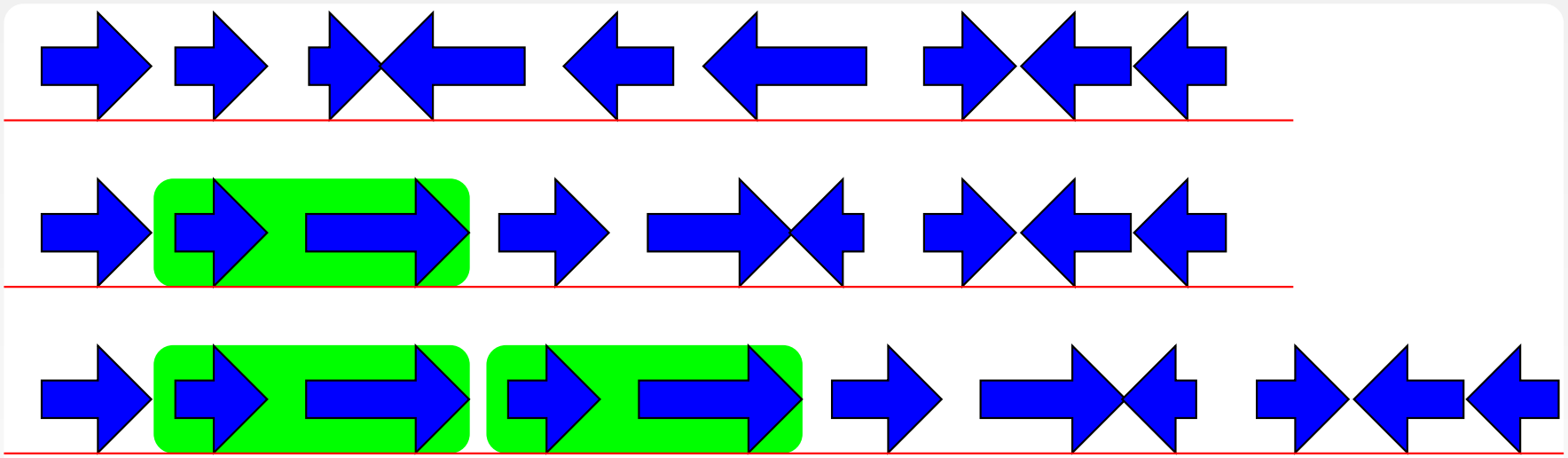
Evolution of a genome

An inversion



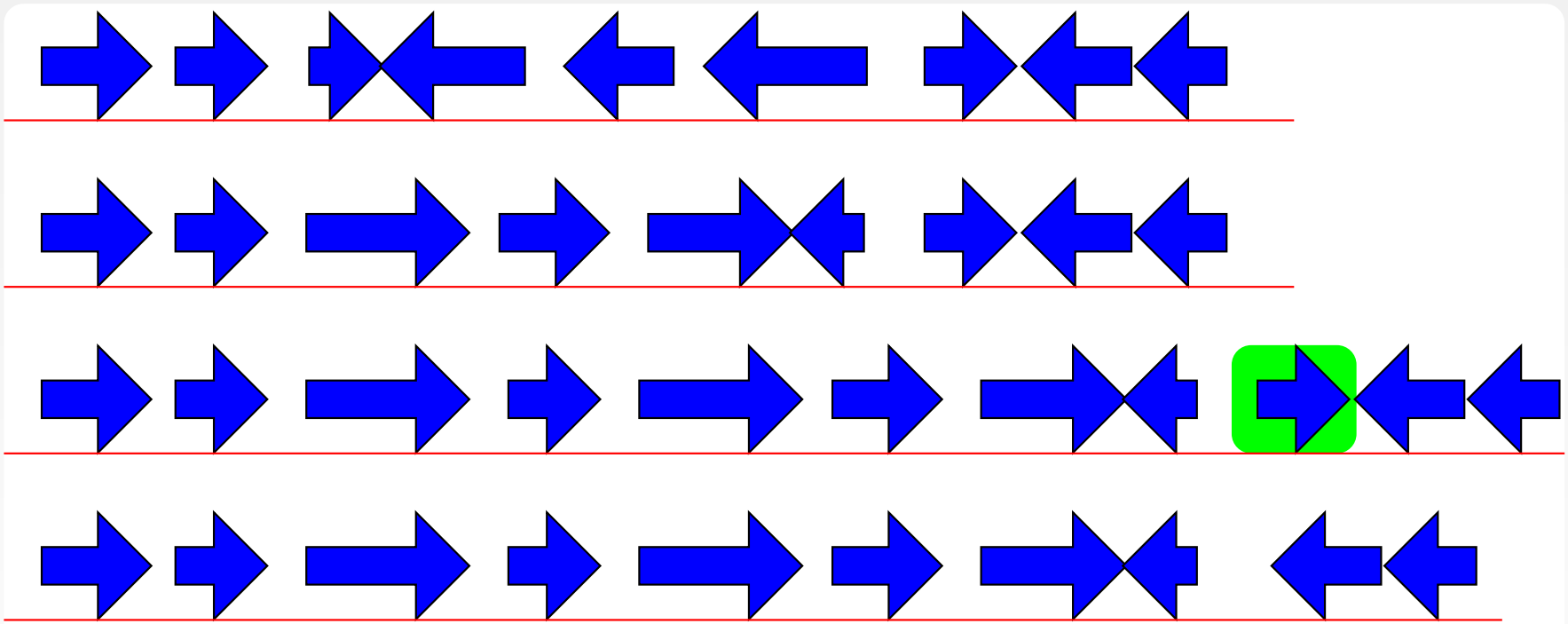
Evolution of a genome

A duplication



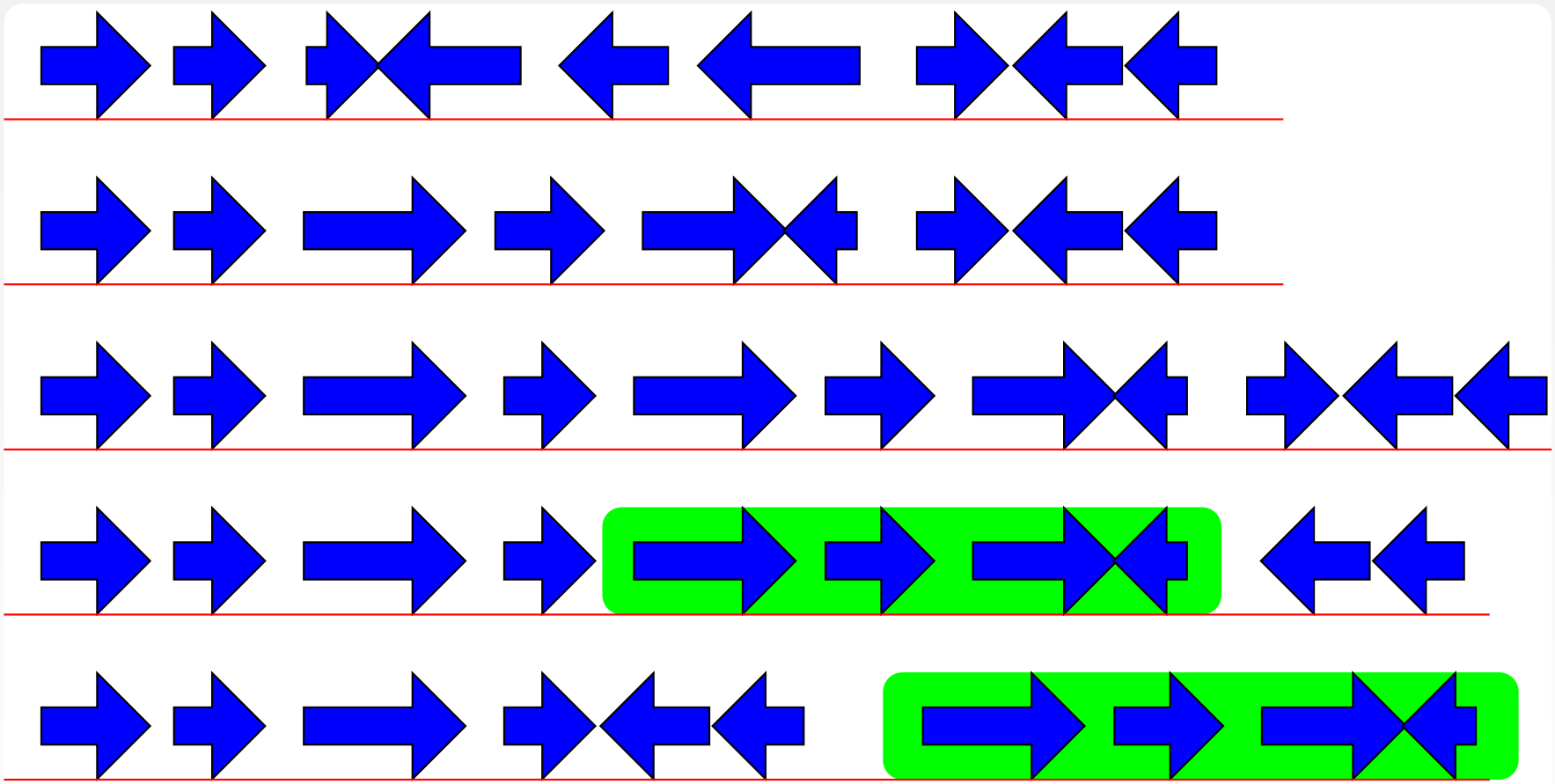
Evolution of a genome

A deletion

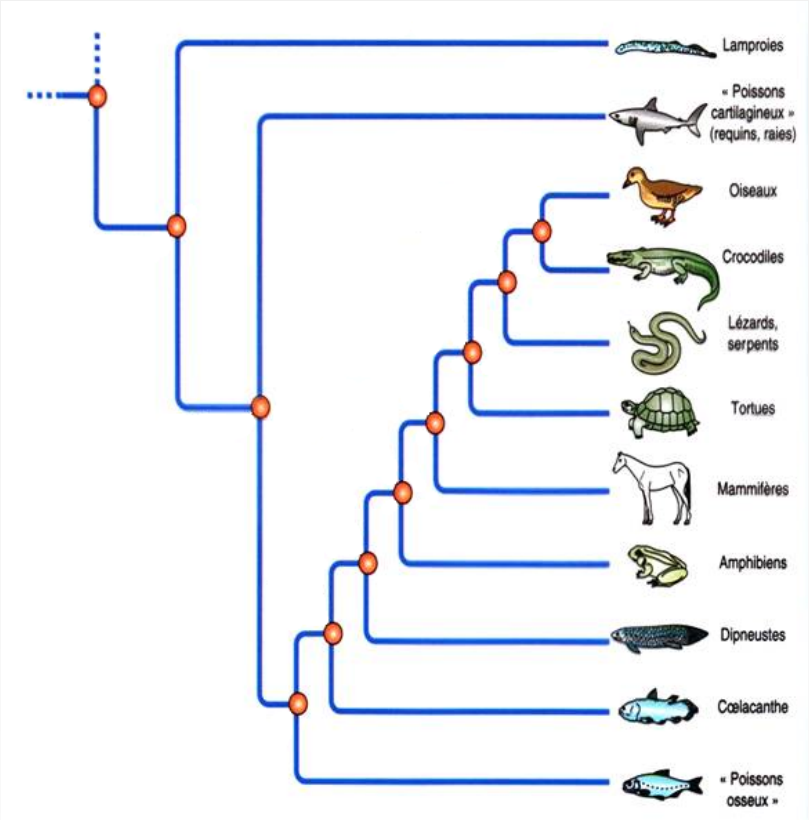


Evolution of a genome

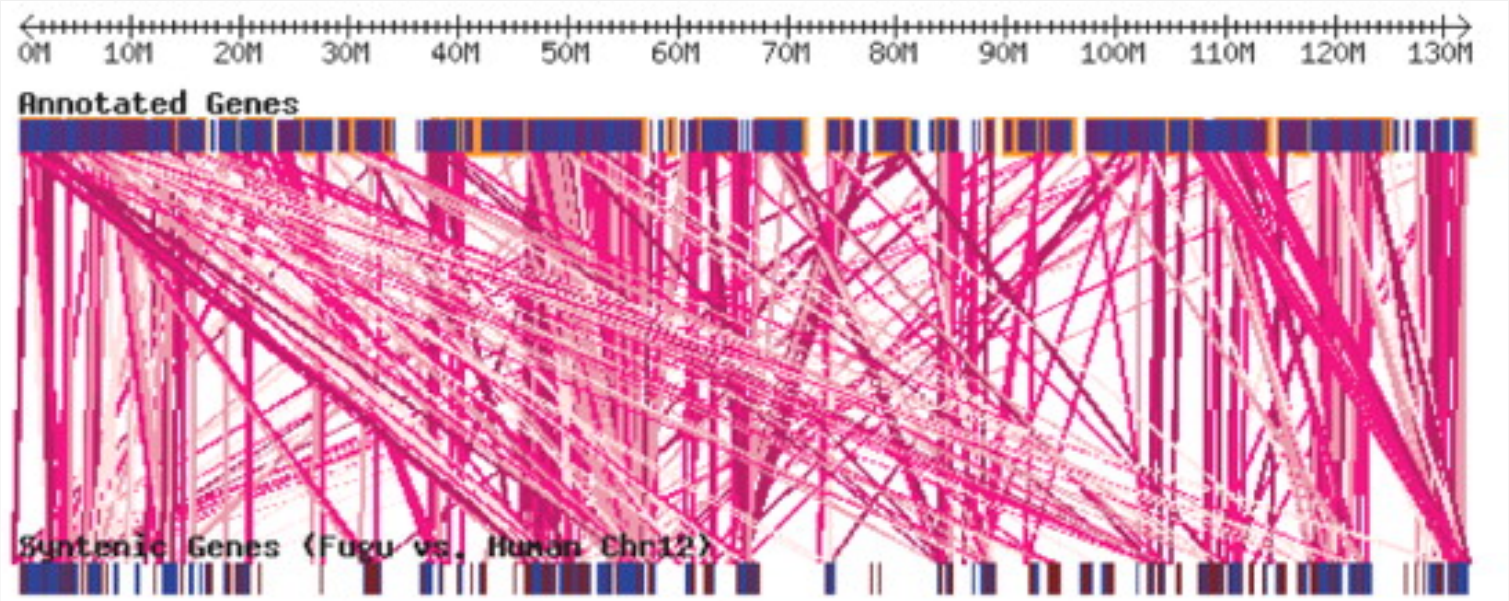
A transposition



Phylogenetic tree

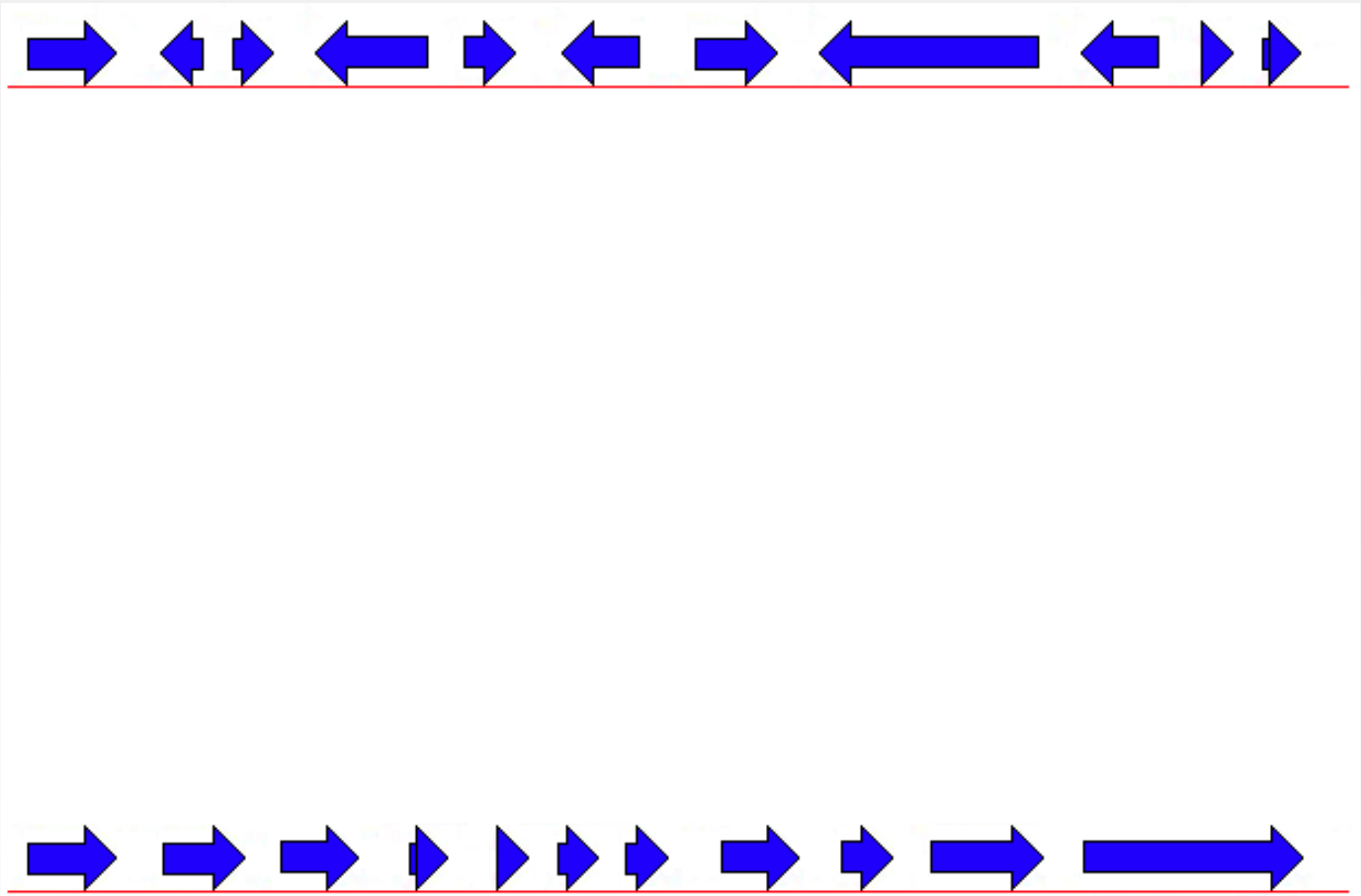


Comparison between two genomes

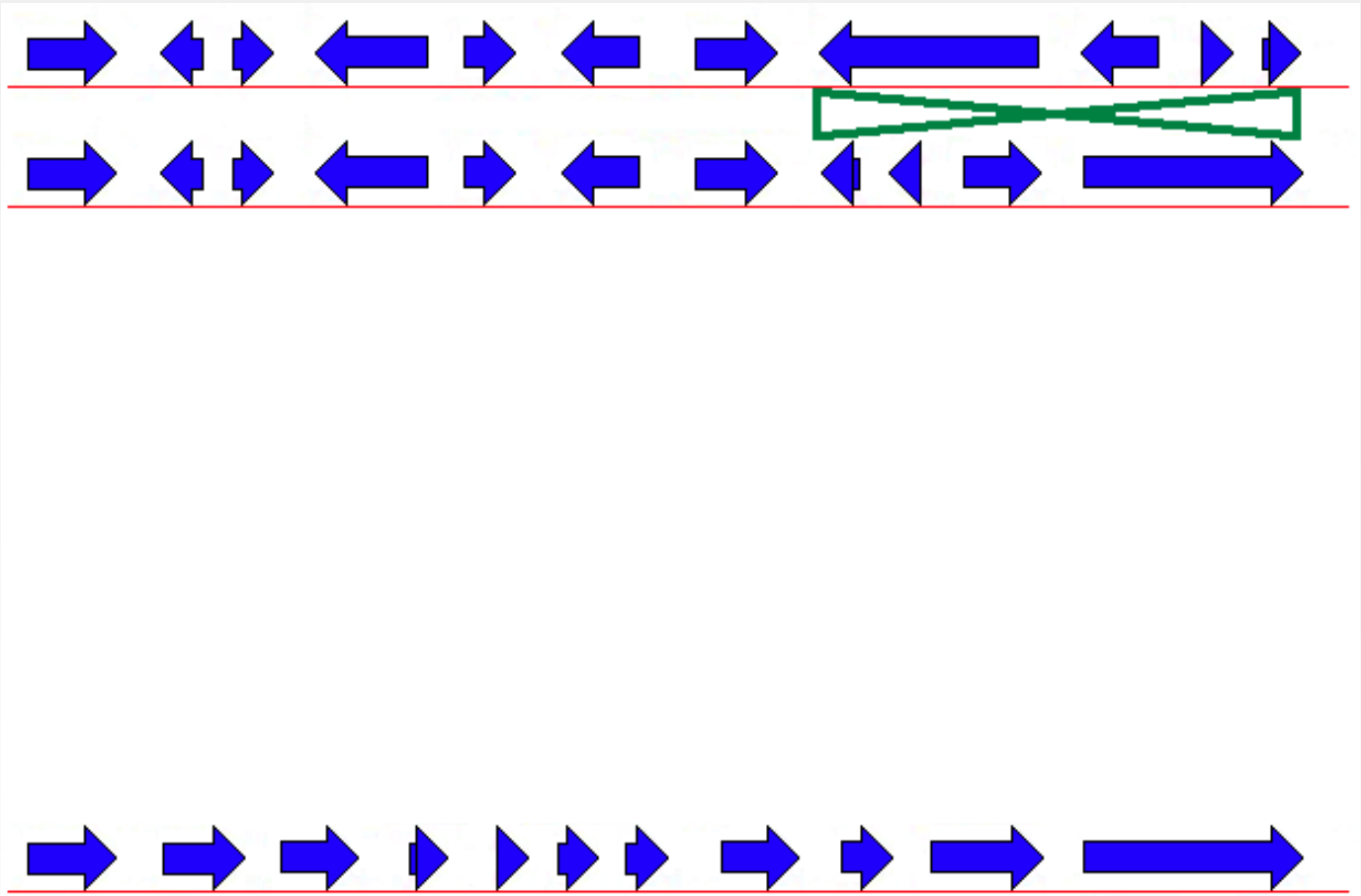


Fugu/*Human* chromosome 12 [Wang et al., 2006]

