ALGORITHMES DE CONSTRUCTION ET CORRECTION **D'ARBRES DE GÈNES** PAR LA **RÉCONCILIATION**

Manuel Lafond, Université de Montréal

Le plan

Introduction

- **Arbres de gènes, arbres d'espèces**
- **Réconciliation**

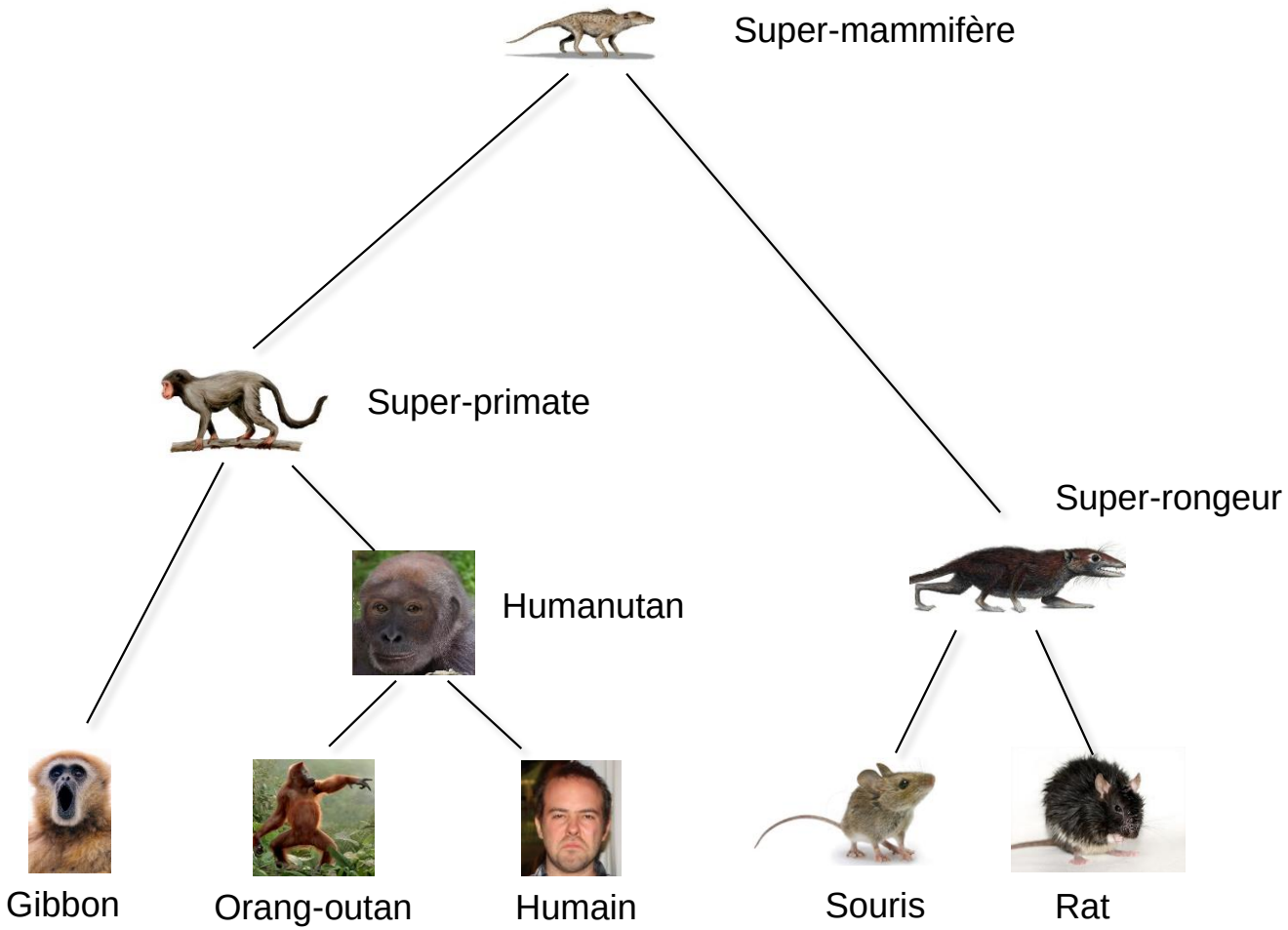
Résolution de polytomies

Correction par relations d'orthologie

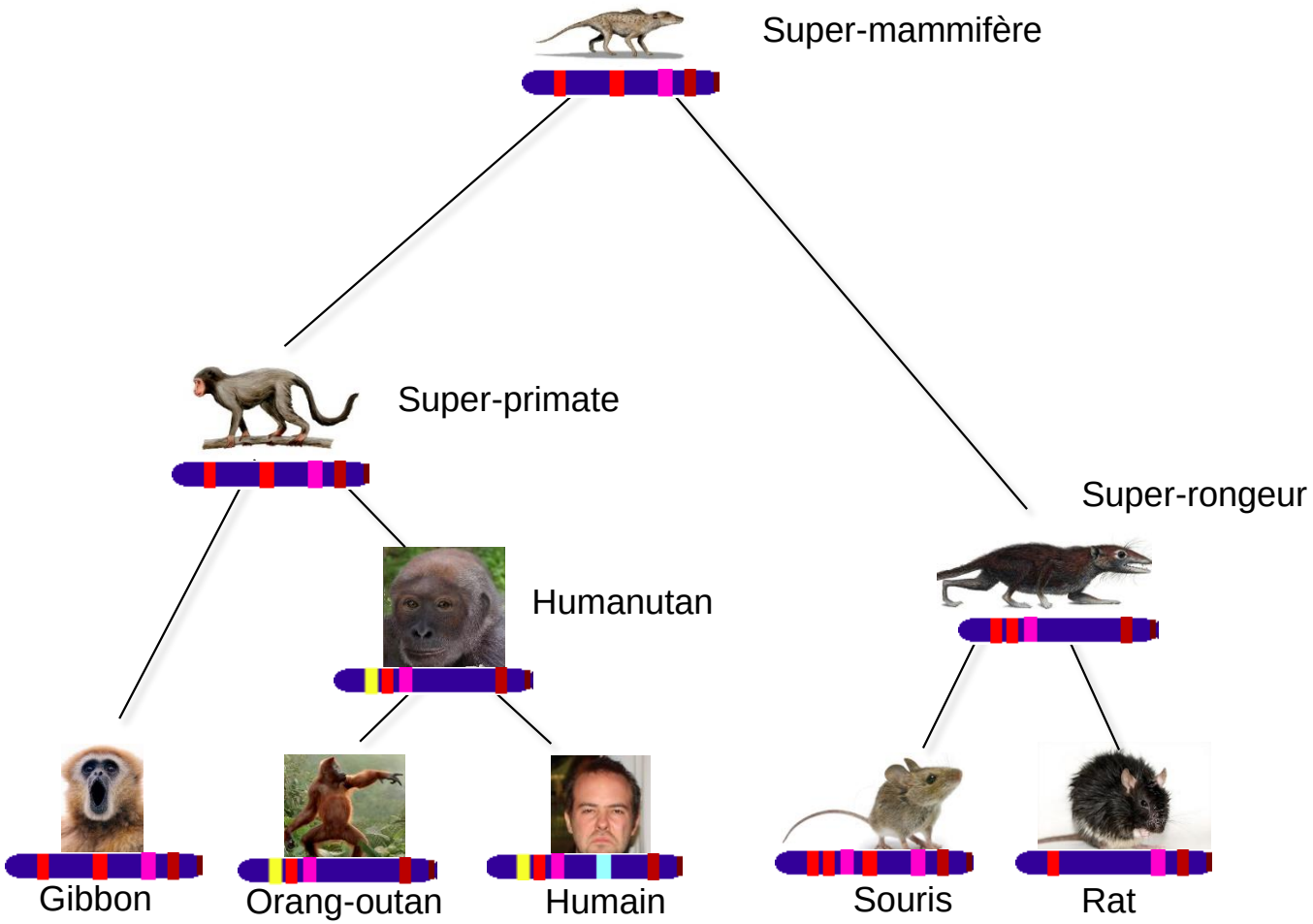
Validation de relations d'orthologie et paralogie

Superarbres de gènes et réconciliation

Arbre d'espèces



Arbre d'espèces



Arbre de gènes

Prenons par exemple le gène RPGR :

Retinitis pigmentosa GTPase regulator

Participe à la coloration des yeux.

Quelle est l'**histoire** du RPGR ?

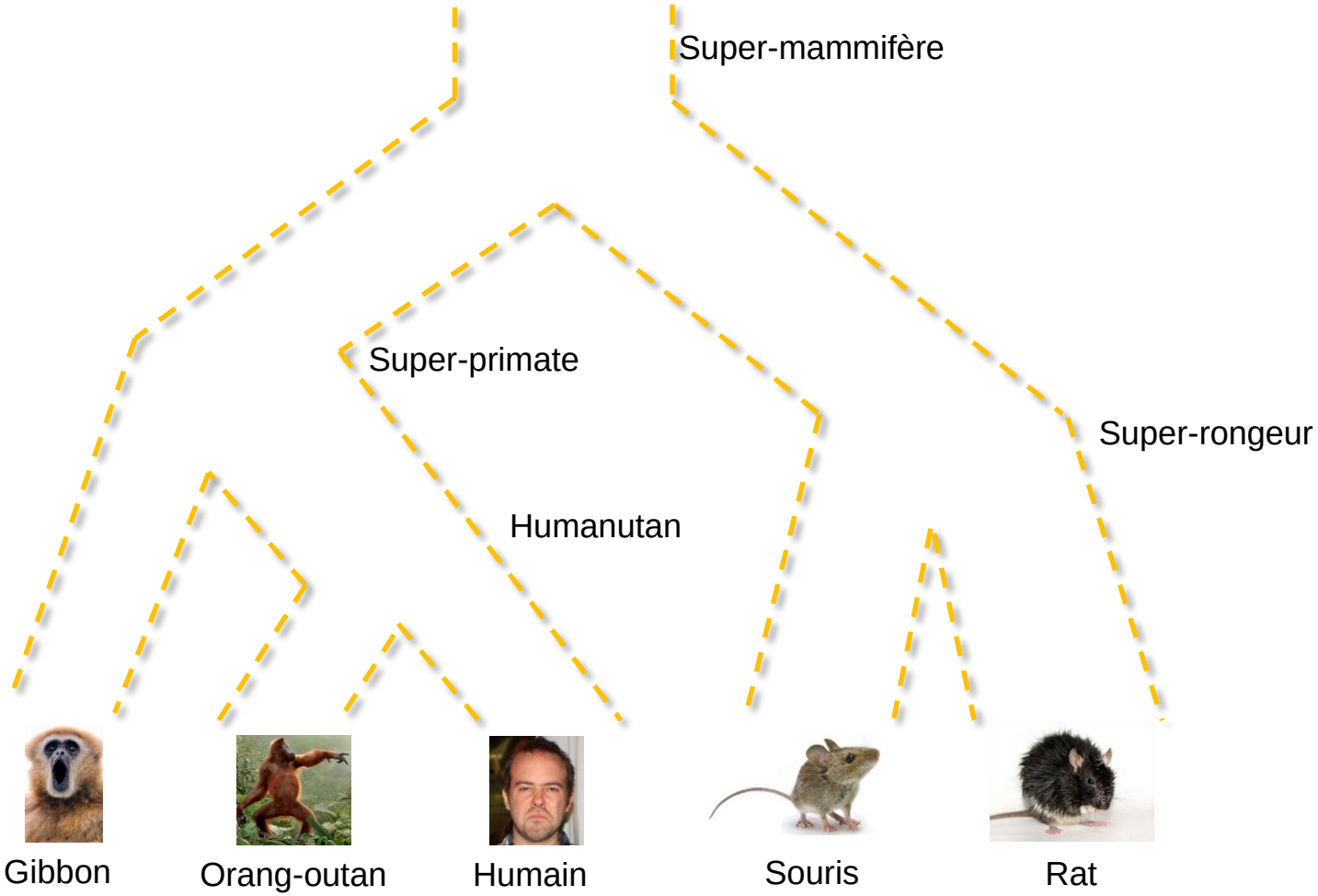
Presque tous les vertébrés ont ce gène. Certaines espèces ont même 2+ copies.

Qu'est-ce qui s'est passé?

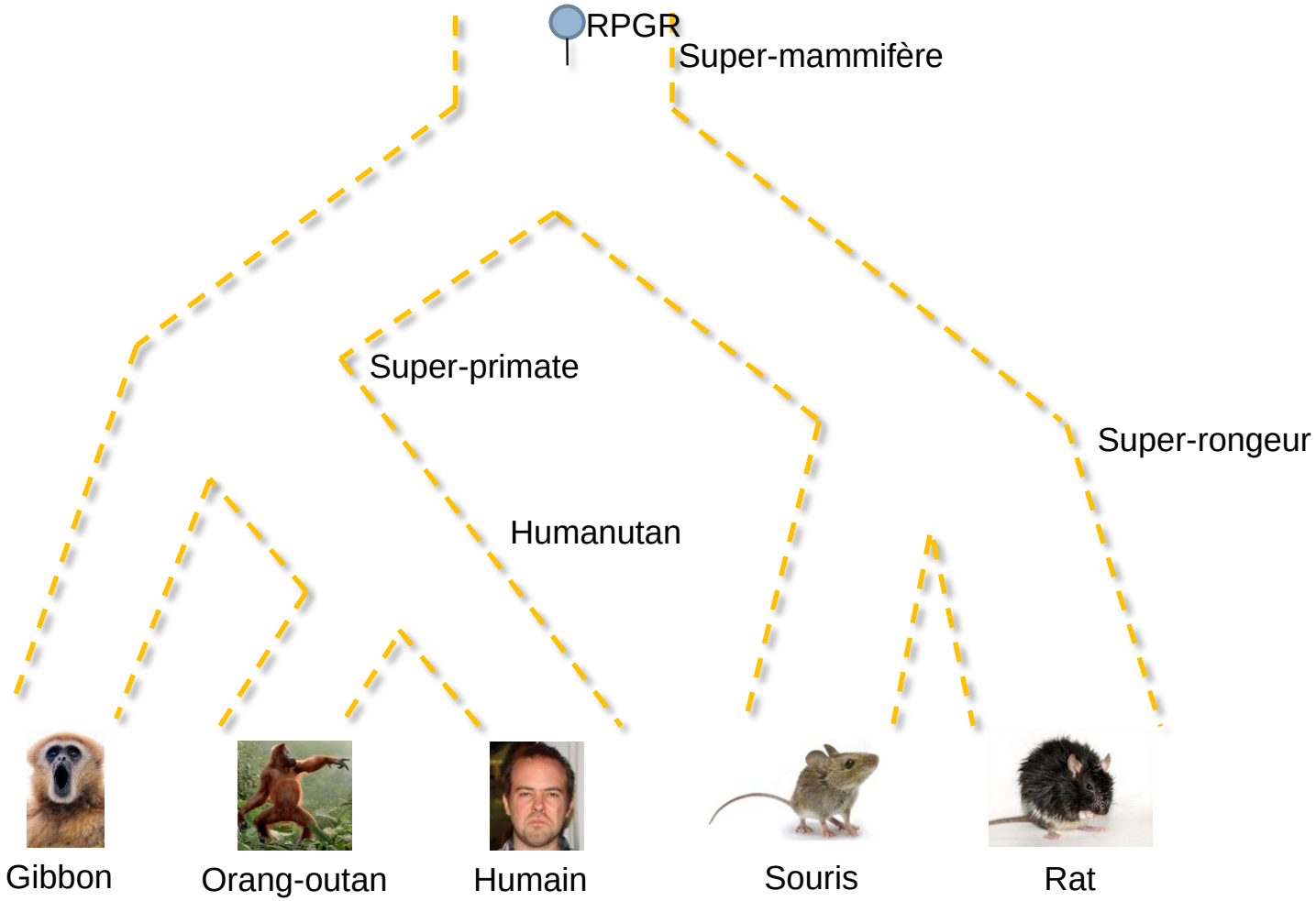
Un gène peut être :

- **Transmis** à des espèces descendantes par **spéciation**
- **Dupliqué**
- **Perdu**

Arbre de gènes

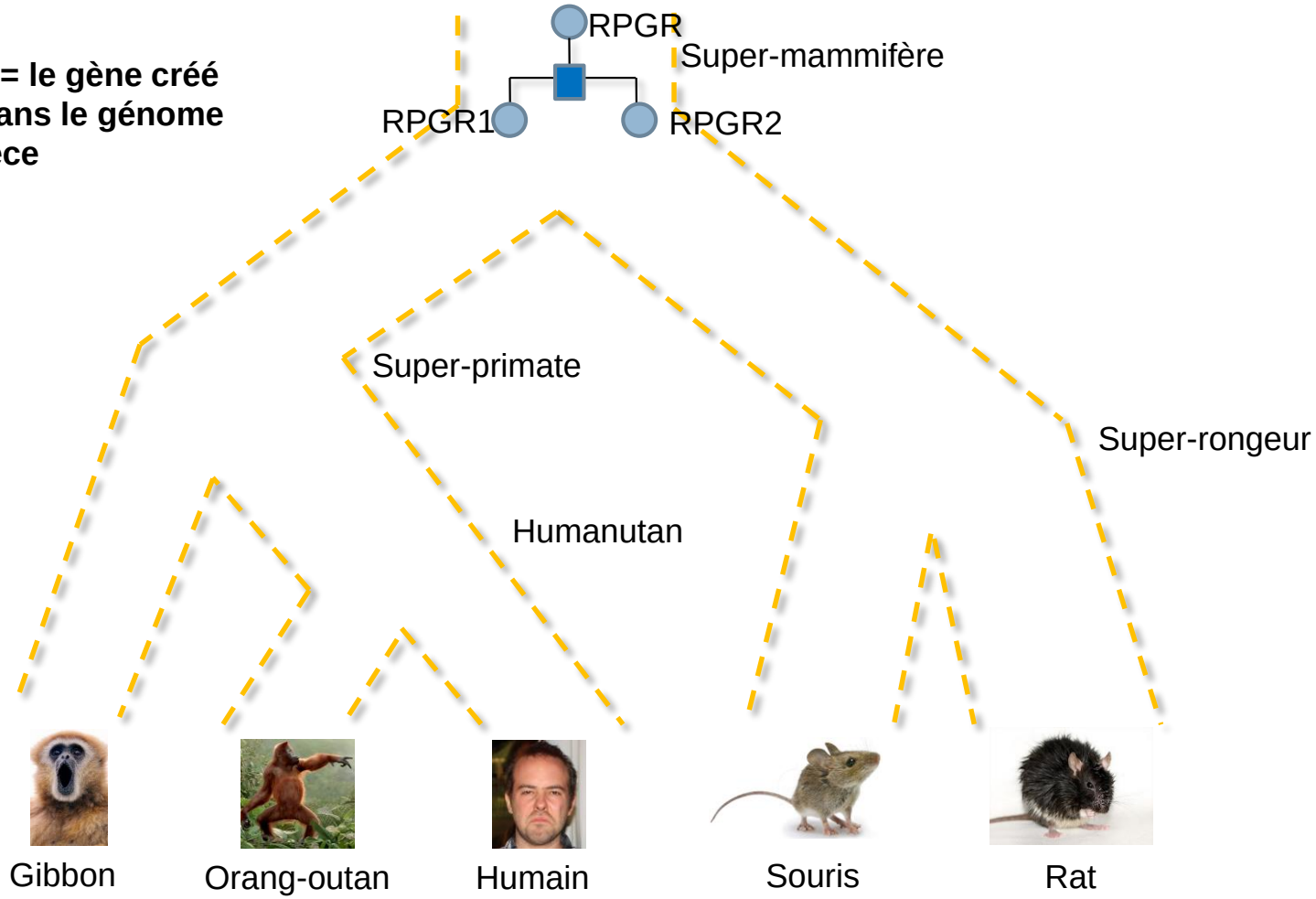


Arbre de gènes



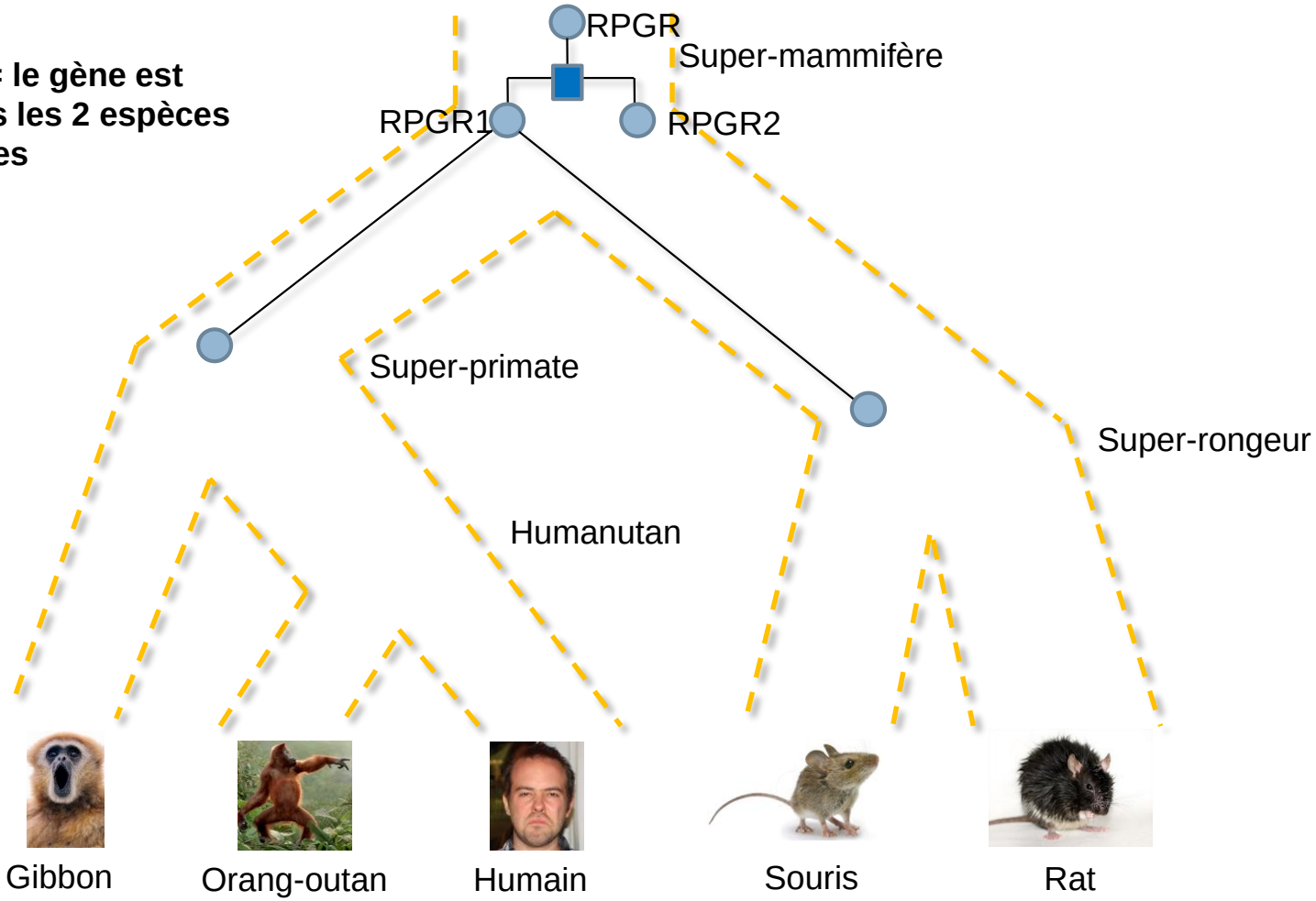
Arbre de gènes

Duplication = le gène créé une copie dans le génome de son espèce

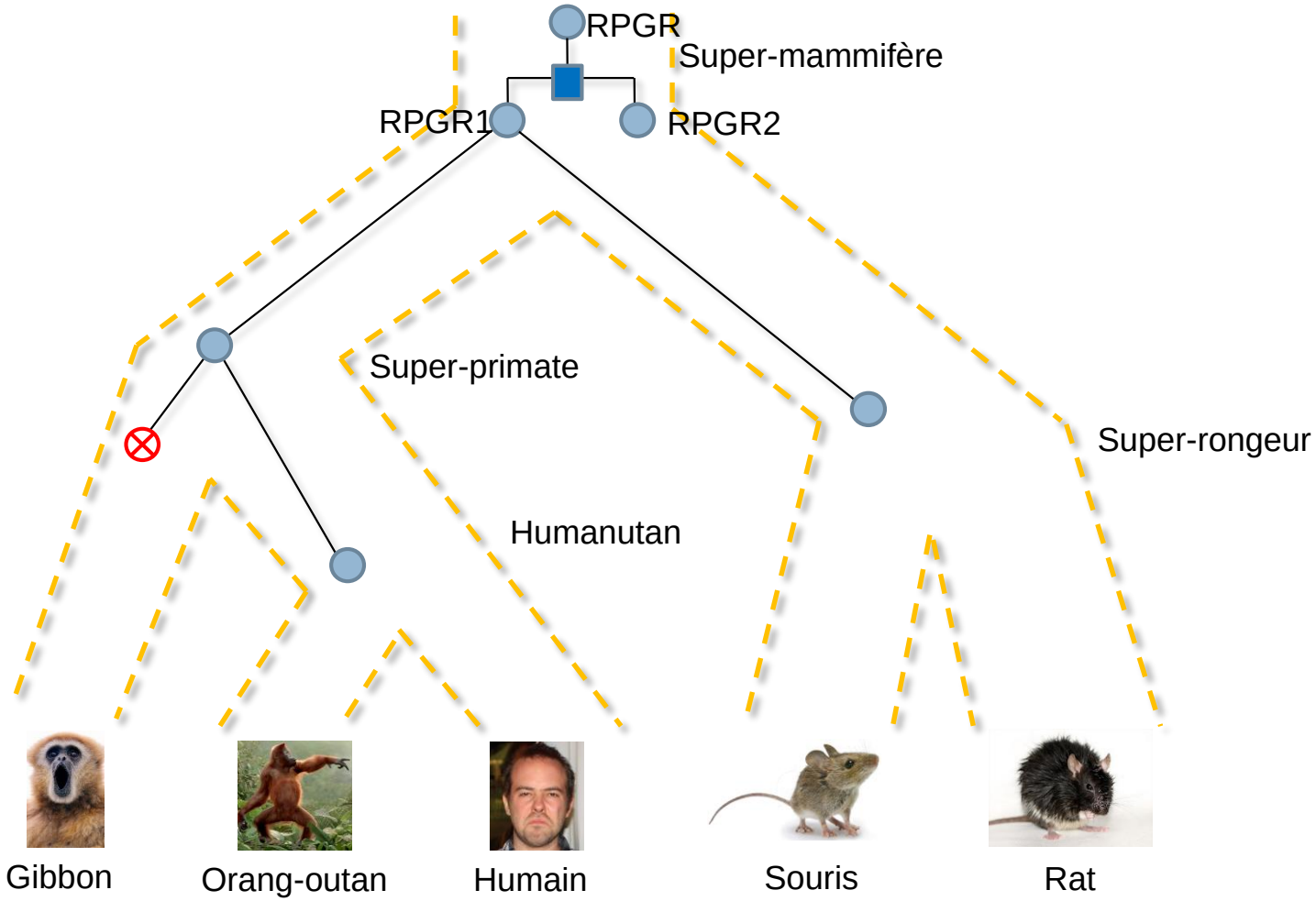


Arbre de gènes

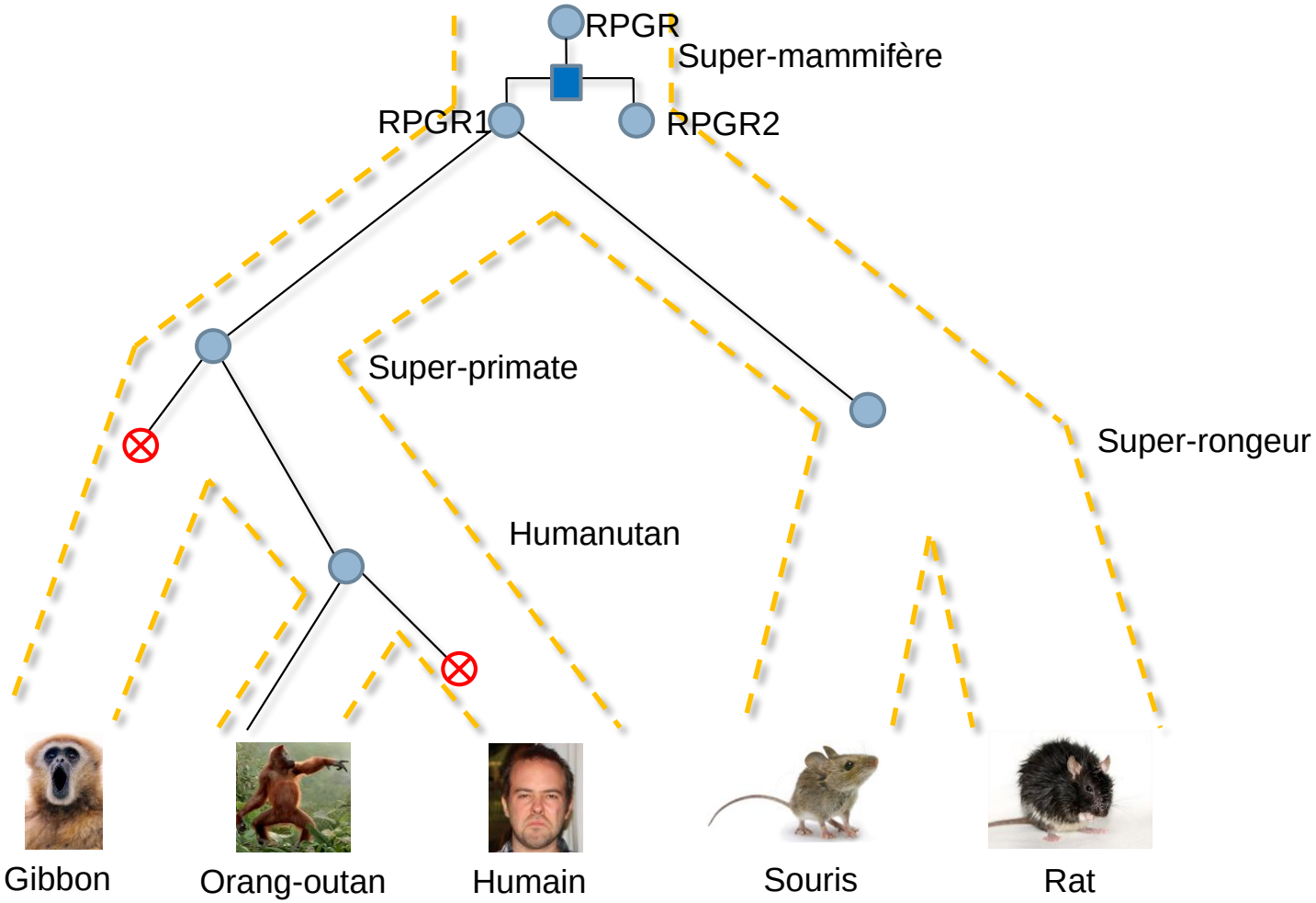
Speciation = le gène est envoyé dans les 2 espèces descendantes