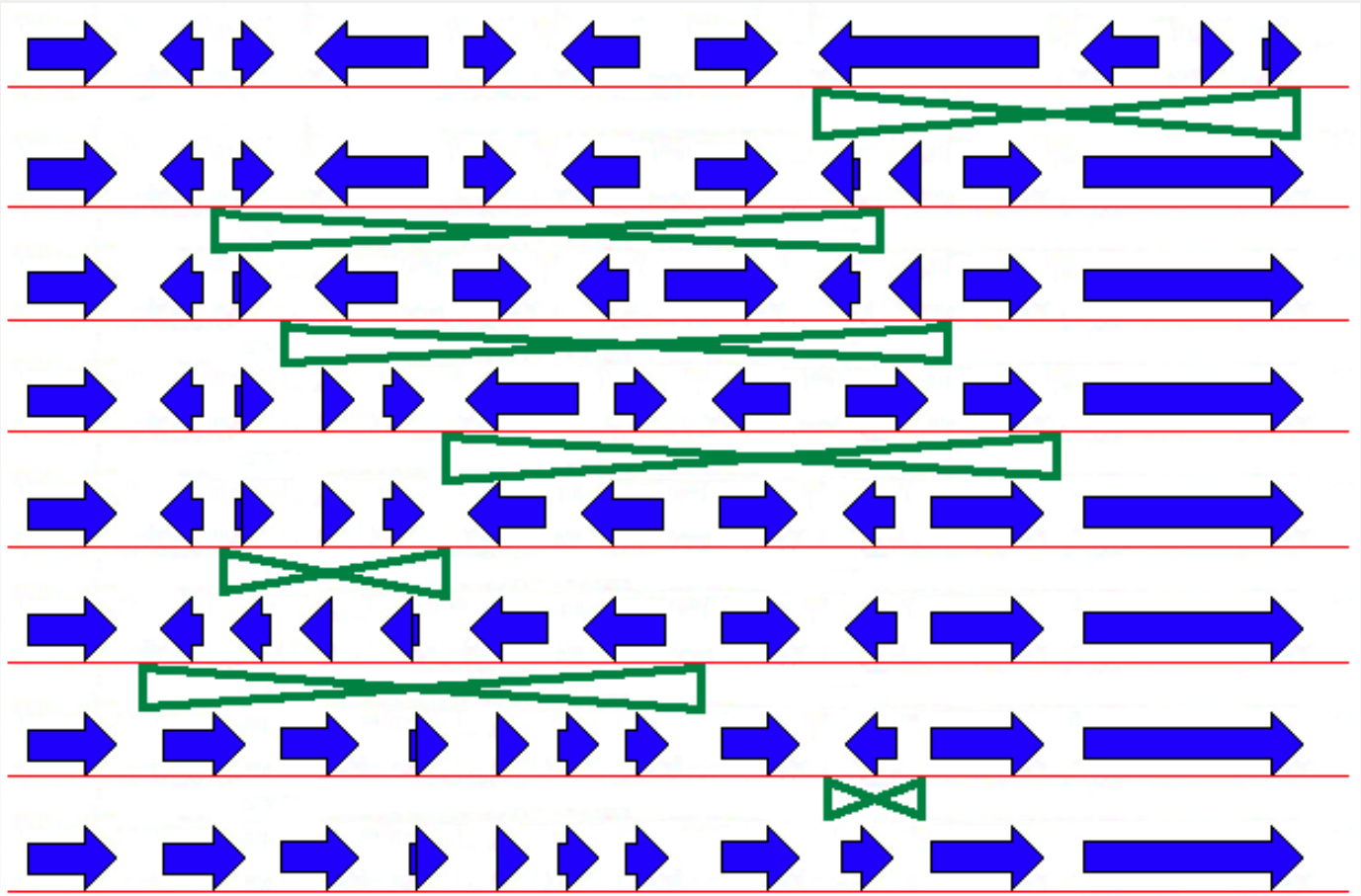
Computation of a scenario based on the principle of parsimony



Computation of a scenario based on the principle of parsimony



Computation of a scenario based on the principle of parsimony

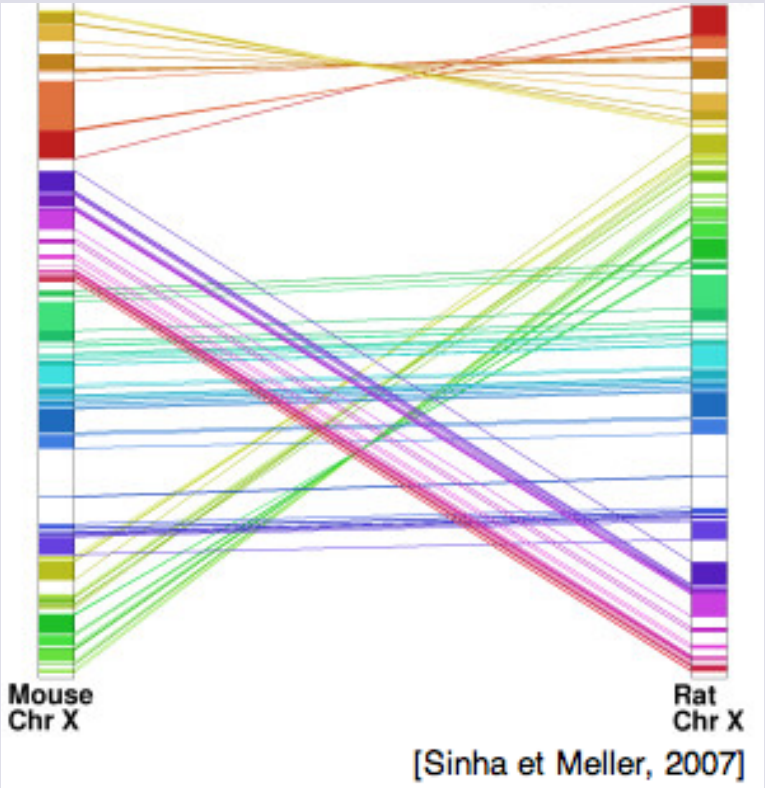


Computation of a measure of (dis)similarity

Compute the (dis)similarities between two genomes.

Synteny

Group of genes conserved between two species.



Comparative genomic

Comparative genomic allows to better understand the evolution of species.

- Scenario or computation of measure of (dis)similarity,
- Study of measures of (dis)similarity: number of common intervals, number of adjacencies, number of breakpoints, MAD, SAD...

Compare two genomes

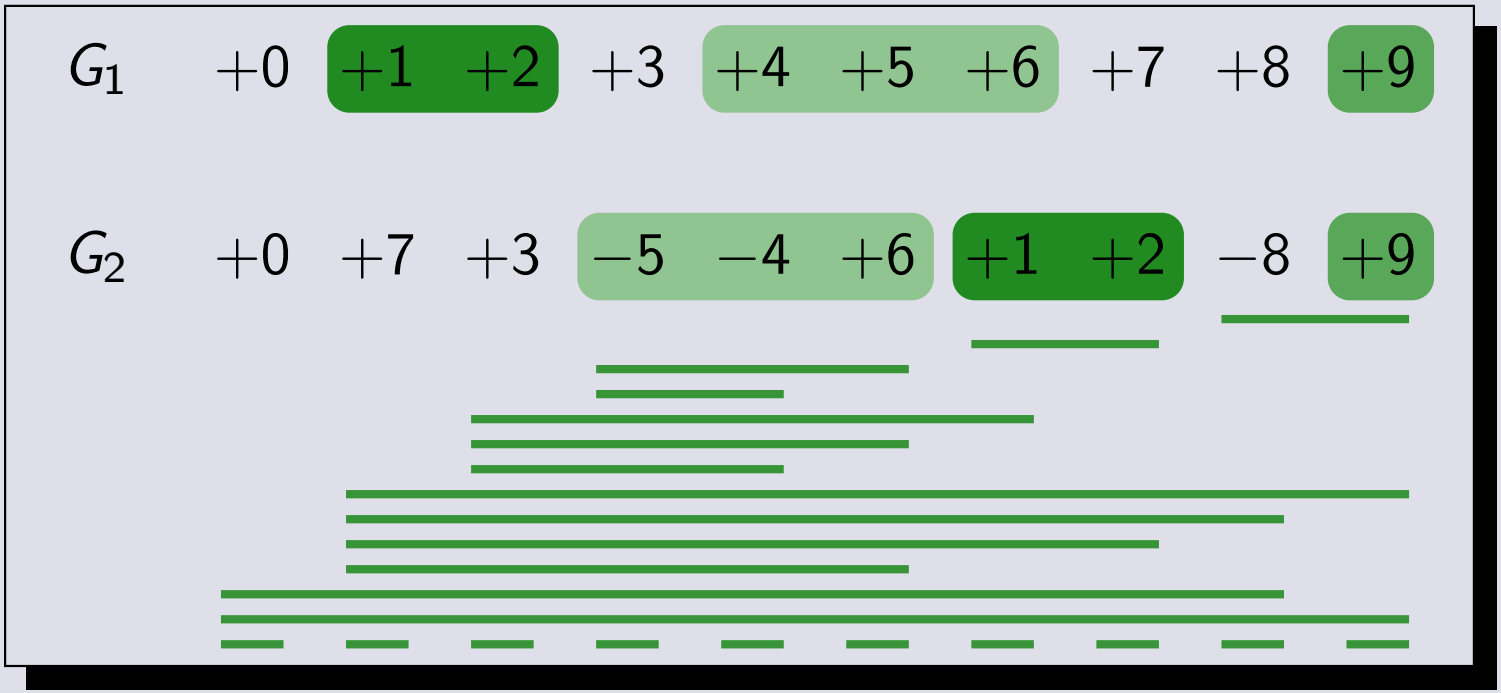
Two genomes without duplication

G_1	$+0$	$+1$	$+2$	$+3$	$+4$	$+5$	$+6$	$+7$	$+8$	$+9$
G_2	$+0$	$+7$	$+3$	-5	-4	$+6$	$+1$	$+2$	-8	$+9$

The genome is a signed permutation.

The number of common intervals [Uno et Yagiura, 2000].

A measure of similarity



$\Rightarrow$ 23 common intervalles between G_1 and G_2.

Number of adjacencies [Angibaud et al., 2008]

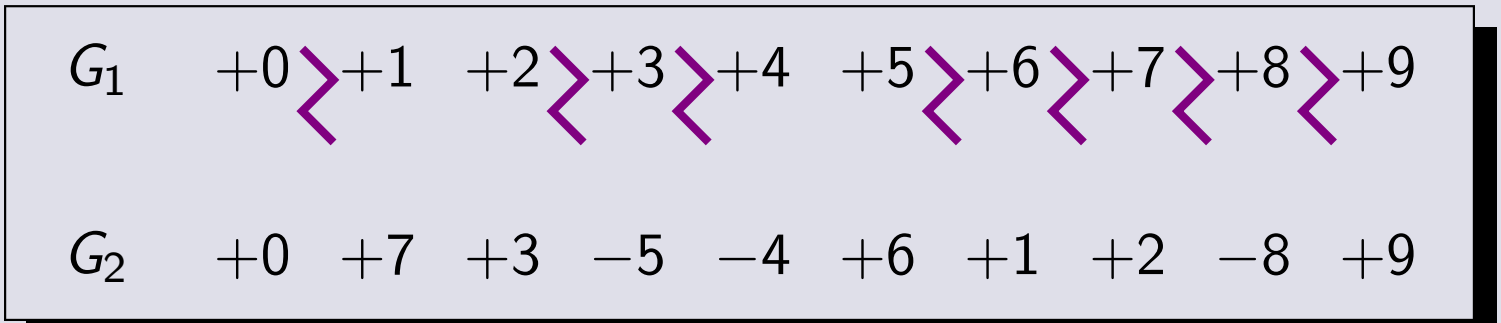
A second measure of similarity

G_1	$+0$	$+1$	$+2$	$+3$	$+4$	$+5$	$+6$	$+7$	$+8$	$+9$
G_2	$+0$	$+7$	$+3$	-5	-4	$+6$	$+1$	$+2$	-8	$+9$

$\Rightarrow$ 2 adjacencies between G_1 and G_2.

Number of breakpoints [Sankoff et Blanchette, 1997]

A measure of dissimilarity



Dual measure to the measure of number of adjacencies:

$$\texttt{bkp}(G_1, G_2) + \texttt{adj}(G_1, G_2) = n - 1$$

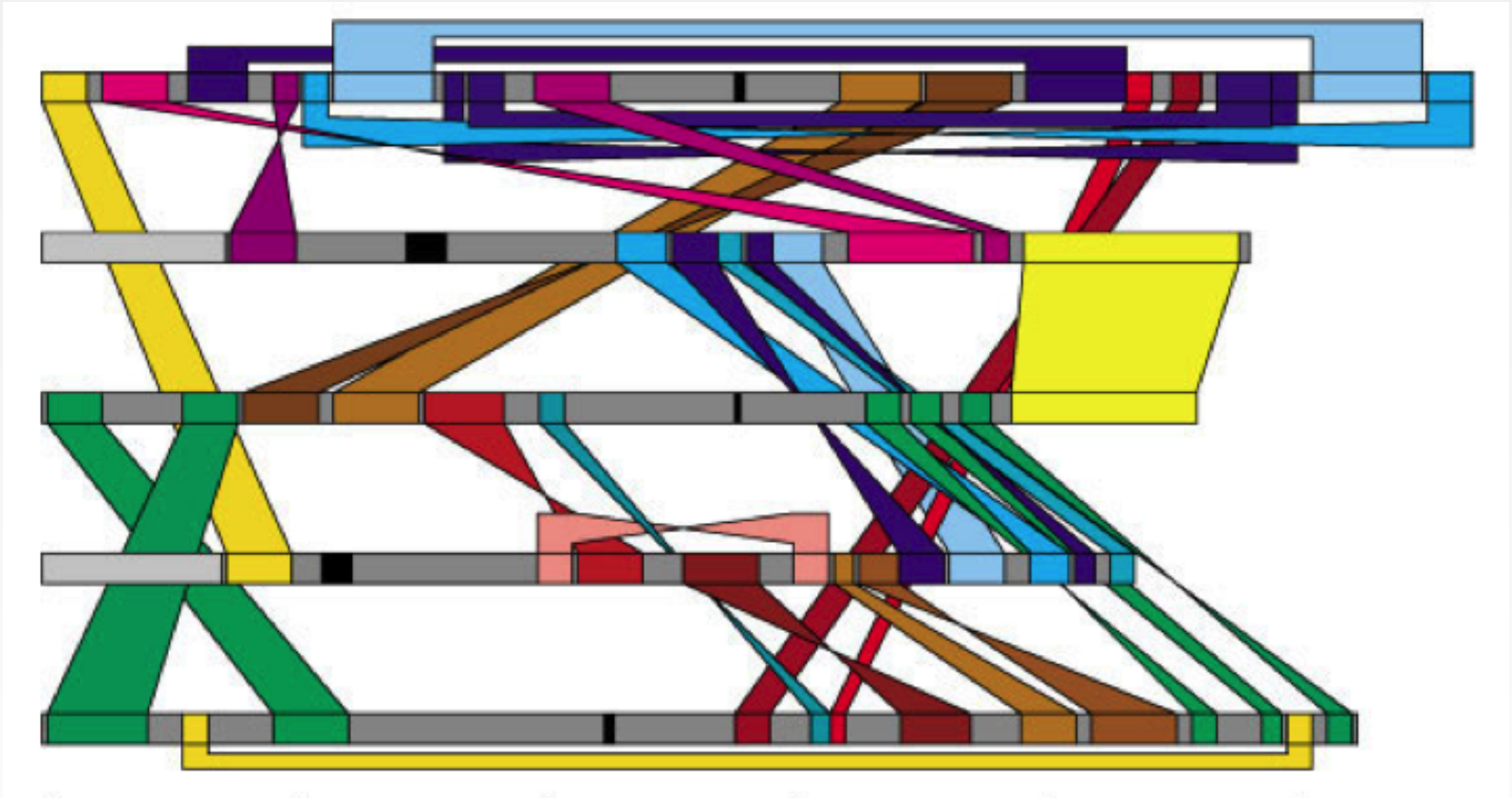
$\Rightarrow$ 7 breakpoints between G_1 and G_2.

Outline

1. Comparison between two genomes with **duplications**
2. Comparison between two **partially ordered** genomes

Comparison between two genomes with **duplications**

Duplications of genes on 5 chromosomes of the *Arabidopsis thaliana* 's genome

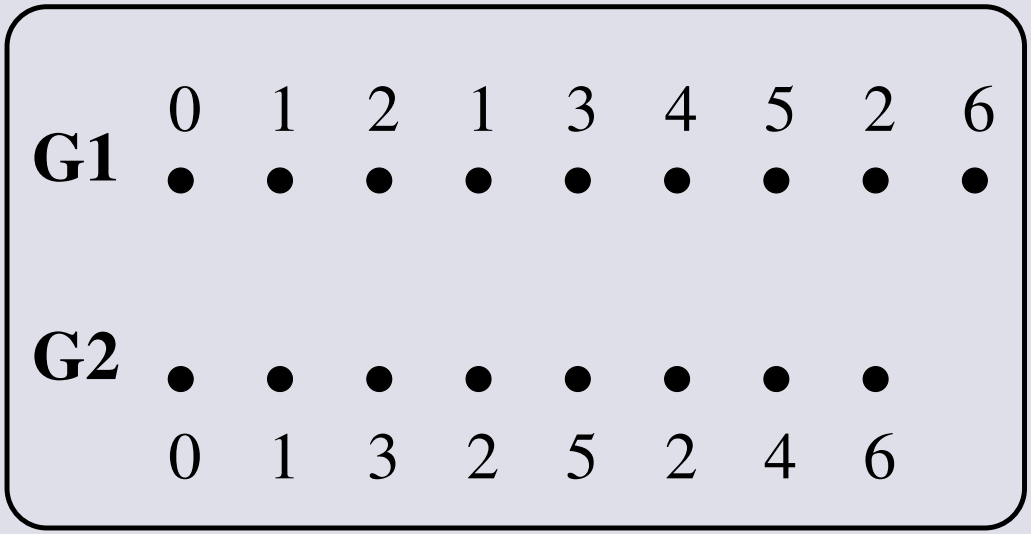


[*Arabidopsis* genome initiative, 2000]

Comparison of two genomes

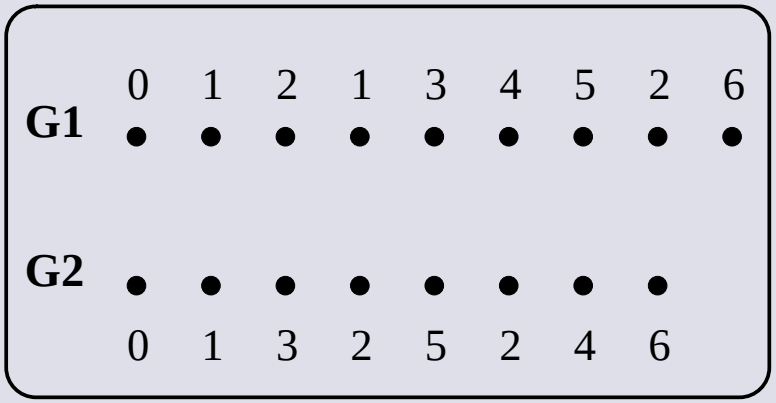
Two genomes with duplications

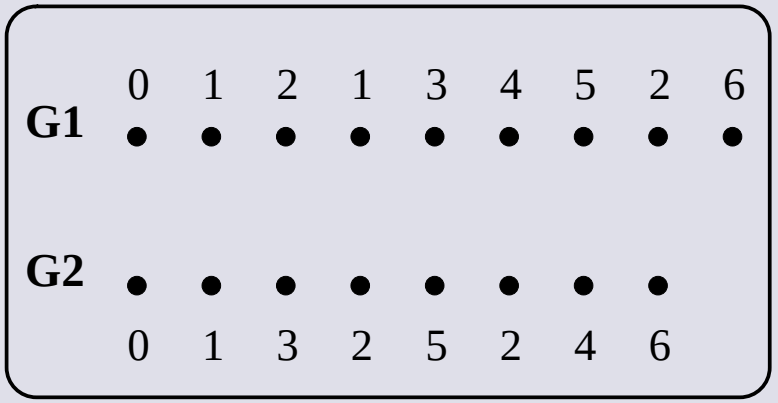
We need to find a matching between both genomes.



Matching model: exemplar

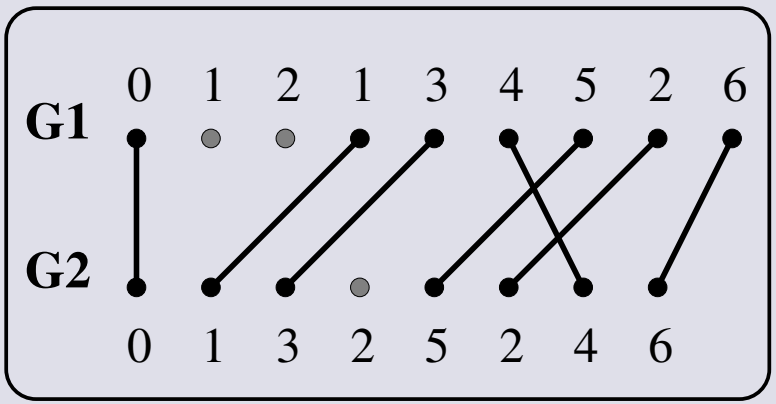
The matching is required to sature exactly one gene of each gene family [Sankoff, 1999].

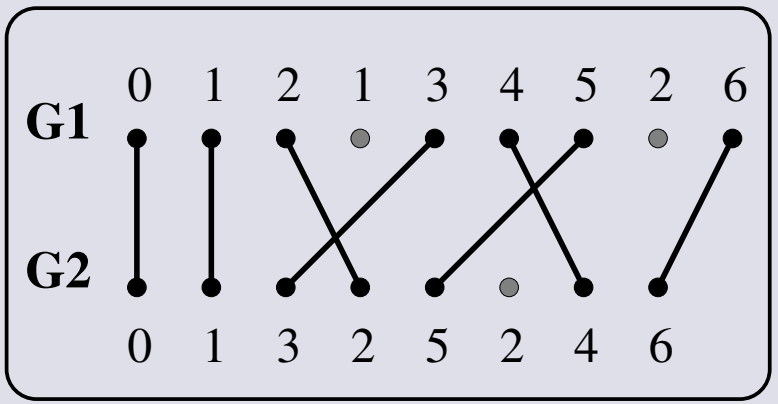




Matching model: exemplar

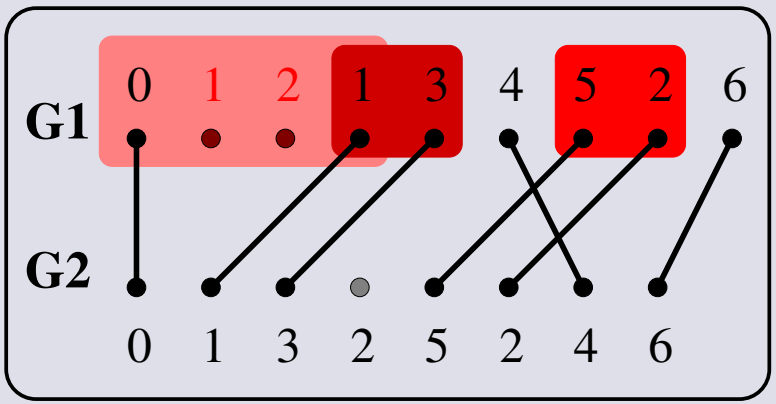
The matching is required to sature exactly one gene of each gene family [Sankoff, 1999].



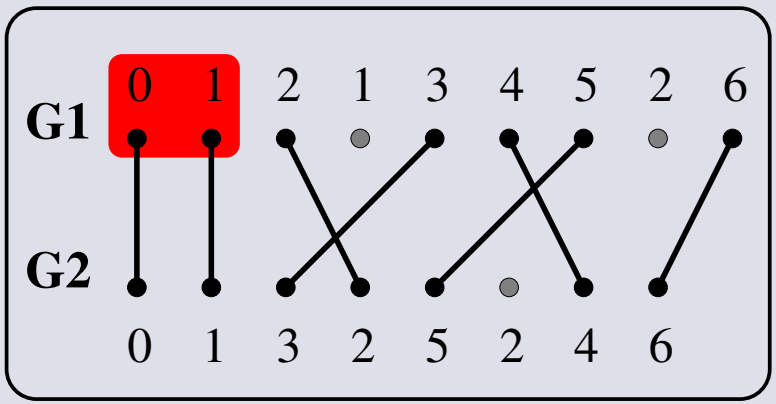


Matching model: exemplar

The matching is required to sature exactly one gene of each gene family [Sankoff, 1999].



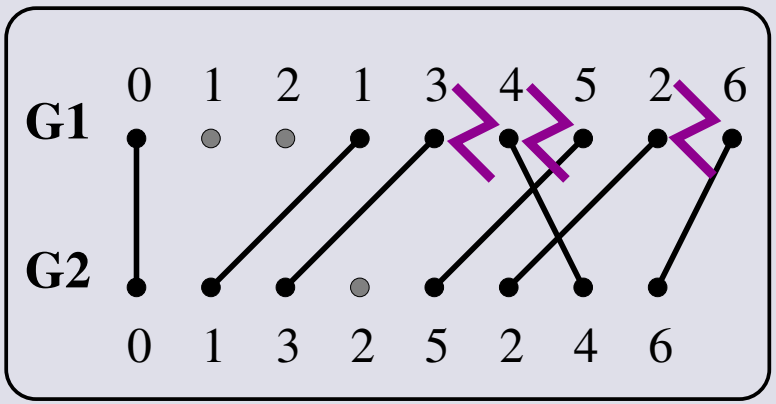
3 adjacencies



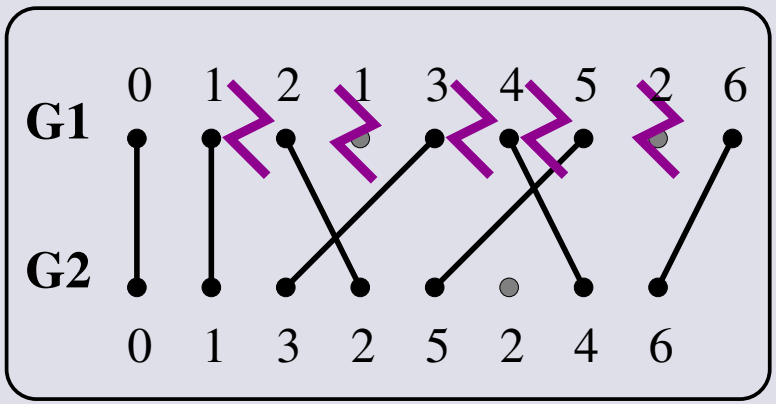
1 adjacence

Matching model: exemplar

The matching is required to sature exactly one gene of each gene family [Sankoff, 1999].



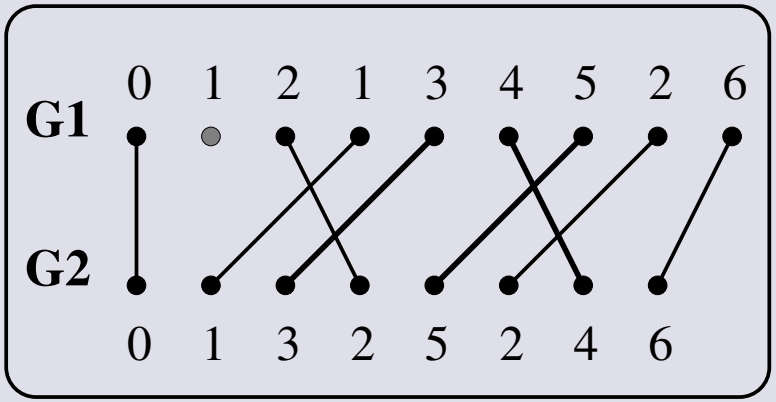
3 breakpoints

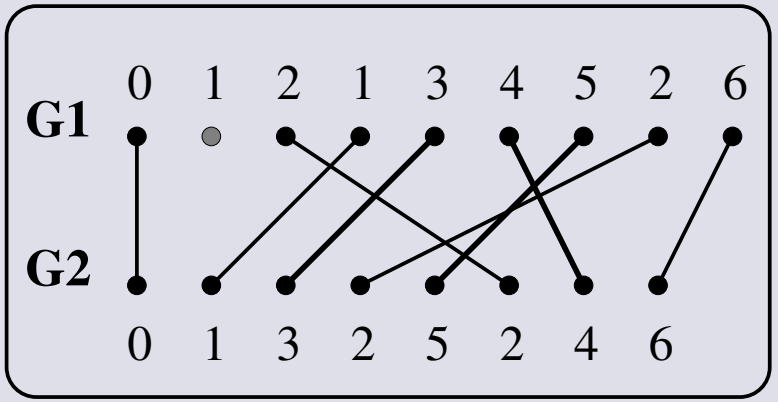


5 breakpoints

Matching model: maximum

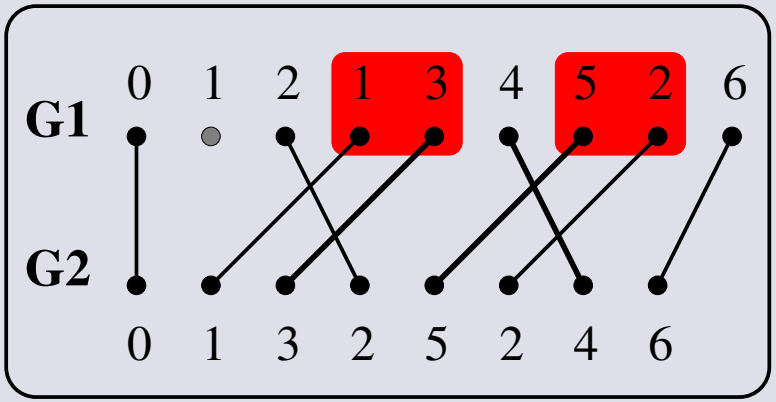
The matching is required to saturate as many genes as possible of each gene family [Tang and Moret, 2003].



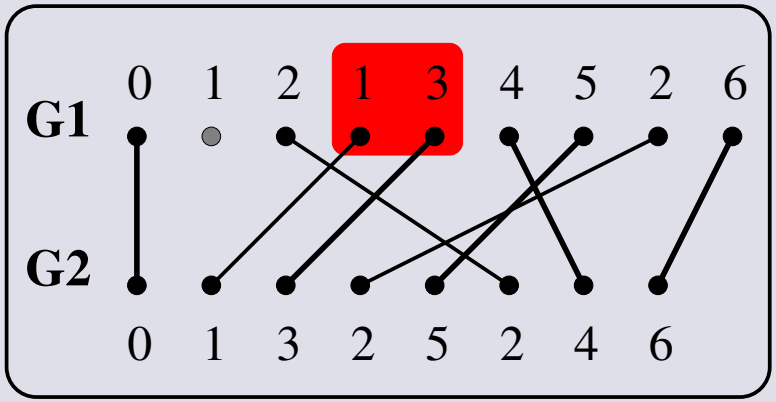


Matching model: maximum

The matching is required to saturate as many genes as possible of each gene family [Tang and Moret, 2003].



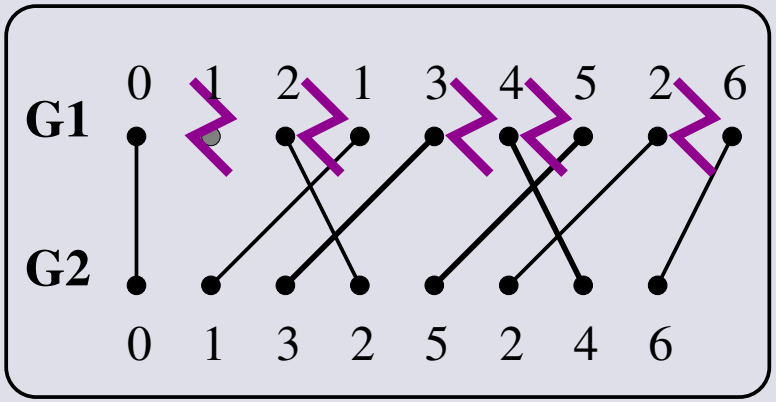
2 adjacencies



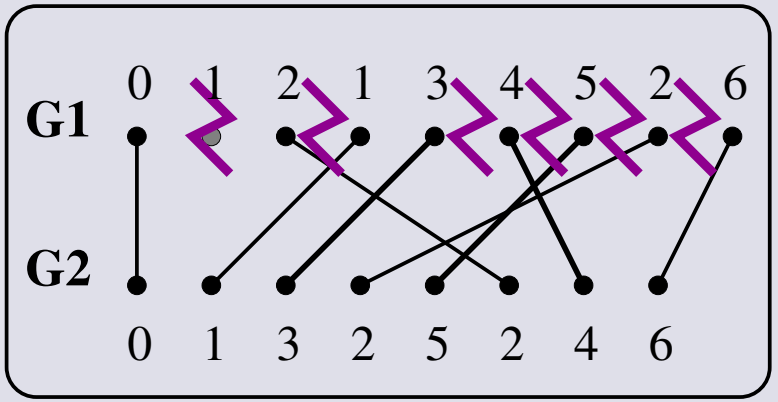
1 adjacence

Matching model: maximum

The matching is required to saturate as many genes as possible of each gene family [Tang and Moret, 2003].



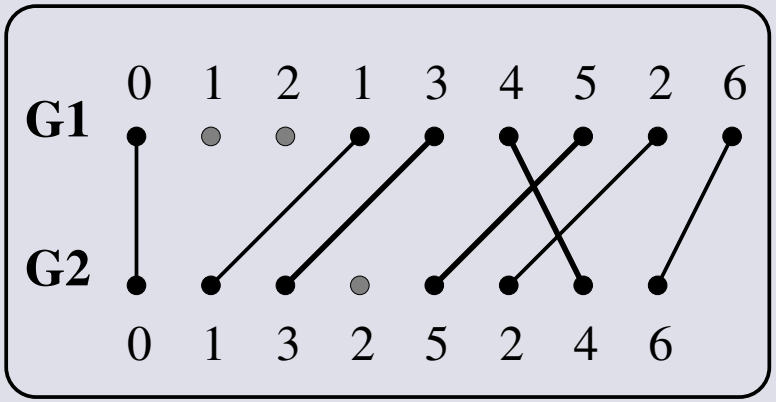
5 breakpoints

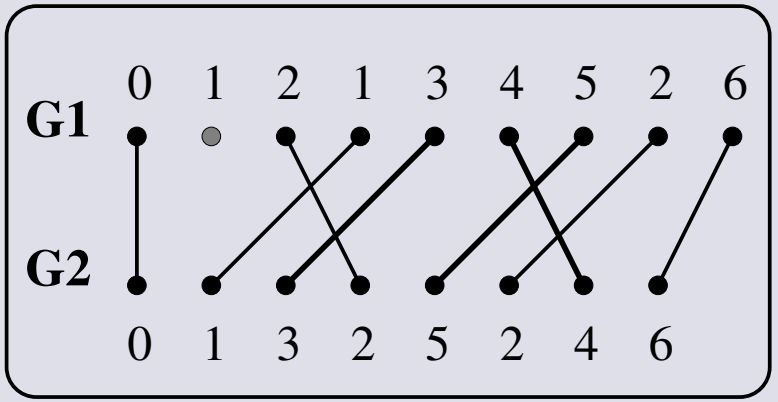


6 breakpoints

New matching model: intermediate

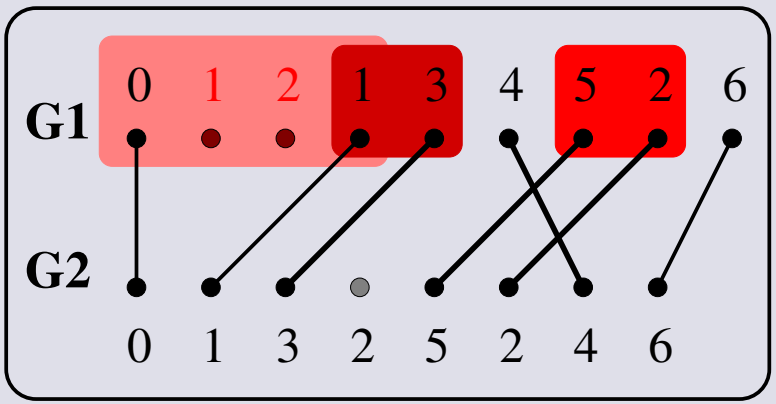
The matching is required to saturate at least one gene of each gene family.



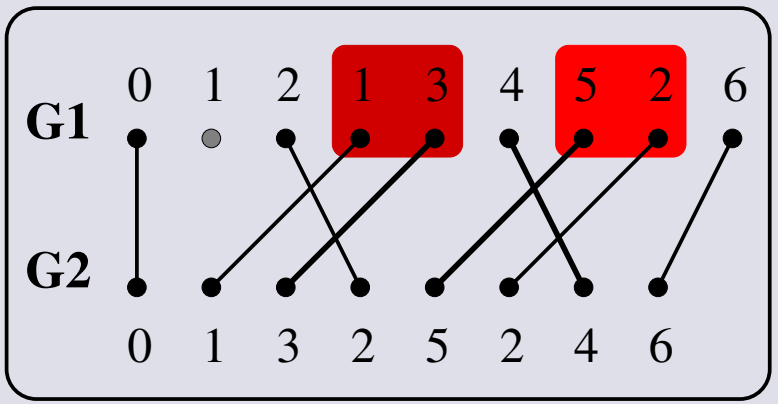


New matching model: intermediate

The matching is required to saturate at least one gene of each gene family.



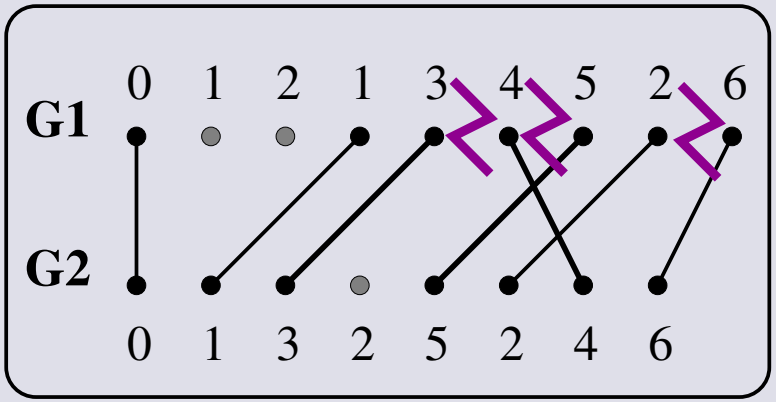
3 adjacencies



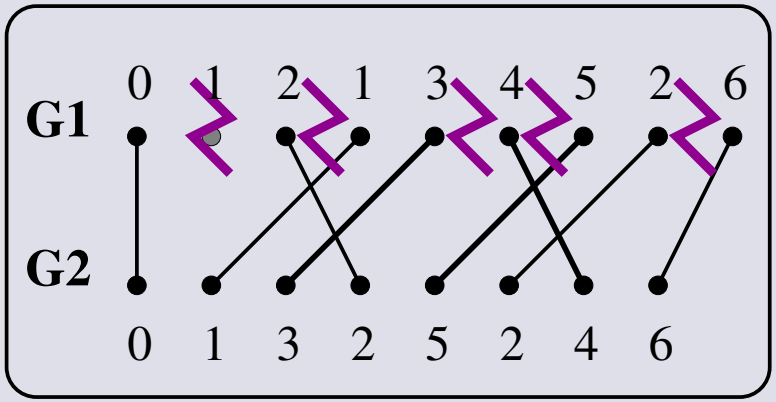
2 adjacencies

New matching model: intermediate

The matching is required to saturate at least one gene of each gene family.



3 breakpoints



5 breakpoints

Strategy

Framework study

Studied models: exemplar, intermediate, maximum.

Studied measures: numbers of adjacencies and breakpoints.

The corresponding problems are **NP**–hard [Bryant, 2000].

$\Rightarrow$ Result of **no-approximation**.

$\Rightarrow$ **Heuristics**.

$\Rightarrow$ **Exact** algorithms.

Comparison between two genomes with **duplication**

Exact algorithms

A pseudo-boolean program

Objective $\max(z);\ z = x_1 + 2x_2 - x_3$

Constraints

$$\begin{aligned} x_1 - 2x_2 + 3x_3 &\geq 1 \\ x_1 + x_2 + x_3 &= 1 \\ 2x_1 + x_2 + x_3 &< 3 \end{aligned}$$

Boolean variables $x_i \in \{0, 1\} \qquad \forall i = 1, 2, 3.$

Variables

We have 4 types of variables:

- $a(i,k)$ denotes a **matching** between two genes;
- $b_x(i)$ denotes the **saturation** of a gene in genome G_x;
- $c_x(i,j)$ denotes two genes **consecutives** in genome G_x;
- $d(i,j,k,l)$ denotes an **adjacency**.

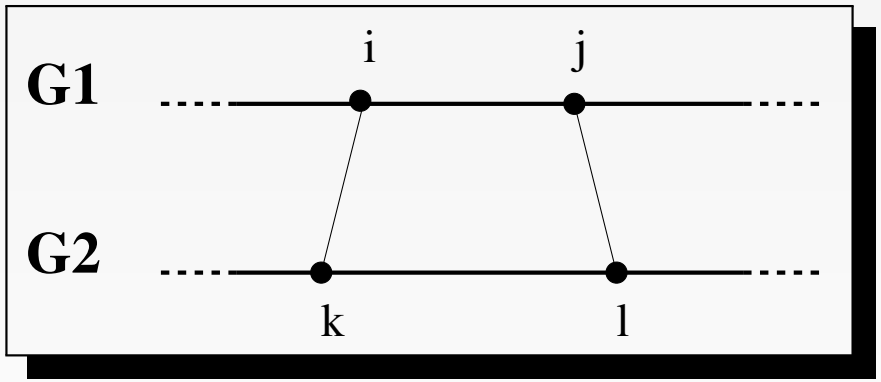
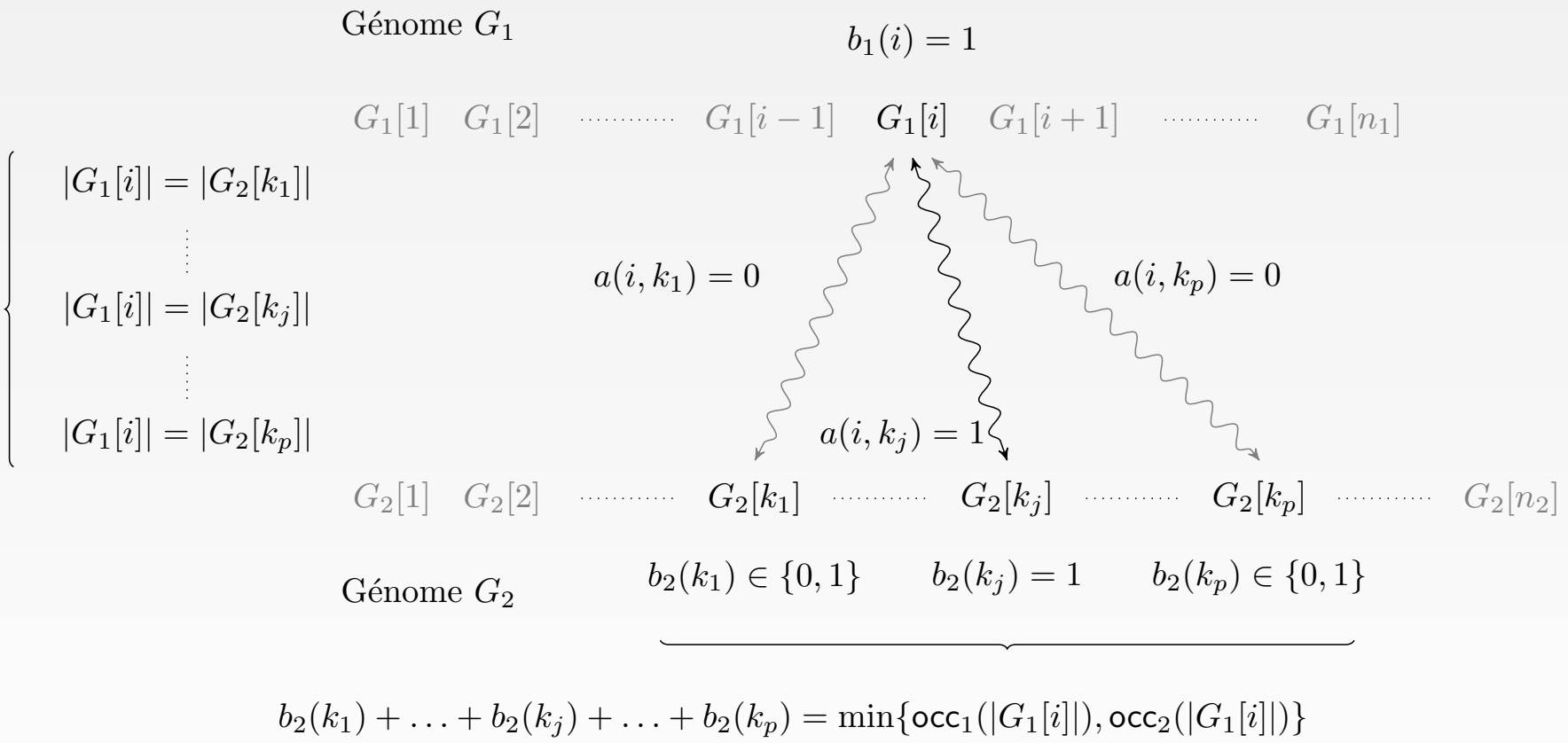