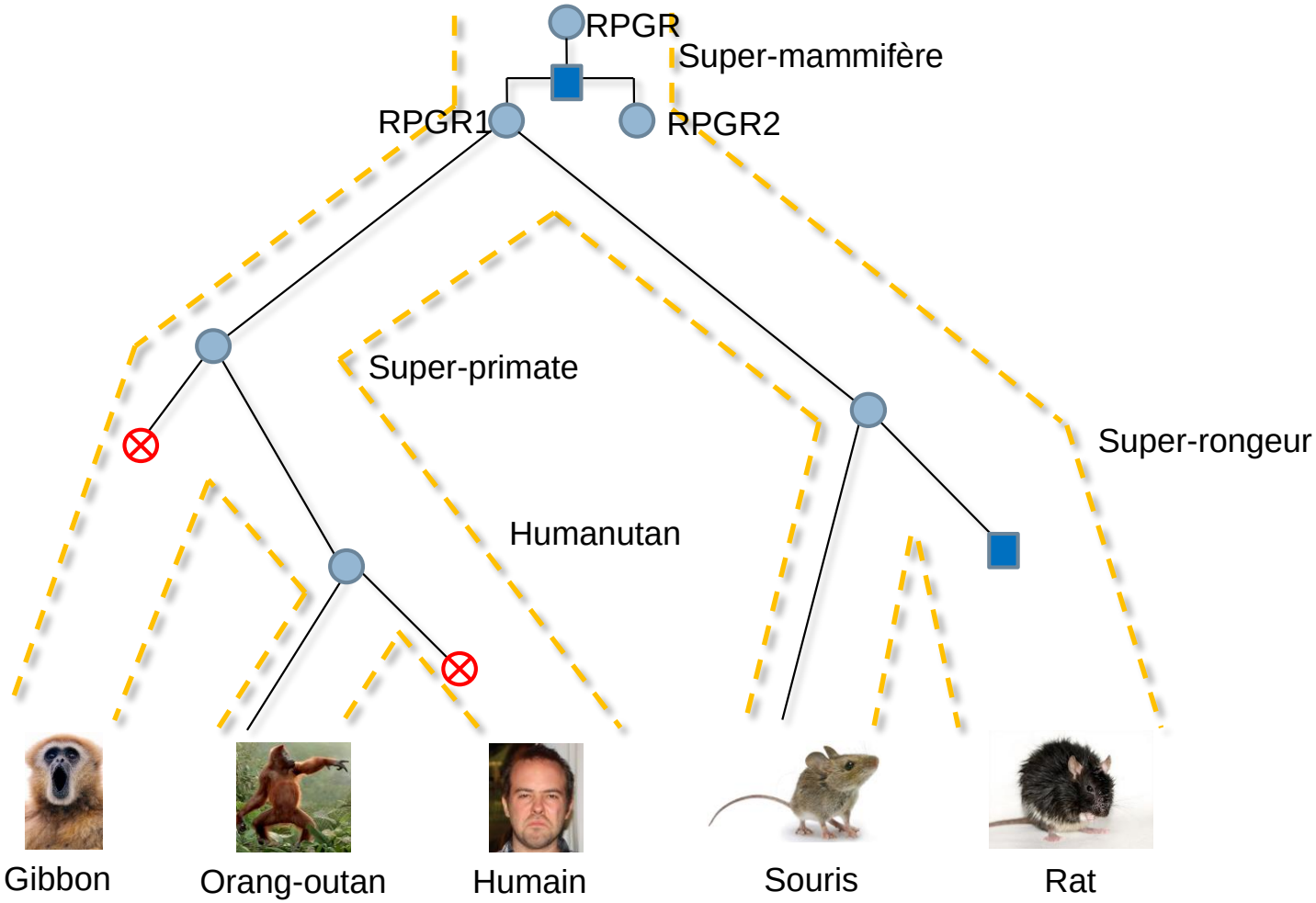
Arbre de gènes



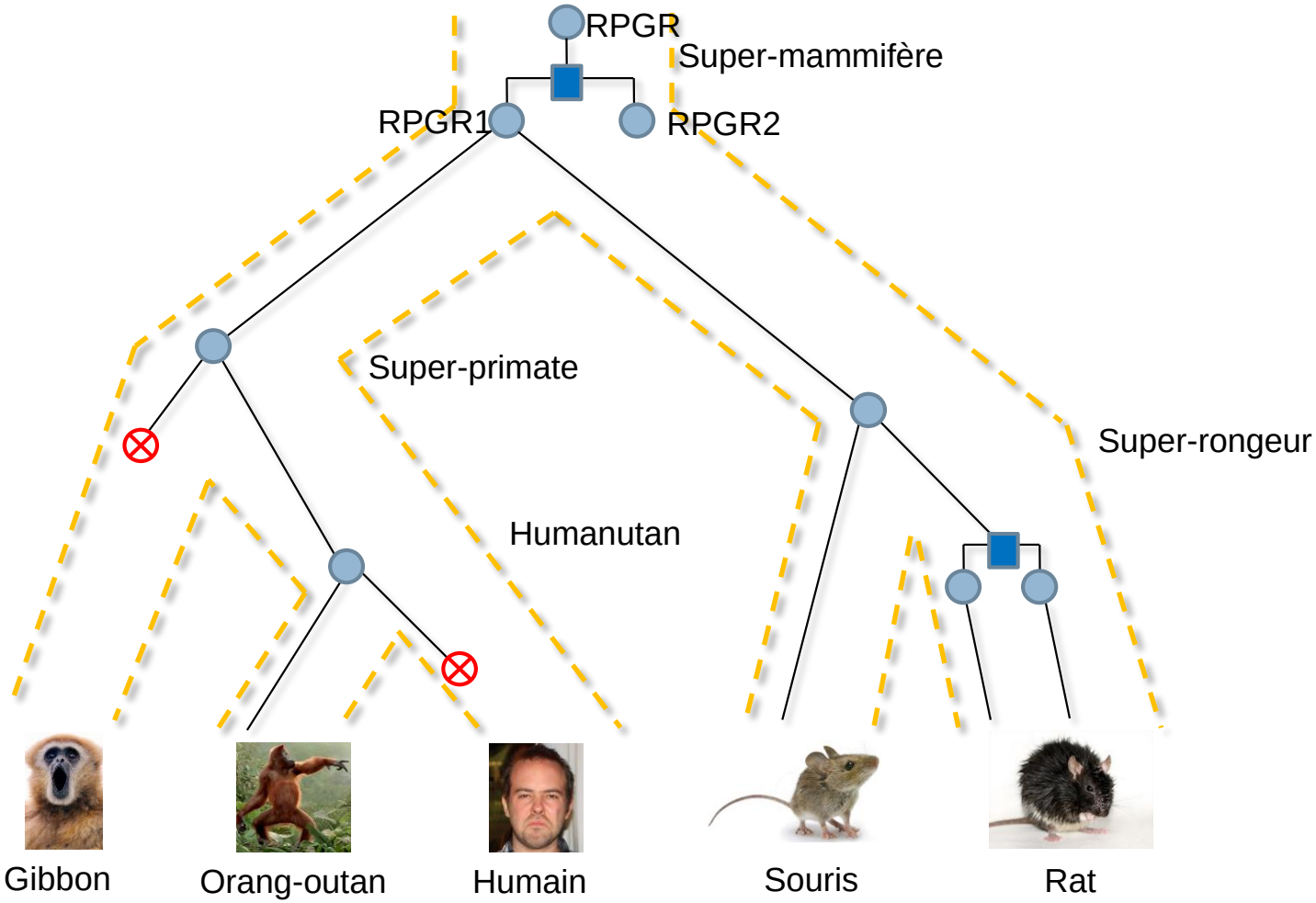
Arbre de gènes



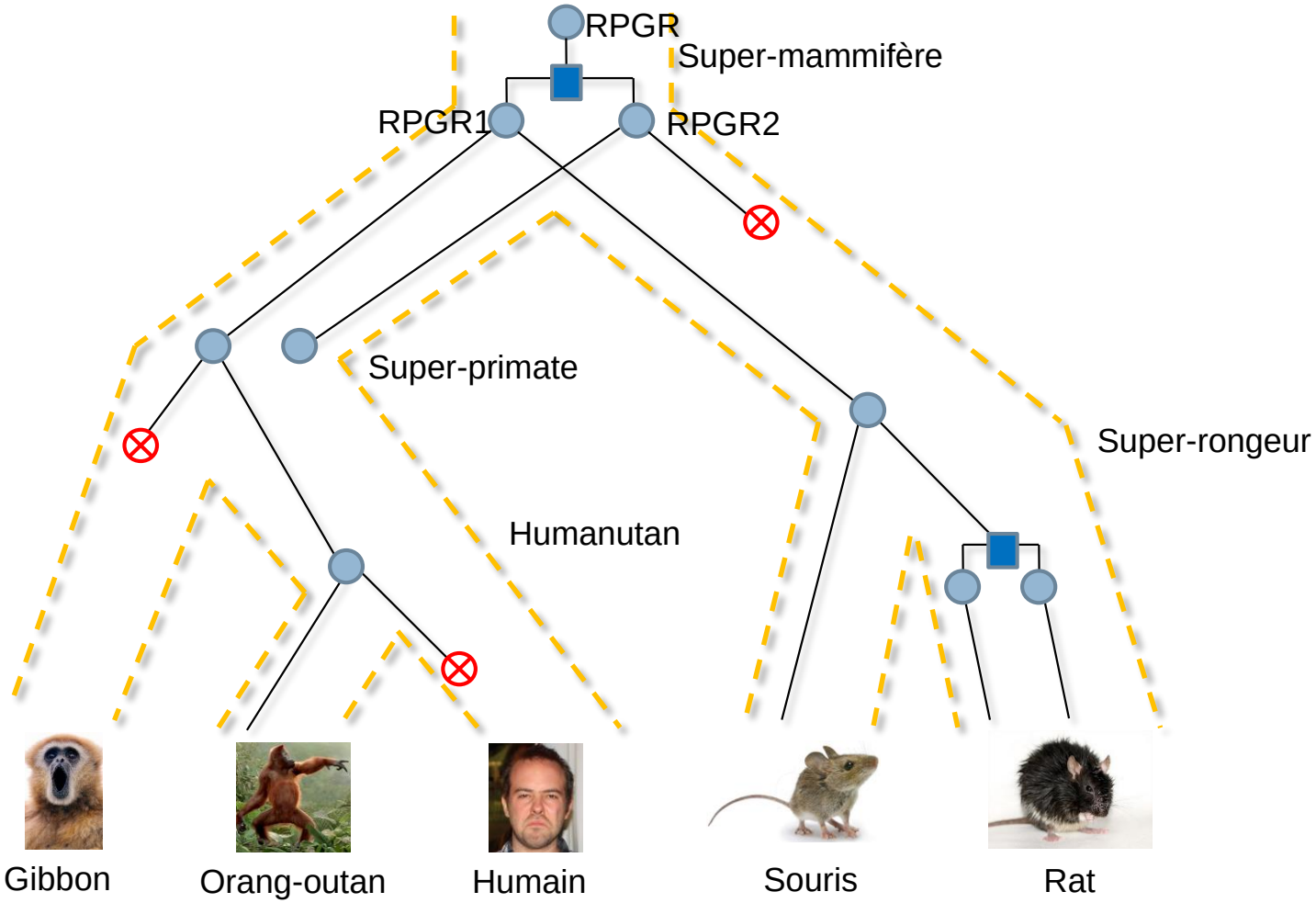
Arbre de gènes



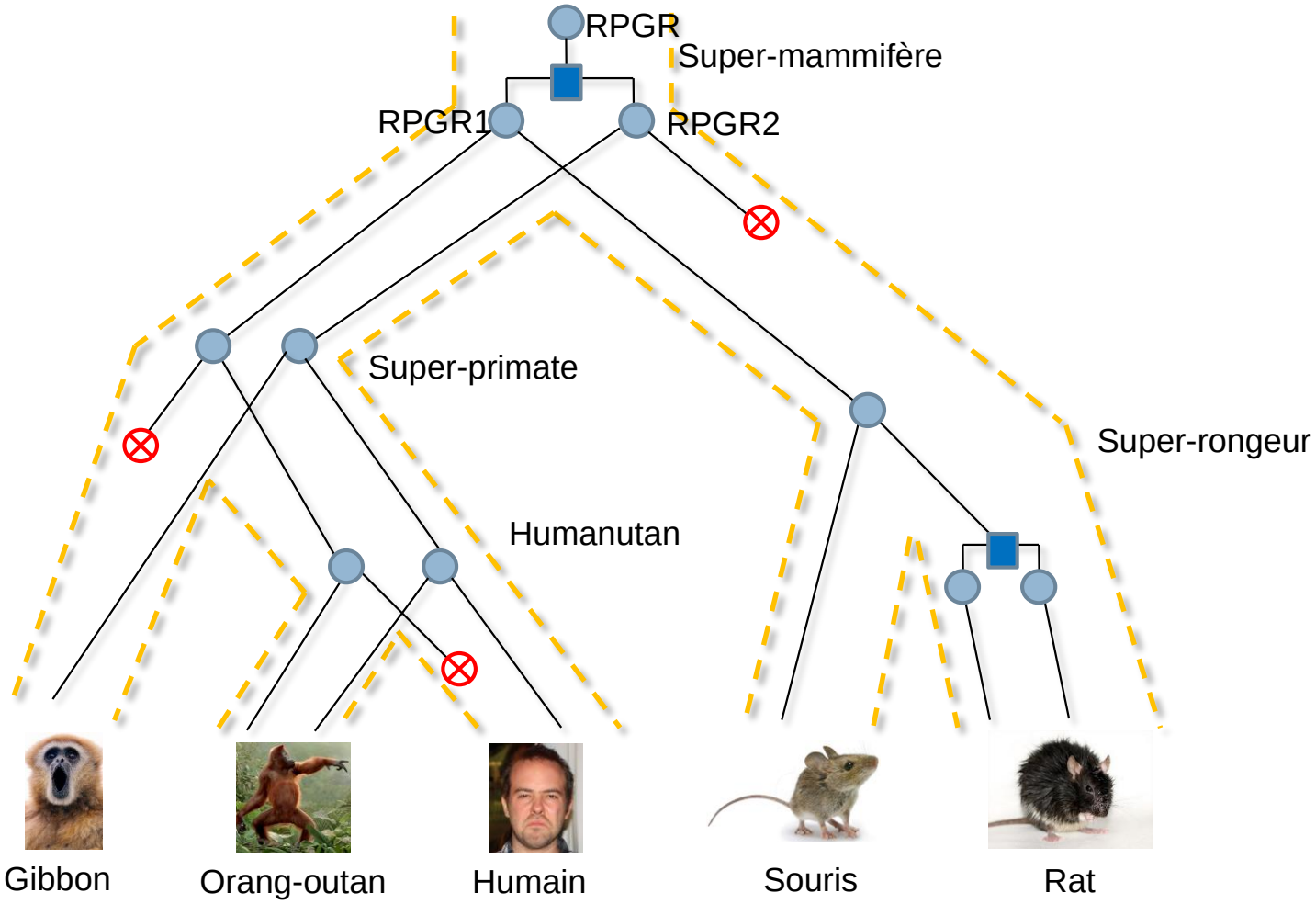
Arbre de gènes



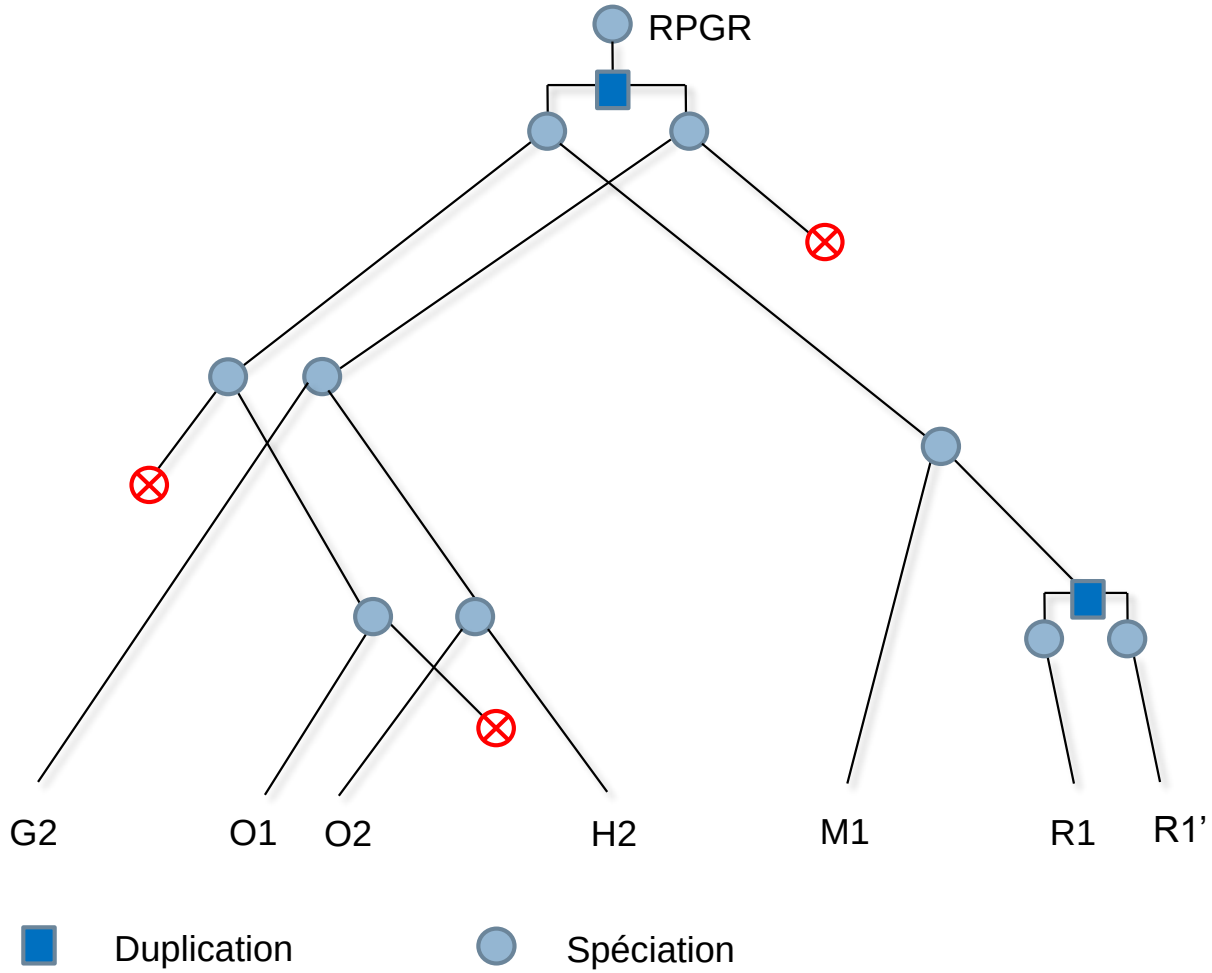
Arbre de gènes



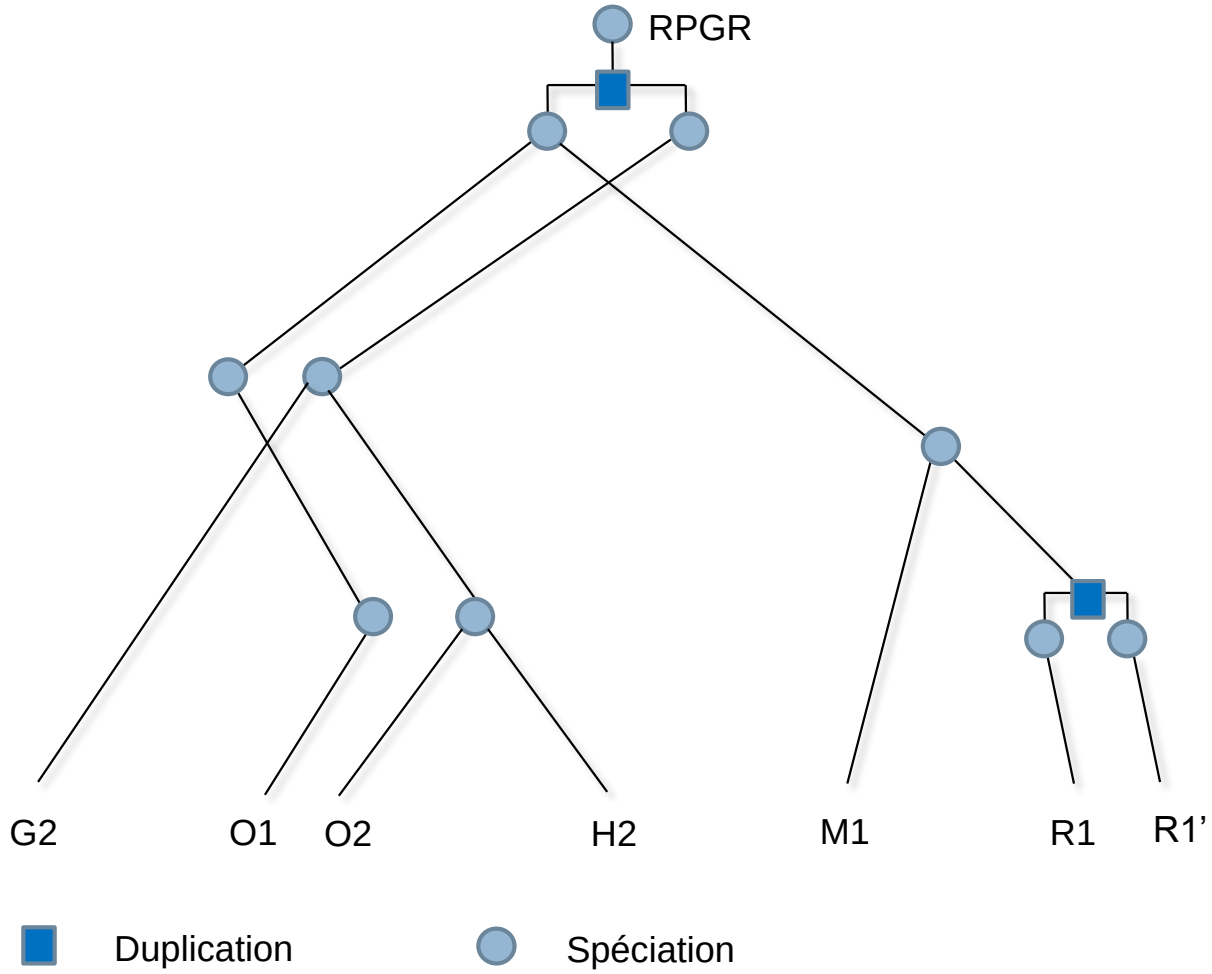
Arbre de gènes



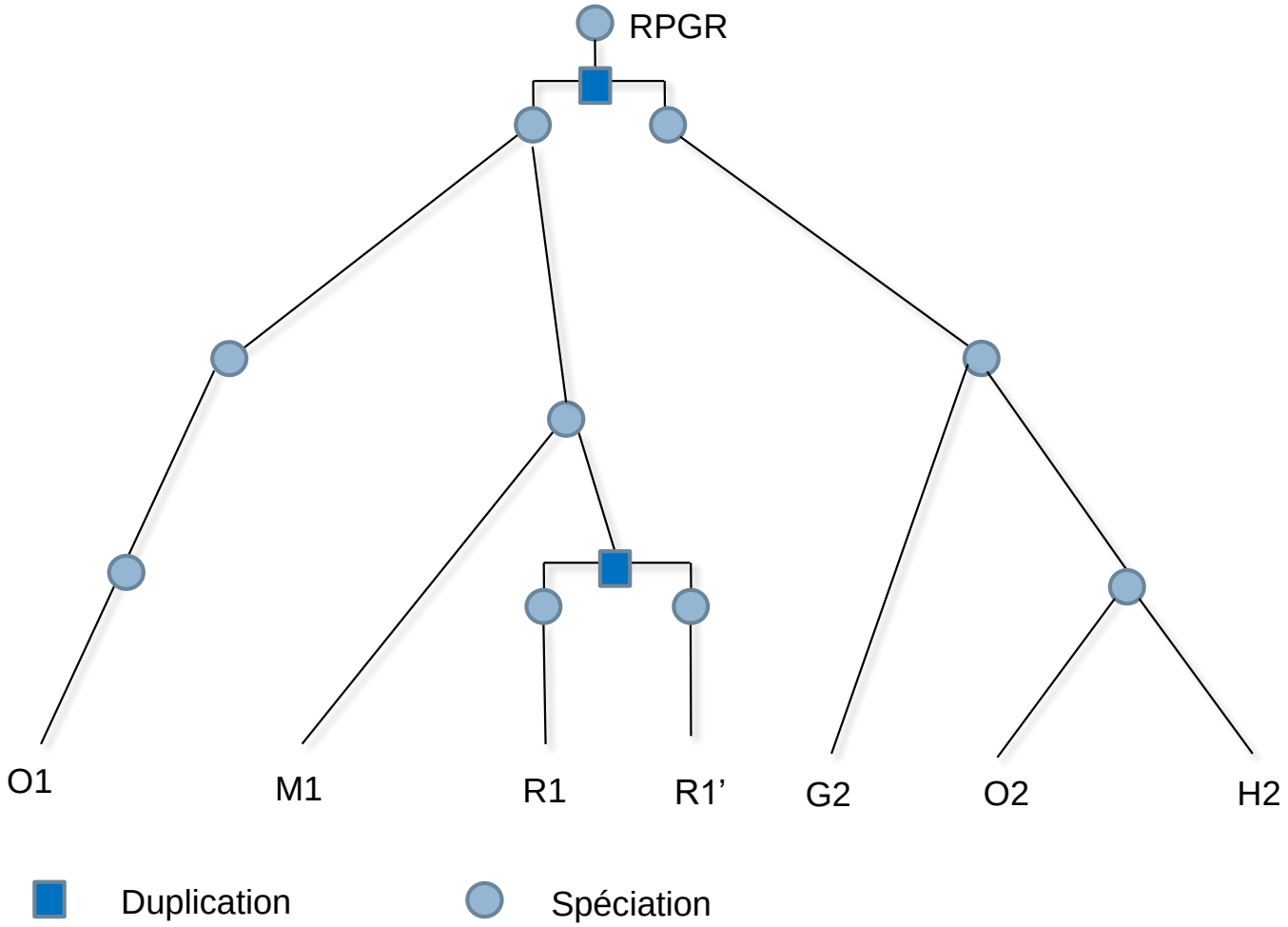
Arbre de gènes



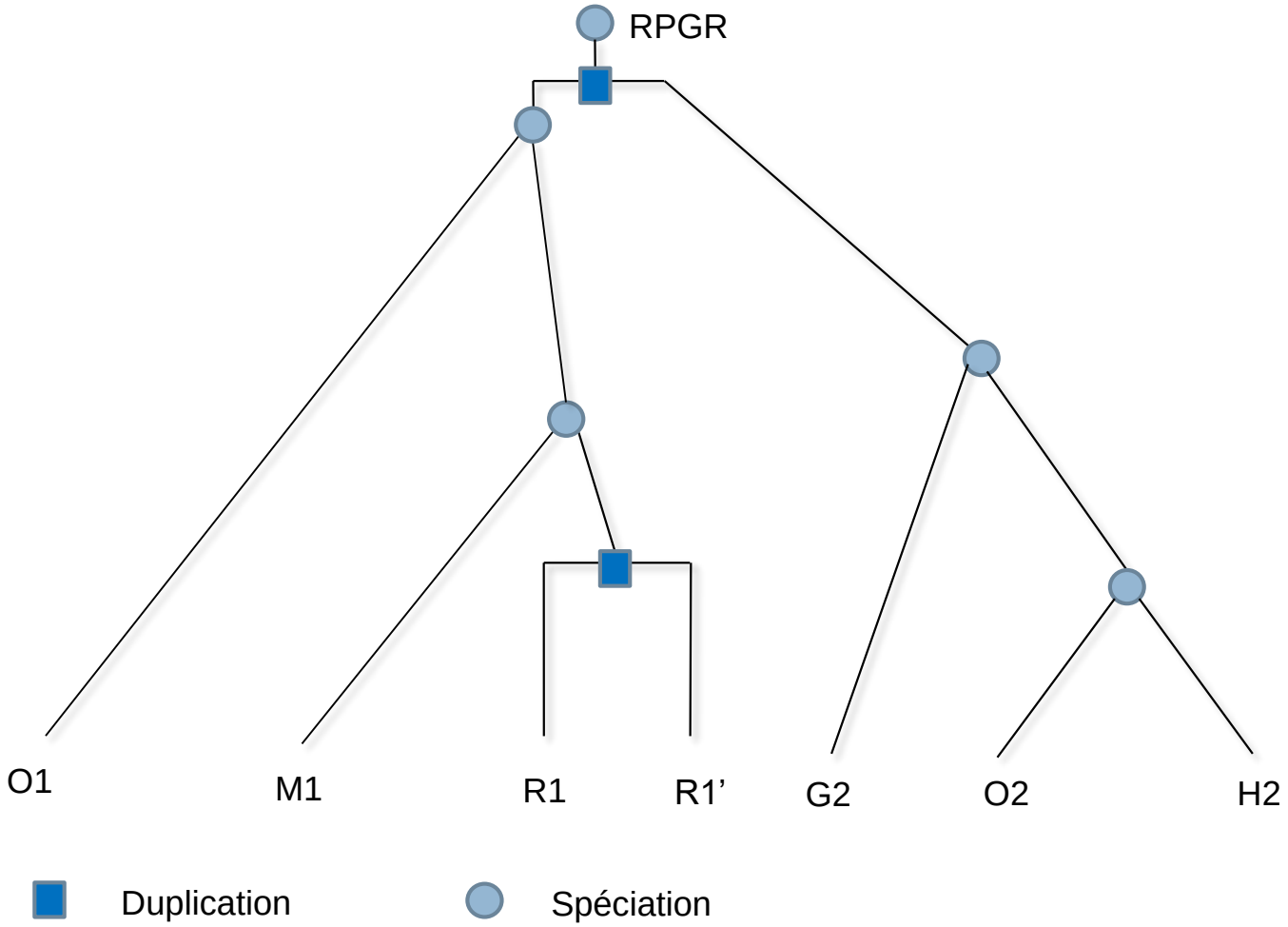
Arbre de gènes



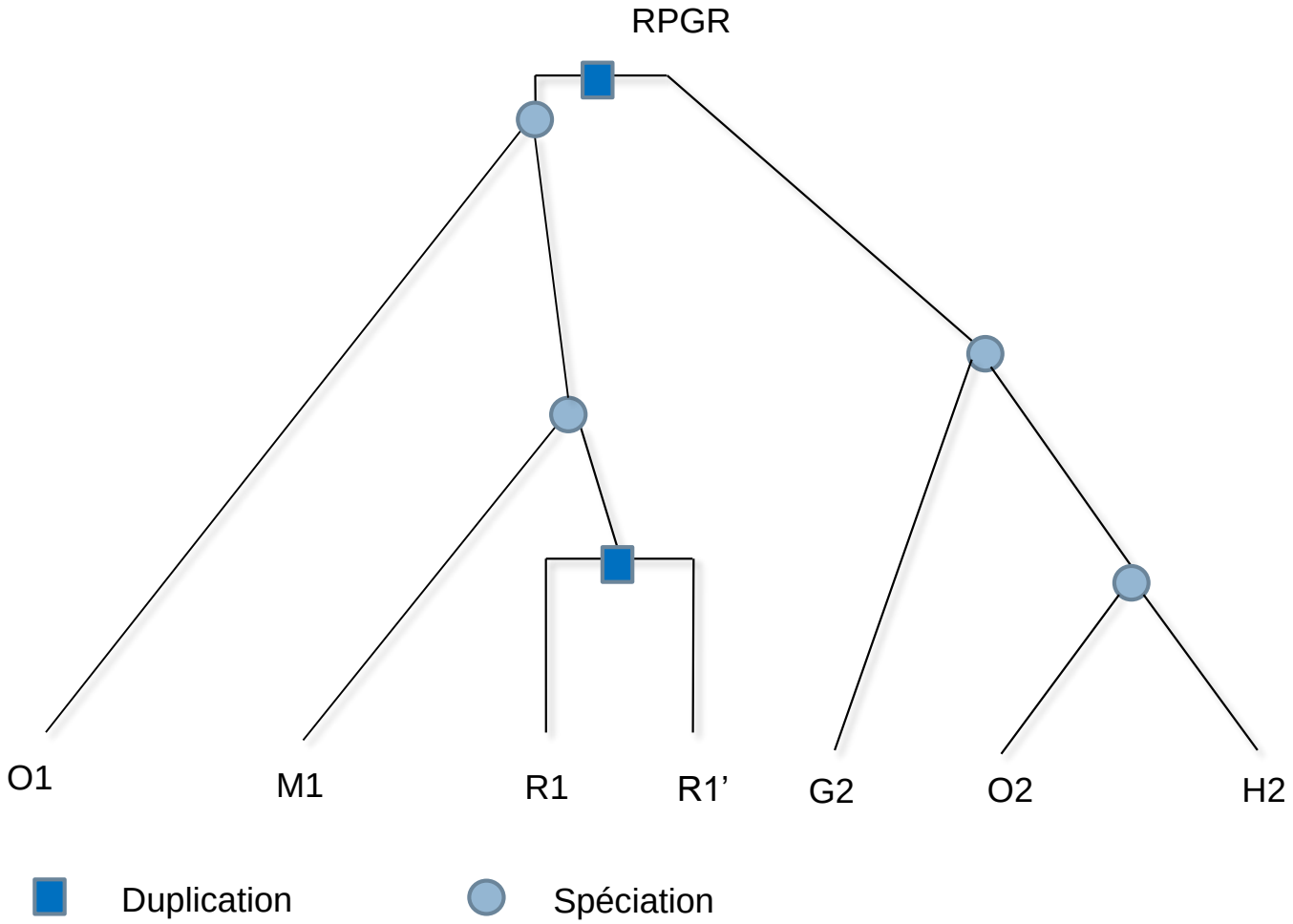
Arbre de gènes



Arbre de gènes



Arbre de gènes



Inférence d'arbres de gènes

Ce qu'on a en réalité:

O1	M1	R1	R1’	G2	O2	H2
AGTTG	ACGTT	AGGTT	AGGTT	CGGAT	CCGAT	CGGAT

Inférence d'arbres de gènes

O1
AGTTG

M1
ACGTT

R1
AGGTT

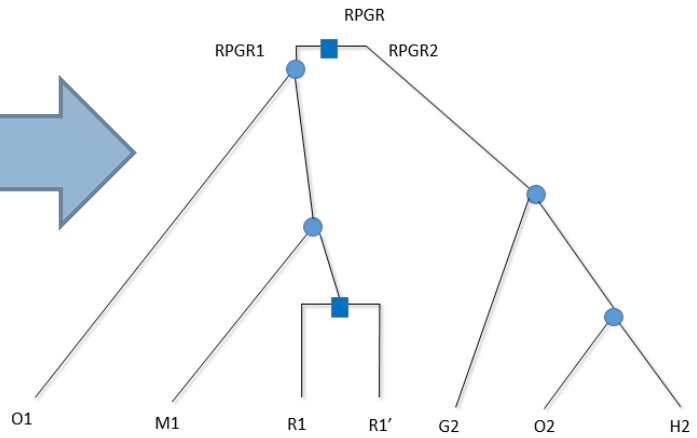
R1’
AGGTT

G2
CGGAT

O2
CCGAT

H2
CGGAT

Algorithme de reconstruction phylogénétique (e.g. maximum parcimonie, maximum vraisemblance, inférence Bayésienne)



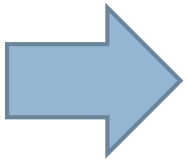
Mauvaise différentiation des séquences

g1
CCCCC

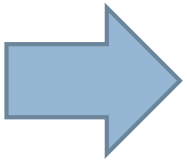
g2
CCCCC

g3
CCCCC

g4
CCCCC



Algorithme de reconstruction phylogénétique (e.g. maximum parcimonie, maximum vraisemblance, inférence Bayésienne)



???

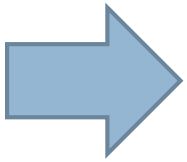
Mauvaise différentiation des séquences

g1
AAAA

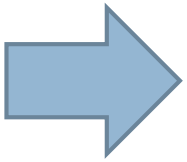
g2
CCCC

g3
GGGG

g4
TTTT



Algorithme de reconstruction phylogénétique (e.g. maximum parcimonie, maximum vraisemblance, inférence Bayésienne)



???

Mauvaise différentiation des séquences

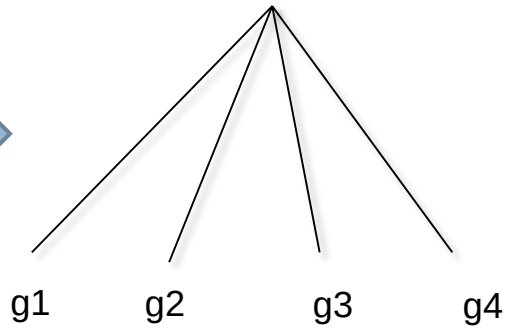
g1
AAAA

g2
CCCC

g3
GGGG

g4
TTTT

Algorithme de reconstruction phylogénétique (e.g. maximum parcimonie, maximum vraisemblance, inférence Bayésienne)



Mauvaise différentiation des séquences

g1
AAAA

g2
CCCC

g3
GGGG

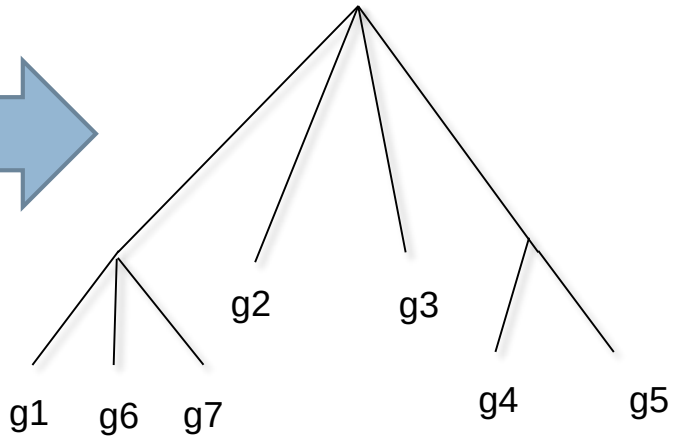
g4
TTTT

g5
TTTT

g6
AAAA

g7
AAAA

Algorithme de reconstruction phylogénétique (e.g. maximum parcimonie, maximum vraisemblance, inférence Bayésienne)



Branches mal supportées

g1
ACGT

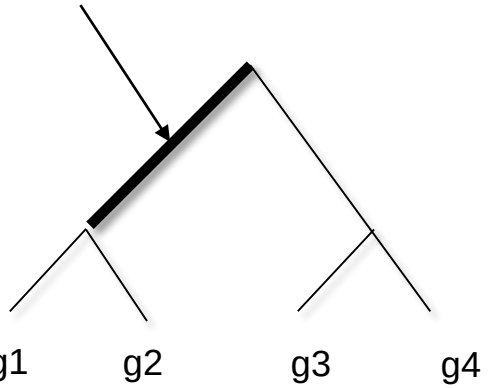
g2
CGTA

g3
CGGT

g4
TGCG

Algorithme de reconstruction phylogénétique (e.g. maximum parcimonie, maximum vraisemblance, inférence Bayésienne)

Revient dans peu d'arbres bootstrap (e.g. <70%)



Branches mal supportées

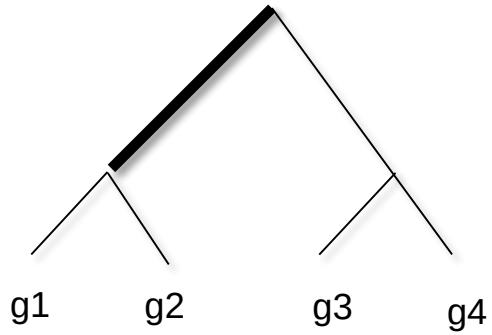
g1
ACGT

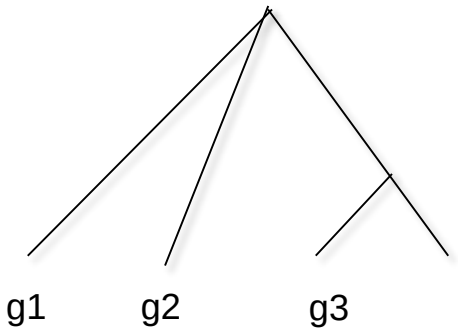
g2
CGTA

g3
CGGT

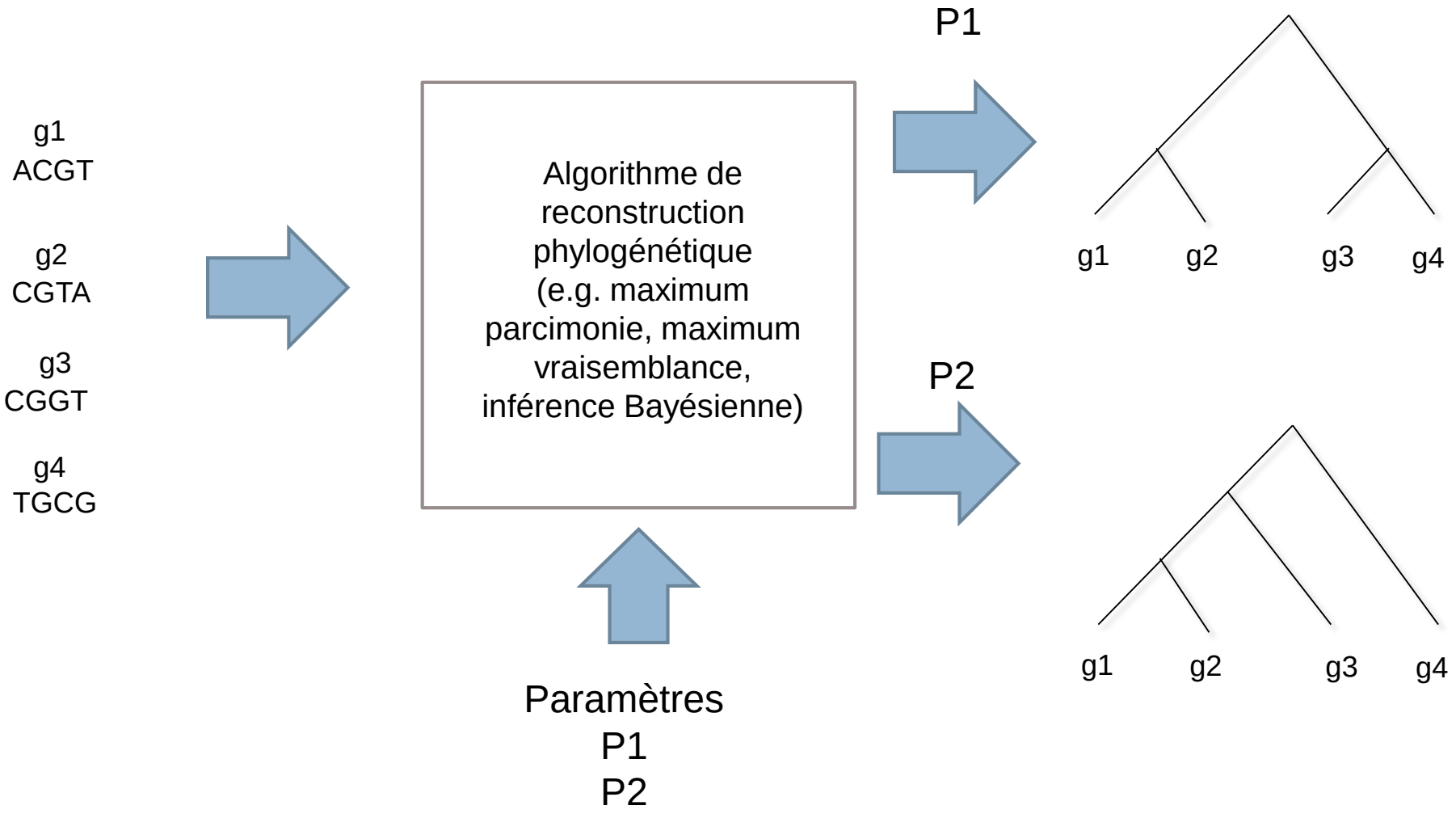
g4
TGCG

Algorithme de reconstruction phylogénétique (e.g. maximum parcimonie, maximum vraisemblance, inférence Bayésienne)





Paramétrisation des algorithmes



Arbres multiples

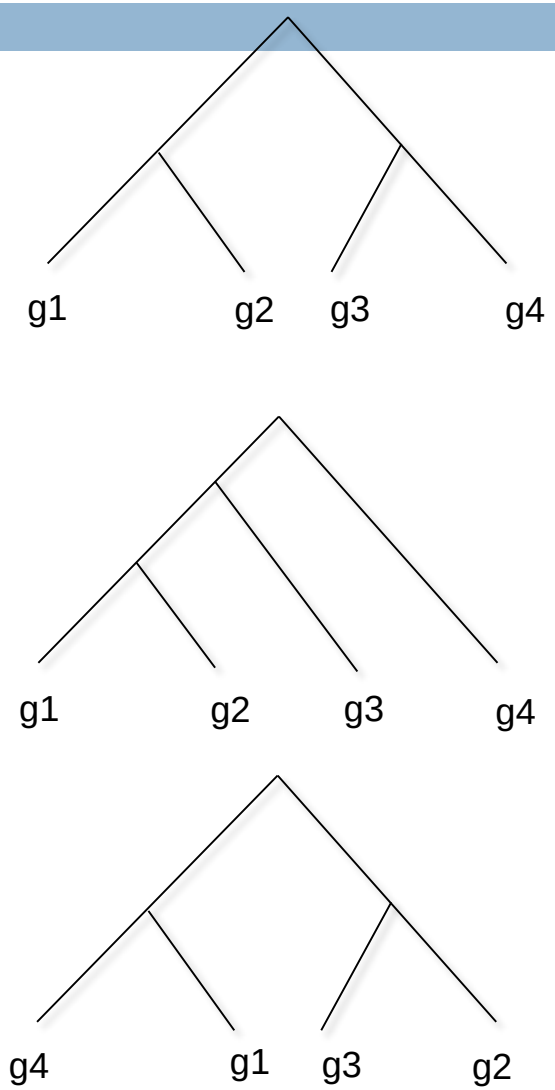
g1
ACGT

g2
CGTA

g3
CGGT

g4
TGCG

Algorithme de reconstruction phylogénétique (e.g. maximum parcimonie, maximum vraisemblance, inférence Bayésienne)



Quelques problèmes

- Arbres partiellement résolus
  - Les arbres contiennent des **polytomies** (des nœuds non-binaires)
- Topologies erronés
- Plusieurs arbres
  - Candidats multiples
  - Petits sous-arbres à combiner

Quelques problèmes

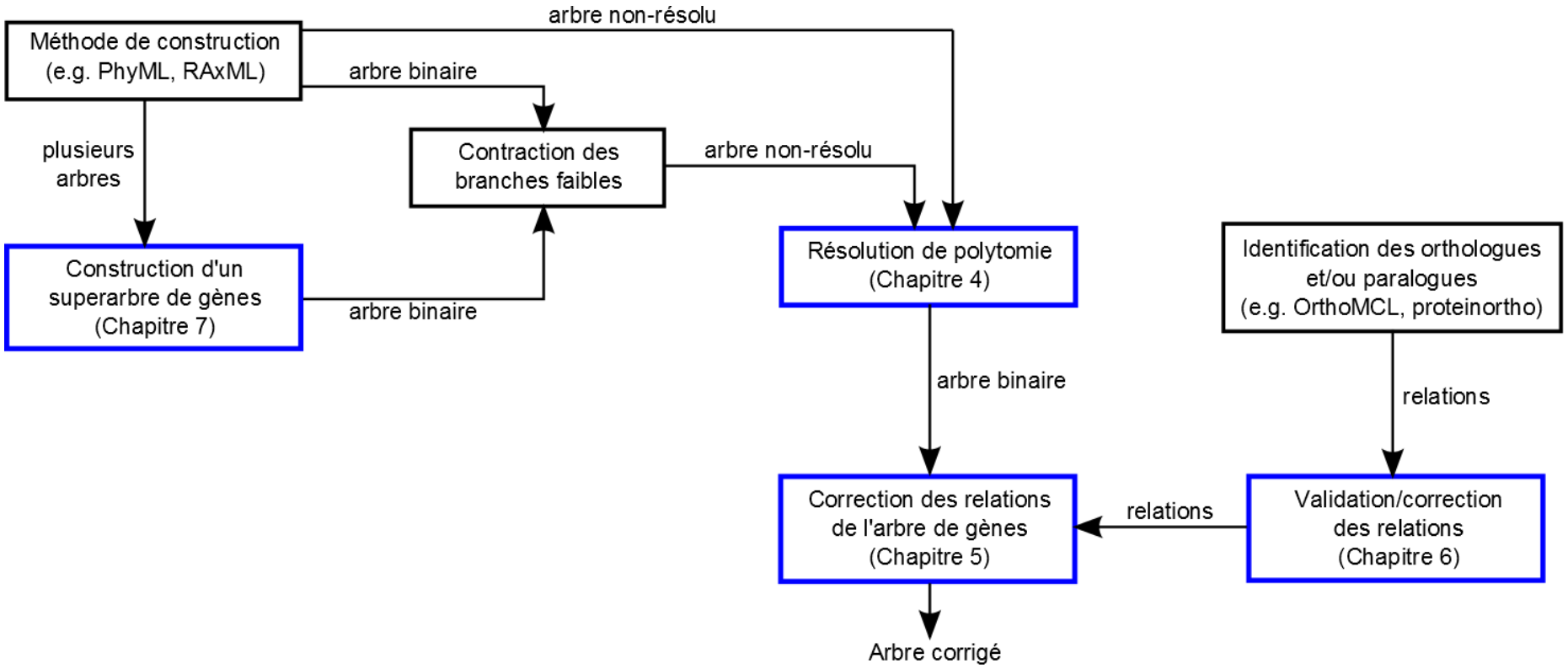
- La quasi-totalité des méthodes de reconstruction phylogénétique sont génériques (i.e. non-spécifiques aux arbres de gènes) et ont souvent été développées pour la reconstruction d'**arbres d'espèces**.

Quelques problèmes

- La quasi-totalité des méthodes de reconstruction phylogénétique sont génériques (i.e. non-spécifiques aux arbres de gènes) et ont souvent été développées pour la reconstruction d'**arbres d'espèces**.

- On veut utiliser les propriétés spécifiques aux gènes pour corriger les arbres de gènes.
  - Arbre d'espèces
  - Réconciliation

Pipeline de correction d'arbres



Articles présentés

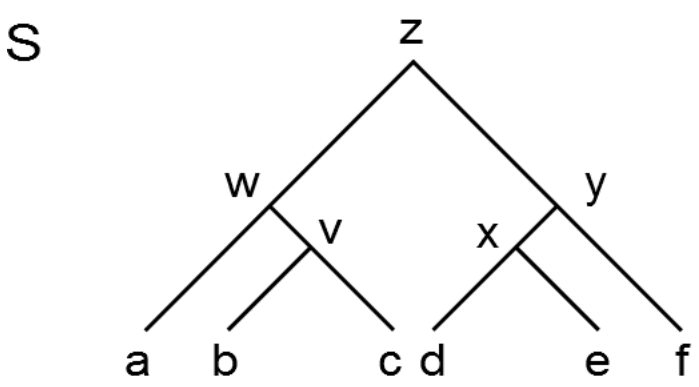
An optimal reconciliation algorithm for gene trees with polytomies
M Lafond, KM Swenson, N El-Mabrouk
International Workshop on Algorithms in Bioinformatics, 2012

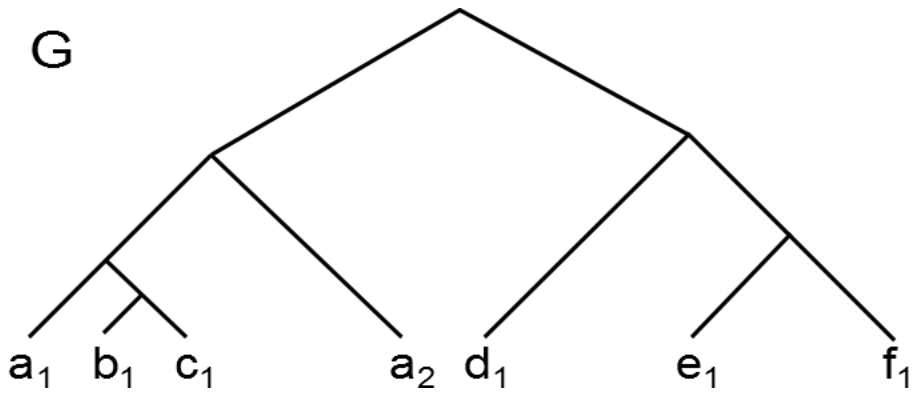
Gene tree correction guided by orthology
M Lafond, M Semeria, KM Swenson, E Tannier, N El-Mabrouk
BMC bioinformatics (2013)

Orthology and paralogy constraints: satisfiability and consistency
M Lafond, N El-Mabrouk
BMC genomics 15 (2014)

Reconstructing a SuperGeneTree minimizing reconciliation
M Lafond, A Ouangraoua, N El-Mabrouk
BMC bioinformatics 16 (2015)

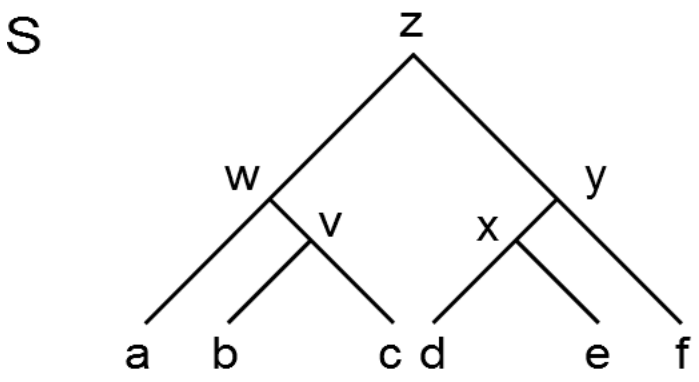
Réconciliation

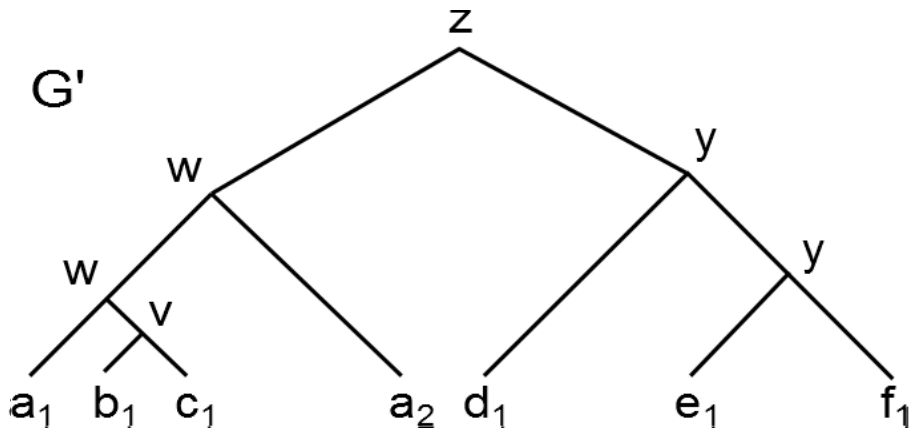




Réconciliation

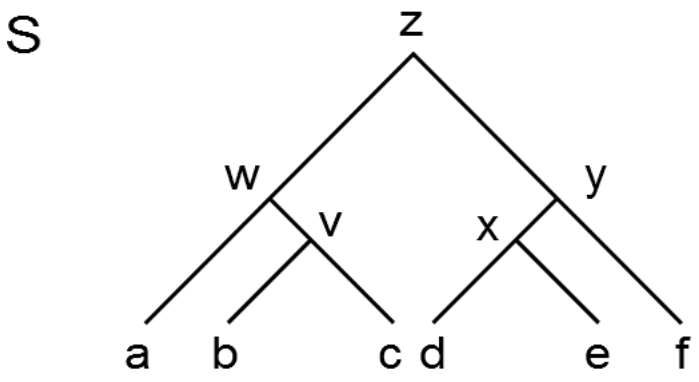
- 1) Mapping gène-espèce (lca mapping)

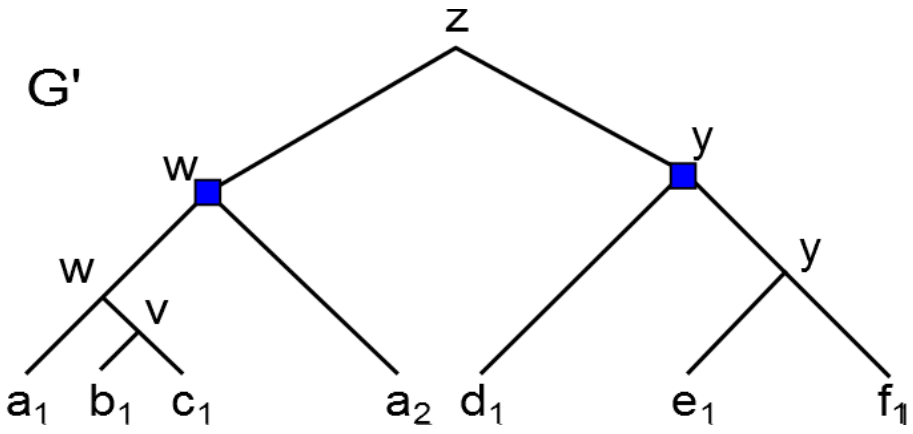




Réconciliation

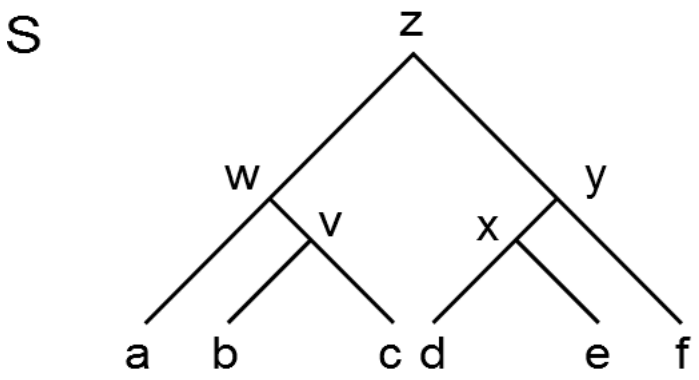
- 1) Mapping gène-espèce (lca mapping)
- 2) Trouver les duplications
  - Nœuds qui ont le même mapping que leur enfant

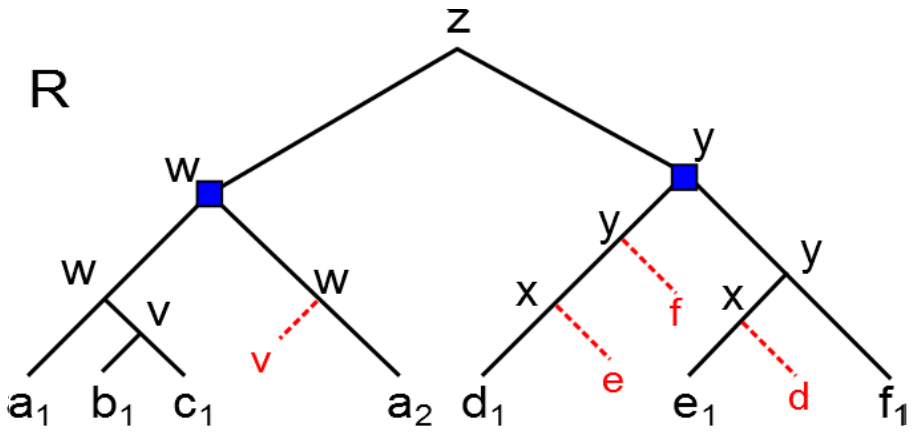




Réconciliation

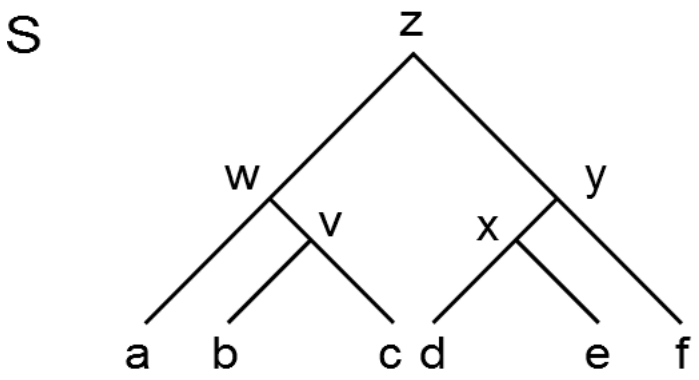
- 1) Mapping gène-espèce (lca mapping)
- 2) Trouver les duplications
  - Nœuds qui ont le même mapping que leur enfant
- 3) Trouver les pertes
  - Compléter les branches avec les espèces manquantes

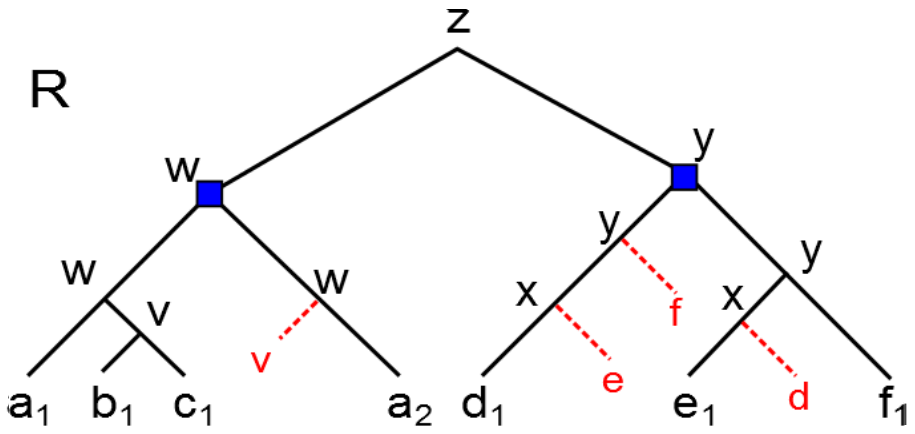




Réconciliation

- Coût d'une réconciliation:
  - Typiquement, # de duplications + # de pertes
  - Parfois, # de duplications seulement





Méthodes de construction/correction

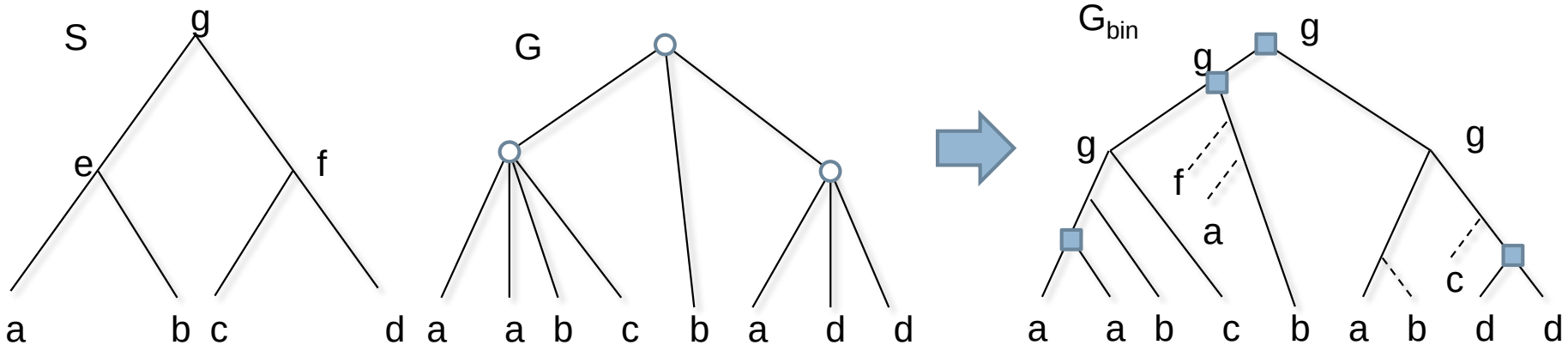
- Construction "de novo"
  - SYNERGY (2007), Prime-GSR (2009), GIGA (2010), SPIMAP (2011), PHYLDOG (2013)
- Construction par amalgamation
  - AngST (2011), TERA (2014)
- Correction par résolution de polytomies
  - NOTUNG (2006), Chang & Eulenstein (2006), Zheng & Zhang (2014)
- Correction par exploration du voisinage
  - NOTUNG (2000), Eulenstein (2012), TreeFix (2012)
- À l'exception de PHYLDOG, toutes ces méthodes nécessitent un arbre d'espèces en entrée, et appliquent une forme de réconciliation.