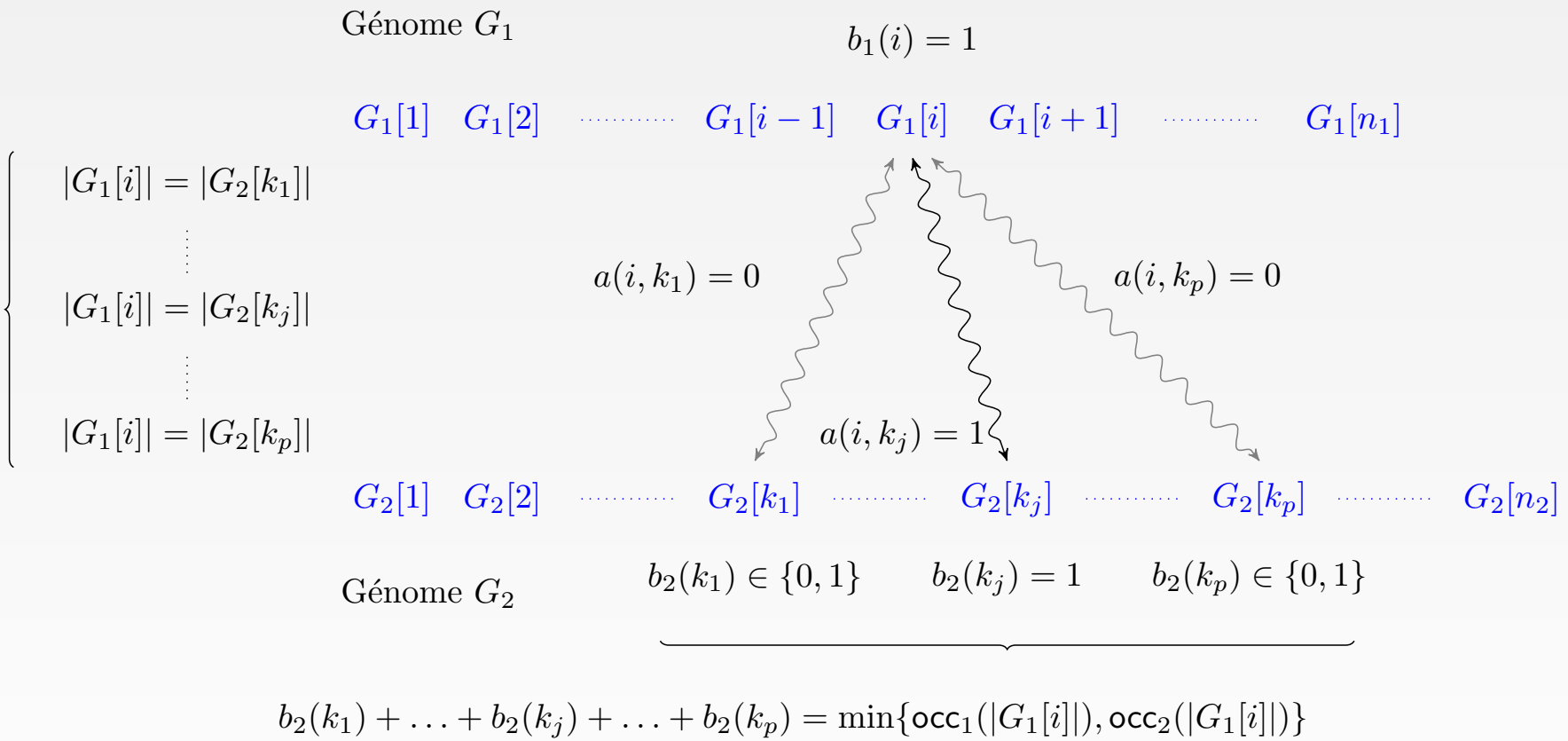
Illustration of the variables $a(i,k)$ and $b_x(i)$



$$b_2(k_1) + \ldots + b_2(k_j) + \ldots + b_2(k_p) = \min\{\mathsf{occ}_1(|G_1[i]|), \mathsf{occ}_2(|G_1[i]|)\}$$

Illustration of the variables $a(i,k)$ and $b_x(i)$

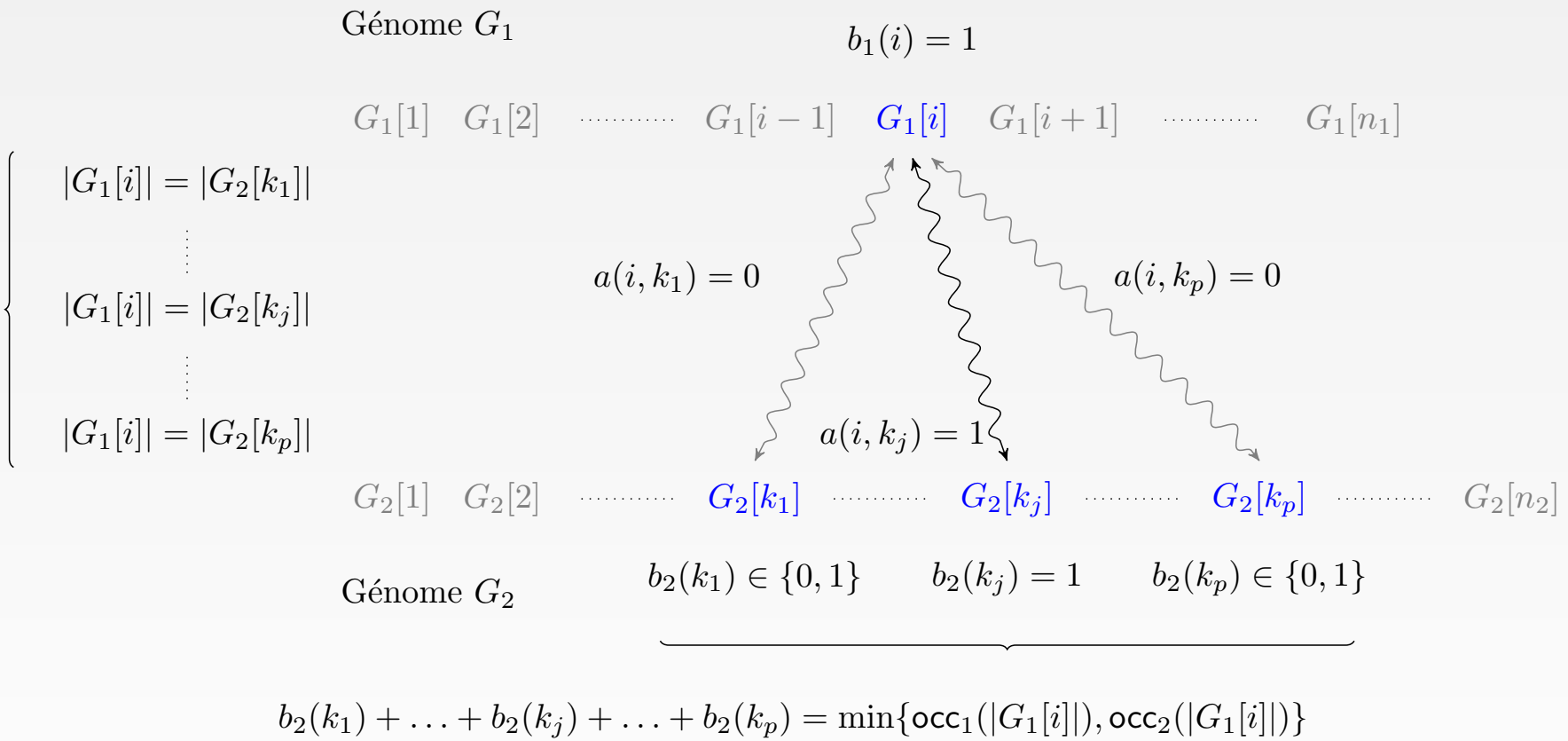
We have two genomes:



$$b_2(k_1) + \ldots + b_2(k_j) + \ldots + b_2(k_p) = \min\{\mathsf{occ}_1(|G_1[i]|), \mathsf{occ}_2(|G_1[i]|)\}$$

Illustration of the variables $a(i,k)$ and $b_x(i)$

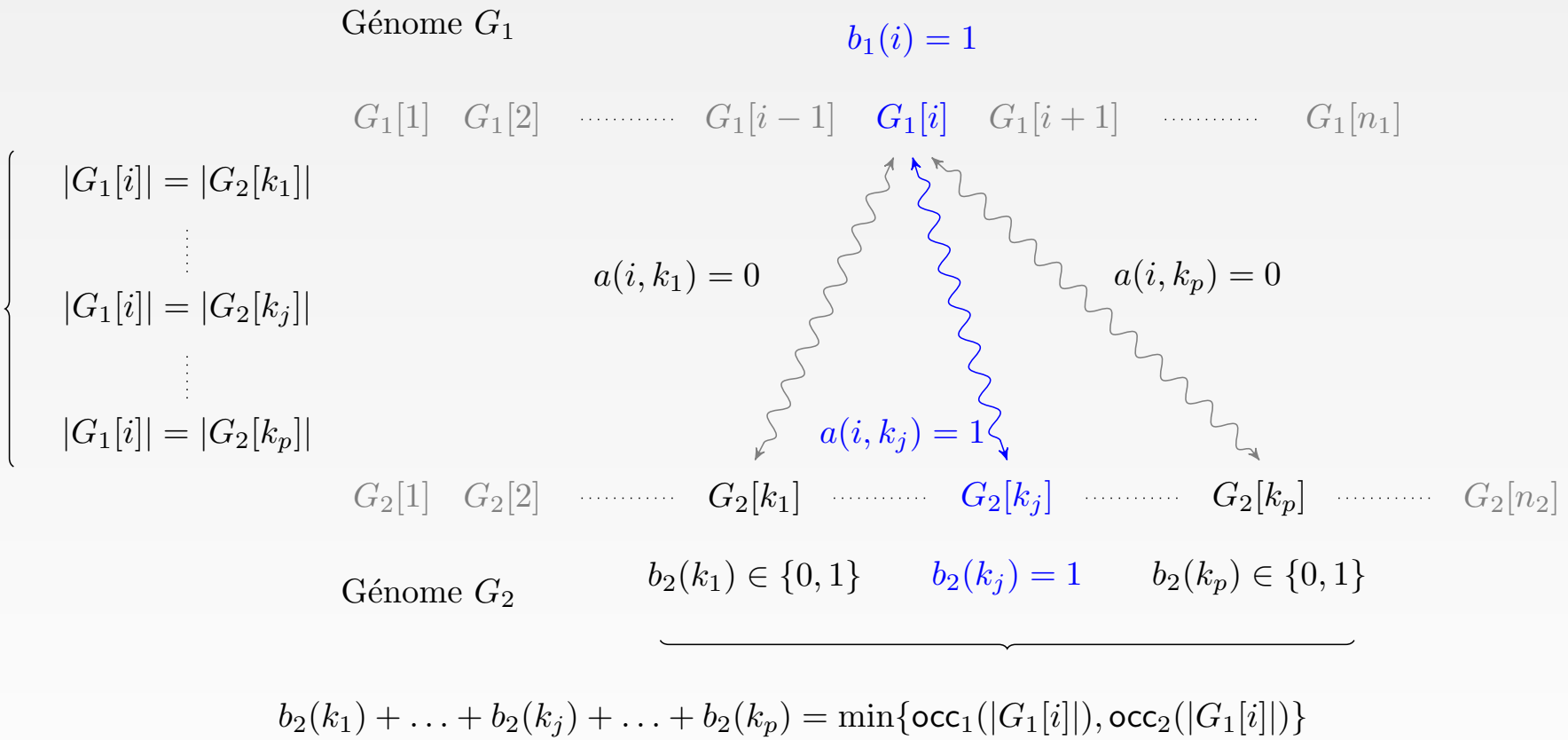
We have one family of duplicated genes:



$$b_2(k_1) + \ldots + b_2(k_j) + \ldots + b_2(k_p) = \min\{\mathsf{occ}_1(|G_1[i]|), \mathsf{occ}_2(|G_1[i]|)\}$$

Illustration of the variables $a(i,k)$ and $b_x(i)$

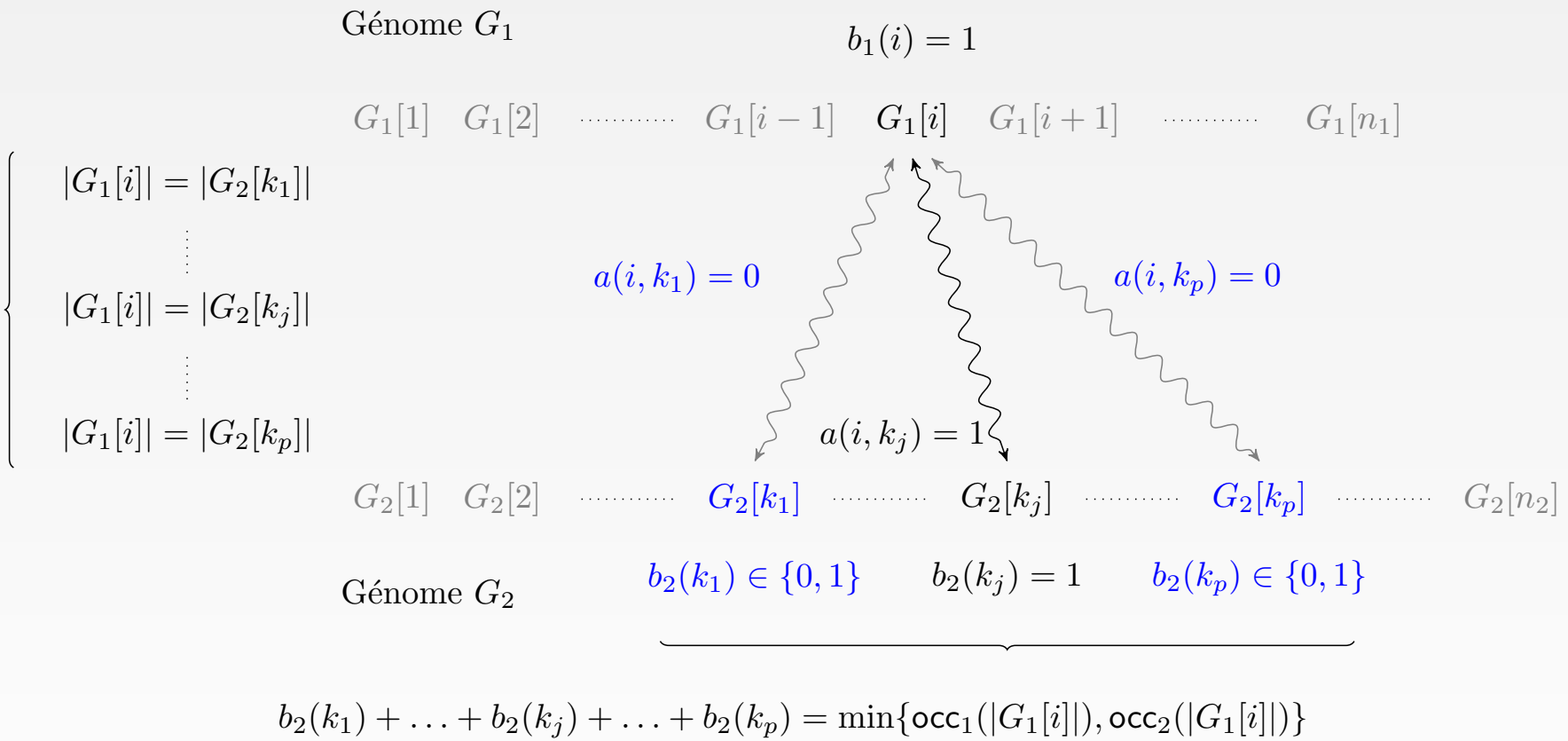
We have a matching between $G_1(i)$ and $G_2(k_j)$:



$$b_2(k_1) + \ldots + b_2(k_j) + \ldots + b_2(k_p) = \min\{\mathsf{occ}_1(|G_1[i]|), \mathsf{occ}_2(|G_1[i]|)\}$$

Illustration of the variables $a(i,k)$ and $b_x(i)$

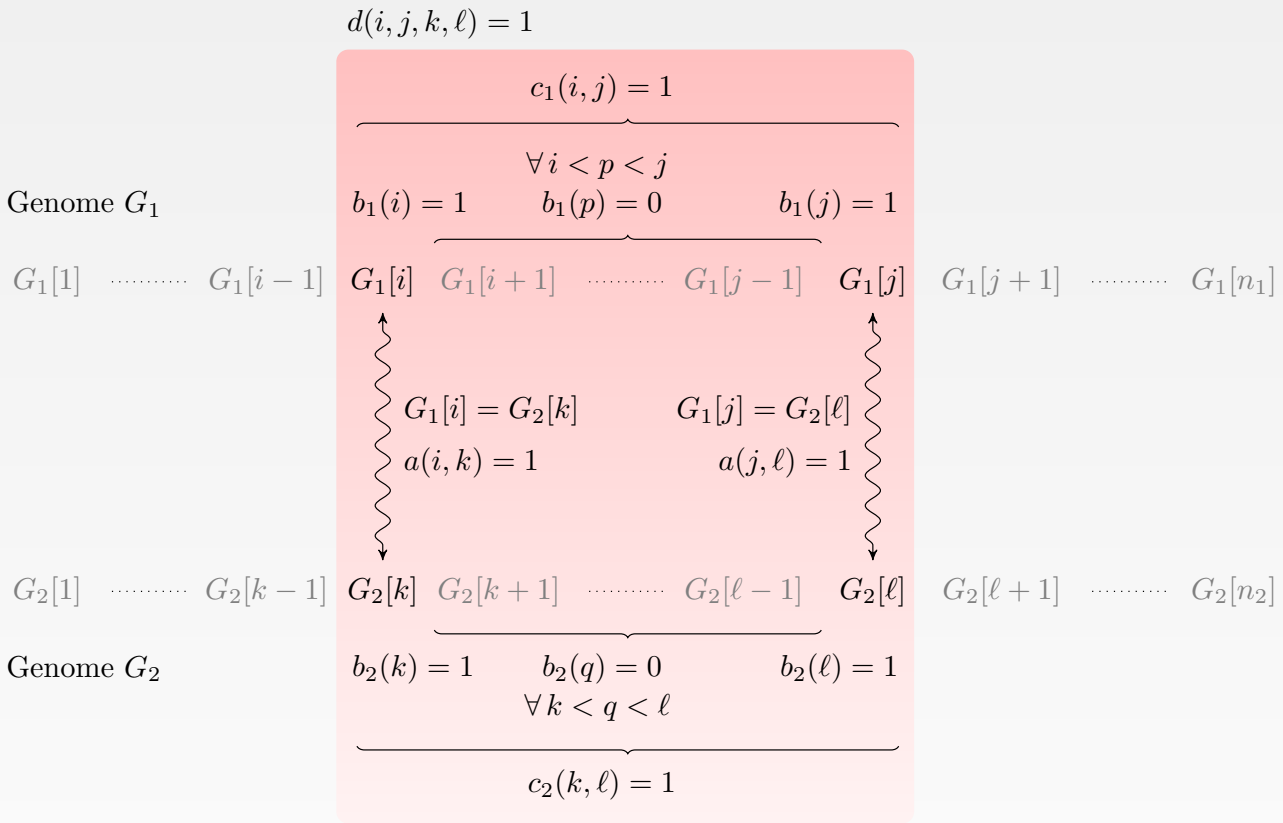
We have a matching between $G_1(i)$ and $G_2(k_j)$:



$$b_2(k_1) + \ldots + b_2(k_j) + \ldots + b_2(k_p) = \min\{\mathsf{occ}_1(|G_1[i]|), \mathsf{occ}_2(|G_1[i]|)\}$$

Illustration of the variables $c_x(i,j)$ and $d(i,j,k,l)$

We have an adjacency between $G_0[i]$ and $G_1[j]$:



Comparison between two genomes with **duplication**

Experimentation

Experimentation

Dataset [Lerat *et al.* 2003]

- 12 γ-proteobacteria complete genomes,
- size: between 565 and 5474 genes,
- 7.6% of duplicated genes.

Solver

- For this work, the solver used is `CPLEX` `http://www.ilog.com/products/cplex`.

Results for the comparison between 12 genomes

Quadri Intel(R) Xeon(TM) CPU 3.00 GHz with 16GB of memory.

Results under the maximum model

- All results: 66 pairs of genomes (**100%**);
- Total time: $\simeq$ **3 minutes**.

Results under the exemplar model

- 65 out of 66 (**98%**) - memory problem
- Total time: $\simeq$ **3 minutes**.

Results under the intermediate model

- Maximization of number of adjacencies:
  - 63 out of 66 (**95%**); Total time: $\simeq$ **16 minutes**.
- Minimization of number of breakpoints:
  - 59 out of 66 (**89%**); Total time: $\simeq$ **1 hour**.

Comparison between the exemplar and maximum models

The choice of model depends on the measure considered

- Between two genomes, under the maximum model there are 8% of adjacencies more than under the exemplar model.
- Between two genomes, under the exemplar model there are 11% of breakpoints less than under the maximum model.

Gain from intermediate model to exemplar and maximum models

During the miminization of the number of breakpoints

Ratio	Intermediate	
	Adjacencies	Breakpoints
Exemplar	2%	0%
Maximum	-5%	10%

During the maximization of the number of adjacencies

Ratio	Intermediate	
	Adjacencies	Breakpoints
Exemplar	10%	-3%
Maximum	1%	8%

Gain from intermediate model to exemplar and maximum models

During the miminization of the number of breakpoints

Ratio	Intermediate	
	Adjacencies	Breakpoints
Exemplar	2%	0%
Maximum	-5%	10%

During the maximization of the number of adjacencies

Ratio	Intermediate	
	Adjacencies	Breakpoints
Exemplar	10%	-3%
Maximum	1%	8%

Gain from intermediate model to exemplar and maximum models

During the miminization of the number of breakpoints

Ratio	Intermediate	
	Adjacencies	Breakpoints
Exemplar	2%	0%
Maximum	-5%	10%

During the maximization of the number of adjacencies

Ratio	Intermediate	
	Adjacencies	Breakpoints
Exemplar	10%	-3%
Maximum	1%	8%

Comparison between two genomes with **duplications**

Heuristics

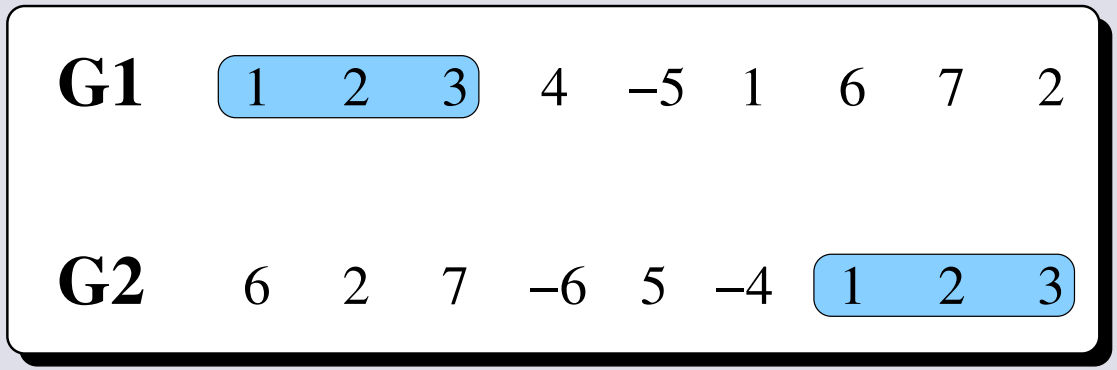
LCS: Longest Common Substring [Marron *et al.*, 2004]

G1	1	2	3	4	−5	1	6	7	2
G2	6	2	7	−6	5	−4	1	2	3

IILCS heuristic under the exemplar model

1. Compute S: the Longest Common Substring up to a reversal
2. Map all the genes of S accordingly
3. Remove genes that cannot be matched any longer according to the model
4. Iterate the process until saturation

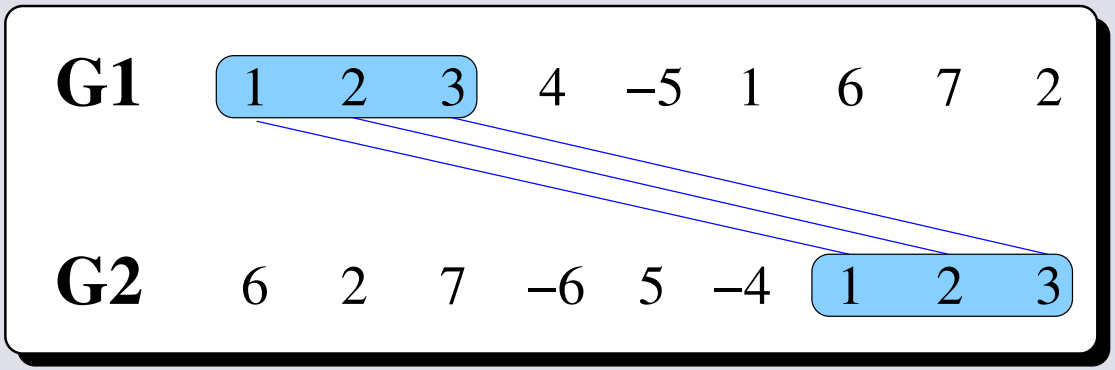
LCS: Longest Common Substring [Marron *et al.*, 2004]



IILCS heuristic under the exemplar model

1. Compute S: the Longest Common Substring up to a reversal
2. Map all the genes of S accordingly
3. Remove genes that cannot be matched any longer according to the model
4. Iterate the process until saturation

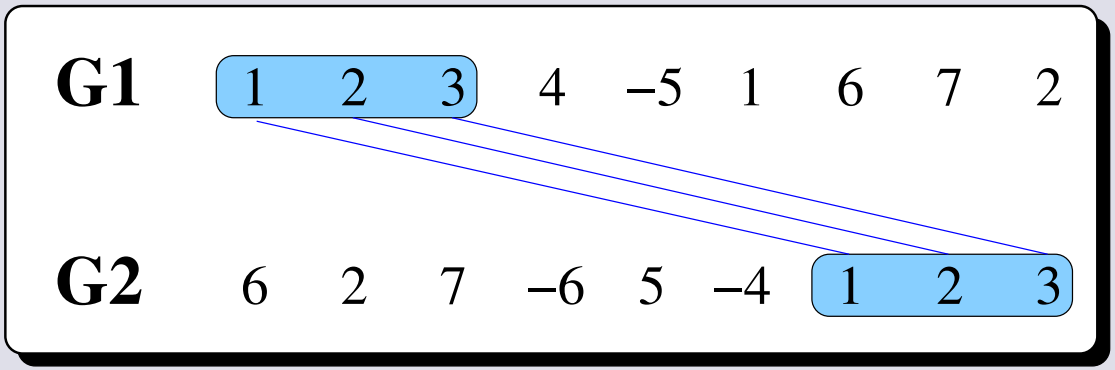
LCS: Longest Common Substring [Marron *et al.*, 2004]



IILCS heuristic under the exemplar model

1. Compute S: the Longest Common Substring up to a reversal
2. Map all the genes of S accordingly
3. Remove genes that cannot be matched any longer according to the model
4. Iterate the process until saturation

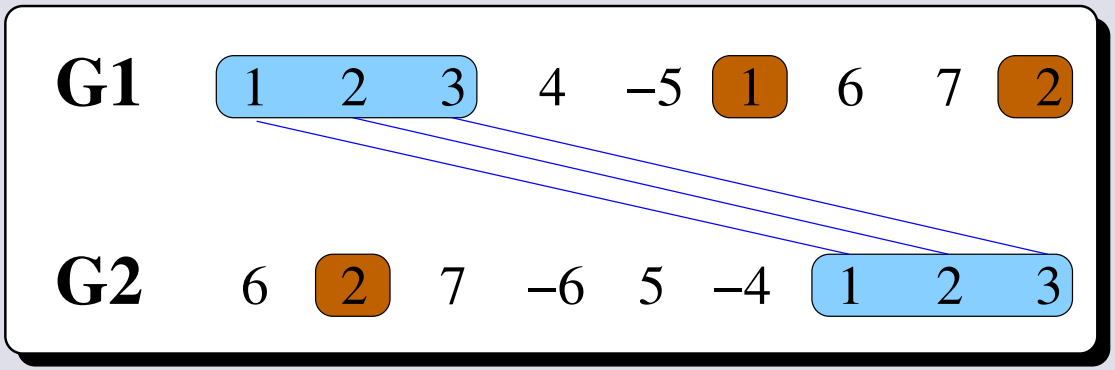
LCS: Longest Common Substring [Marron *et al.*, 2004]



IILCS heuristic under the exemplar model

1. Compute S: the Longest Common Substring up to a reversal
2. Map all the genes of S accordingly
3. Remove genes that cannot be matched any longer according to the model
4. Iterate the process until saturation

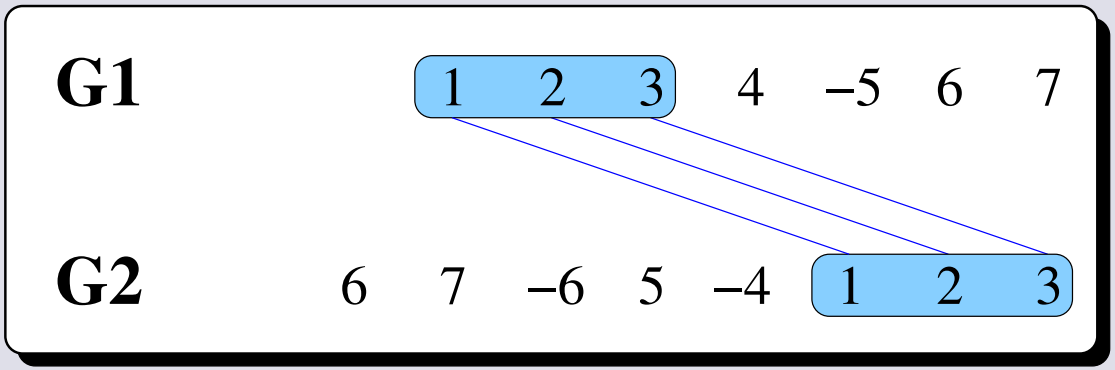
LCS: Longest Common Substring [Marron *et al.*, 2004]



IILCS heuristic under the exemplar model

1. Compute S: the Longest Common Substring up to a reversal
2. Map all the genes of S accordingly
3. Remove genes that cannot be matched any longer according to the model
4. Iterate the process until saturation

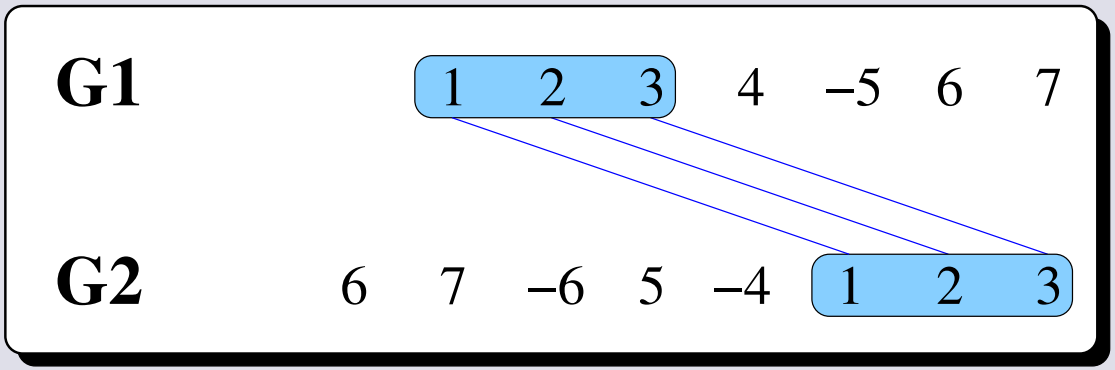
LCS: Longest Common Substring [Marron *et al.*, 2004]



IILCS heuristic under the exemplar model

1. Compute S: the Longest Common Substring up to a reversal
2. Map all the genes of S accordingly
3. Remove genes that cannot be matched any longer according to the model
4. Iterate the process until saturation

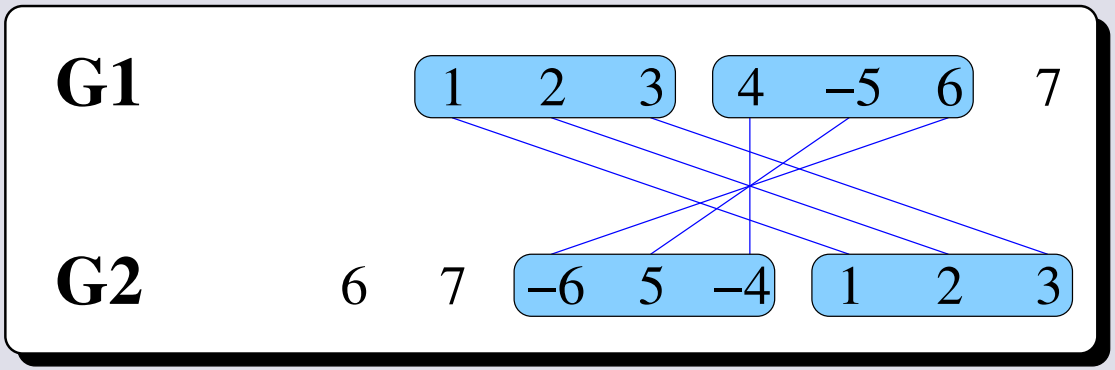
LCS: Longest Common Substring [Marron *et al.*, 2004]



IILCS heuristic under the exemplar model

1. Compute S: the Longest Common Substring up to a reversal
2. Map all the genes of S accordingly
3. Remove genes that cannot be matched any longer according to the model
4. Iterate the process until saturation

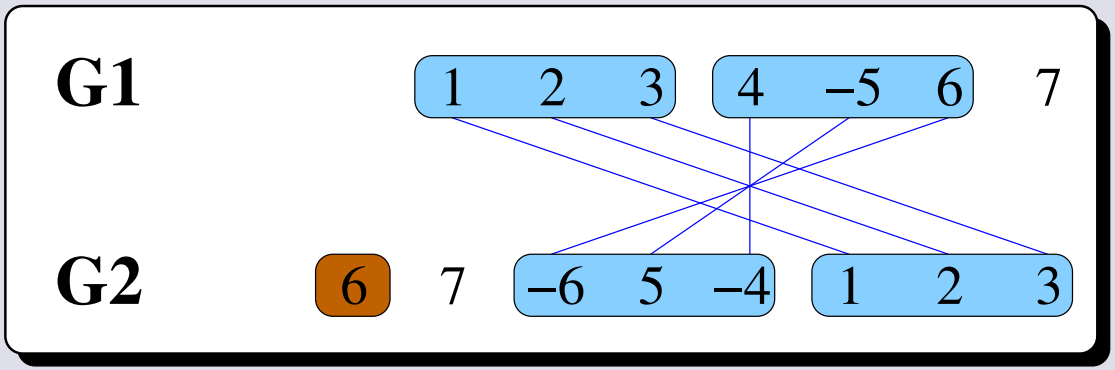
LCS: Longest Common Substring [Marron *et al.*, 2004]



IILCS heuristic under the exemplar model

1. Compute S: the Longest Common Substring up to a reversal
2. Map all the genes of S accordingly
3. Remove genes that cannot be matched any longer according to the model
4. Iterate the process until saturation

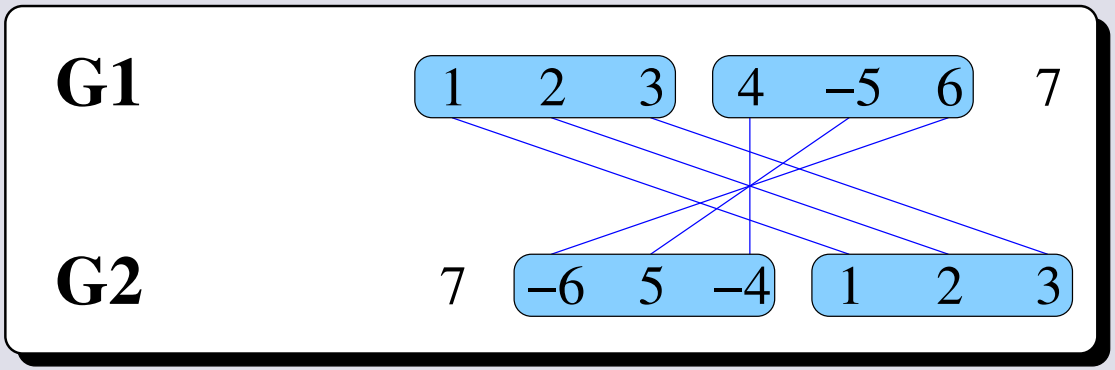
LCS: Longest Common Substring [Marron *et al.*, 2004]



IILCS heuristic under the exemplar model

1. Compute S: the Longest Common Substring up to a reversal
2. Map all the genes of S accordingly
3. Remove genes that cannot be matched any longer according to the model
4. Iterate the process until saturation

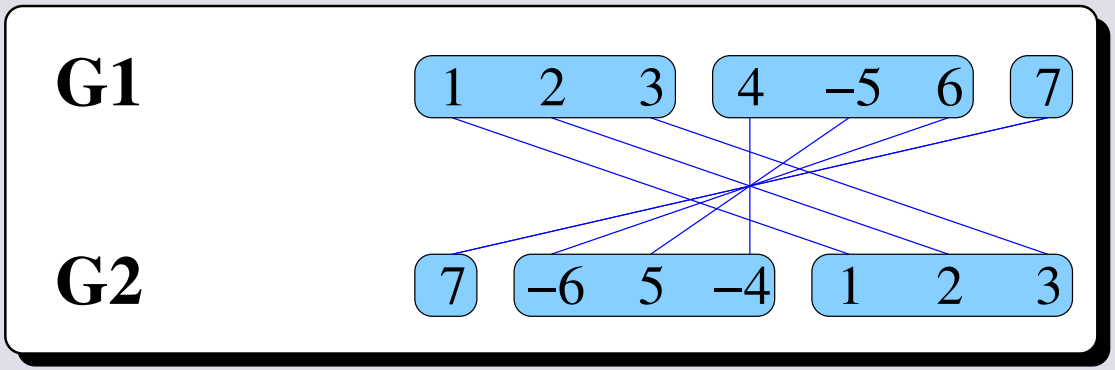
LCS: Longest Common Substring [Marron *et al.*, 2004]



IILCS heuristic under the exemplar model

1. Compute S: the Longest Common Substring up to a reversal
2. Map all the genes of S accordingly
3. Remove genes that cannot be matched any longer according to the model
4. Iterate the process until saturation

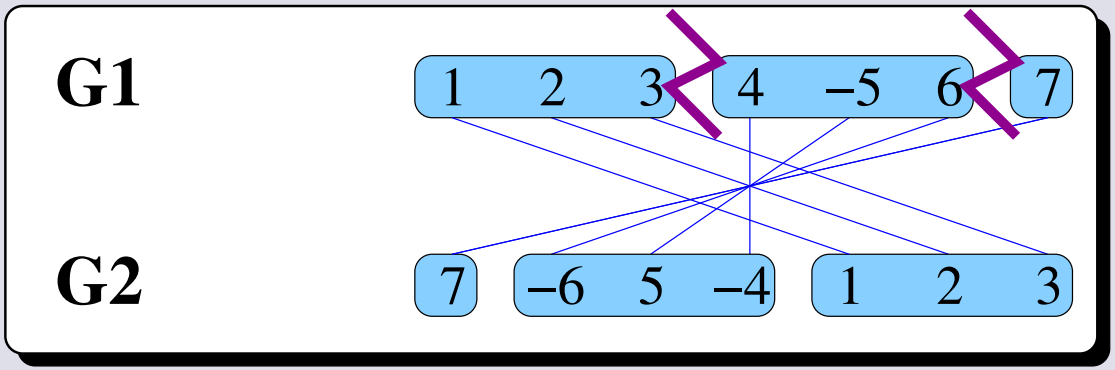
LCS: Longest Common Substring [Marron *et al.*, 2004]



IILCS heuristic under the exemplar model

1. Compute S: the Longest Common Substring up to a reversal
2. Map all the genes of S accordingly
3. Remove genes that cannot be matched any longer according to the model
4. Iterate the process until saturation

LCS: Longest Common Substring [Marron *et al.*, 2004]



IILCS heuristic under the exemplar model

1. Compute S: the Longest Common Substring up to a reversal
2. Map all the genes of S accordingly
3. Remove genes that cannot be matched any longer according to the model
4. Iterate the process until saturation

Hybrid: IILCS then exact algorithm

- Partial matching by iteration of `IILCS`;
- Stopping criterion: size of a LCS less than a parameter k;
- Then, total matching using our exact algorithm.