Résolution de polytomies

An optimal reconciliation algorithm for gene trees with polytomies
M Lafond, KM Swenson, N El-Mabrouk
International Workshop on Algorithms in Bioinformatics, 2012

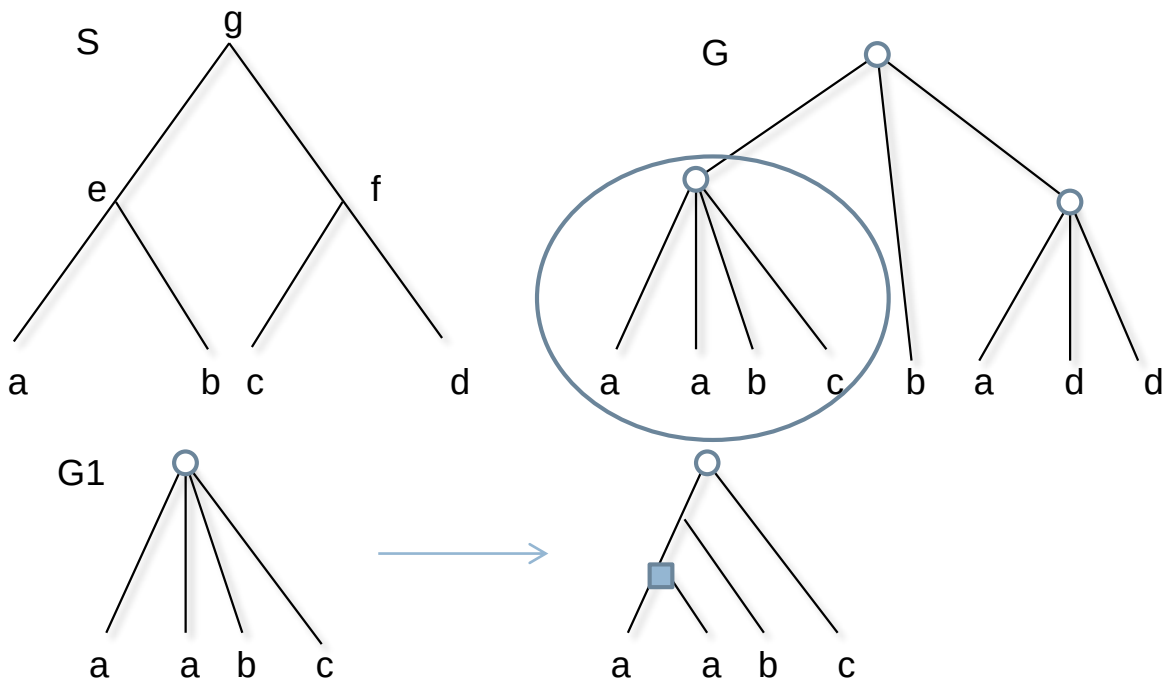
Résolution de polytomies

- **Entrée**: une arbre d'espèces S binaire, et un arbre de gènes G non-binaire
- **Sortie**: une "binarisation" de G qui minimise le coût de réconciliation avec S



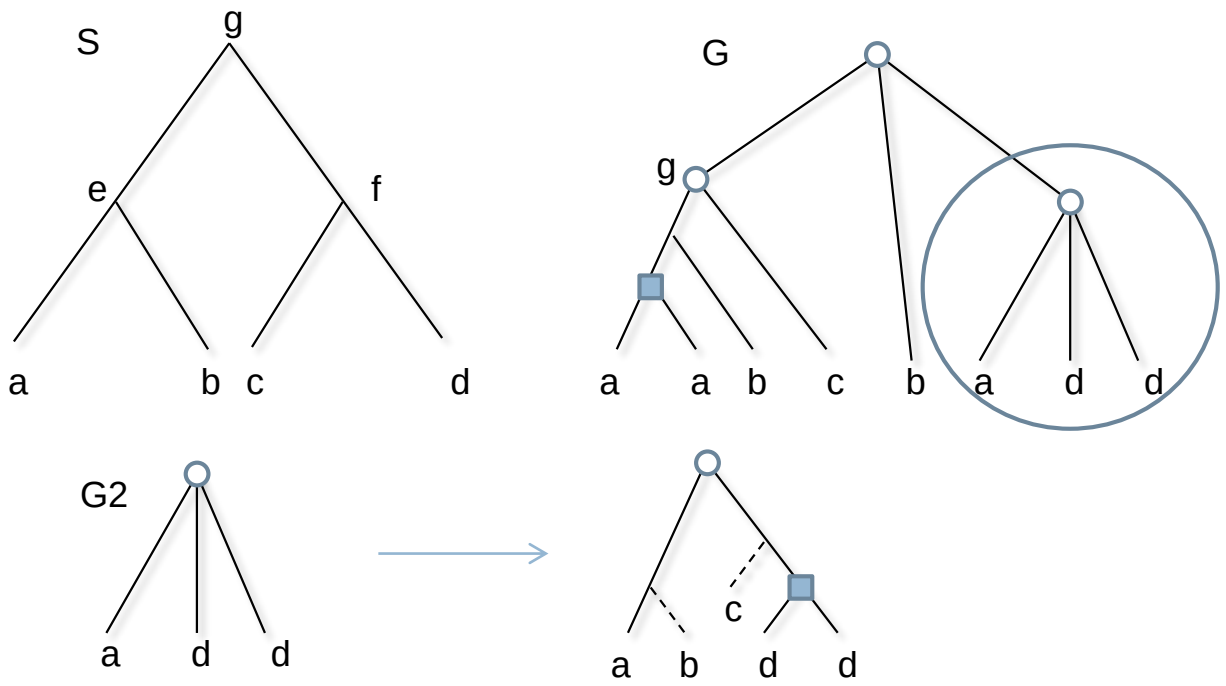
Résolution de polytomies

- Chaque polytomie peut être résolue indépendamment *(Chang & Eulenstein, 2006)*
  - Algorithme O(|G||S|^2) par polytomie



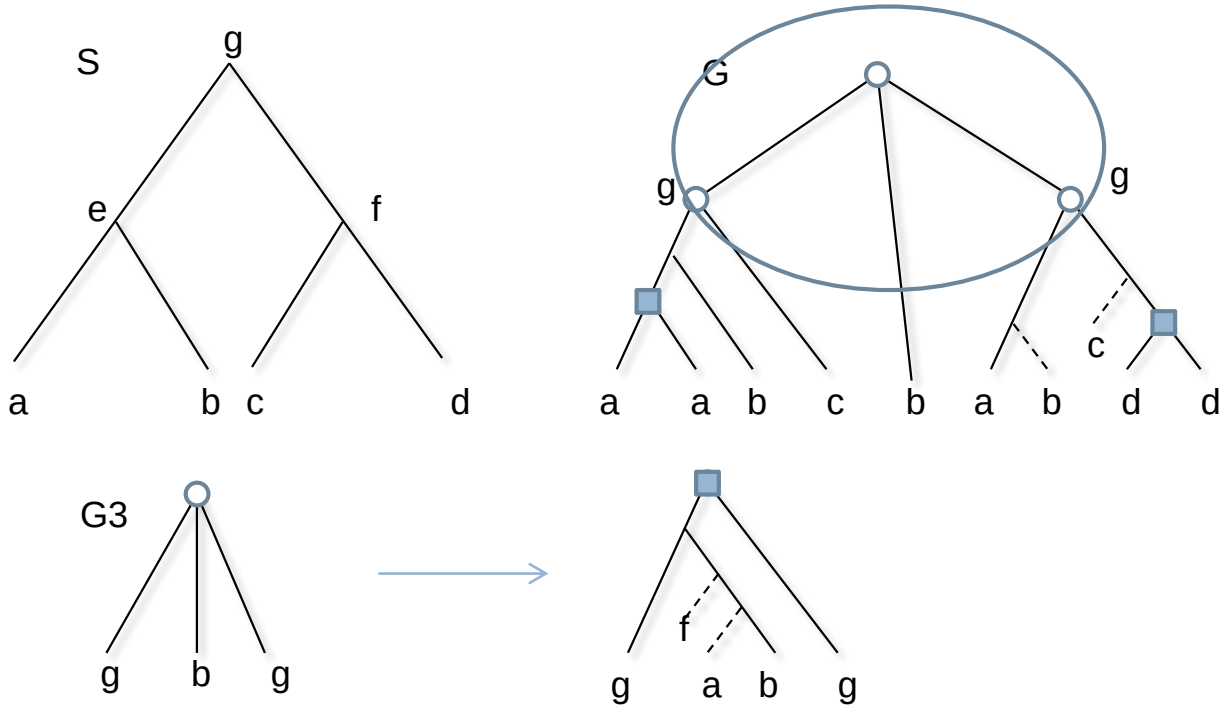
Résolution de polytomies

- Chaque polytomie peut être résolue indépendamment *(Chang & Eulenstein, 2006)*
  - Algorithme O(|G||S|^2) par polytomie



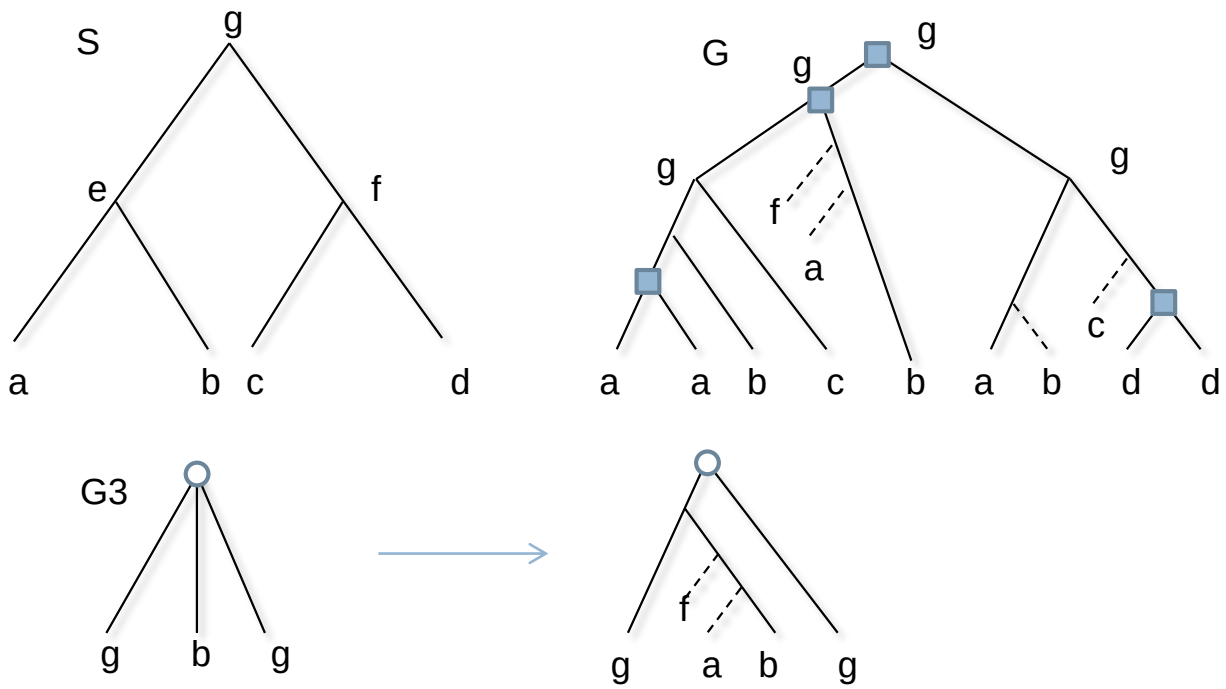
Résolution de polytomies

- Chaque polytomie peut être résolue indépendamment *(Chang & Eulenstein, 2006)*
  - Algorithme O(|G||S|^2) par polytomie



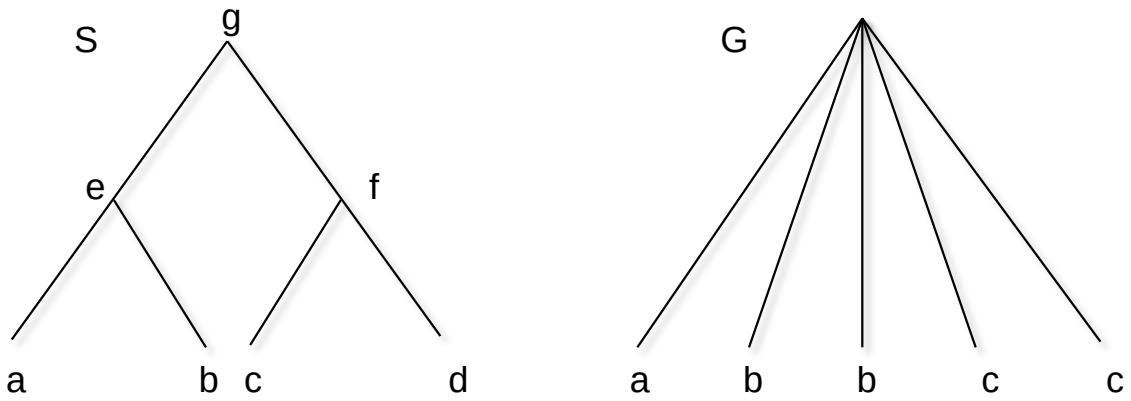
Résolution de polytomies

- Chaque polytomie peut être résolue indépendamment *(Chang & Eulenstein, 2006)*
  - Algorithme O(|G||S|^2) par polytomie



Le vrai problème

- On réduit le problème à la résolution d'une seule polytomie.

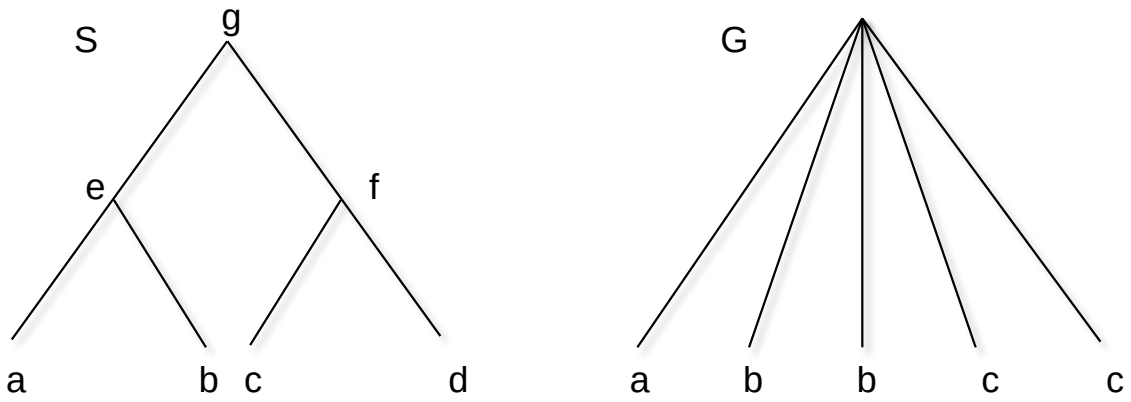


Travaux antérieurs

- Notung (2006)
  - $O(|S||G|^3)$ pour résoudre tout un arbre. Supporte coût différents dup et pertes.
- Chang & Eulenstein (2006)
  - $O(|S||P|^2)$ par polytomie P, $O(|S||G|^2)$ pour tout un arbre G
- **Lafond, Swenson & El-Mabrouk (2012)**
  - $O(|S| + |P|)$ par polytomie P, $O(|S||G|)$ pour tout l'arbre
- Zhang & Zheng (2014)
  - $O(|S| +|G|)$ pour résoudre tout un arbre G
- Lafond, Noutahi & El-Mabrouk (2016)
  - $O(|S| + |G|)$ pour tout l'arbre G, $O(|G|^2)$ avec coûts différents dup et pertes.

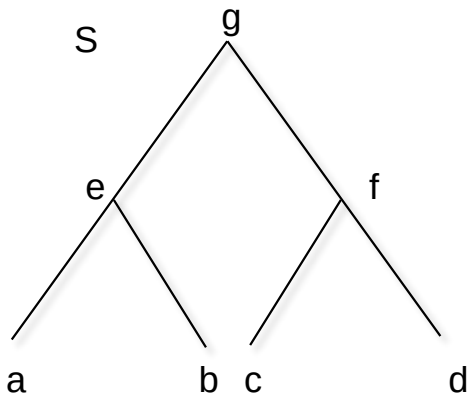
Programmation dynamique

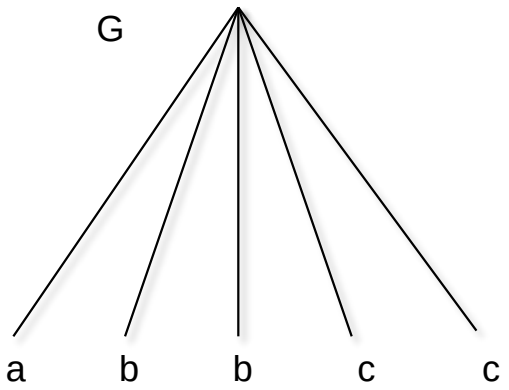
- On réduit le problème à la résolution d'une seule polytomie.
- Algorithme en temps O(|S|) par programmation dynamique.



Programmation dynamique

- Pour une espèce s et un entier k:
- M(s, k): si on se restreint aux gènes des descendant de s, quel est le coût minimum pour avoir une forêt de k arbres tous enracinés en un gène de s.



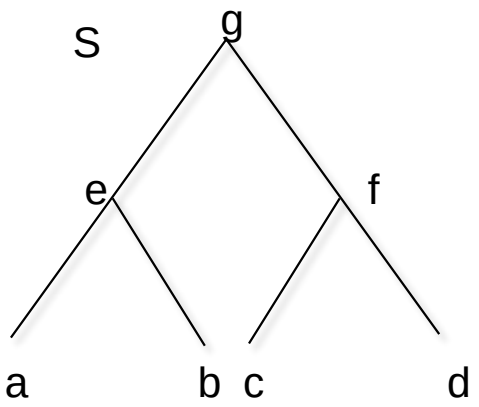


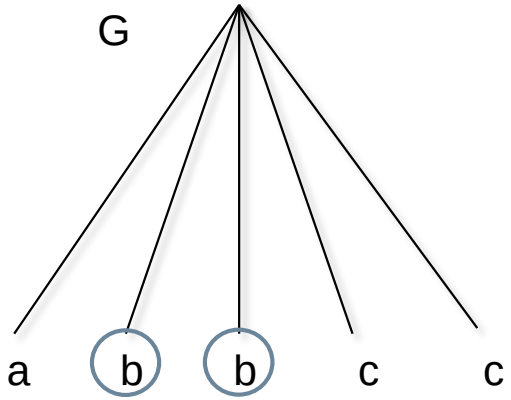
Programmation dynamique

- Pour une espèce s et un entier k:
- M(s, k): si on se restreint aux gènes des descendant de s, quel est le coût minimum pour avoir une forêt de k arbres tous enracinés en un gène de s.
  - Chaque gène retenu devient un sous-arbre
  - On peut connecter deux feuilles/sous-arbres
  - On doit utiliser tous les gènes dans notre restriction
  - On peut insérer des pertes, ce qui créé de nouveaux sous-arbres

Programmation dynamique

- Exemple: M(b, 2) = 0



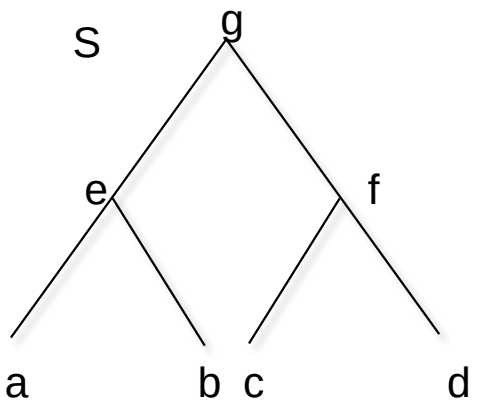


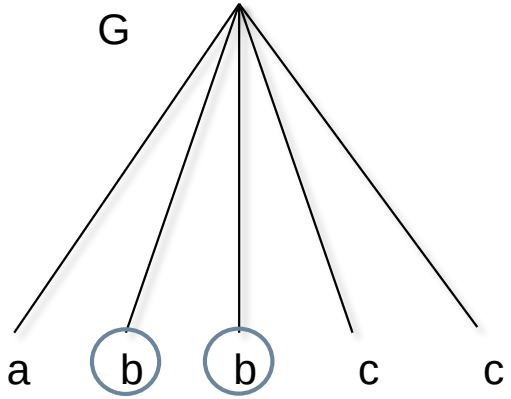
Restriction à b
(et ses descendants)

Forêt de 2 arbres

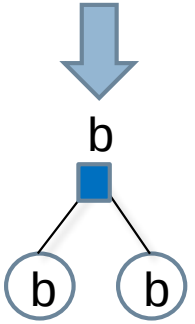
Programmation dynamique

- Exemple: M(b, 1) = 1





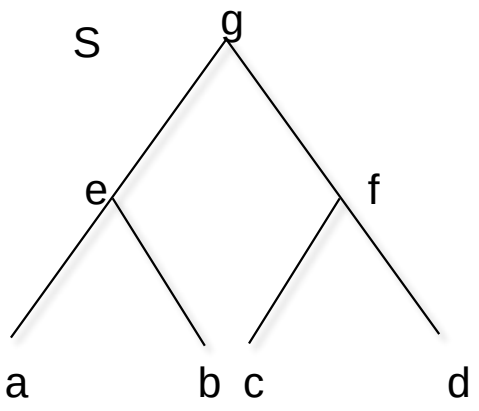
Restriction à b (et ses descendants)

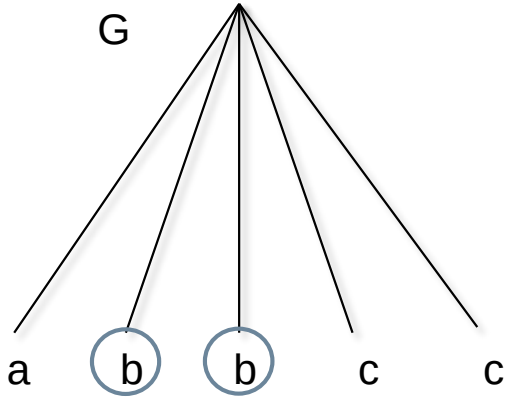


Forêt de 1 arbre

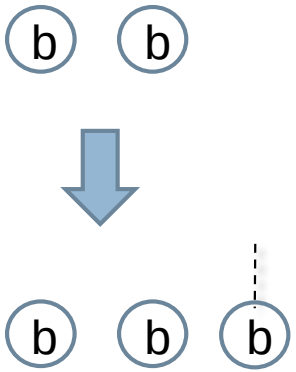
Programmation dynamique

- Exemple: M(b, 3) = 1





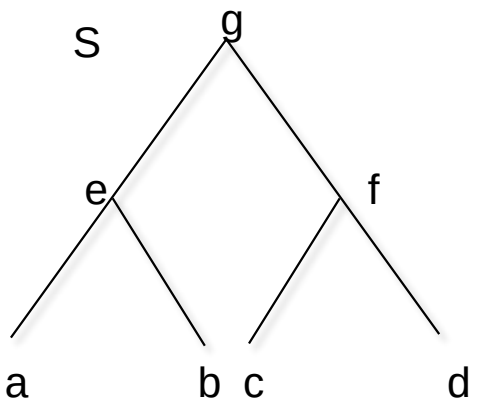
Restriction à b (et ses descendants)

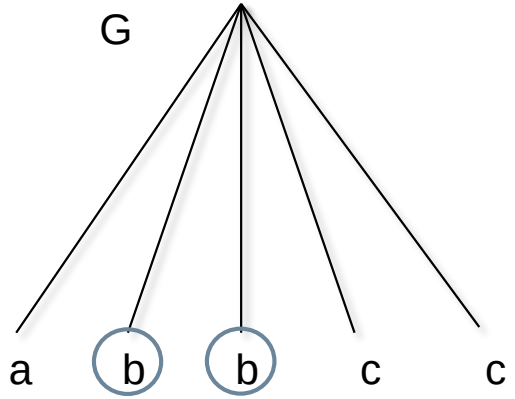


Forêt de 3 arbres (dont 1 perte)

Programmation dynamique

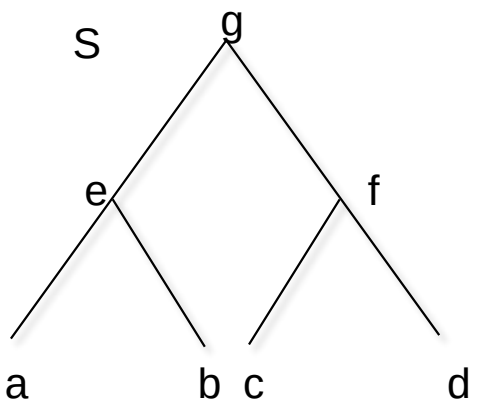
- En général, pour une feuille s présente #(s) fois dans la polytomie,
- M(s, k) = |k - #(s)|

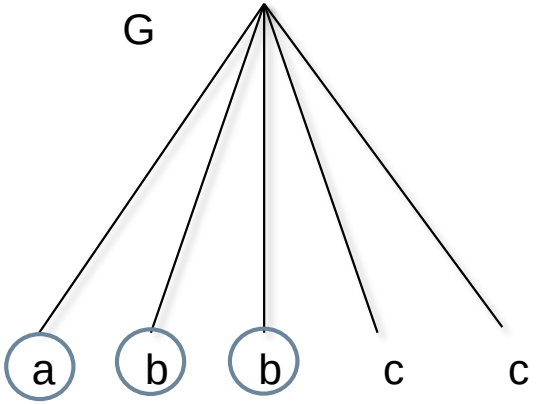




Programmation dynamique

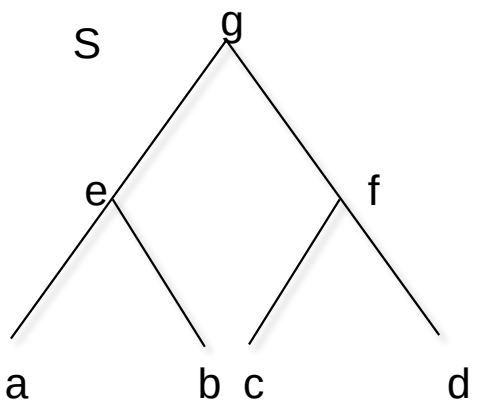
- M(e, 1) = ?

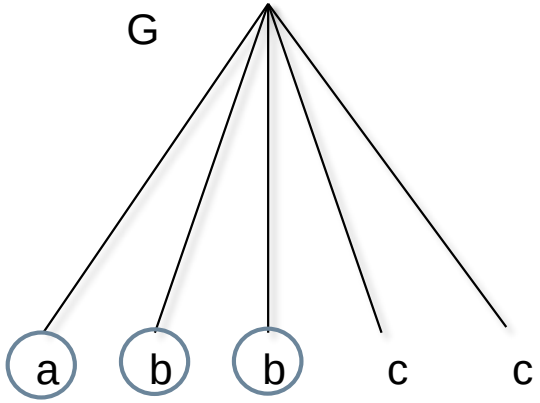


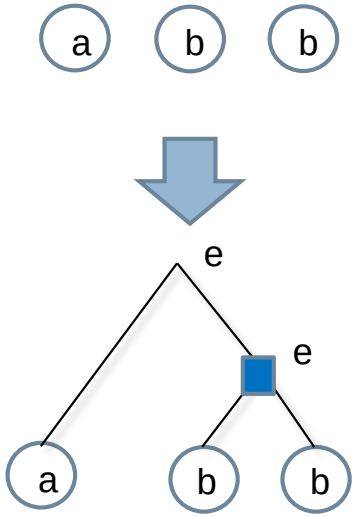


Programmation dynamique

- M(e, 1) = 1

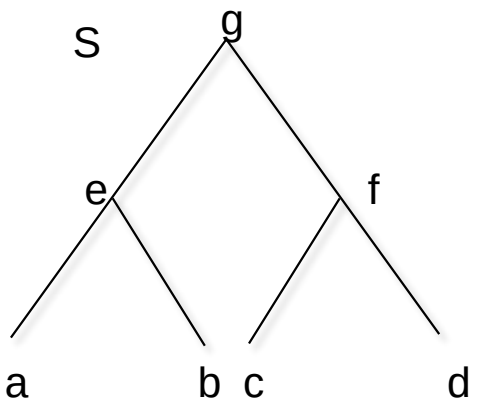


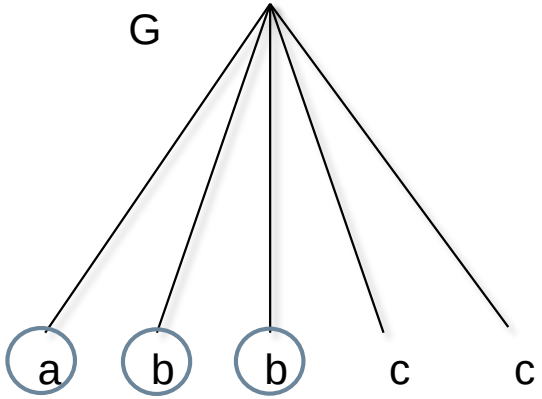


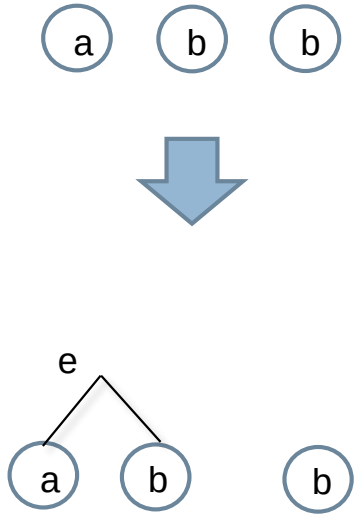


Programmation dynamique

- M(e, 2) = ?

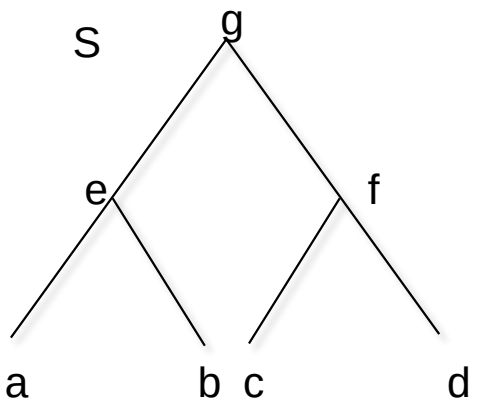


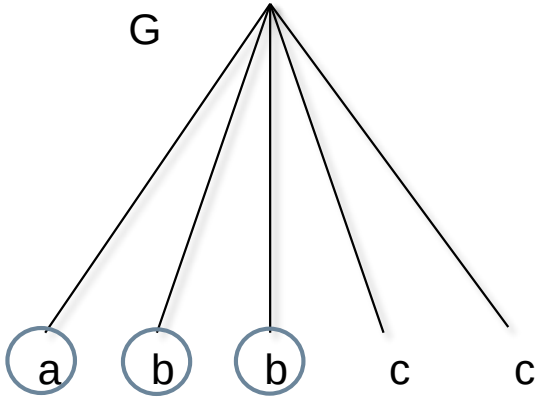


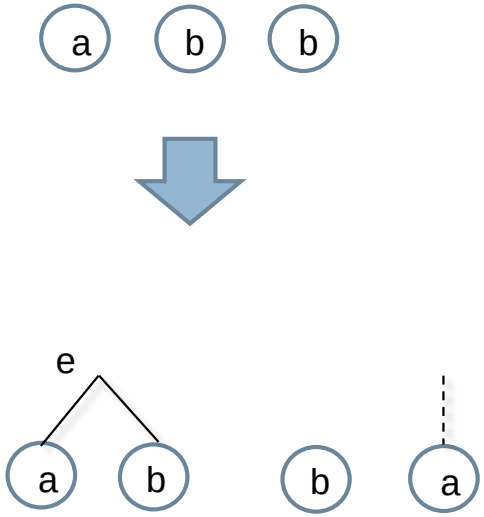


Programmation dynamique

- M(e, 2) = ?

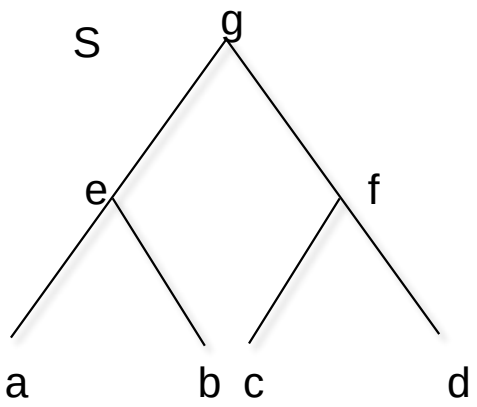


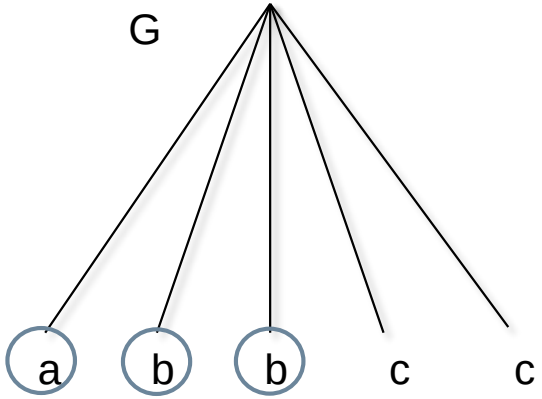


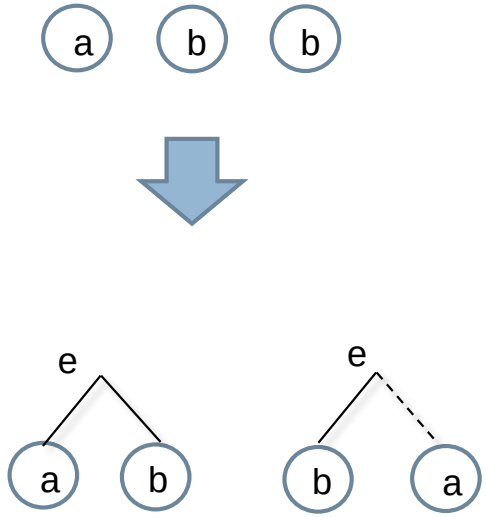


Programmation dynamique

- M(e, 2) = 1







Récurrence

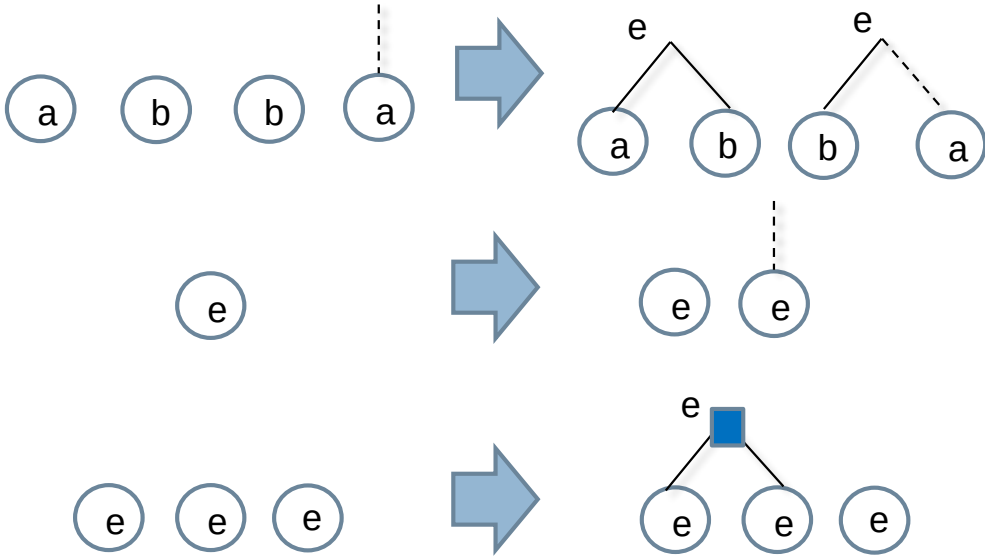
- M(e, 2) = min(

  M(a, 2) + M(b, 2),

  M(e, 2 – 1) + 1,

  M(e, 2 + 1) + 1

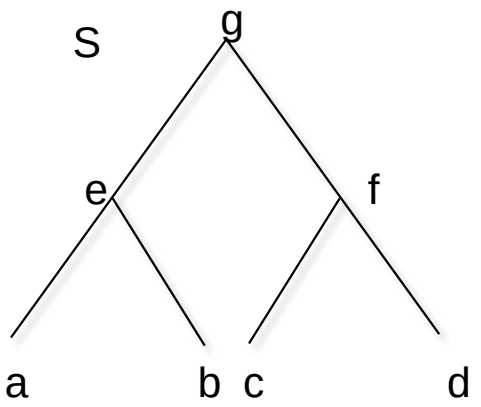
)

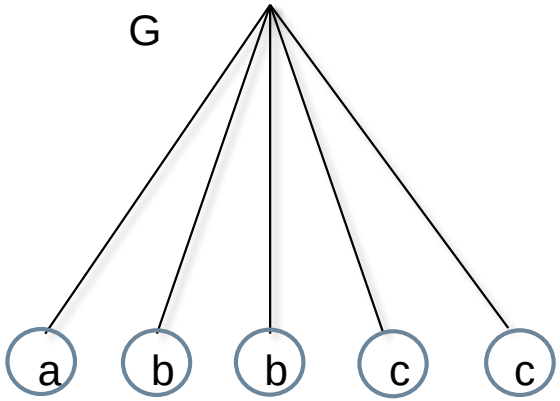


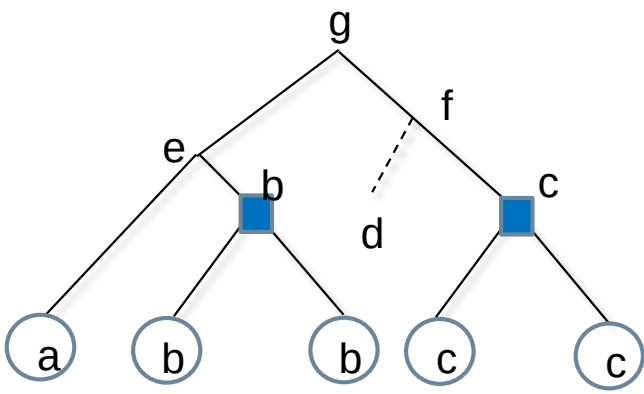
La table M peut être remplie en temps O(|S||G|)

Cas final

- M(g, 1) = si on se restreint à tous les gènes, coût min. pour avoir un seul arbre
- (i.e. coût min. d'une solution)







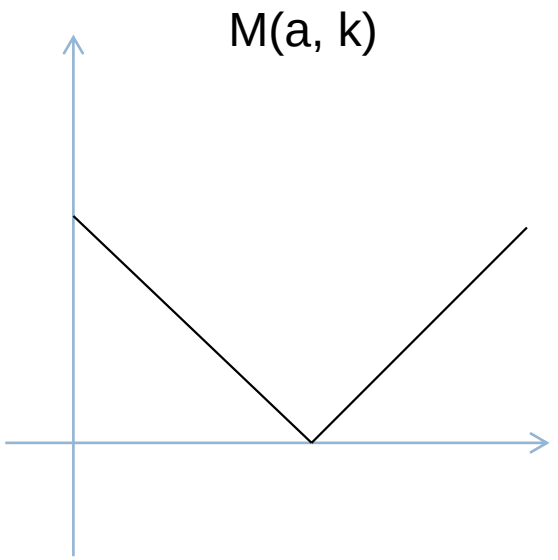
Calcul de la table M

k =	1	2	3	4	5	6
M(a, k)	3	2	1	0	1	2
M(b, k)	1	0	1	2	3	4
M(c, k)	0	1	2	3	4	5
M(d, k)	1	2	3	4	5	6
M(e, k)	3	2	2	2	3	4
M(f, k)	1	2	3	4	5	6
M(g, k)	4	4	5	6	7	8

Calcul de la table M

- Les valeurs des rangées ne sont pas aléatoires.

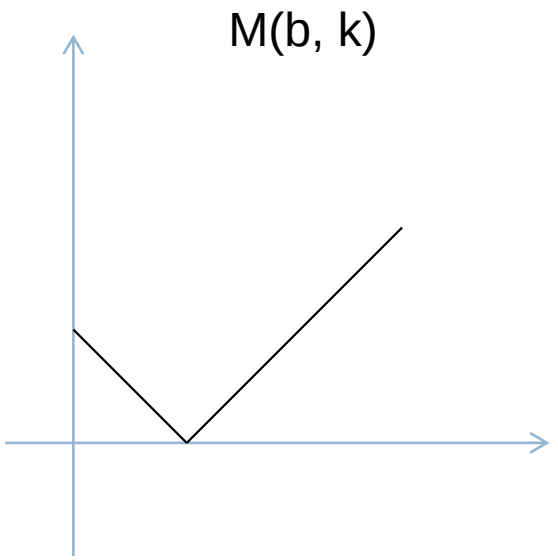
k =	1	2	3	4	5	6
M(a, k)	3	2	1	0	1	2
M(b, k)	1	0	1	2	3	4
M(c, k)	0	1	2	3	4	5
M(d, k)	1	2	3	4	5	6
M(e, k)	3	2	2	2	3	4
M(f, k)	1	2	3	4	5	6
M(g, k)	4	4	5	6	7	8



Calcul de la table M

- Les valeurs des rangées ne sont pas aléatoires.

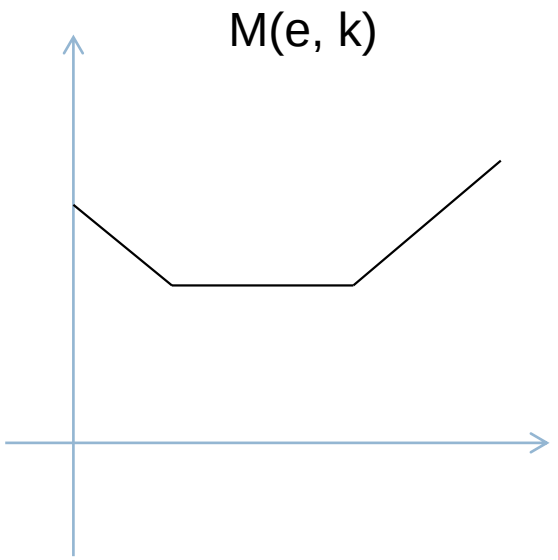
k =	1	2	3	4	5	6
M(a, k)	3	2	1	0	1	2
M(b, k)	1	0	1	2	3	4
M(c, k)	0	1	2	3	4	5
M(d, k)	1	2	3	4	5	6
M(e, k)	3	2	2	2	3	4
M(f, k)	1	2	3	4	5	6
M(g, k)	4	4	5	6	7	8



Calcul de la table M

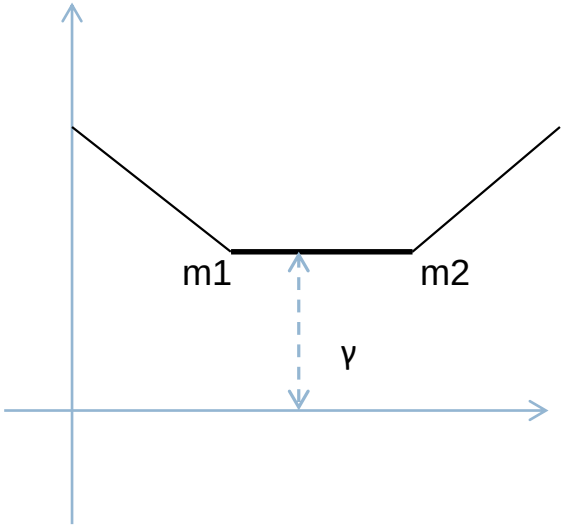
- Les valeurs des rangées ne sont pas aléatoires.

k =	1	2	3	4	5	6
M(a, k)	3	2	1	0	1	2
M(b, k)	1	0	1	2	3	4
M(c, k)	0	1	2	3	4	5
M(d, k)	1	2	3	4	5	6
M(e, k)	3	2	2	2	3	4
M(f, k)	1	2	3	4	5	6
M(g, k)	4	4	5	6	7	8



Calcul de la table M

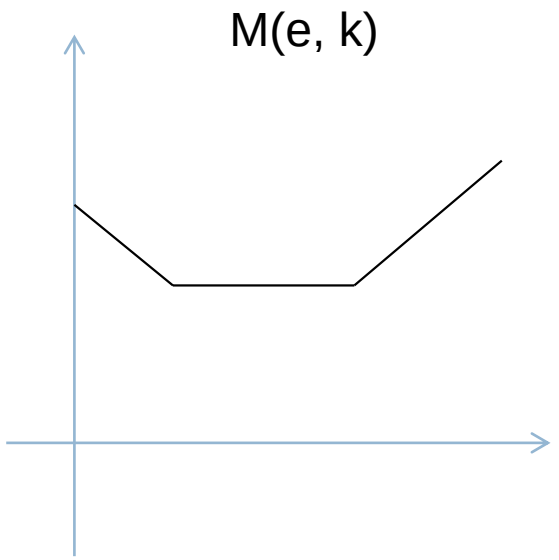
- Chaque rangée peut être représentée par une "fonction de coupe", qui est entièrement décrite par seulement 3 = O(1) valeurs.



Calcul de la table M

- Les valeurs des rangées ne sont pas aléatoires.

k =	1	2	3	4	5	6
M(a, k)	3	2	1	0	1	2
M(b, k)	1	0	1	2	3	4
M(c, k)	0	1	2	3	4	5
M(d, k)	1	2	3	4	5	6
M(e, k)	3	2	2	2	3	4
M(f, k)	1	2	3	4	5	6
M(g, k)	4	4	5	6	7	8



Chaque rangée peut être remplie (i.e. représentée par ses 3 valeurs) en temps O(1)

=> La table M peut être remplie en temps O(|S|)

Extensions de l'algorithme

- Plusieurs solutions possibles:
  - Choisir selon un critère Neighbor-Joining
  - Sortir toutes les solutions possibles

  E. Noutahi, M. Semeria, **M. Lafond**, J. Seguin, B. Boussau, L. Guéguen, N. El-Mabrouk, E. Tannier, "Efficient gene tree correction guided by species and synteny evolution". PLOS ONE (à paraître)

- Coût différents entre pertes et duplications
  - De plus, coût pouvant varier selon l'espèce

  M. Lafond, E. Noutahi, N. El-Mabrouk, "Efficient Non-Binary Gene Tree Resolution with Weighted Reconciliation Cost". CPM 2016

Correction d'arbres par relations d'orthologie

Gene tree correction guided by orthology
M Lafond, M Semeria, KM Swenson, E Tannier, N El-Mabrouk
BMC bioinformatics (2013)

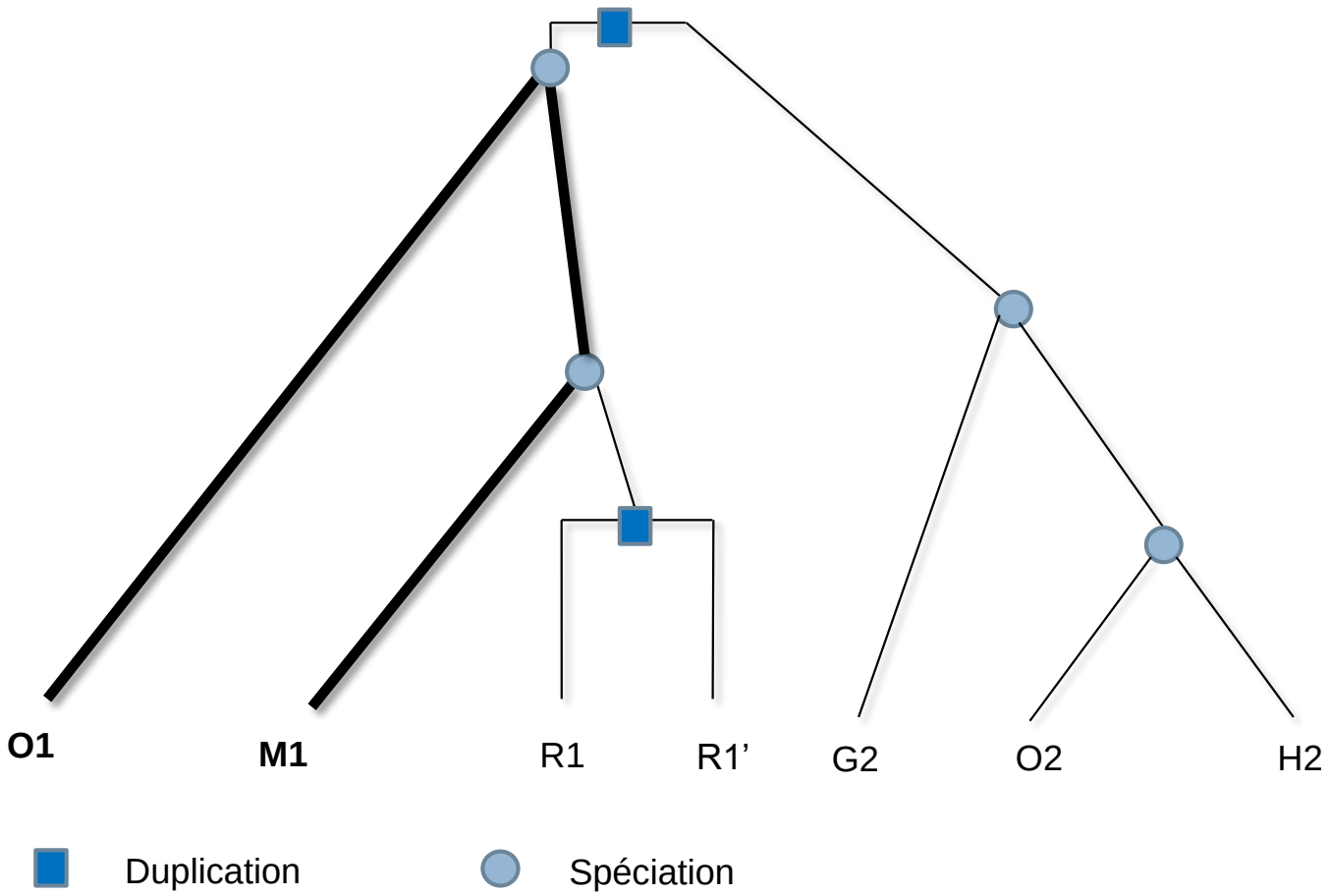
Orthologues et paralogues

Deux gènes sont:

Orthologues si leur dernier ancêtre commun a subi une **spéciation**

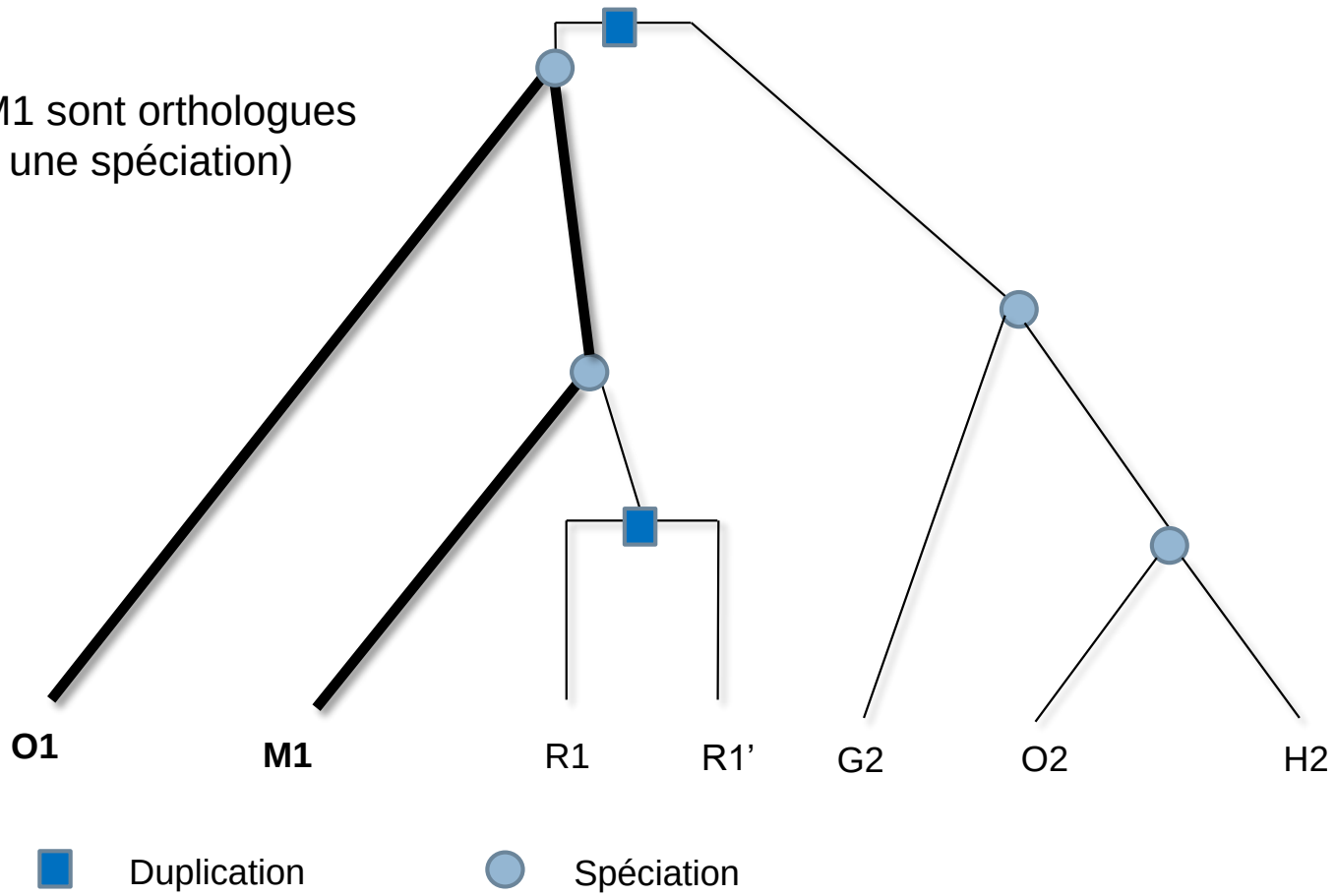
Paralogues si leur dernier ancêtre commun a subi une **duplication**

Orthologues et paralogues



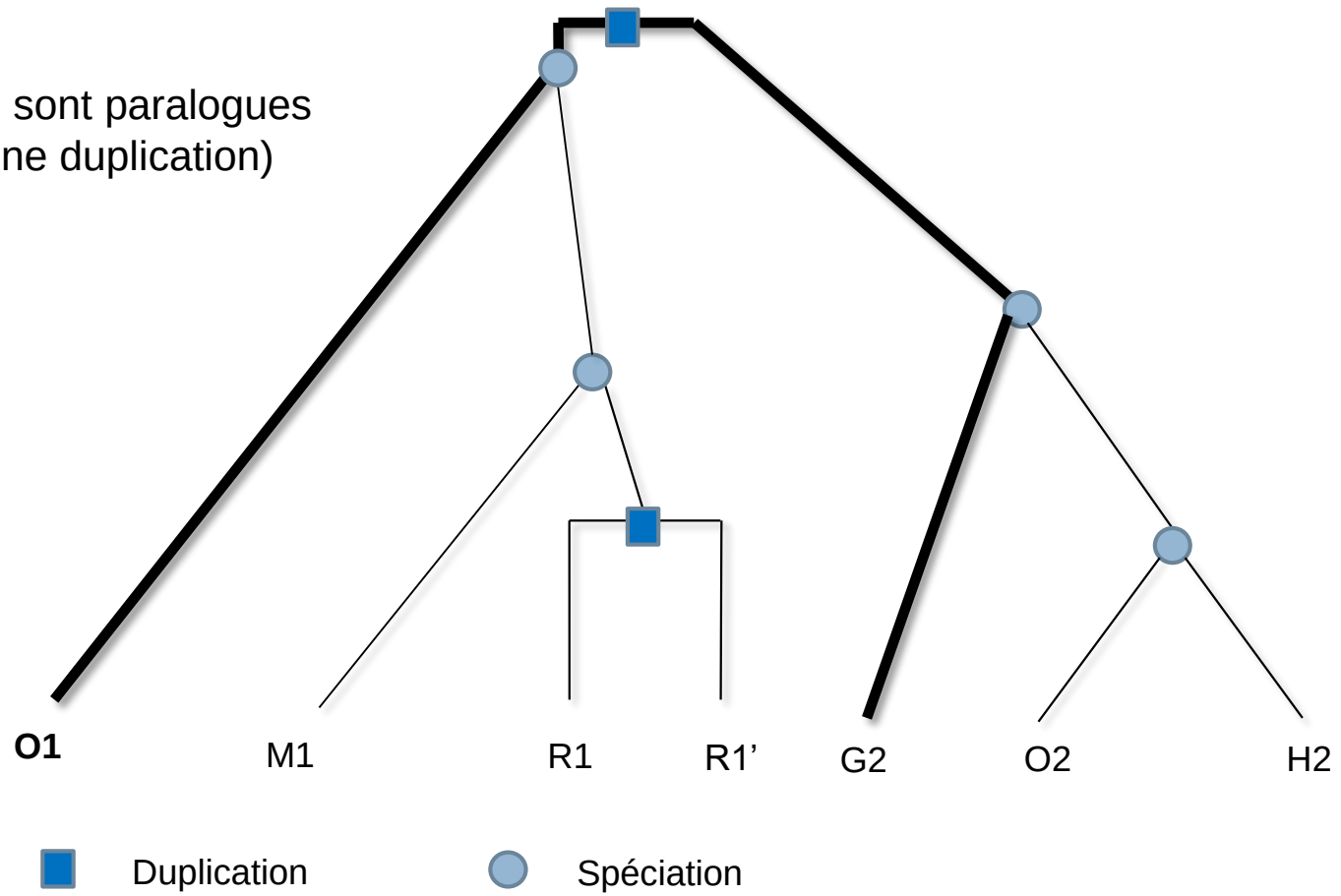
Orthologues et paralogues

O1 et M1 sont orthologues
(lca est une spéciation)



Orthologues et paralogues

O1 et G2 sont paralogues
(lca est une duplication)



Pourquoi les relations?

Les relations d'orthologie/paralogie sont liées aux fonctions des gènes.

Certaines bases de données **d'annotation fonctionnelle** supposent que les orthologues effectuent des fonctions similaires

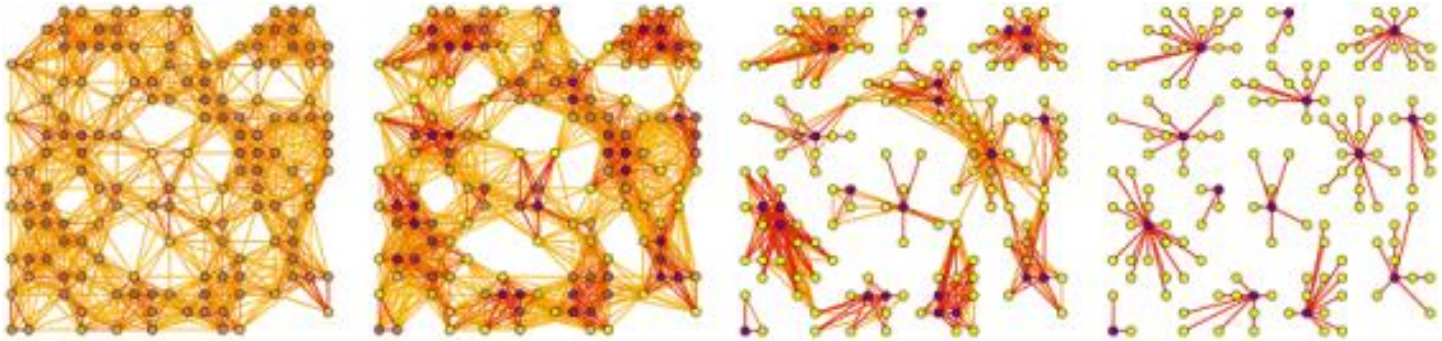
(e.g. COG, eggNOG)

Quest For Orthologs consortium: "a joint effort to benchmark, improve and standardize orthology predictions through collaboration, the use of shared reference datasets, and evaluation of emerging new methods".