Results of heuristics

Maximum (66 cases)	Heuristic	IILCS_M	HYB_M(2)	HYB_M(3)
	Average	99.05%	99.83%	99.94%
	Worst case	97.43%	99.38%	99.47%
	Best case	100%	100%	100%
	Exact result	16,67%	45,45%	75,76%

Exemplar (65 cases)	Heuristic	IILCS_E	HYB_E(2)	HYB_E(3)
	Average	99.36%	99.97%	99.99%
	Worst case	97.89%	99.73%	99.73%
	Best case	100%	100%	100%
	Exact result	20%	83,08%	95,38%

Intermediate (63 cases)	Heuristic	IILCS_IA	HYB_IA(2)	HYB_IA(3)
	Average	90.56%	99.43%	99.82%
	Worst case	82.09%	98.20%	98.78%
	Best case	98.52%	100%	100%
	Exact result	0%	28,57%	55,56%

Results of heuristics

Maximum (66 cases)	Heuristic	IILCS_M	HYB_M(2)	HYB_M(3)
	Average	99.05%	99.83%	99.94%
	Worst case	97.43%	99.38%	99.47%
	Best case	100%	100%	100%
	Exact result	16,67%	45,45%	75,76%

Exemplar (65 cases)	Heuristic	IILCS_E	HYB_E(2)	HYB_E(3)
	Average	99.36%	99.97%	99.99%
	Worst case	97.89%	99.73%	99.73%
	Best case	100%	100%	100%
	Exact result	20%	83,08%	95,38%

Intermediate (63 cases)	Heuristic	IILCS_IA	HYB_IA(2)	HYB_IA(3)
	Average	90.56%	99.43%	99.82%
	Worst case	82.09%	98.20%	98.78%
	Best case	98.52%	100%	100%
	Exact result	0%	28,57%	55,56%

Results of heuristics

Maximum (66 cases)	Heuristic	IILCS_M	HYB_M(2)	HYB_M(3)
	Average	99.05%	99.83%	99.94%
	Worst case	97.43%	99.38%	99.47%
	Best case	100%	100%	100%
	Exact result	16,67%	45,45%	75,76%

Exemplar (65 cases)	Heuristic	IILCS_E	HYB_E(2)	HYB_E(3)
	Average	99.36%	99.97%	99.99%
	Worst case	97.89%	99.73%	99.73%
	Best case	100%	100%	100%
	Exact result	20%	83,08%	95,38%

Intermediate (63 cases)	Heuristic	IILCS_IA	HYB_IA(2)	HYB_IA(3)
	Average	90.56%	99.43%	99.82%
	Worst case	82.09%	98.20%	98.78%
	Best case	98.52%	100%	100%
	Exact result	0%	28,57%	55,56%

Results of heuristics

Maximum (66 cases)	Heuristic	IILCS_M	HYB_M(2)	HYB_M(3)
	Average	99.05%	99.83%	99.94%
	Worst case	97.43%	99.38%	99.47%
	Best case	100%	100%	100%
	Exact result	16,67%	45,45%	75,76%

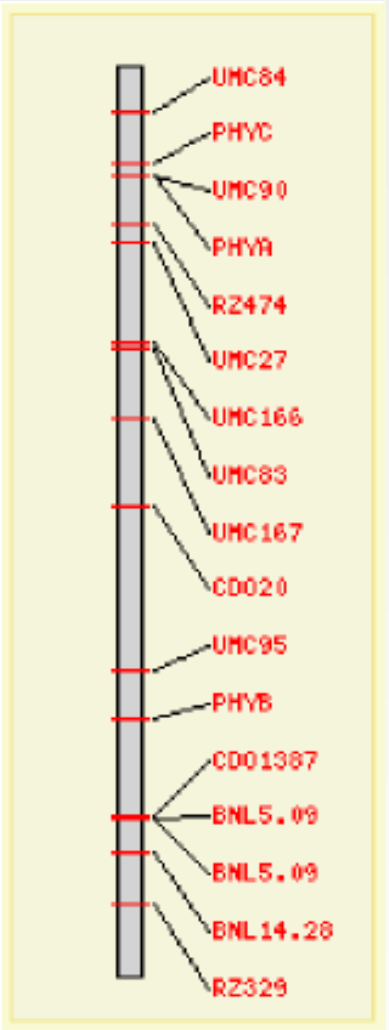
Exemplar (65 cases)	Heuristic	IILCS_E	HYB_E(2)	HYB_E(3)
	Average	99.36%	99.97%	99.99%
	Worst case	97.89%	99.73%	99.73%
	Best case	100%	100%	100%
	Exact result	20%	83,08%	95,38%

Intermediate (63 cases)	Heuristic	IILCS_IA	HYB_IA(2)	HYB_IA(3)
	Average	90.56%	99.43%	99.82%
	Worst case	82.09%	98.20%	98.78%
	Best case	98.52%	100%	100%
	Exact result	0%	28,57%	55,56%

Comparison between two **partially ordered** genomes

A partially ordered genome

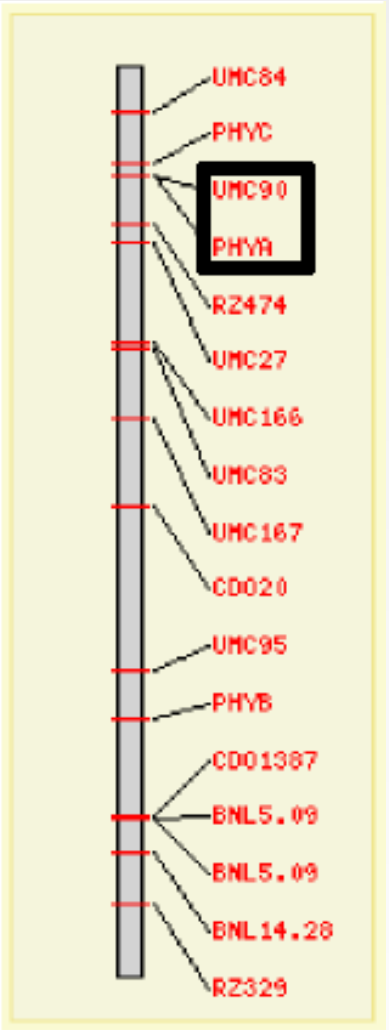
One chromosome of Sorghum:



[Klein, 2004]

A partially ordered genome

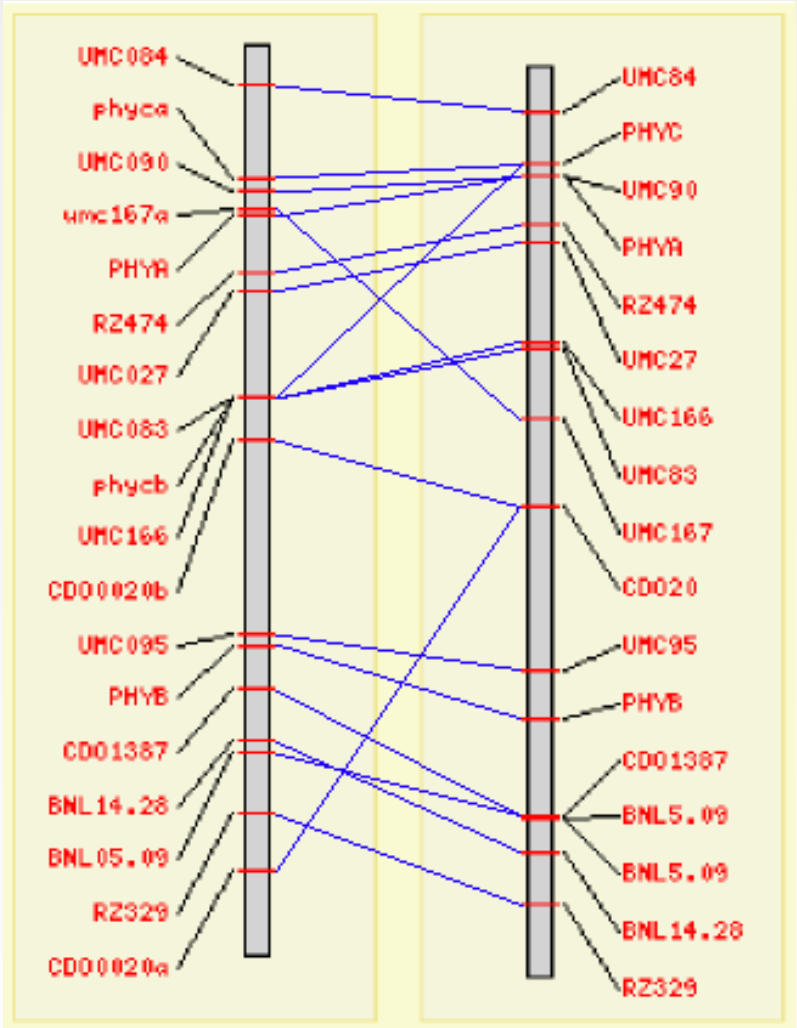
A part of partially ordered Sorgum Genome:



[Klein, 2004]

A partially ordered genome

Two studies of a same part of Sorghum genome:



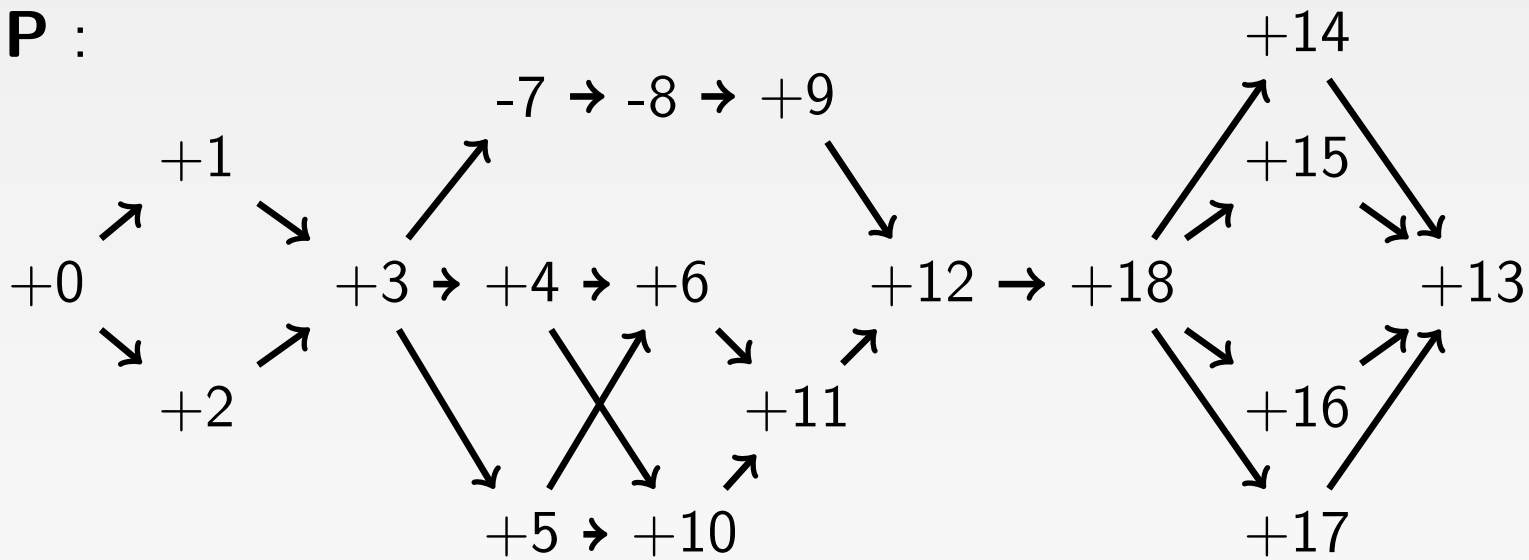
[Paterson, 2003]

[Klein, 2004]

Comparison between two **partially ordered** genomes

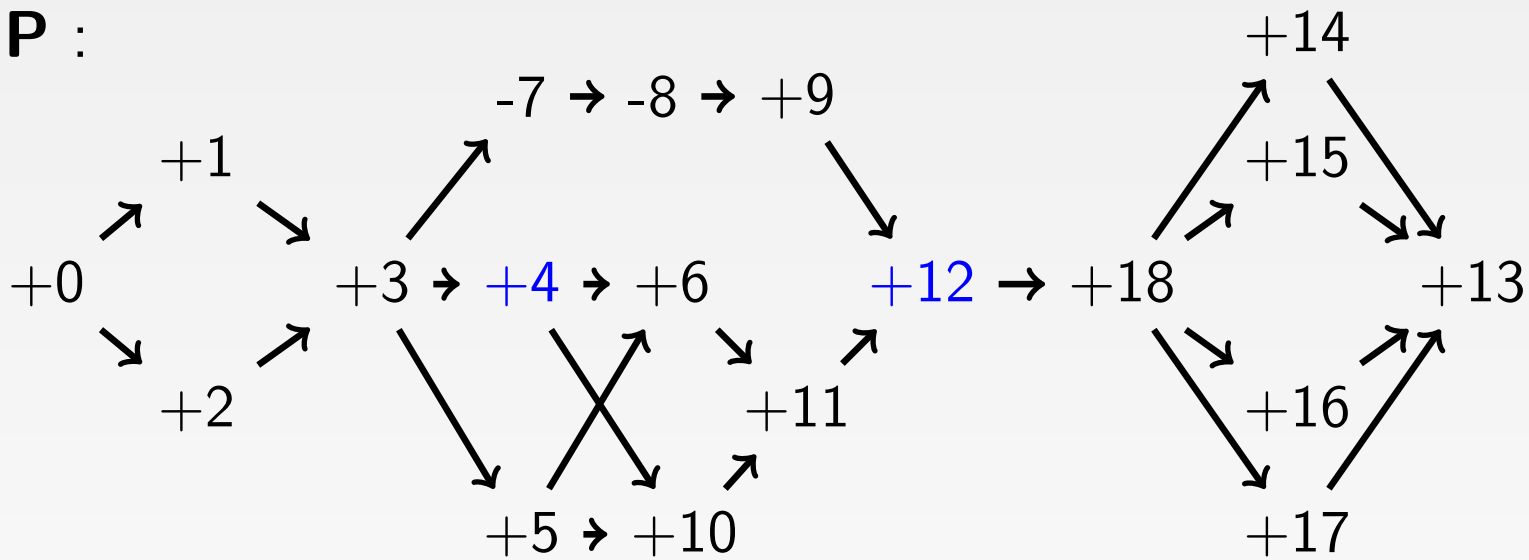
Notations

A partial order



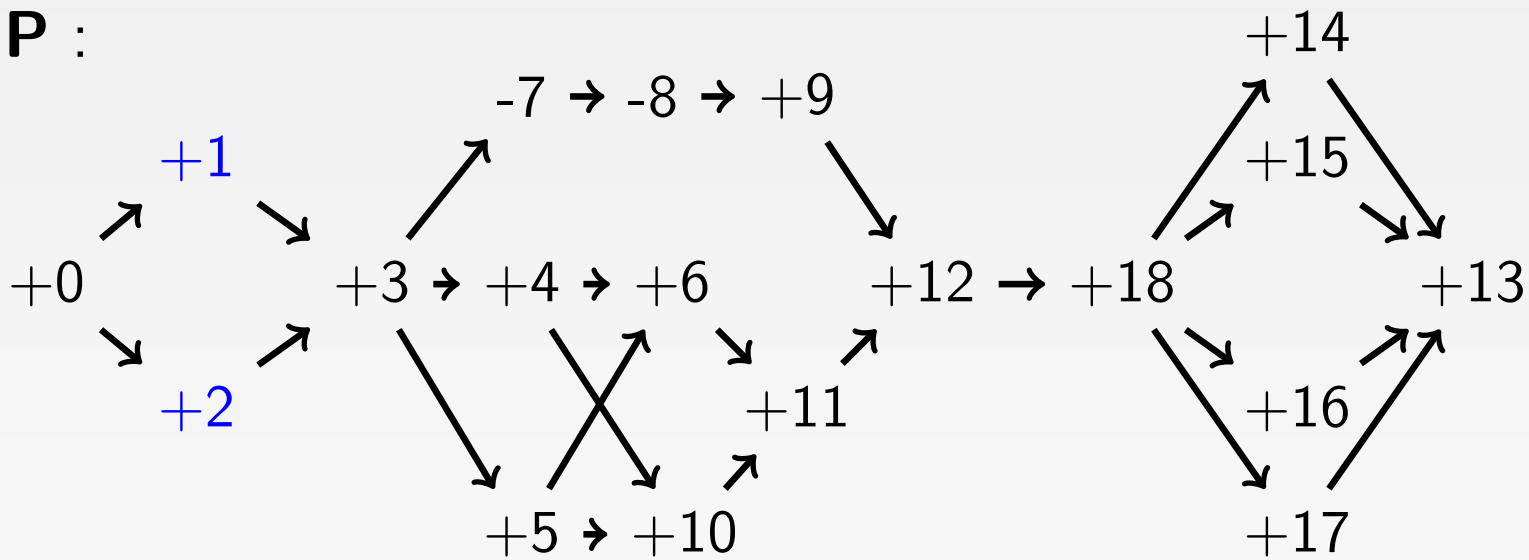
A partially ordered genome represented by a **DAG**.

A partial order



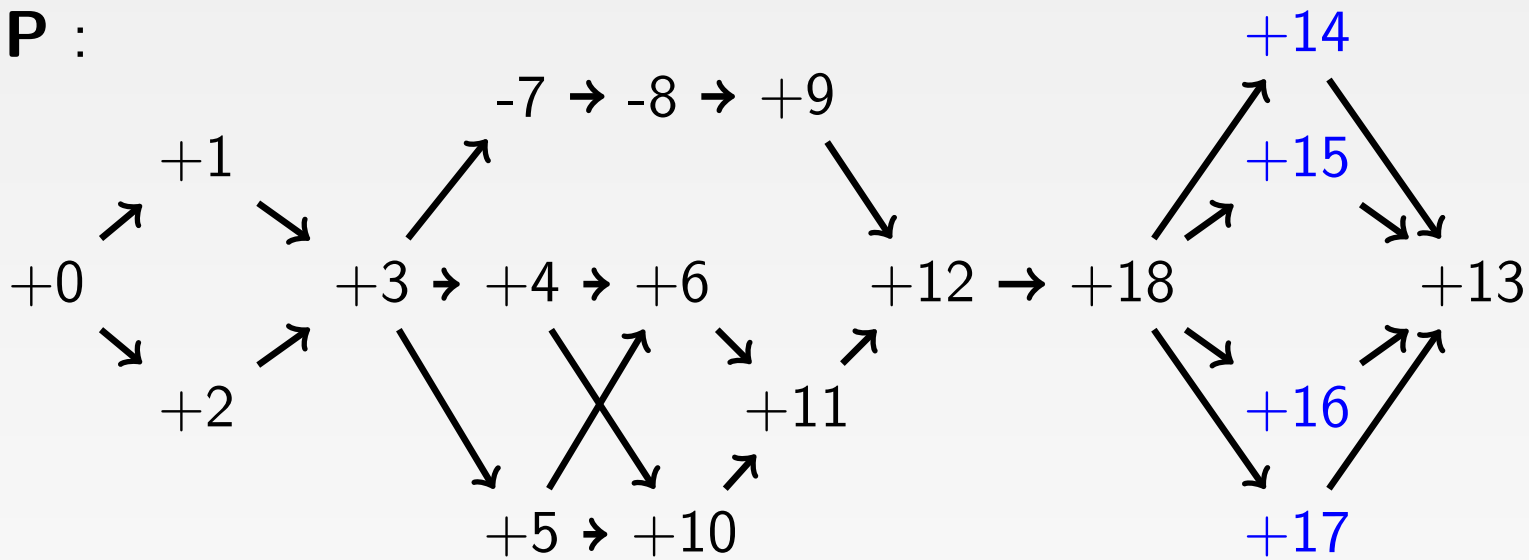
The gene +4 precedes the gene +12, both genes are **comparables**.

A partial order



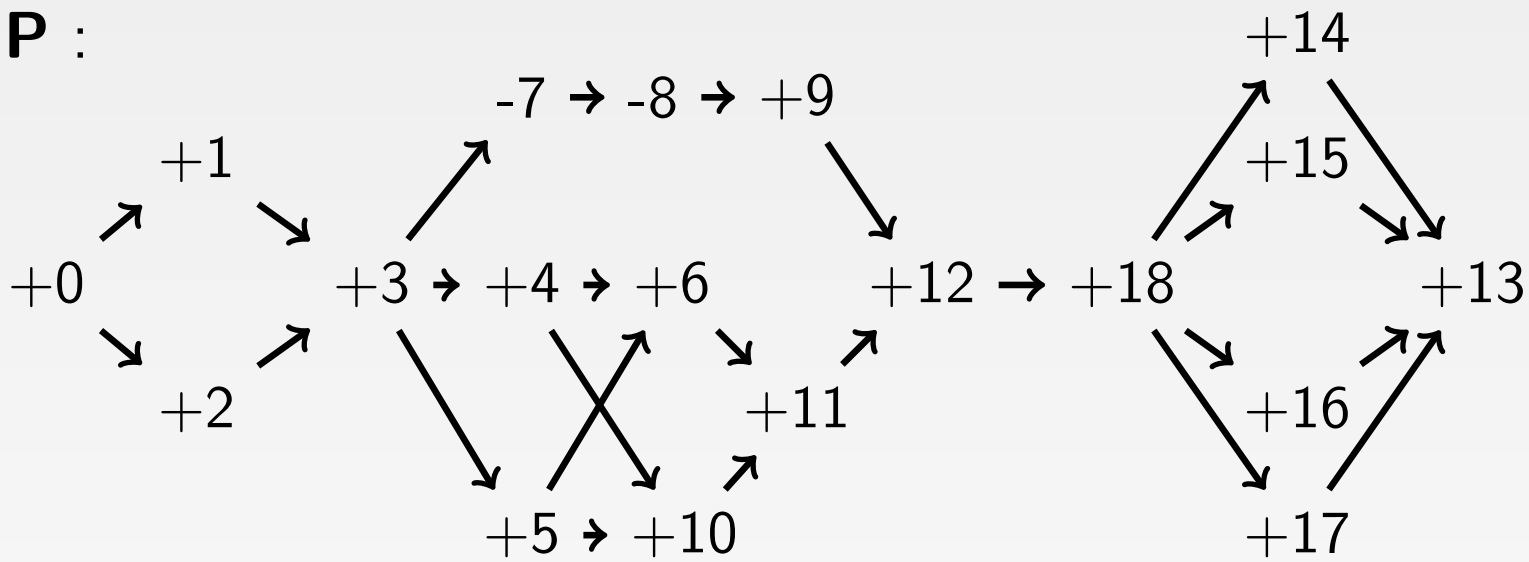
The genes +1 and +2 are **incomparables**.

A partial order



The **width** of **P** is the size of the maximal set of incomparable genes: 4.

A linear extension



T :

+0 +1 +2 +3 +4 +5 +6 -7 -8 +9 +10 +11 +12 +18 +17 +16 +15 +14 +13

- **Studied measures**: number of common intervals, number of adjacencies.
- **NP-hard problems** [Blin *et al.* 2007, Fu and Jiang 2006].
- We want, in the future, to **evaluate** some heuristics.
- We express our problems as **pseudo-boolean programs**.

$\Rightarrow$ **Exact** algorithms.

Comparison between two **partially ordered** genomes

Exact algorithms

Three studied problems:

The number of common intervals

- **MCIL-1PO**: Confront **one** partially ordered genome P_1 and a reference totally ordered genome *Id* and maximize the number of common intervals.

The number of adjacencies

- **MAL-1PO**: Confront **one** partially ordered genome P_1 and a reference totally ordered genome *Id* and maximize the number of adjacencies.
- **MAL-2PO**: Confront **two** partially ordered genomes P_1 and P_2 and maximize the number of adjacencies.

We have 1 common variable 's type:

$a^x_{g,i}$ denotes the gene g at the position i ($i \in \{1, n\}$) in the linear extension T_x ($x \in \{1, 2\}$).

3 constraints:

C.a $\forall\ 0 \leq g \leq n, \sum_{0 \leq i \leq n} a^x_{g,i} = 1$

C.b $\forall\ 0 \leq i \leq n, \sum_{0 \leq g \leq n} a^x_{g,i} = 1$

C.c $\forall\ 0 \leq g_1 \leq n,\ 0 \leq g_2 \leq n,\ g_1 \prec_x g_2,\ 0 < j \leq i \leq n,\ a^x_{g_1,i} + a^x_{g_2,j} \leq 1$

We have 4 specific variable's types:

$b_{g,i,t}$ denotes the gene g at a position between $T_1[i]$ and $T_1[i+t]$;

$c_{g,i,t}$ denotes the common interval compose of the genes $\{g, g+1, \ldots, g+t\}$ at positions $\{i, i+1, \ldots, i+t\}$ in T_1;

$d_{g,i}$ denotes an adjacency between g and $g+1$ with g at the position $T_1[i]$;

e_{g_1,i,j,g_2} denotes an adjacency between g_1 and g_2 with g_1 at positions $T_1[i]$ and $T_2[j]$.