Inférence de relations

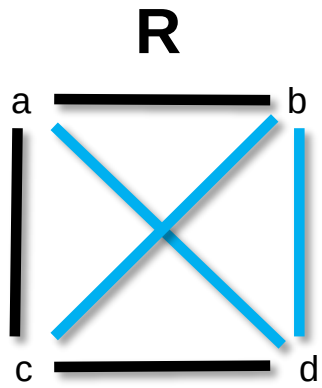
Clustering de gènes en groupes d'orthologues:

- Si g1 and g2 sont "**assez similaires**" en terme de séquences, g1 and g2 sont dits "orthologues candidats".
- Faire une graphe G des orthologues candidats.
- Partitionner G en clusters, i.e. composantes fortement connexes
  - Sinon, trop de faux-positifs
- OrthoMCL, InParanoid, proteinortho, …



Graphe de relations

Sequences
et autres…



Orthologues = (a,b) (a, c) (c, d)

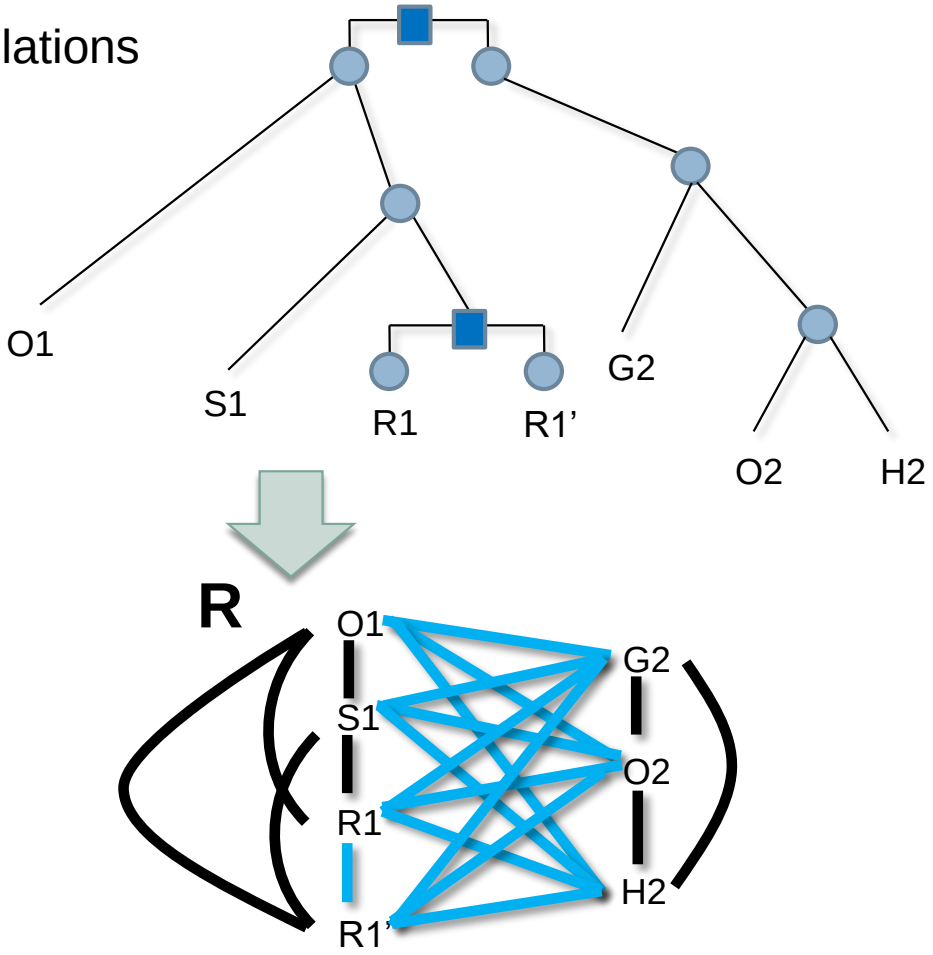
Paralogues = (a, d) (b, c) (b, d)

Orthologues

Paralogues

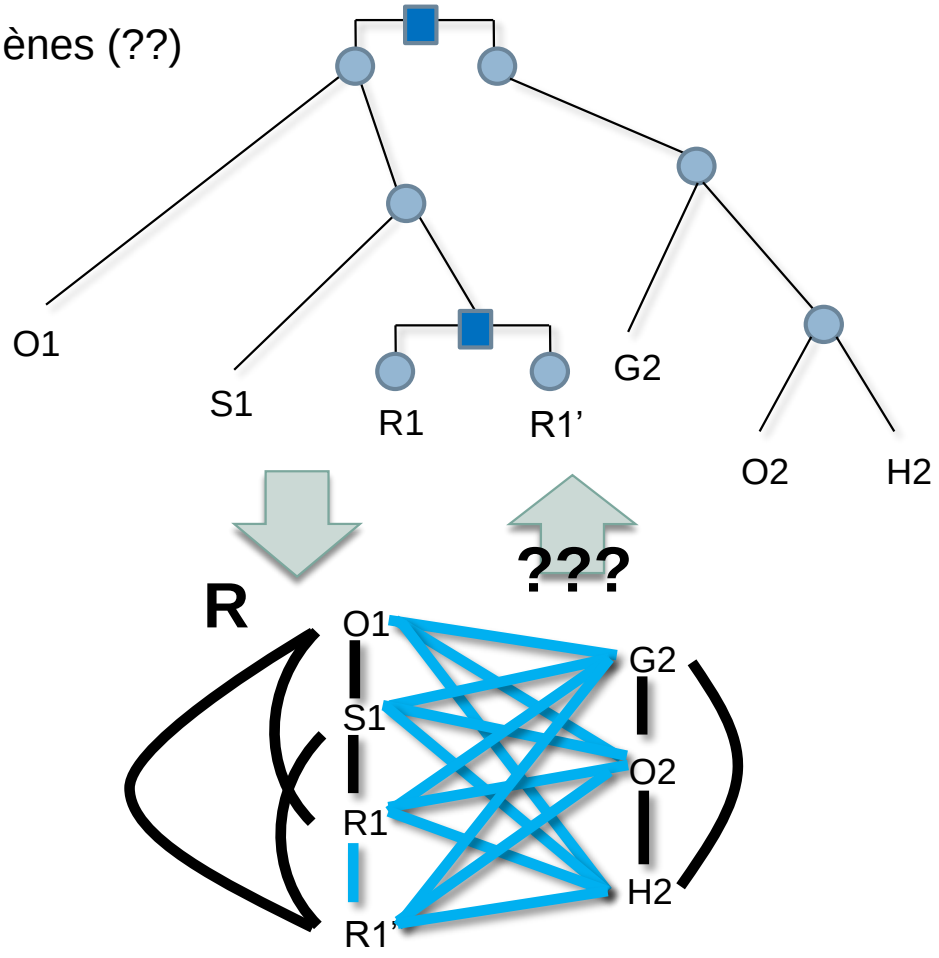
Relations et arbres de gènes

Arbre de gènes => relations



Relations et arbres de gènes

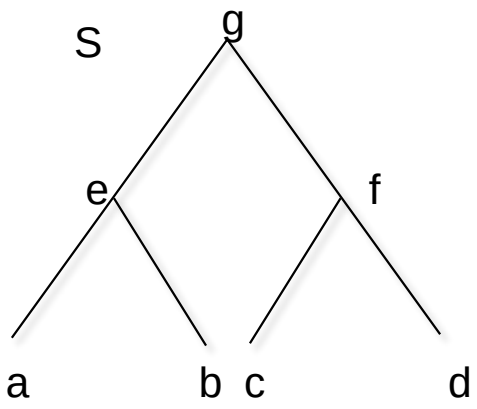
Relations => Arbre de gènes (??)

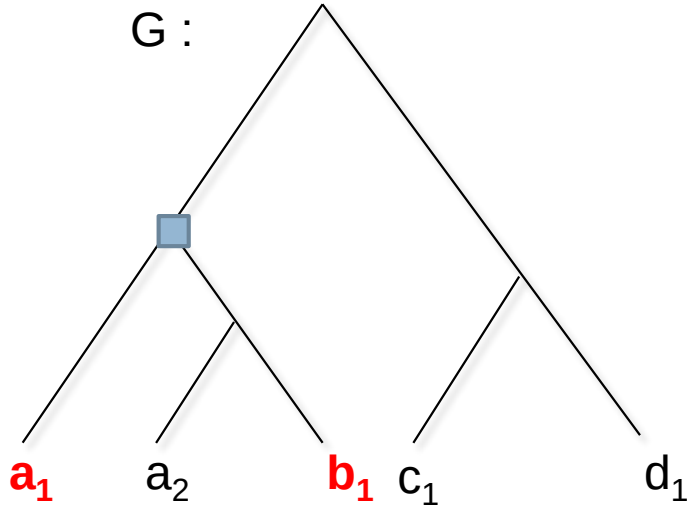


Correction par orthologie

Sequences
et autres…

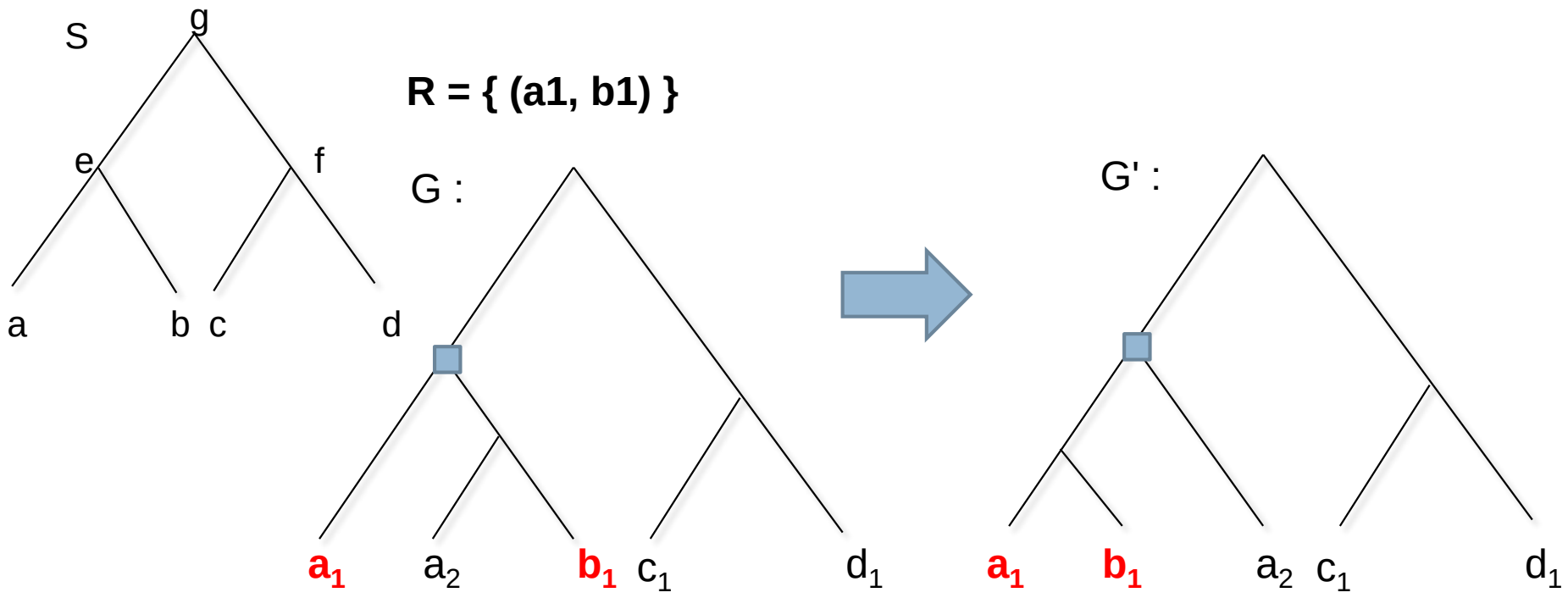
a_1 et b_1 sont
orthologues





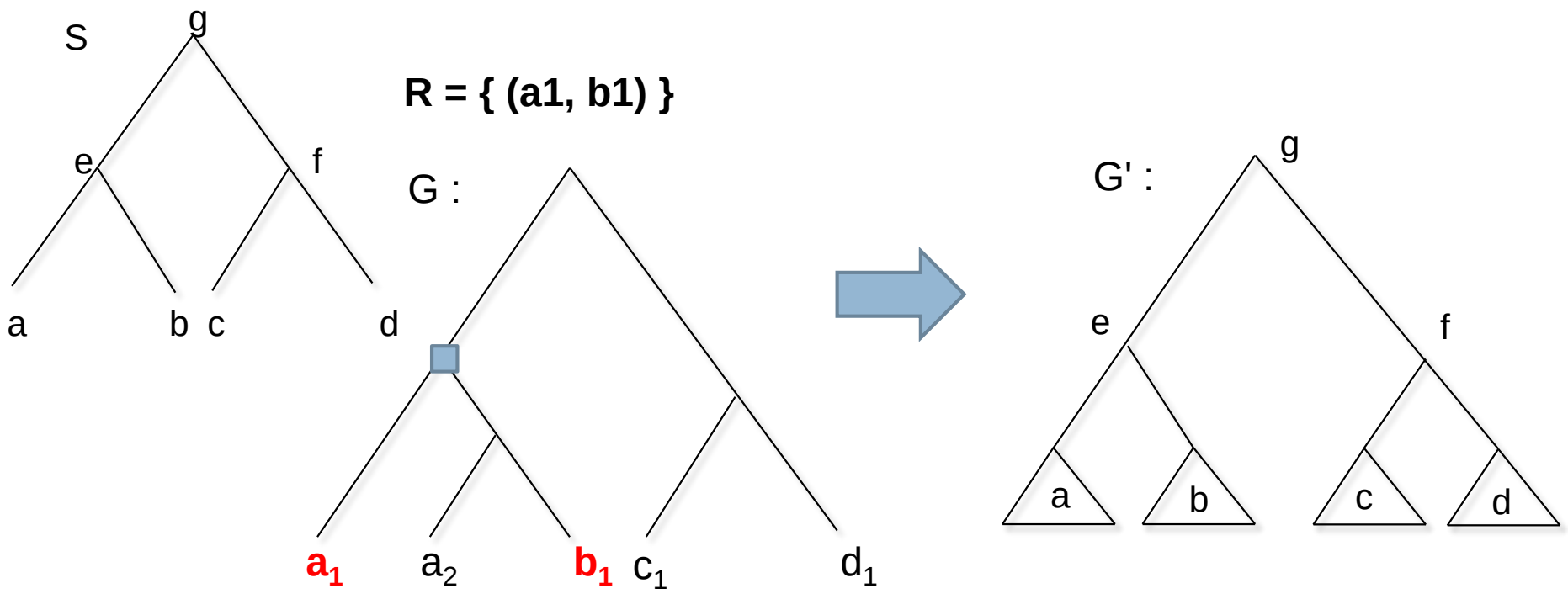
Correction par orthologie

- **Entrée**: une arbre d'espèces S binaire, et un arbre de gènes G binaire, et des relations d'orthologie R à satisfaire.
- **Sortie**: un arbre G' qui satisfait R en "s'éloignant le moins possible" de G.



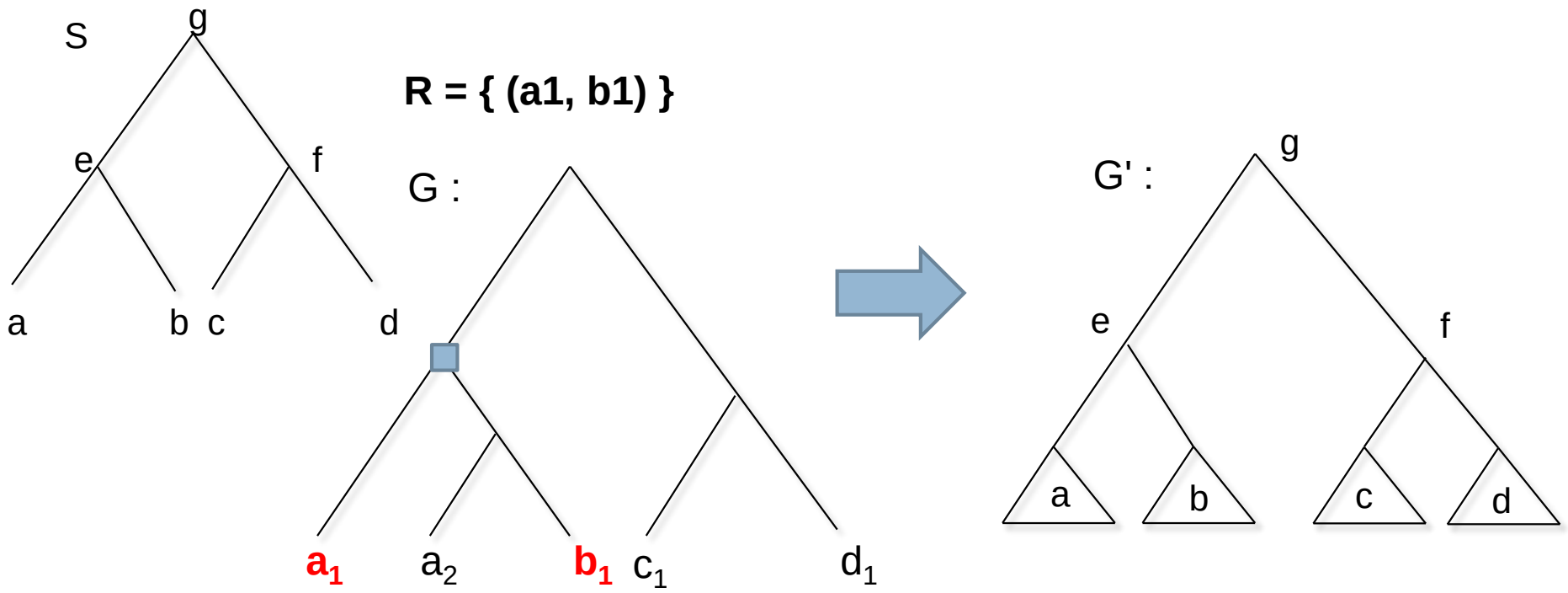
Correction par orthologie

- Note: il existe toujours une solution – on met tous les gènes de la même espèce ensemble.



Correction par orthologie

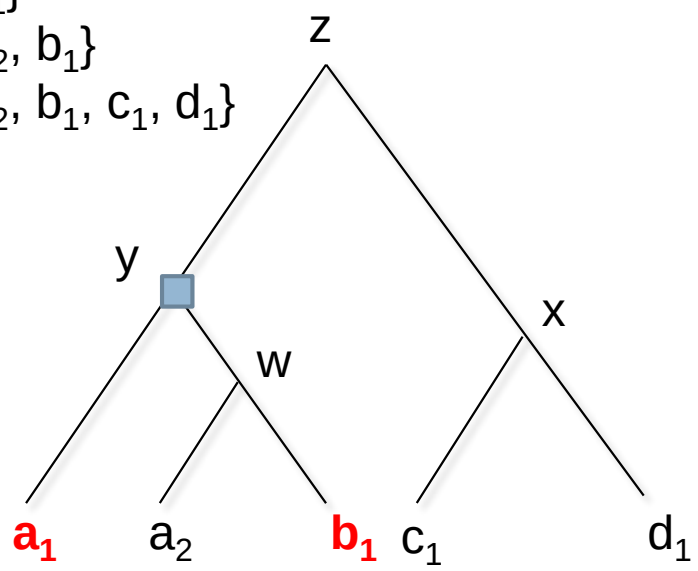
- Note: il existe toujours une solution – on met tous les gènes de la même espèce ensemble.
- Mais on voudrait ne pas trop changer G.



Préservation des clades

- On va préserver un maximum de clades de G.
- À chaque nœud x correspond un clade, dénoté clade(x), qui est l'ensemble des feuilles descendantes de x.

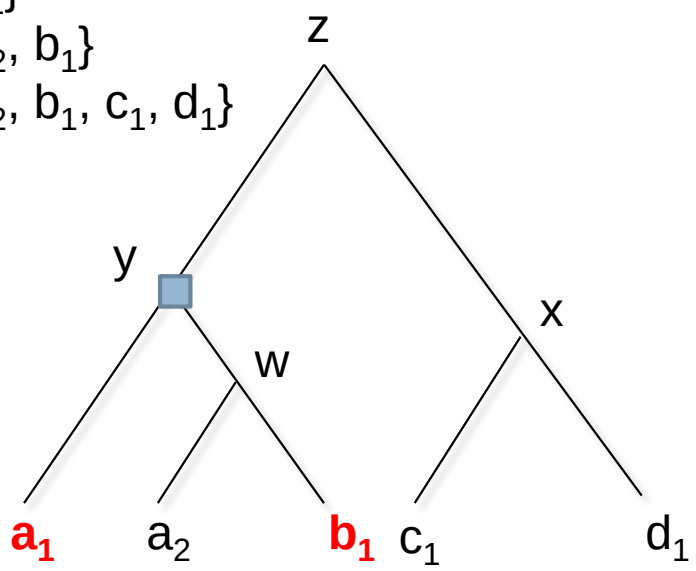
clade(w) = {a_2, b_1}
clade(x) = {c_1, d_1}
clade(y) = {a_1, a_2, b_1}
clade(z) = {a_1, a_2, b_1, c_1, d_1}



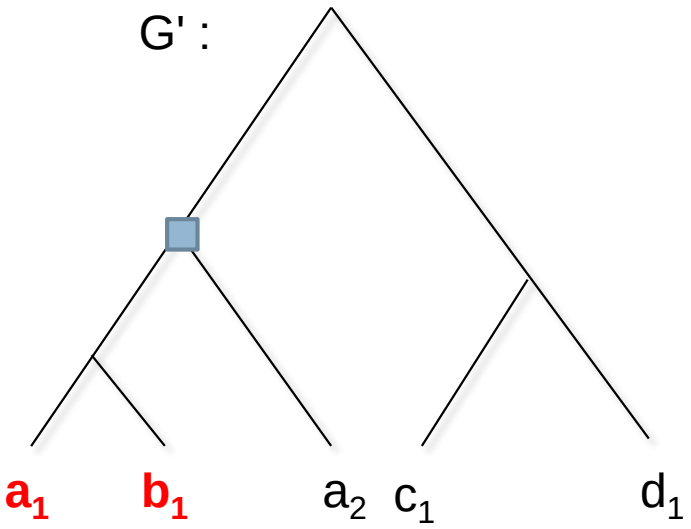
Préservation des clades

- On va préserver un maximum de clades de G.
- À chaque nœud x correspond un clade, dénoté clade(x), qui est l'ensemble des feuilles descendantes de x.

clade(w) = {a_2, b_1}
clade(x) = {c_1, d_1}
clade(y) = {a_1, a_2, b_1}
clade(z) = {a_1, a_2, b_1, c_1, d_1}

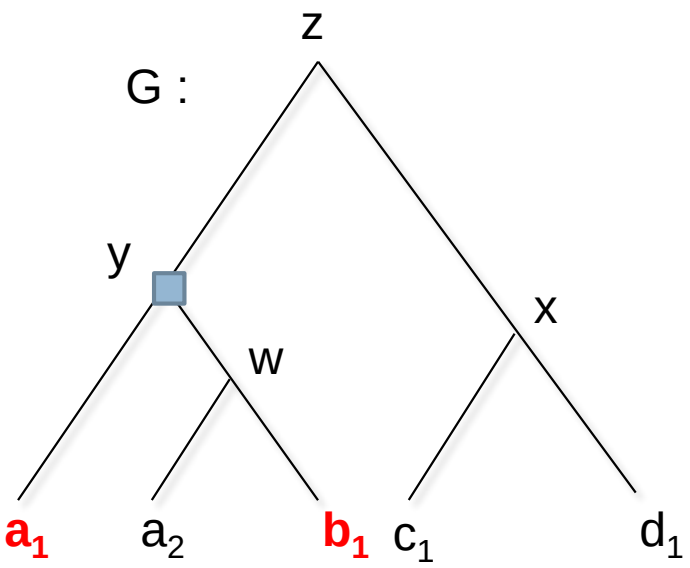


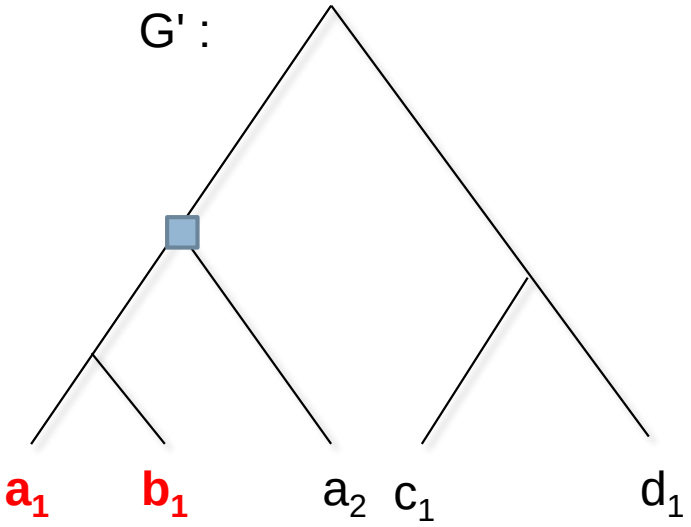
Clades non-préservés: {a_2,b_1}



Préservation des clades

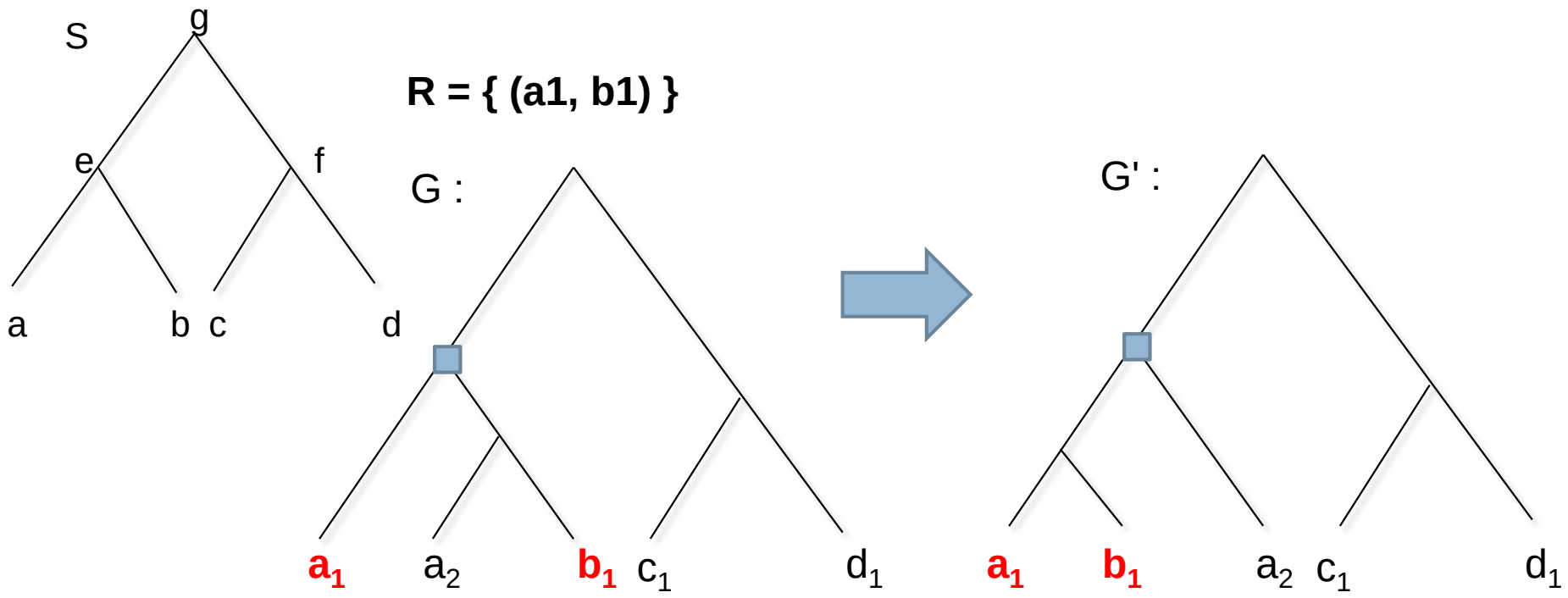
- Préserver un maximum de clades est équivalent à minimiser la distance **Robinson-Foulds** entre G et G'.





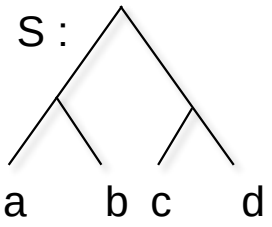
Correction par orthologie

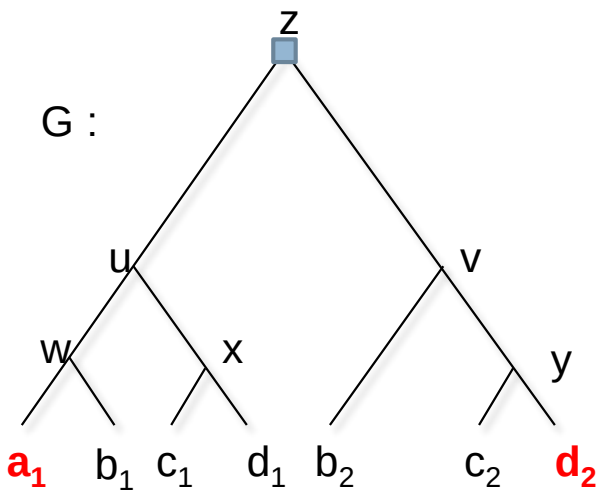
- **Entrée**: une arbre d'espèces S binaire, et un arbre de gènes G binaire, et des relations d'orthologie R à satisfaire.
- **Sortie**: un arbre G' qui satisfait R et **qui préserve un maximum de clades de G**.



L'algo en un exemple

- a_1, d_2 ne veulent pas être paralogues
  - lca(a_1, d_2) devrait être une spéciation
- Quels clades peut-on preserver?

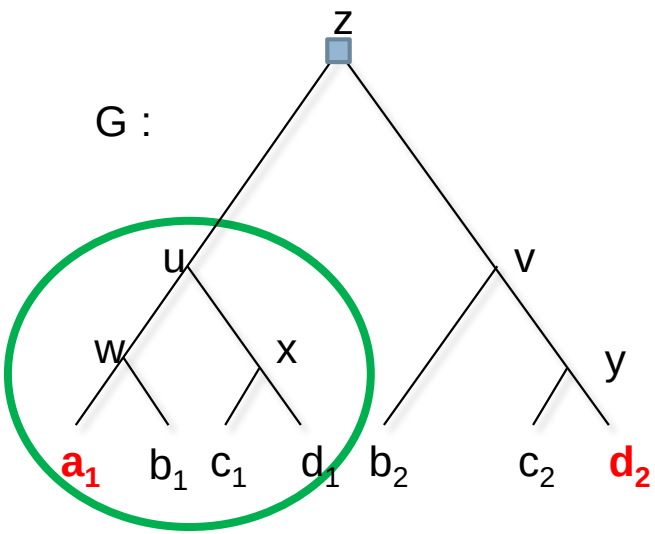


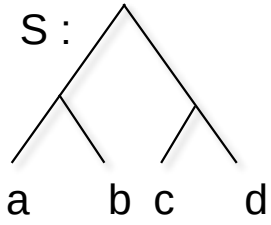


R = {(a_1, d_2)}

L'algo en un exemple

- Peut-on preserver clade(u) = $\{a_1, b_1, c_1, d_1\}$?

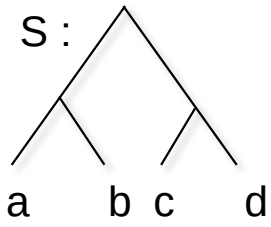


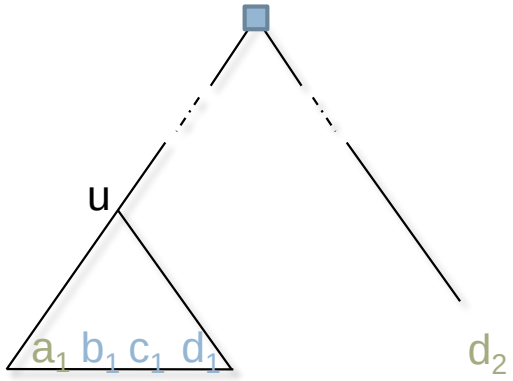


$R = \{(a_1, d_2)\}$

L'algo en un exemple

- Peut-on preserver clade(u) = {a_1,b_1,c_1,d_1} ?
  - On peut montrer que NON, c'est impossible.
  - Si on conserve clade(u), a_1 et d_2 resteront paralogues

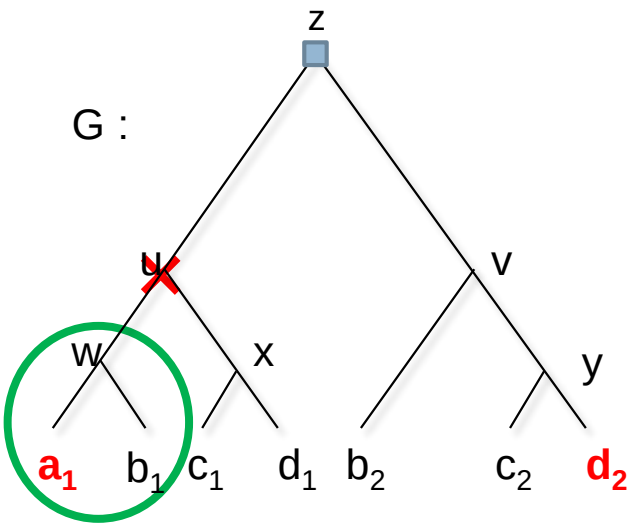


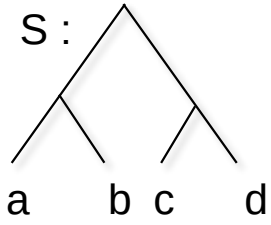


R = {(a_1, d_2)}

L'algo en un exemple

- Peut-on preserver clade(w) = $\{a_1, b_1\}$?

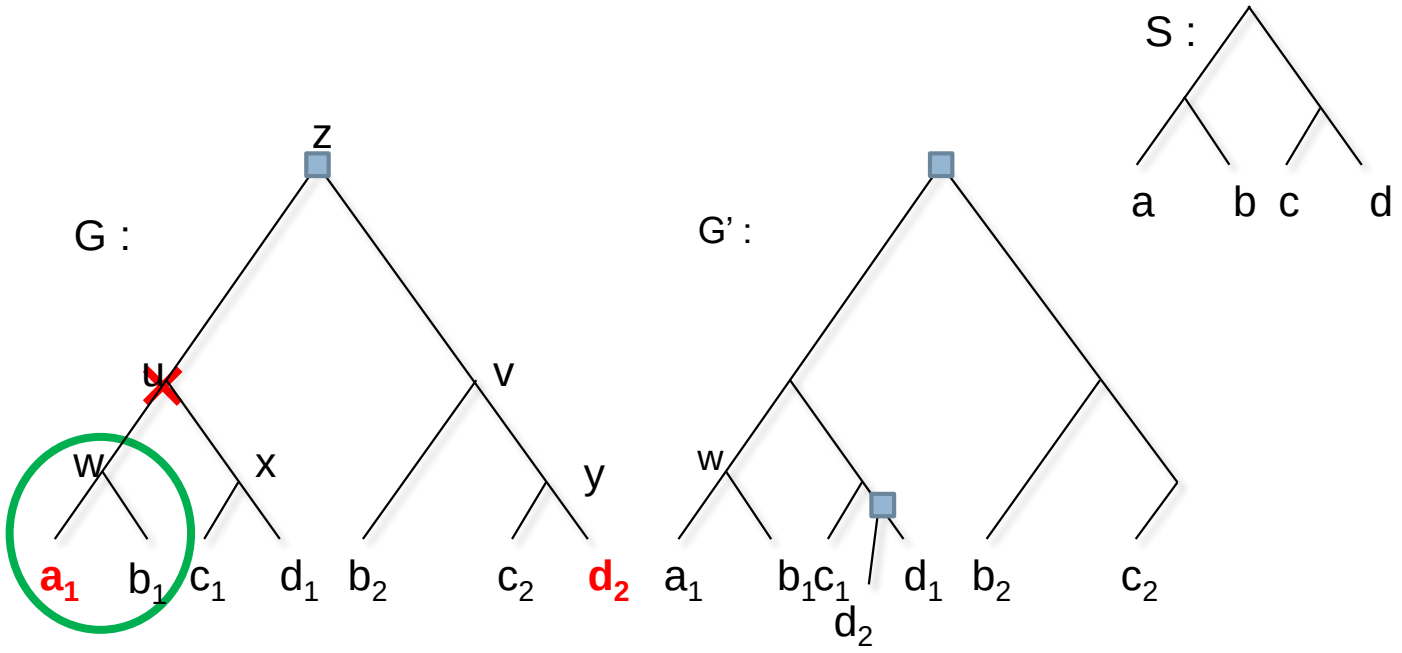




$R = \{(a_1, d_2)\}$

L'algo en un exemple

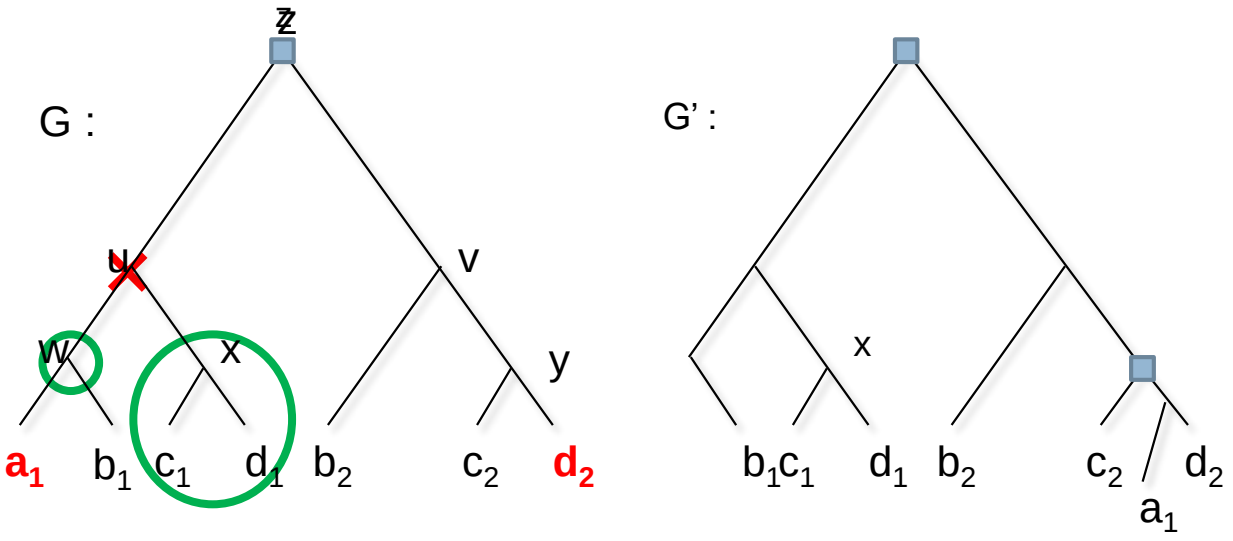
- Peut-on preserver clade(w) = $\{a_1, b_1\}$?
  - Oui, il existe une solution préserve clade(w).



$R = \{(a_1, d_2)\}$

L'algo en un exemple

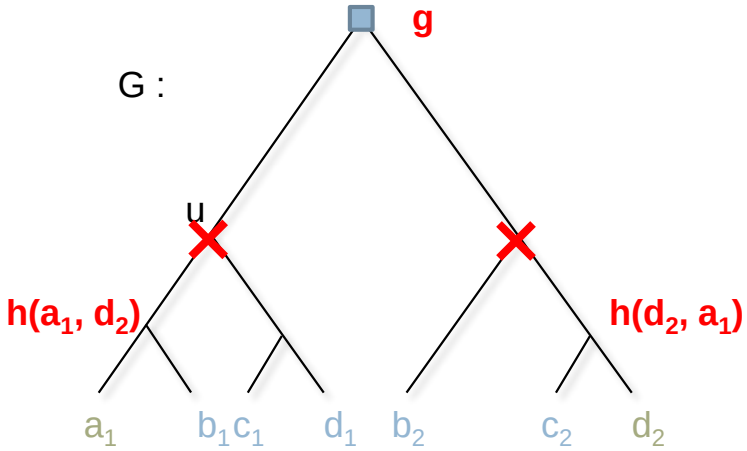
- Peut-on preserver clade(x) = {a_1,b_1} ?
  - Oui, il existe une solution préserve clade(x).

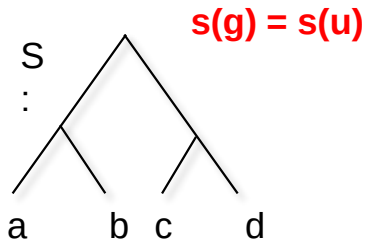


R = {(a_1, d_2)}

L'algo en un exemple

- For some constraint (a, b) in P :
  - Let g = LCA(a,b)
  - Let $h_{a,b}$ be the highest node on the path from a to g such that $s(h_{a,b})$ is a descendant of s(g).
  - Every node on the path from $h_{a,b}$ to g (excluding h and g) corresponds to an unpreservable clade.
  - Define $h_{b,a}$ analogously





$R = \{(a_1, d_2)\}$

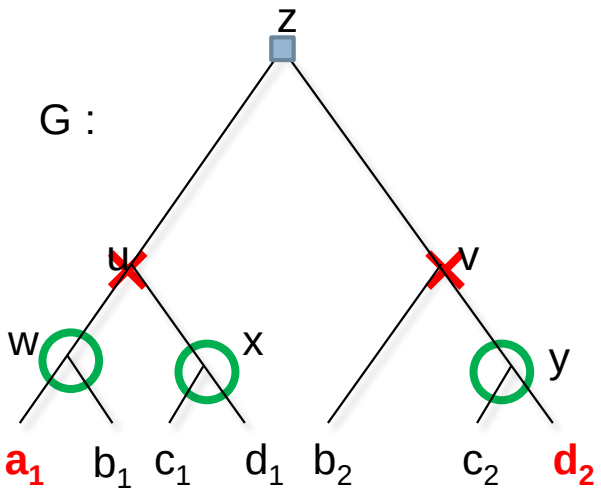
L'algo en un exemple

- For some constraint (a, b) in P :
  - Let g = LCA(a,b)
  - Let $h_{a,b}$ be the highest node on the path from a to g such that $s(h_{a,b})$ is a descendant of s(g).
  - Every node on the path from $h_{a,b}$ to g (excluding h and g) corresponds to an unpreservable clade.
  - Define $h_{b,a}$ analogously

- For every constraint (a, b) in P
  - Compute $h_{a,b}$ and $h_{b,a}$
  - Find the unpreservable clades they imply
- Identifies all unpreservable clades.
- Can be done in time O(|P| |V(G)|)

L'algo en un exemple

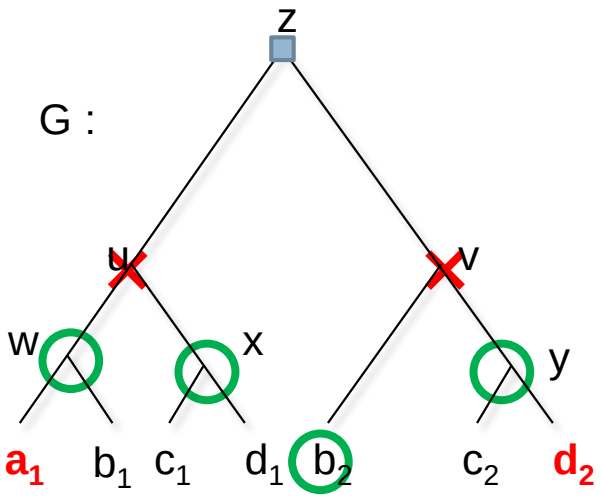
- On sait dire si on clade peut être préservé ou non, individuellement.
- Mais peut on tous les préserver simultanément?



$R = \{(a_1, d_2)\}$

L'algo en un exemple

- Oui! On prend tous les sous-arbres maximaux préservables (i.e. qui n'ont pas d'ancêtre préservable mis-à-part la racine).



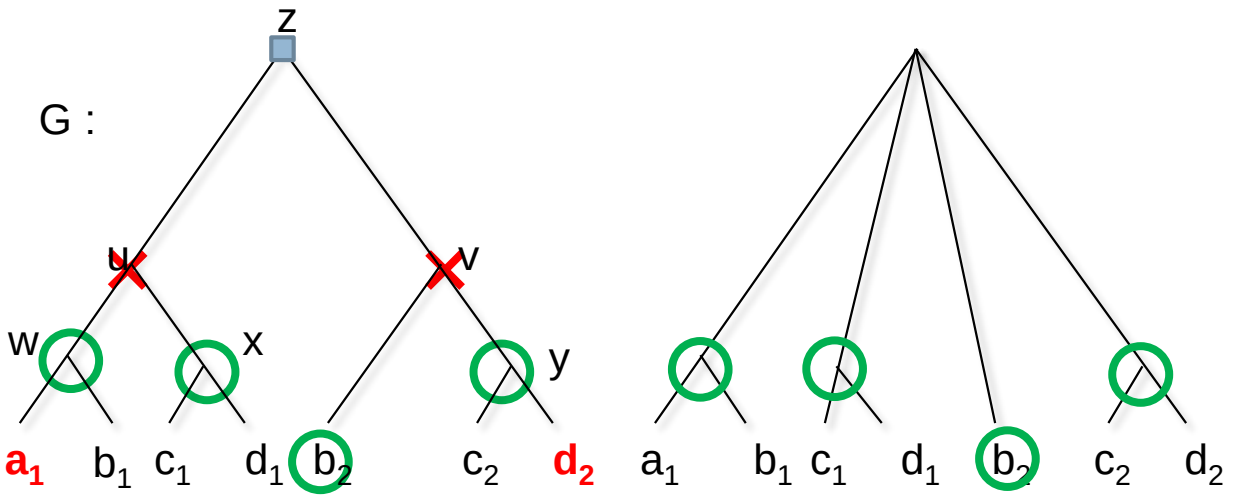
- Ici, les sous-arbres enracinés en
- {w, x, b_2, y}

  (les feuilles sont toujours préservables)

R = {(a_1, d_2)}

L'algo en un exemple

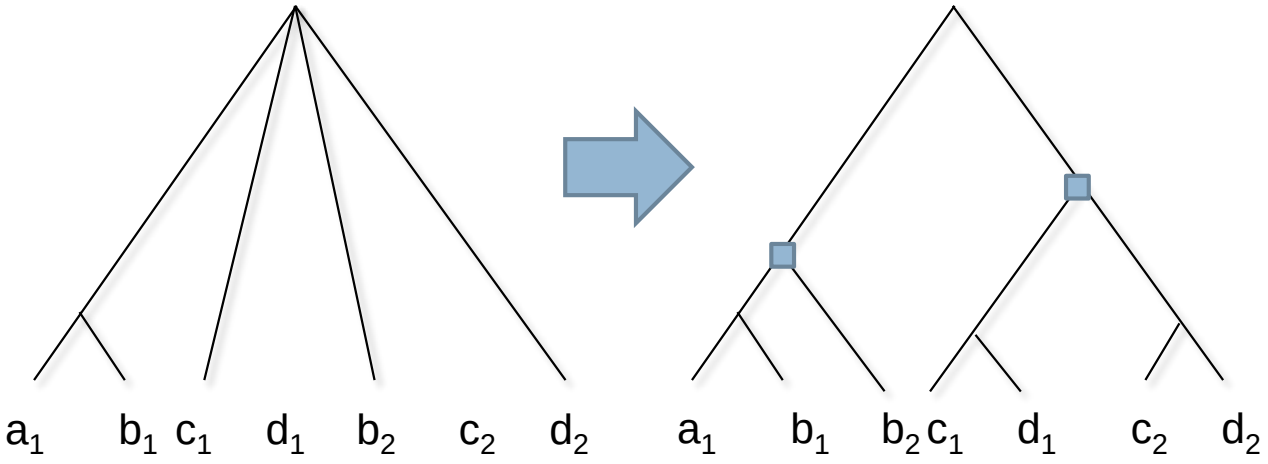
- On joint ces sous-arbres sous une polytomie.



$R = \{(a_1, d_2)\}$

L'algo en un exemple

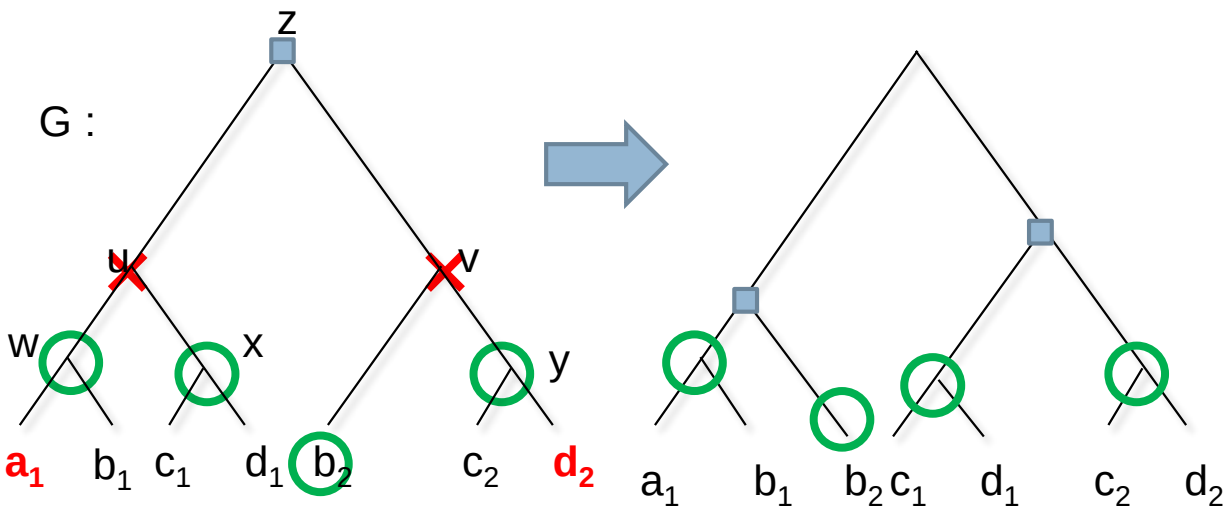
- Trouver la resolution qui maximize le nombre d'orthologies (faisable de façon "greedy").
- R sera satisfait.



$R = \{(a_1, d_2)\}$

L'algo en un exemple

- Cette résolution permet de satisfaire R tout en préservant un maximum de clades.
- Il se peut qu'on doivent répéter récursivement dans les sous-arbres considérés.



$R = \{(a_1, d_2)\}$

Résultats

- Avec une méthode basée sur la synténie (Lafond, Swenson et El-Mabrouk, 2013), nous avons inféré des relations d'orthologie sur 1000 familles de genes.
- Nous avons corrigé les arbres d'Ensembl sur ces 1000 familles de gènes.
- Nous avons comparé nos arbres corrigé en utilisant le test statistique **"AU Test"**
  - 82.3% des arbres corrigés étaient statistiquement viables
  - 17.7% de nos arbres ont donc été rejetés statistiquement
  - 14.8% des arbres Ensembl ont été rejetés

Extension à la paralogie

- Cet algorithme permet seulement de satisfaire des **relations d'orthologie**.
- Et si des relations **d'orthologie + paralogie** étaient données?
- Problème: il se pourrait qu'il n'existe même pas de solution possible.
  - Comment savoir?

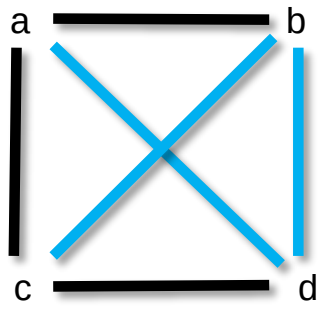
Validation de relations d'orthologie et paralogie

Orthology and paralogy constraints: satisfiability and consistency
M Lafond, N El-Mabrouk
BMC genomics 15 (2014)

Graphe d'orthologie/paralogie

Orthologues = (a,b) (a, c) (c, d)

Paralogues = (a, d) (b, c) (b, d)

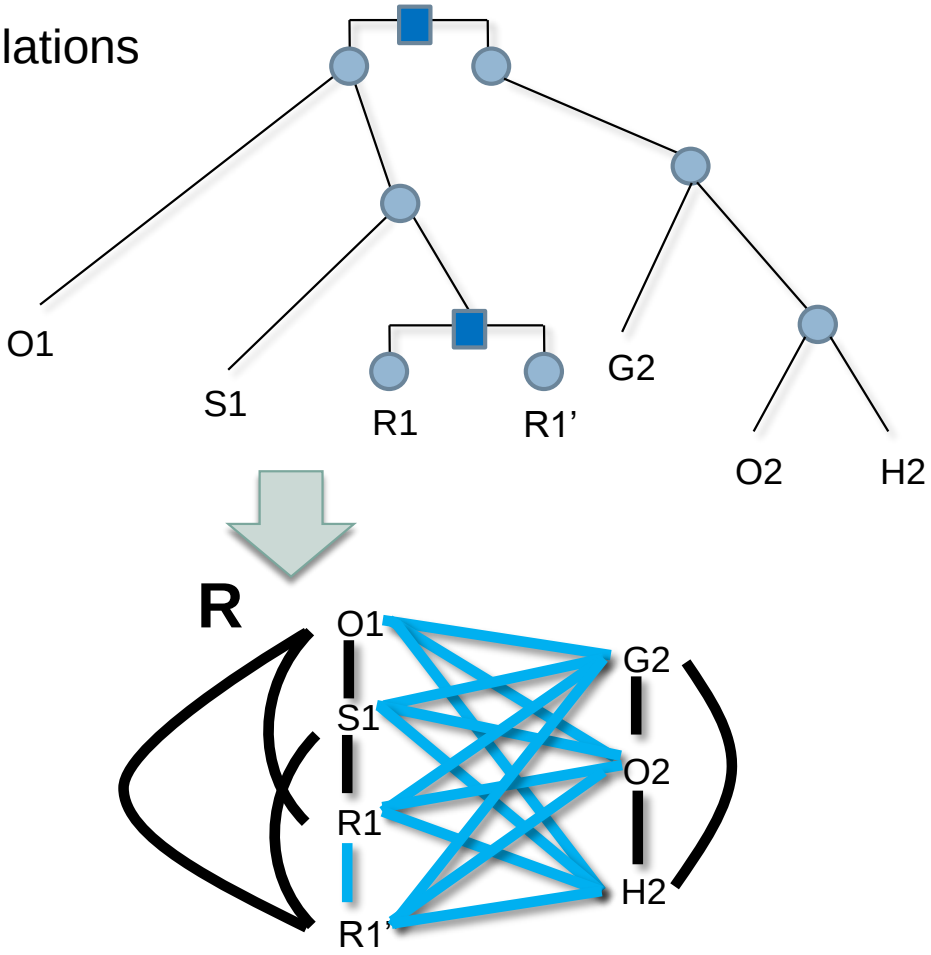


Orthologues

Paralogues

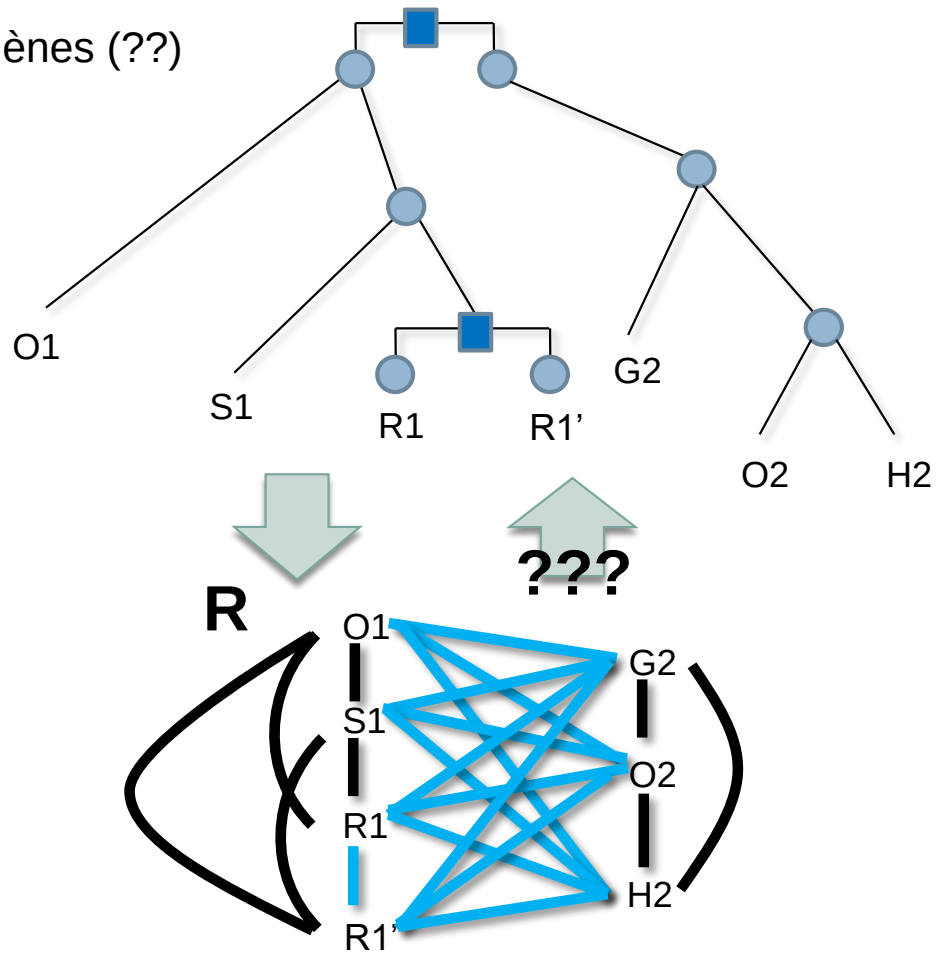
Graphe d'orthologie/paralogie

Arbre de gènes => relations

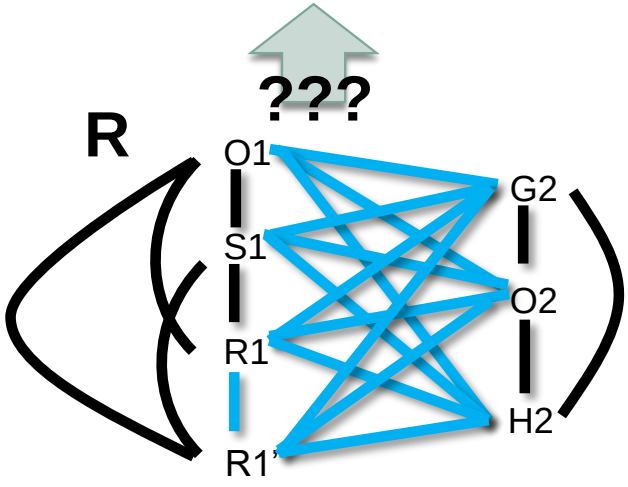


Graphe d'orthologie/paralogie

Relations => Arbre de gènes (??)



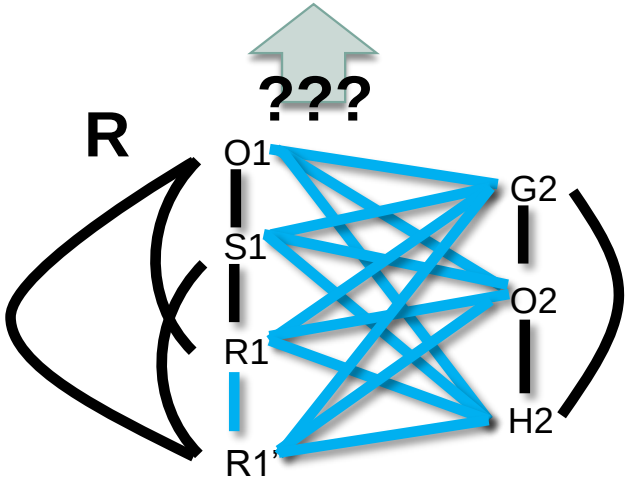
Graphe d'orthologie/paralogie



Satisfaisabilité de relations

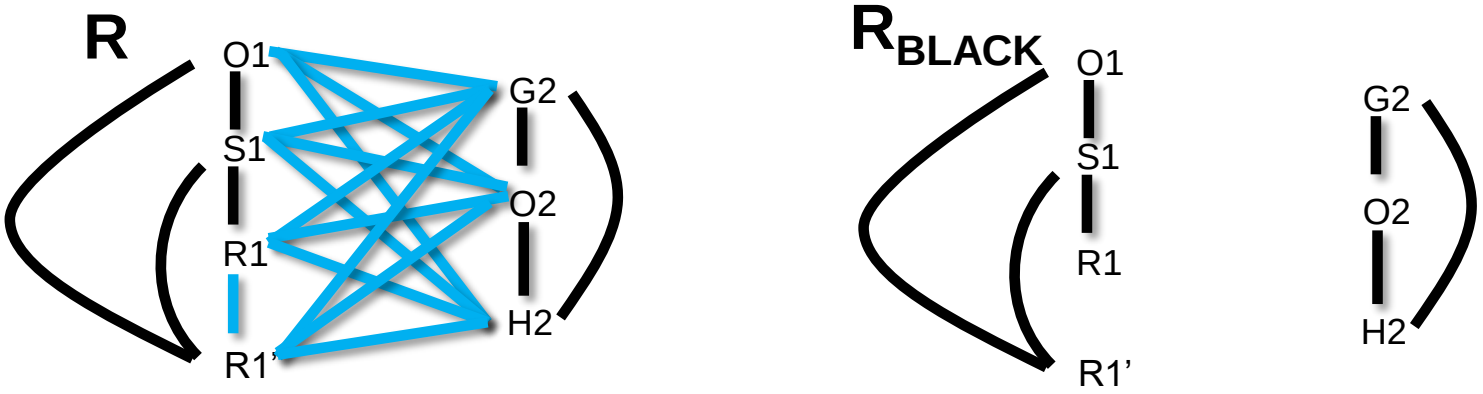
Problème: est-ce qu'un graphe de relations R donné est **satisfaisable**?

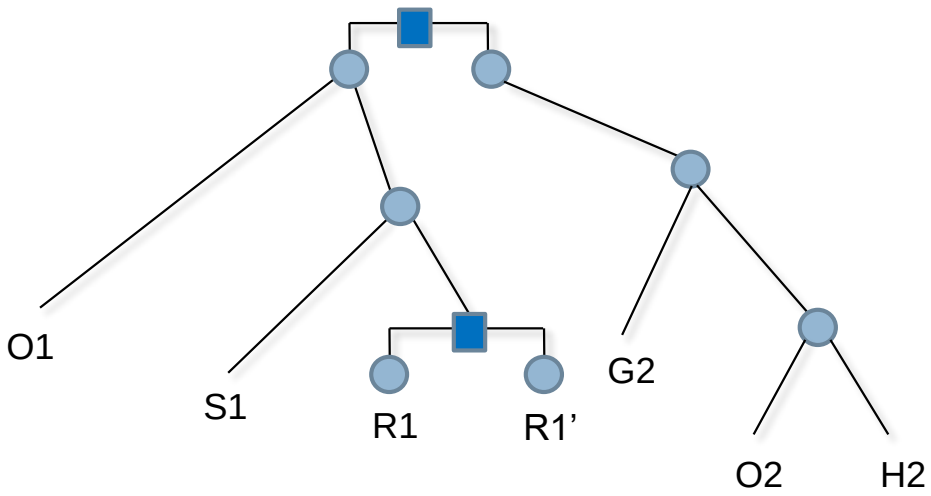
Est-ce qu'il existe un arbre de gènes contenant les relations de R ?

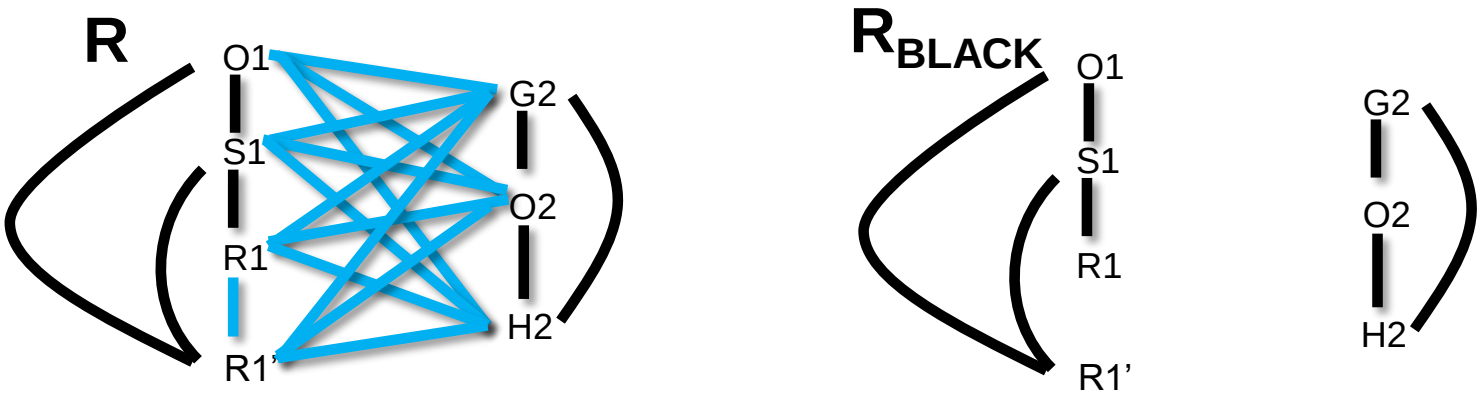


Theorème (Hernandez-Rosalez & al., 2012):

Un graphe de relations R est satisfiable SSI R_{BLACK} est "P_4-free" (n'a pas de chemin **induit** sur 4 sommets).