Illustration of the variables $a^x_{g,i}$

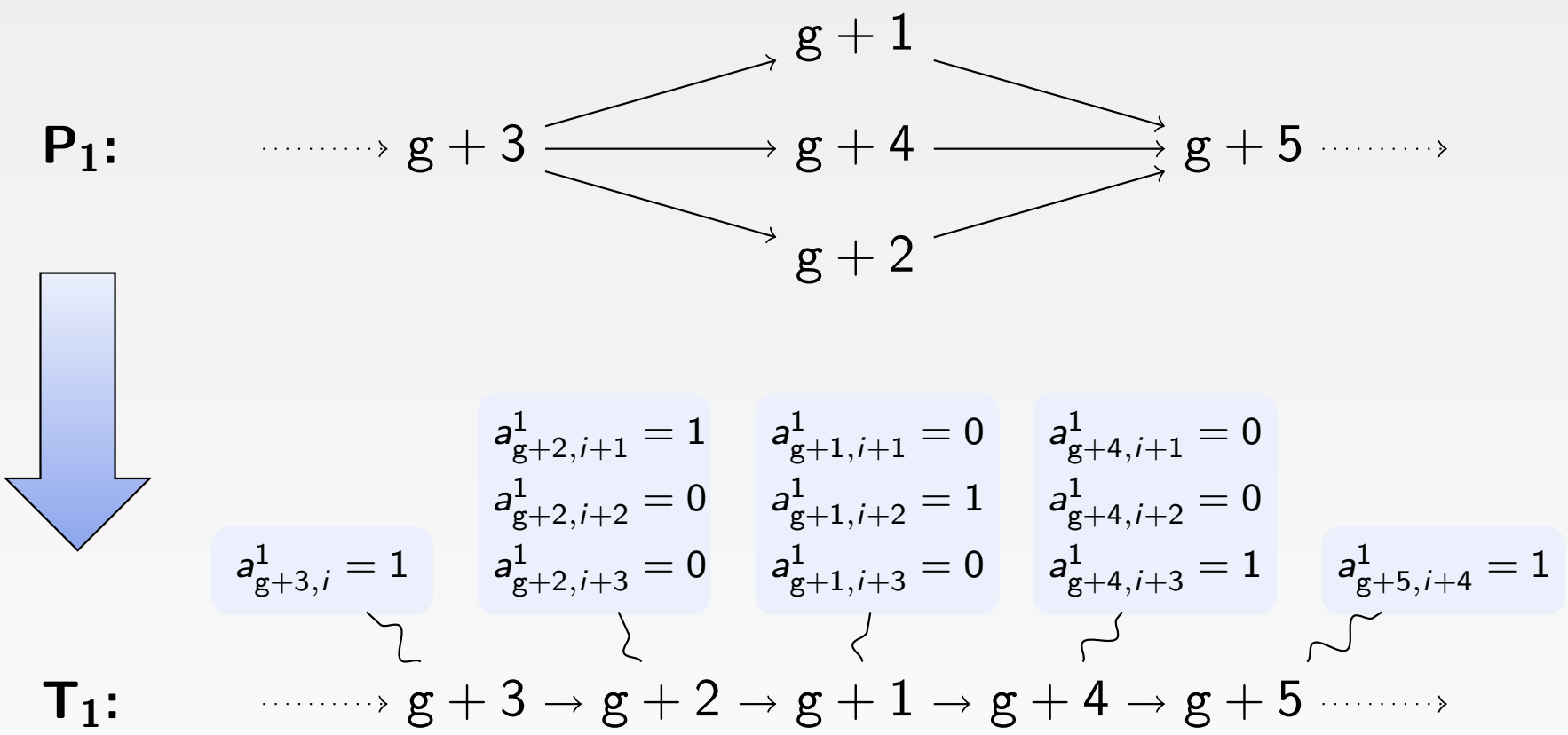
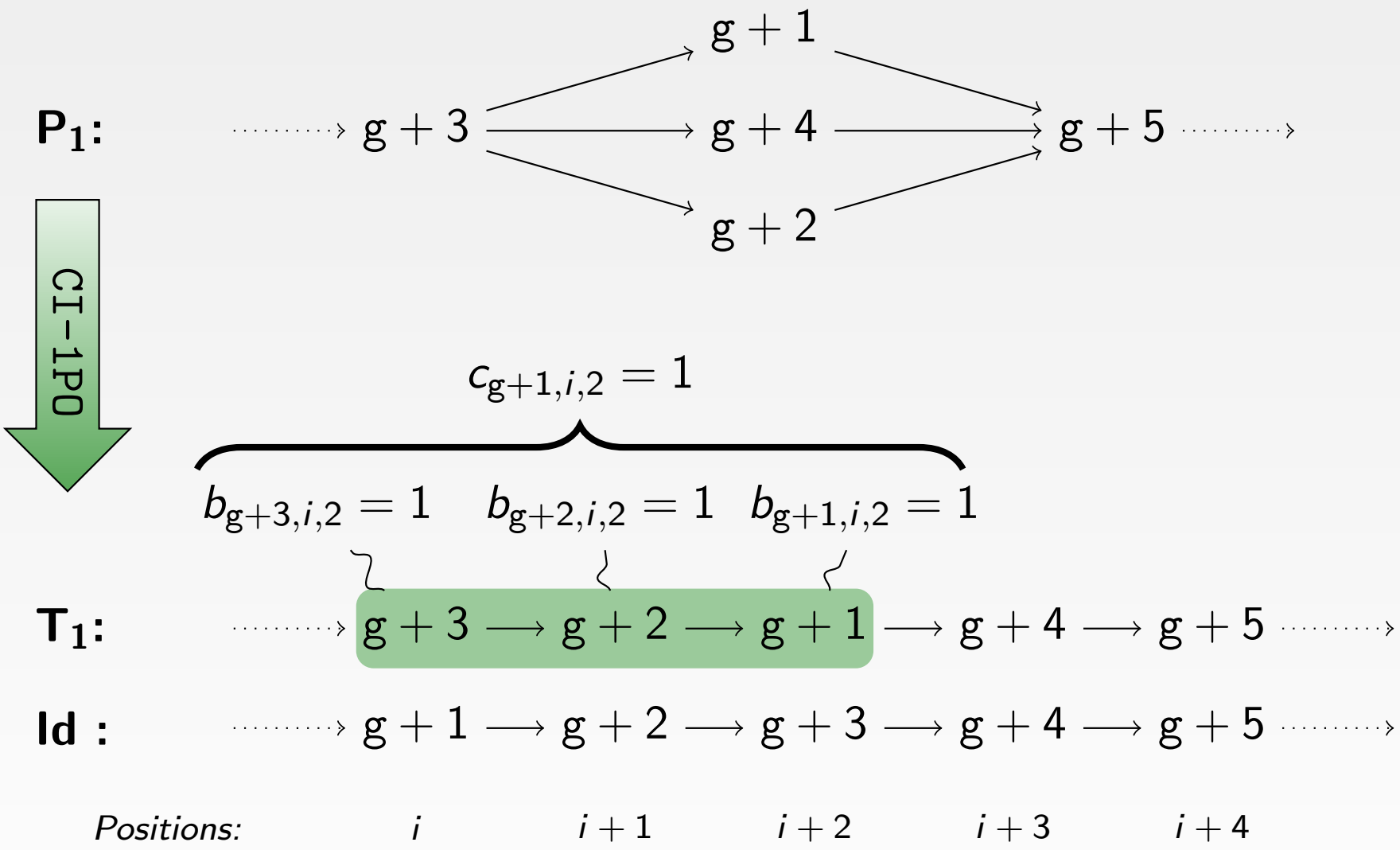


Illustration of the variables $b_{g,i,t}$ and $c_{g,i,t}$



Comparison between two **partially ordered** genomes

Experimentation

Experimentation

Simulated dataset [Blin *et al.* 2006]:

- The *size* $n \in \{30, 40, 50, 60, 70, 80, 90\}$;
- The *order rate* $p \in \{0.7, 0.9\}$;
- The *gene distribution* $q \in \{0.4, 0.6, 0.8\}$;
- 19 unsigned genomes for each triplet (n, p, q).

Solver

- For this work, the solver used is `MiniSat+` [Een et Sorensson, 2006].

Results for the three programs

$q \in \{0.4, 0.6, 0.8\}$
Quadri Intel(R) Xeon(TM) CPU 3.00 GHz with 16GB of memory.

MCIL-1PO $\Rightarrow$ `CI-1PO`

- 494 results out of 570 (**87%**), $n \in \{30, \ldots, 90\}$, for $p = 0.9$
  , $n \in \{30, \ldots, 50\}$, for $p = 0.7$
- **2 hours** in average (6% case $>$ 1 hour).

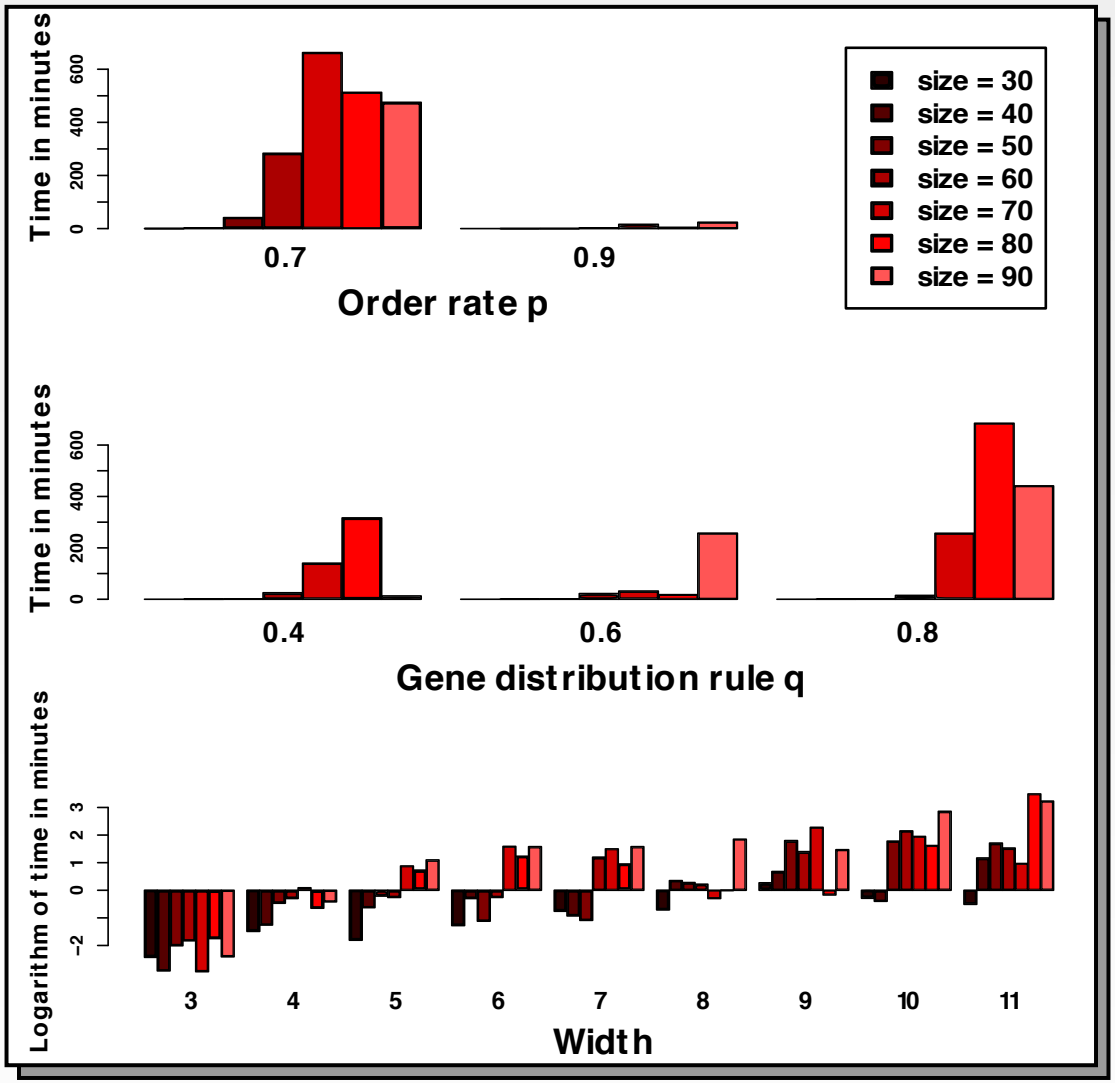
MAL-1PO $\Rightarrow$ `Adjacency-1PO`

- 778 results out of 798 (**97%**), $n \in \{30, \ldots, 90\}$, $p \in \{0.7, 0.9\}$
- in average **2 hours** (9% case $>$ 1 hour).

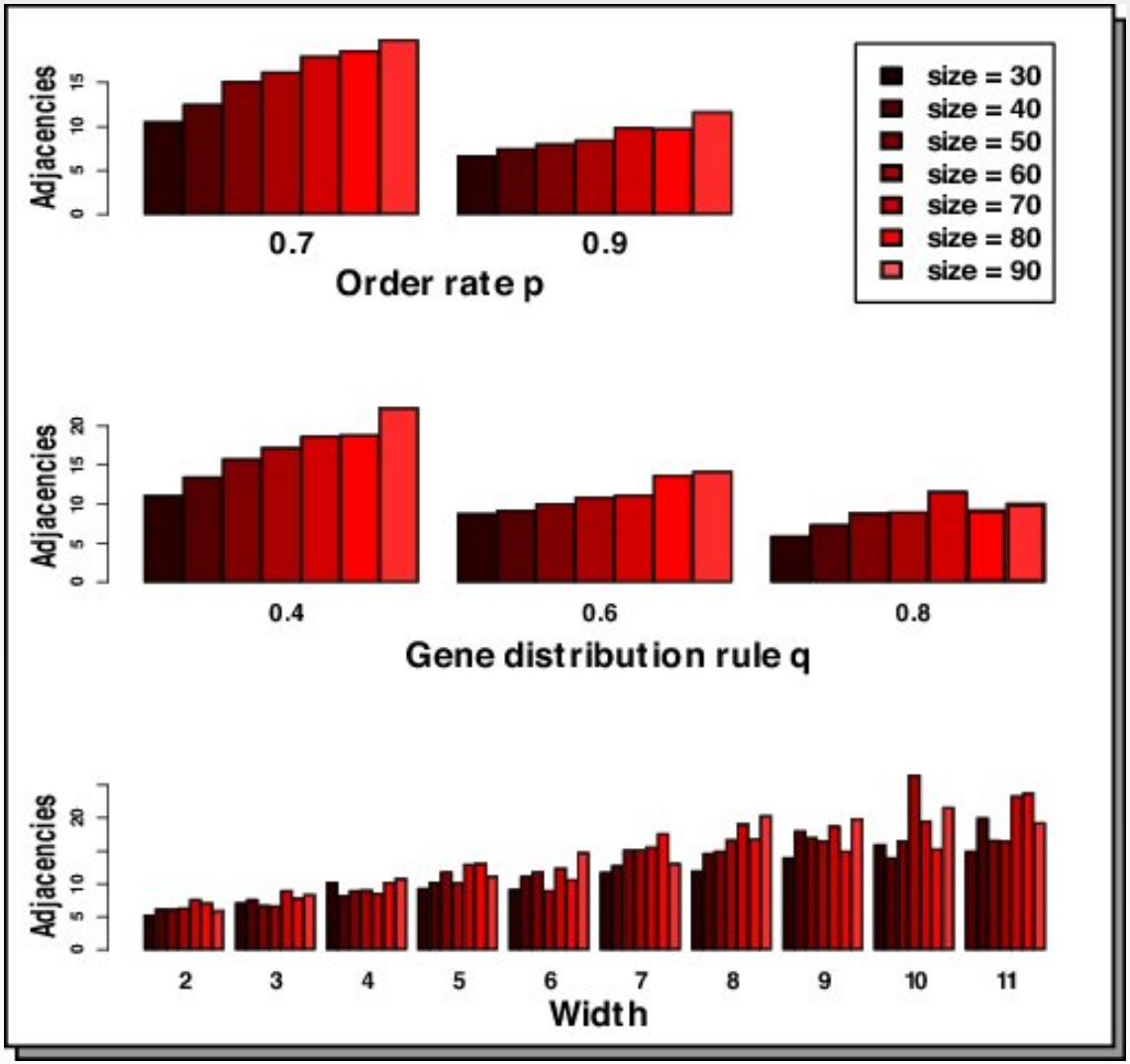
MAL-2PO $\Rightarrow$ `Adjacency-2PO`

- 1 852 results out of 2 052 (**90%**), $n \in \{30, 40, 50\}$, $p = 0.9$
- **1 hour** in average (8% case $>$ 1 hour).

Influence on time, Adjacency-1PO



Influence on the measure, Adjacency-1OP



Comparison between two measures

CI-1PO

90% of results give the maximum number of adjacencies.

Adjacency-1PO

16% of results give the maximum number of common intervals.

Conclusion

Conclusion

Genomes with duplicated genes

- A **pseudo-boolean program** to compute two distances (the number of adjacencies and number of breakpoints) between two genomes with duplication under three models (*exemplar*, *maximum* and *intermediate* matching).
- **Rules of reduction** to speed-up the programs.
- Two **heuristics** for each model: simple, fast and efficient on the dataset we studied.

Conclusion

Partially ordered genomes

- Dealing with **partially ordered genomes**.
- **Exact algorithms** for 3 problems (`MICL-1PO`, `MAL-1PO` and `MAL-2PO`).
- **Rules of reduction** to speed-up the programs.
- Influence of **parameters**.

Future works

General

- Test **other datasets**,
- **Improve the running time** of the programs,
- Study **other (dis)similarity measures**: MAD, SAD.

Genomes with duplicated genes

- **Double objective**: minimize the number of breakpoints and maximize the number of adjacencies at the same time.
- See in **details** the differences and similitudes between each model and measure.
- **Direct analysis** project: No homology assignement.
- **Supermarket** project: Comparison of two ways.

Future works

Partially ordered genomes

- Define and evaluate **heuristics**,
- **Generalize** the programs (`CI-2PO`, genomes with duplications).
- Compare a **set of contigs** and a reference genome.

Collaborators

Université Paris-Est (LIGM)

Stéphane Vialette

Université de Nantes (LINA)

Guillaume Fertin

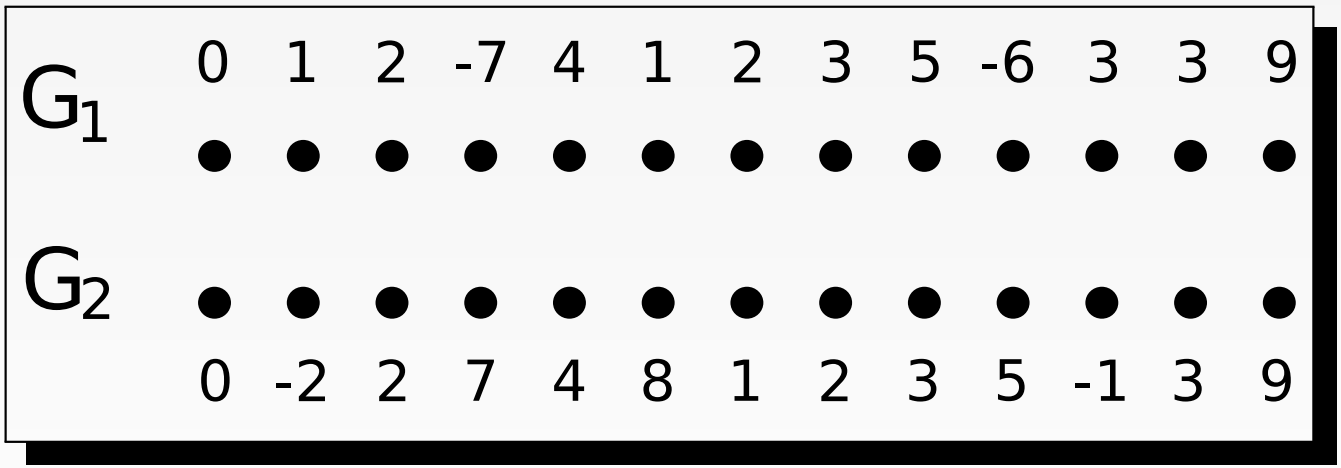
Irena Rusu

Sébastien Angibaud

Rules to speed-up the resolution

Genomes with duplications

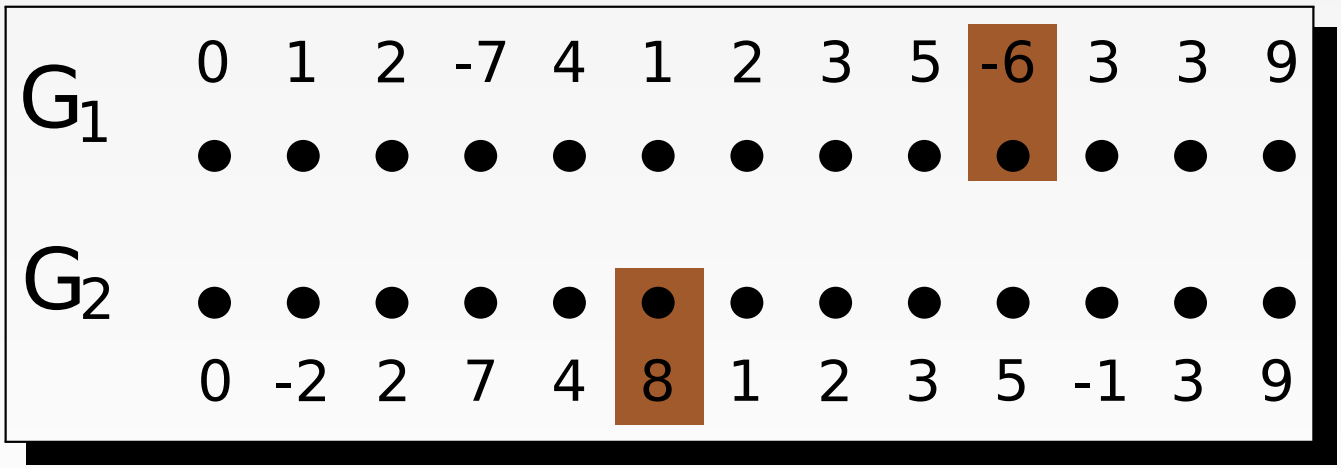
- **Pre-processing**: suppression of genes present in only one genome, specific suppressions under the exemplar model.
- **Rule of reduction**: matching between no-duplicate genes.
- **Pre-matching**: between two no-duplicated genes.



Rules to speed-up the resolution

Genomes with duplications

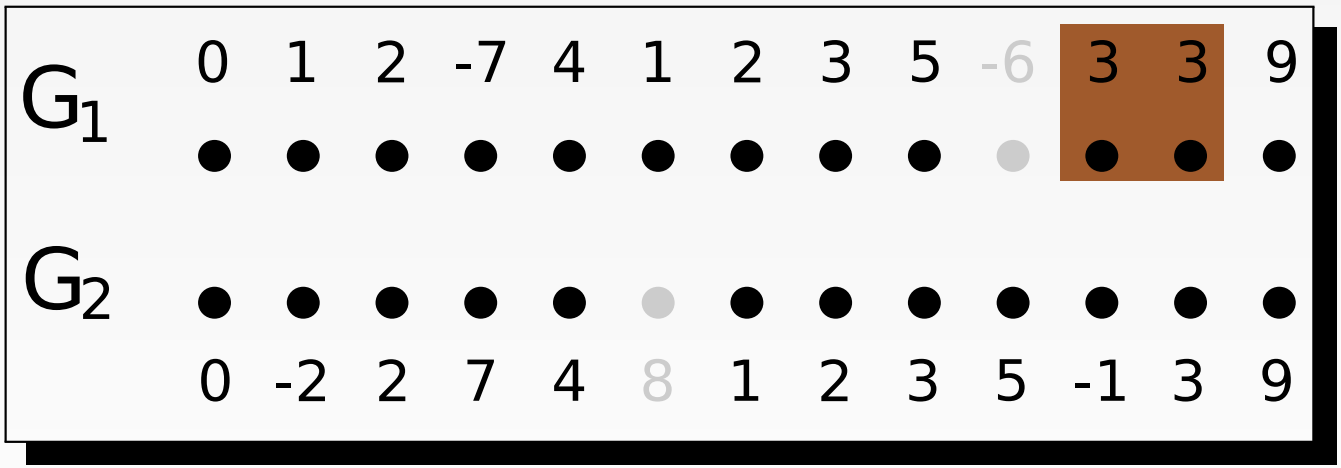
- **Pre-processing**: suppression of genes present in only one genome, specific suppressions under the exemplar model.
- **Rule of reduction**: matching between no-duplicate genes.
- **Pre-matching**: between two no-duplicated genes.



Rules to speed-up the resolution

Genomes with duplications

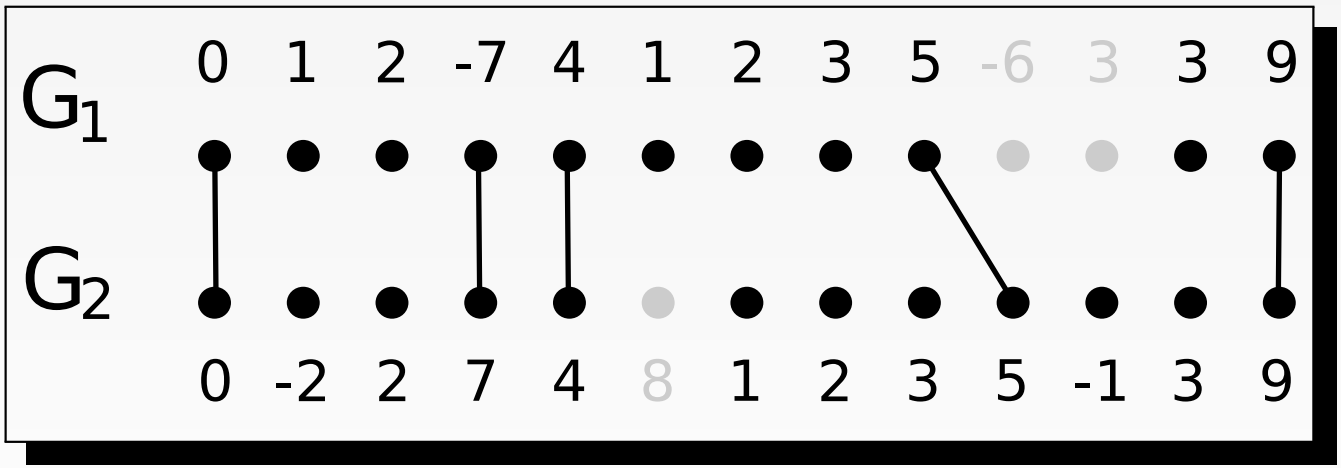
- **Pre-processing**: suppression of genes present in only one genome, specific suppressions under the exemplar model.
- **Rule of reduction**: matching between no-duplicate genes.
- **Pre-matching**: between two no-duplicated genes.



Rules to speed-up the resolution

Genomes with duplications

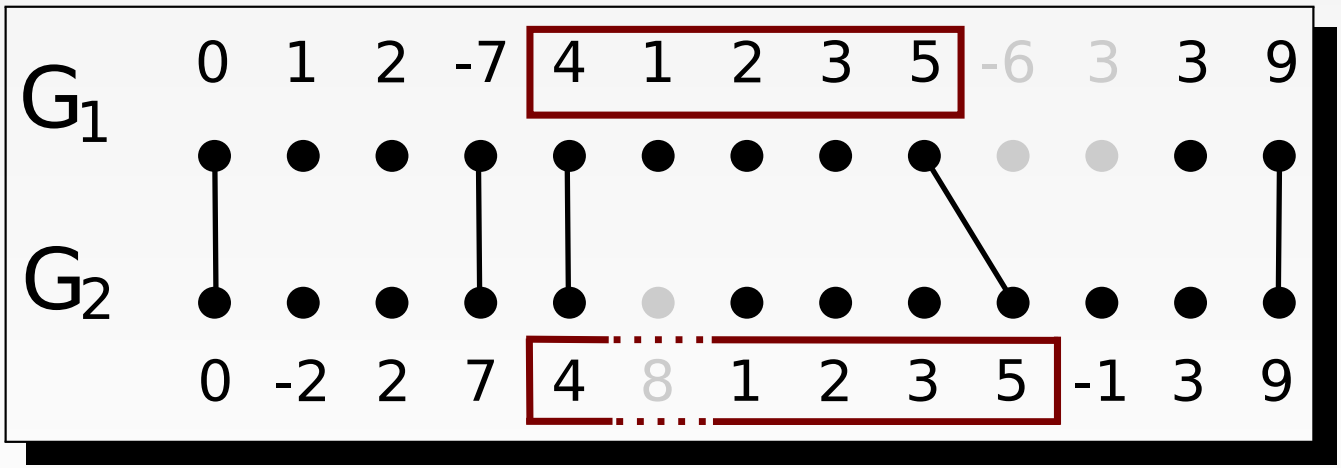
- **Pre-processing**: suppression of genes present in only one genome, specific suppressions under the exemplar model.
- **Rule of reduction**: matching between no-duplicate genes.
- **Pre-matching**: between two no-duplicated genes.



Rules to speed-up the resolution

Genomes with duplications

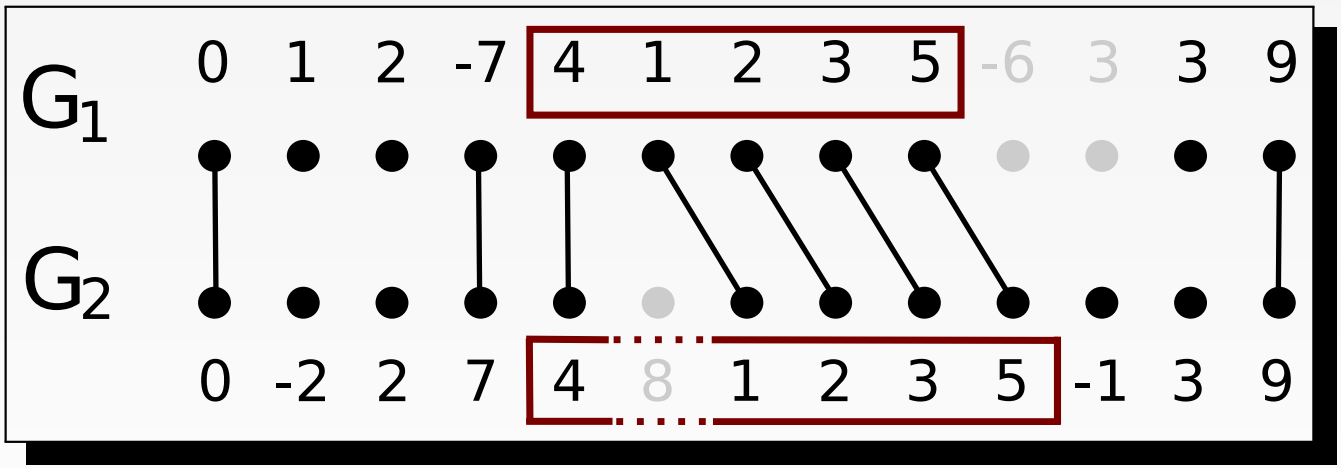
- **Pre-processing**: suppression of genes present in only one genome, specific suppressions under the exemplar model.
- **Rule of reduction**: matching between no-duplicate genes.
- **Pre-matching**: between two no-duplicated genes.



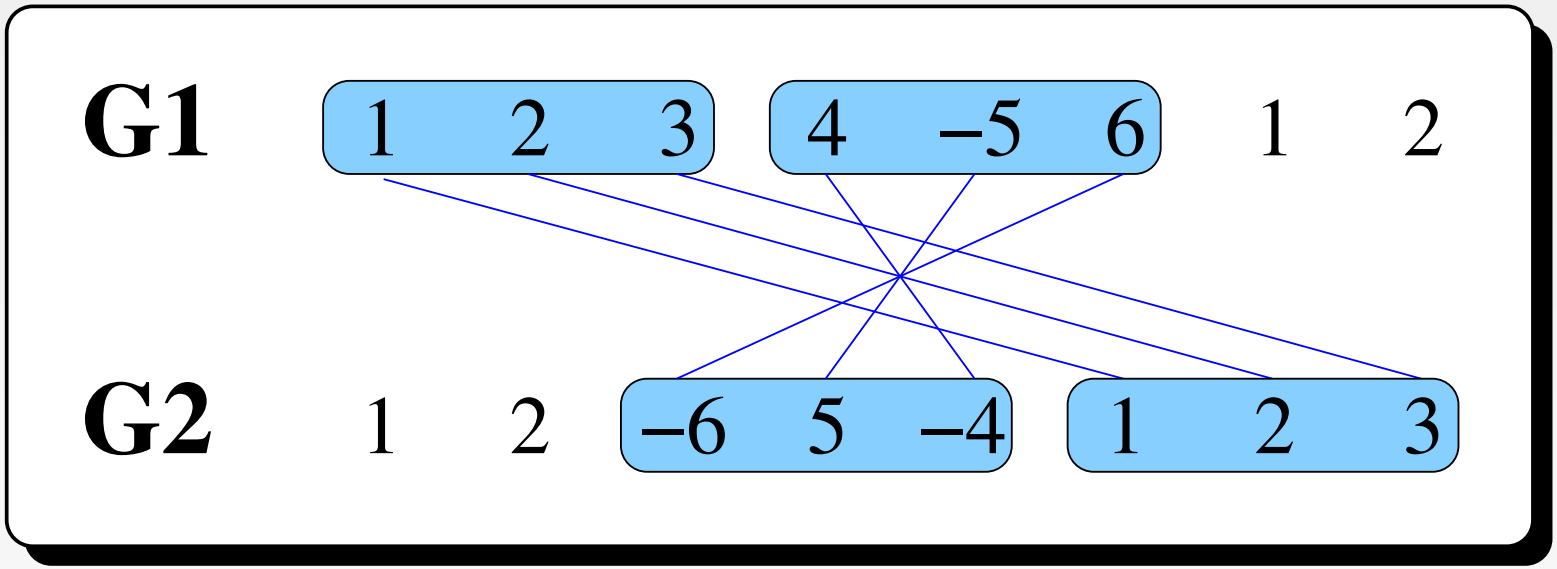
Rules to speed-up the resolution

Genomes with duplications

- **Pre-processing**: suppression of genes present in only one genome, specific suppressions under the exemplar model.
- **Rule of reduction**: matching between no-duplicate genes.
- **Pre-matching**: between two no-duplicated genes.

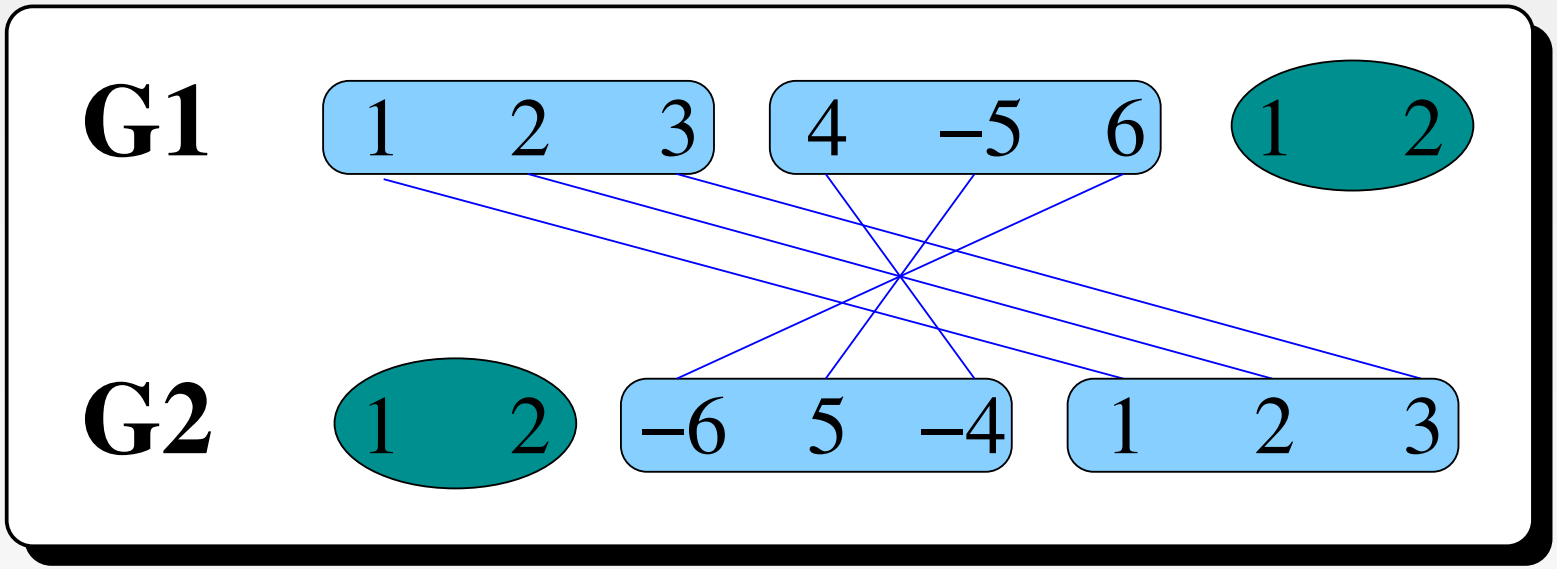


Heuristic IILCS_IA



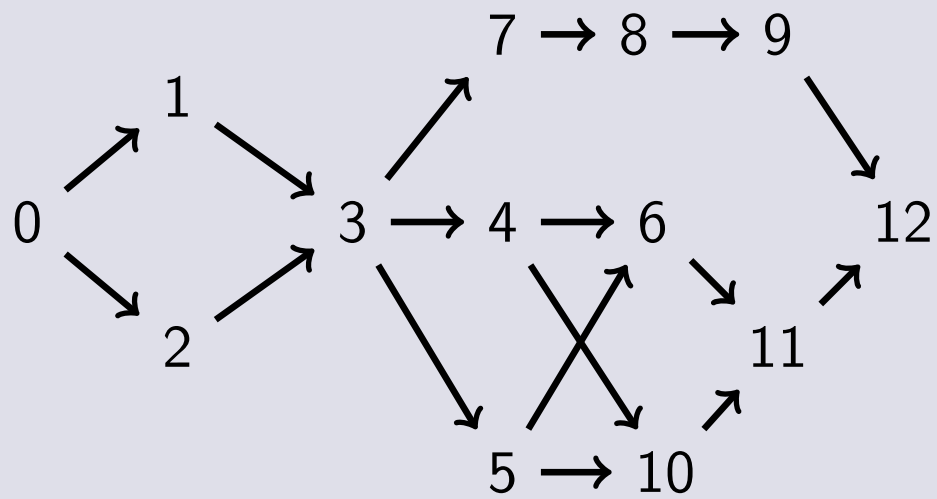
Each family of genes have at least one gene matched
$\Rightarrow$ The heuristic IILCS_IA stop.

Heuristic IILCS_IA



We can again increase the number of adjacencies until the size of LCS if superior than 1.

Parameters of partial order [Blin et al., 2007]



$$(0 \rightarrow (1,2) \rightarrow 3 \rightarrow ((7 \rightarrow 8 \rightarrow 9),(4,5) \rightarrow (6,10) \rightarrow 11) \rightarrow 12$$

- The **gene distribution** q: define the disorder;
- The **order rate** p: probability to have "$\rightarrow$".

The heuristics can be bad

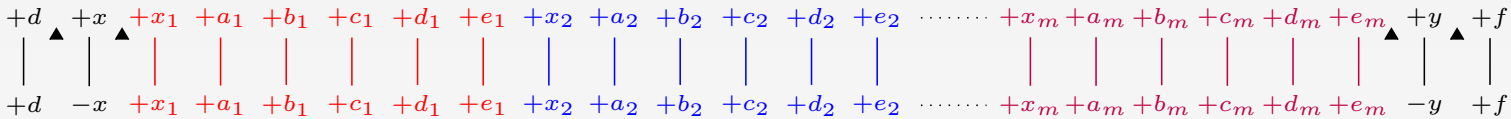
ENTRÉES

$$G_1 \;\; +d \;\; +x \;\; A_1 \;\; A_2 \;\; \cdots\cdots \;\; A_m \;\; +y \;\; +f$$
$$G_2 \;\; +d \;\; -x \;\; B_1 \;\; B_2 \;\; \cdots\cdots \;\; B_m \;\; -y \;\; +f$$

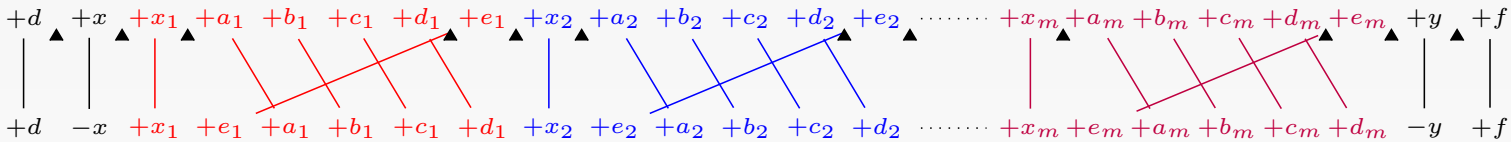
LCS de taille 4

$$\forall\ 1 \leq i \leq m \qquad \begin{array}{l} A_i = x_i \; + a_i \; + b_i \; + c_i \; + d_i \; + e_i \\ B_i = x_i \; + a_i \; + b_i \; + x_i \; + c_i \; + d_i \; + e_i \; + a_i \; + b_i \; + c_i \; + d_i \end{array}$$

$\mathcal{OPT}$



$\mathcal{H}$

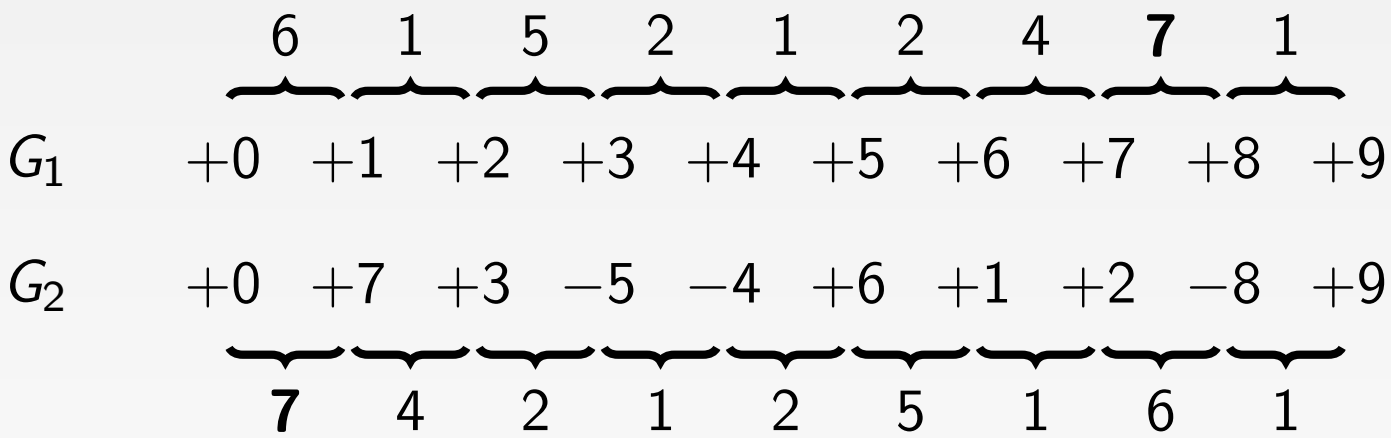


Nous obtenons $3m + 3$ points de cassures et $3m + 2$ adjacences via $\mathcal{H}$ contre 4 et $6m$ avec un programme exacte.

Measures MAD and SAD

G_1	$+0$	$+1$	$+2$	$+3$	$+4$	$+5$	$+6$	$+7$	$+8$	$+9$
G_2	$+0$	$+7$	$+3$	-5	-4	$+6$	$+1$	$+2$	-8	$+9$

Measures MAD and SAD



State of the art

Previous results [Chen et al., 2006]

Under the exemplar model, there do not exist an approximation allows to minimize the number of breakpoints between two genomes even if every gene is present at most three times in each genome.

Result for the exemplar model

Lemma

Under the exemplar model, define if there exist a matching without breakpoint is a **NP**-complet problem, even if every gene is present at most twice on one genome.

Demonstration.

Reduction from the VERTEX COVER problem.□

Result for the exemplar model

Lemma

Under the exemplar model, define if there exist a matching without breakpoint is a **NP**-complet problem, even if every gene is present at most twice on one genome.

Theorem

Under the exemplar model, minimize the number of breakpoints is not approximable even if every gene is present at most twice on one genome.

Results under the intermediate and maximum models

Lemma

Under exemplar and intermediate models, the problems to define if there exist a matching without breakpoint are equivalent problems.

Results under the intermediate and maximum models

Lemma

Under exemplar and intermediate models, the problems to define if there exist a matching without breakpoint are equivalent problems.

Theorem

Minimize the number of breakpoints, under the intermediate model, is a non approximable problem even if every gene is present at more twice on one genome.

$\Rightarrow$ No equivalence under the **maximum** model.
$\Rightarrow$ Currently: equivalent result for 2 genomes where every gene is present **at more twice on each genome** [Sikora, 2009].