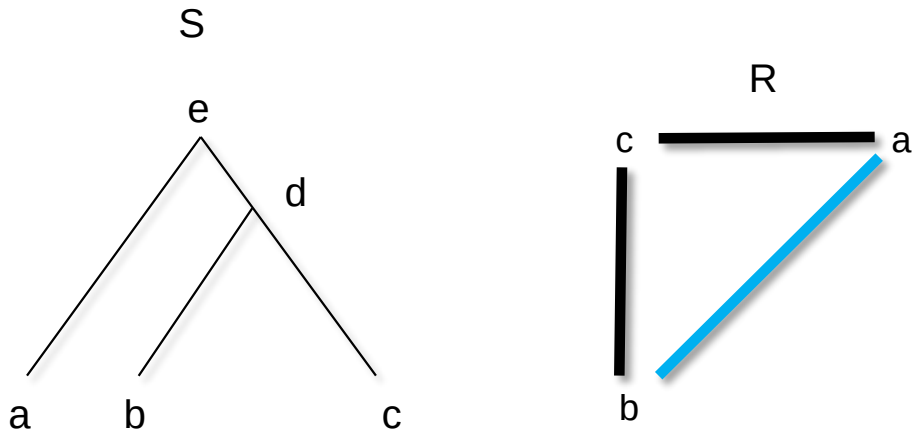



O1
S1
R1
R1'
G2
O2
H2


R
O1
S1
R1
R1'
G2
O2
H2
R BLACK
O1
S1
R1
R1'
G2
O2
H2

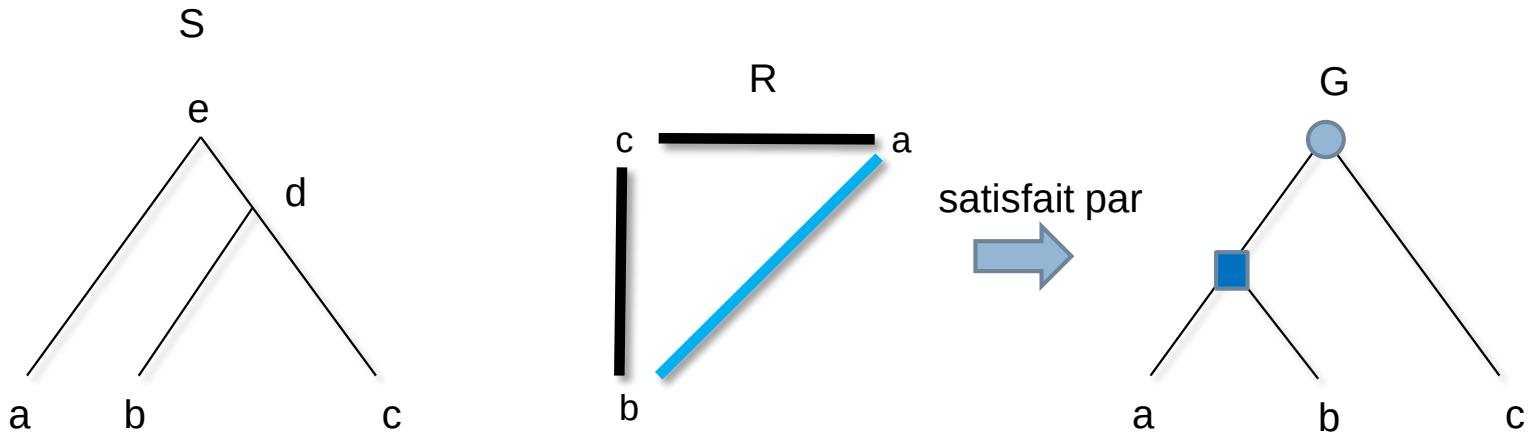
S-Consistance

On pourrait exiger que l'arbre de gènes construit soit consistant avec S



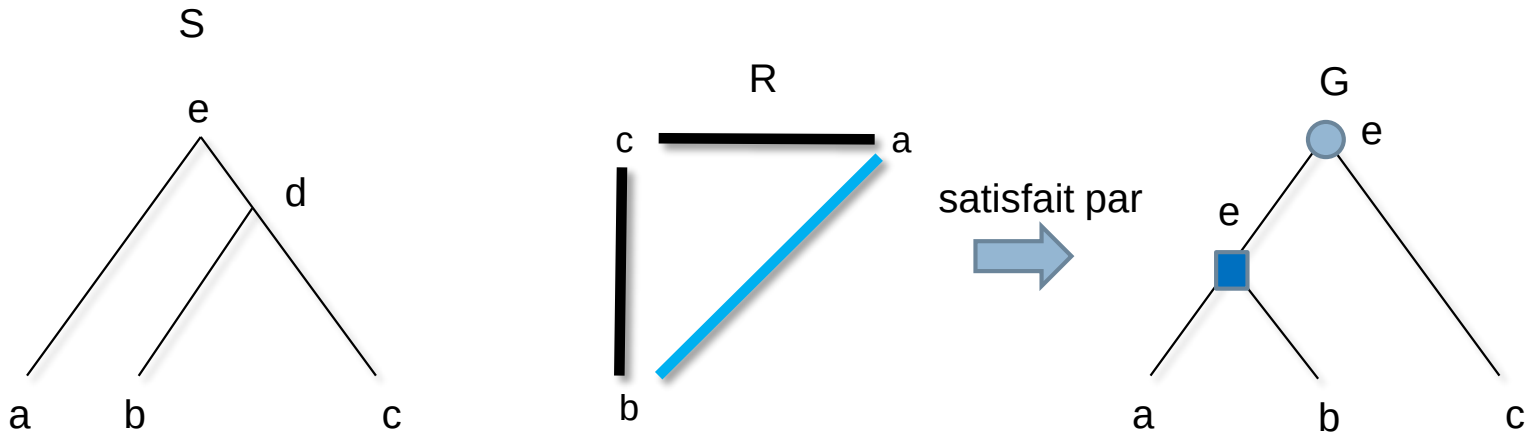
S-Consistance

On pourrait exiger que l'arbre de gènes construit soit consistant avec S



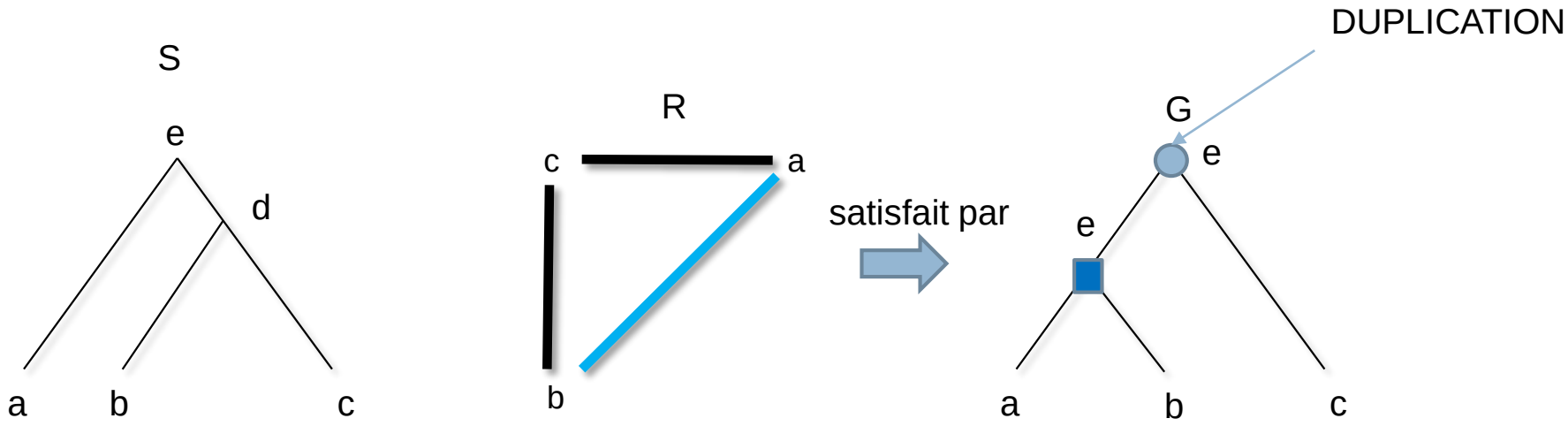
S-Consistance

On pourrait exiger que l'arbre de gènes construit soit consistant avec S



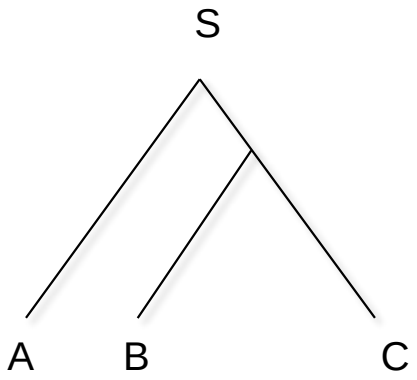
S-Consistance

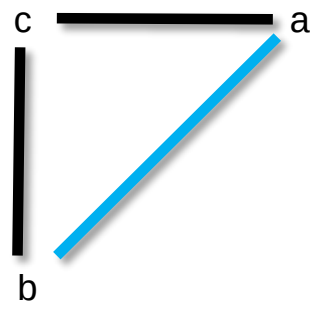
On pourrait exiger que l'arbre de gènes construit soit consistant avec S



Theorème:

Un graphe de relations R est S-Consistant SSI R est satisfaisable, **et tout sous-graphe de 3 sommets de R est "en accord" avec S.**

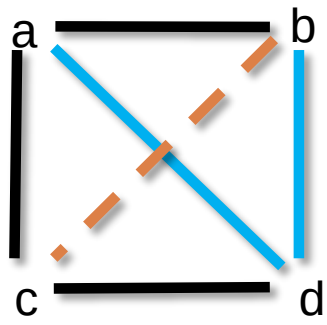




Relations inconnues

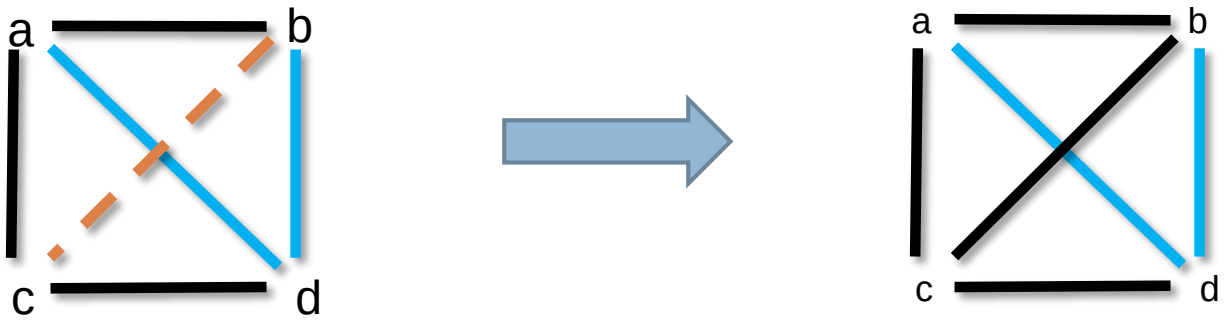
Il se peut que l'on n'ait pas assez d'informations sur une relation d'orthologie/paralogie

e.g. gènes qui ont un seuil BLAST "borderline"



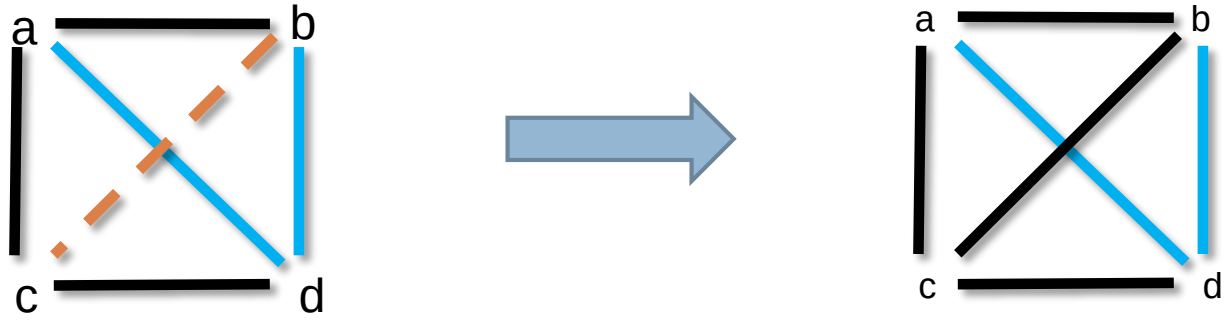
Problème : soit R une graphe de relations avec des relations "inconnues", peut-on choisir ces inconnues pour rendre R:

- **satisfaisable**?
- **S-Consistant**?



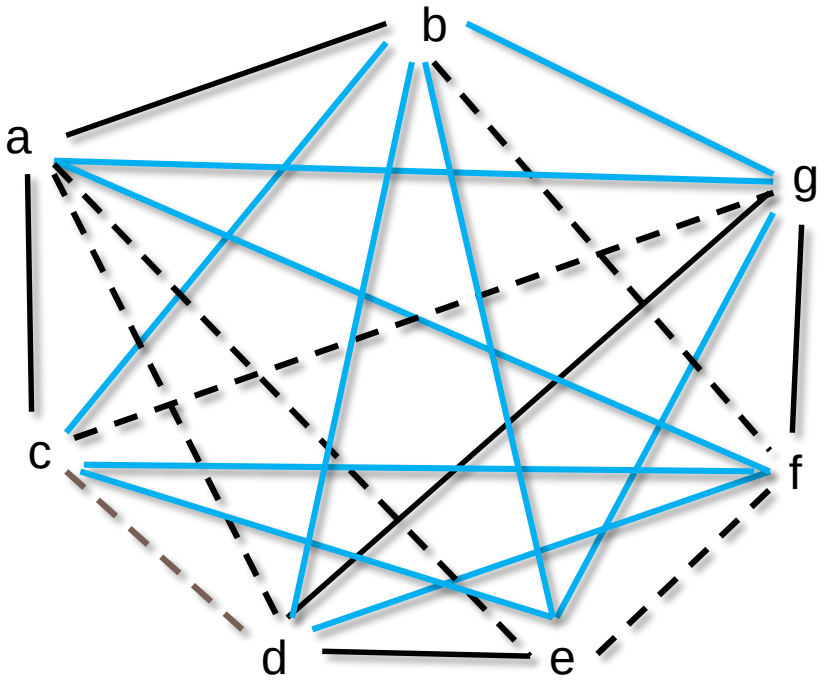
Problème : soit R une graphe de relations avec des relations "inconnues", peut-on choisir ces inconnues pour rendre R:

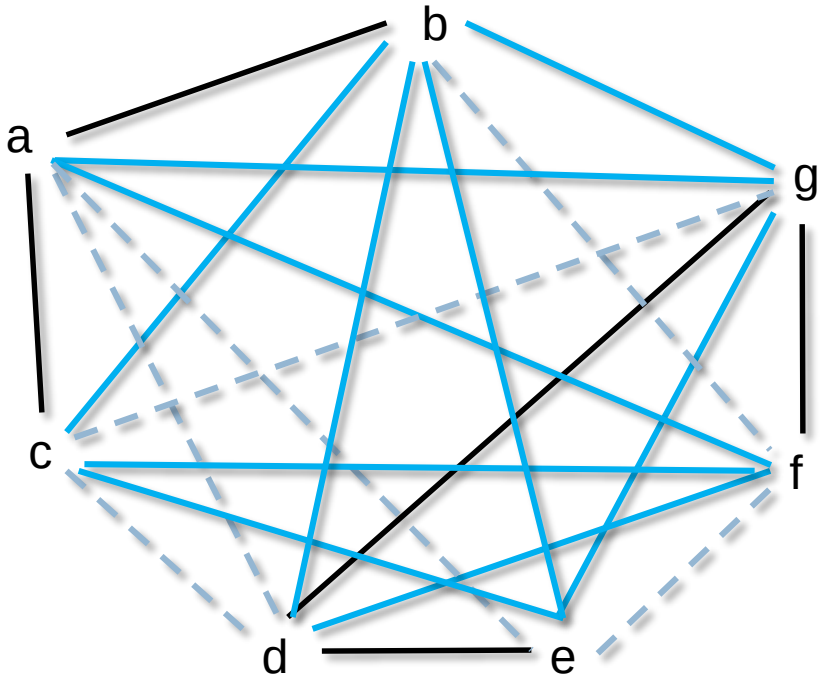
- **satisfaisable**? O(n^3)
- **S-Consistent**? O(n^3)



Idée de l'algorithme

Essayer "tout noir", et "tout bleu".

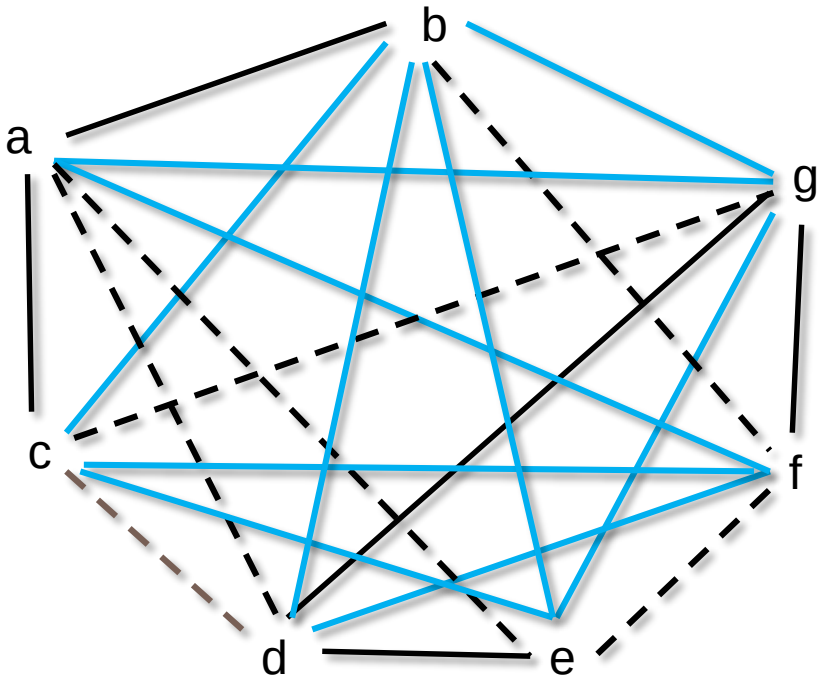


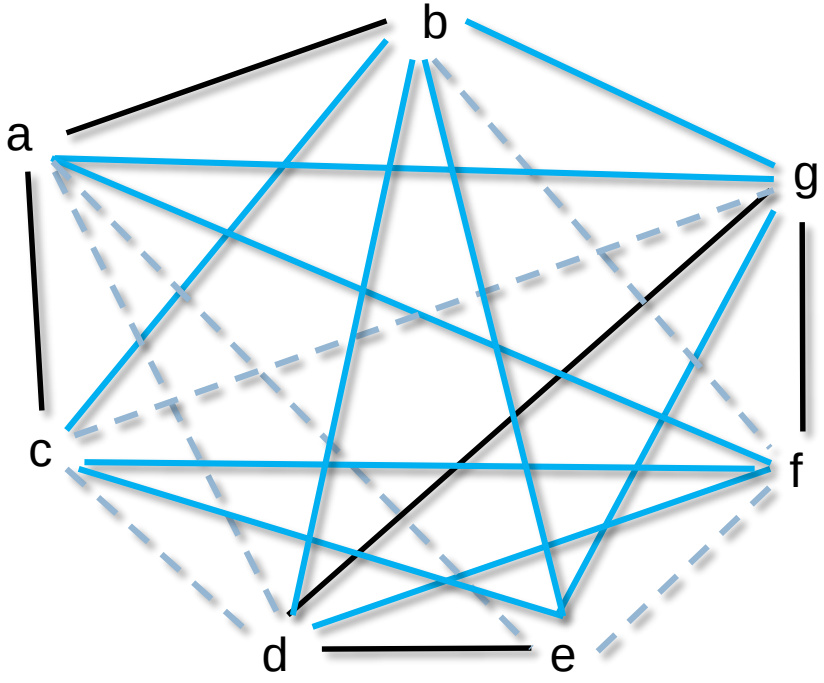


Idée de l'algorithme

Essayer "tout noir", et "tout bleu".

Si R_{BLACK} et R_{BLUE} sont connexes dans les 2 cas, pas de solution.





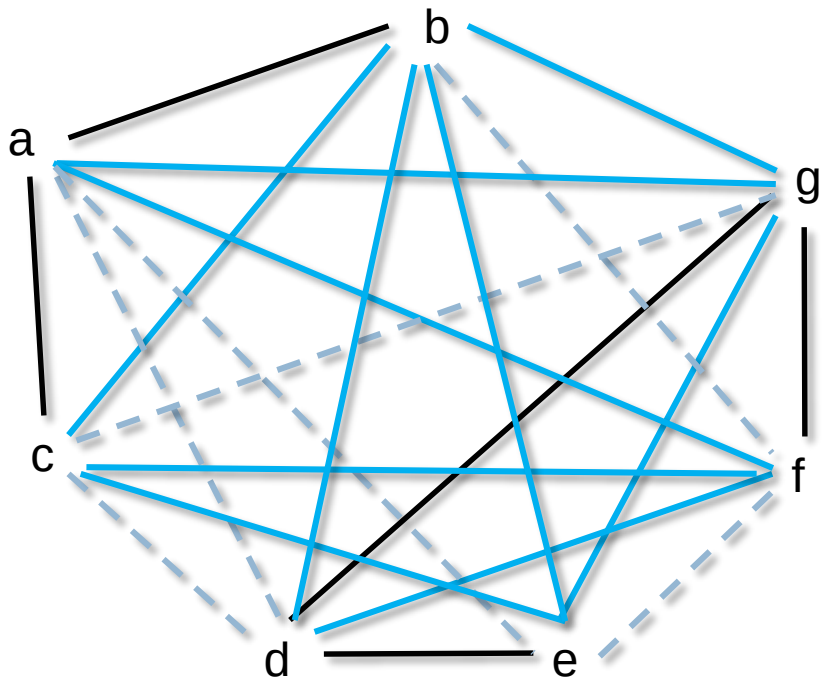
Idée de l'algorithme

Essayer "tout noir", et "tout bleu".

Si R_{BLACK} et R_{BLUE} sont connexes dans les 2 cas, pas de solution.

Ici, R_{BLACK} de "tout bleu"

n'est pas connexe.



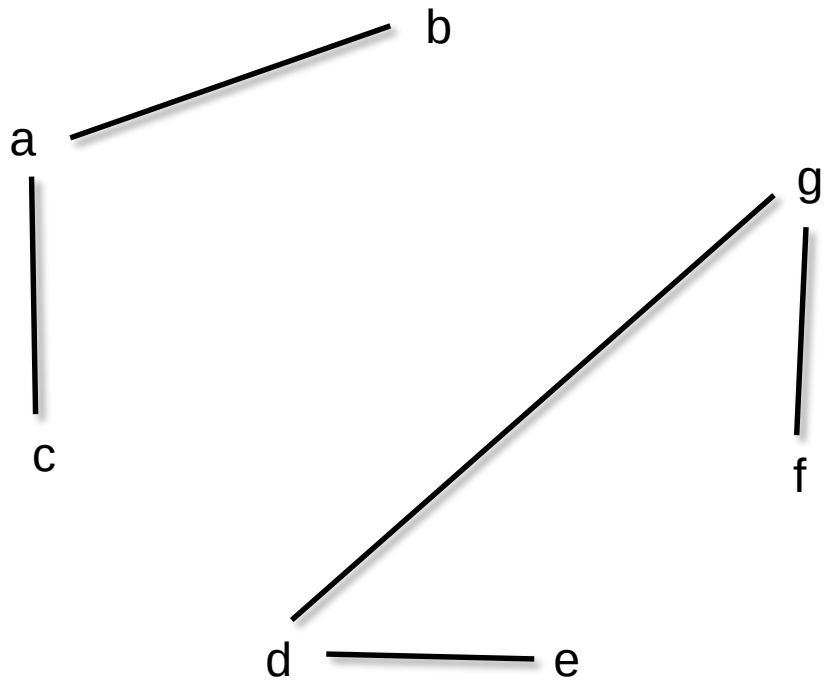
Idée de l'algorithme

Essayer "tout noir", et "tout bleu".

Si R_{BLACK} et R_{BLUE} sont connexes dans les 2 cas, pas de solution.

Ici, R_{BLACK} de "tout bleu"

n'est pas connexe.



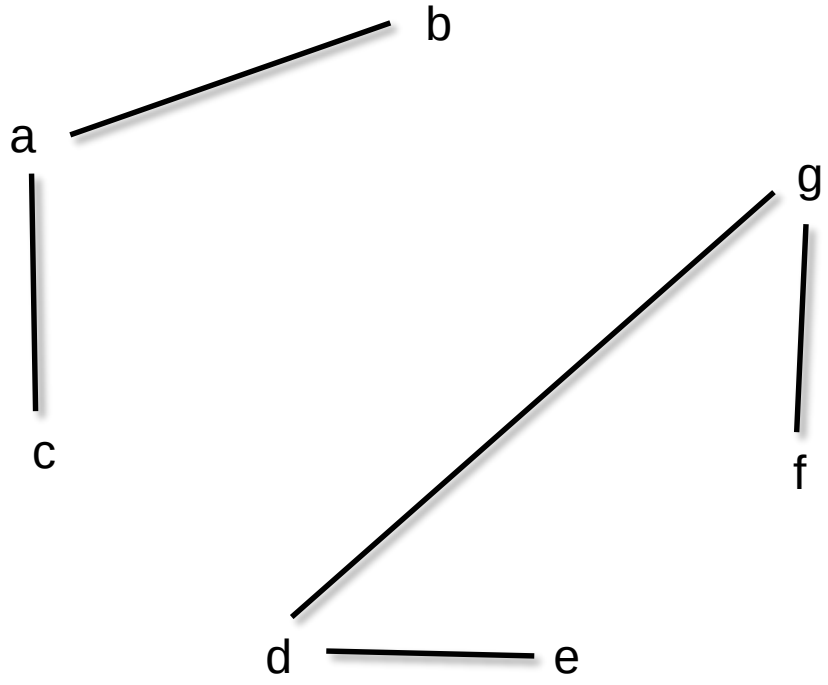
Idée de l'algorithme

Essayer "tout noir", et "tout bleu".

Si R_{BLACK} et R_{BLUE} sont connexes dans les 2 cas, pas de solution.

Ici, R_{BLACK} de "tout bleu" n'est pas connexe.

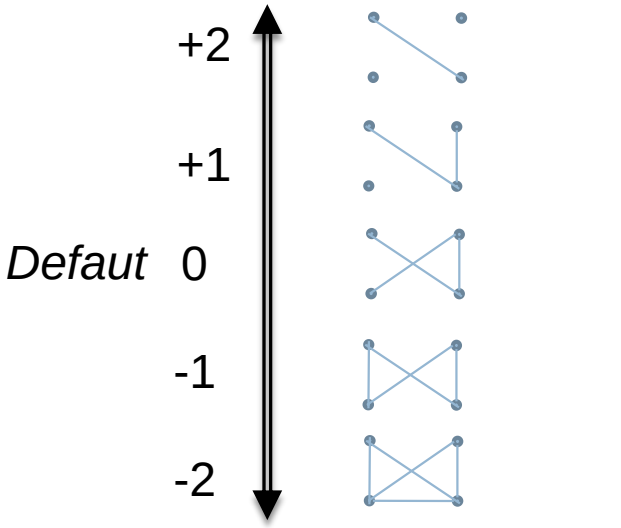
Bien! On répète sur chaque composante connexe.



Expériences

Nous avons inféré 265 graphes d'orthologie/paralogie avec **ProteinOrtho**, avec 5 jeux de paramètres {-2, -1, 0, +1, +2}.

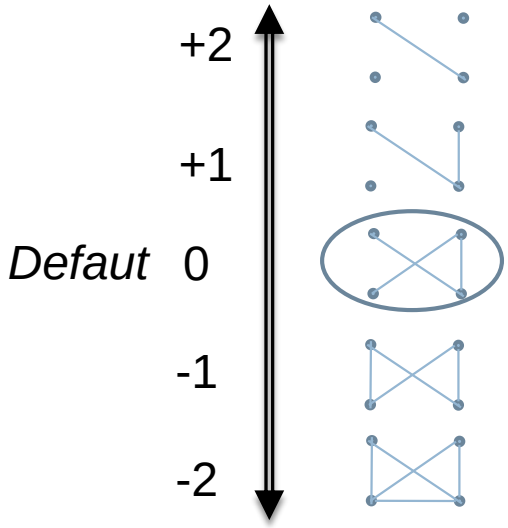
Strict => Moins d'orthologies



Lousse => Plus d'orthologies

Expériences

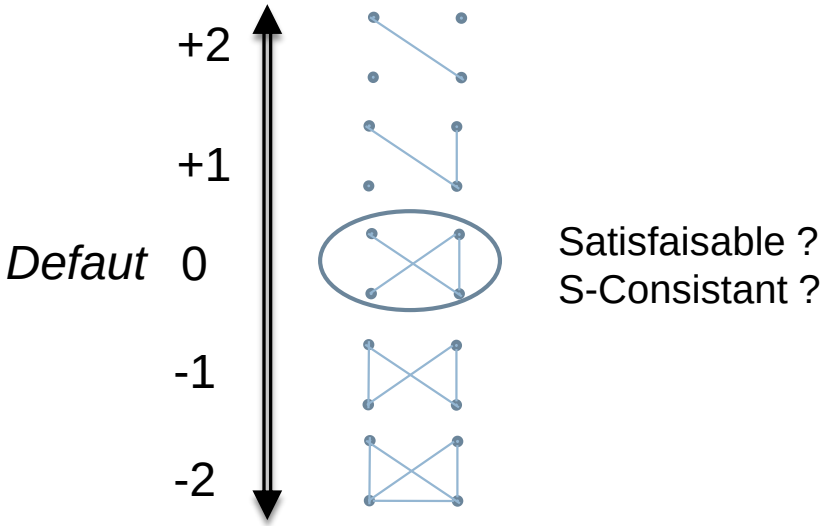
Strict => Moins d'orthologies



Lousse => Plus d'orthologies

Expériences

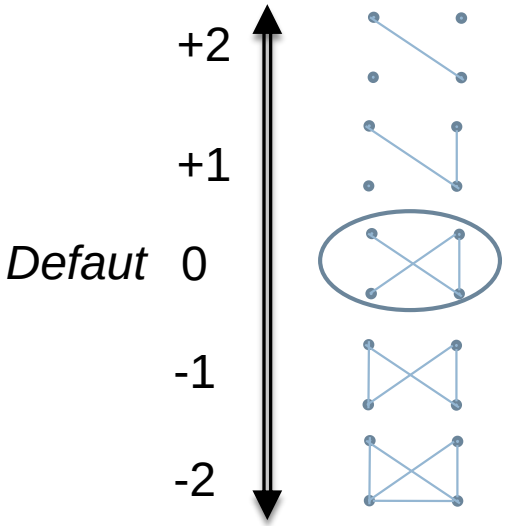
Strict => Moins d'orthologies



Lousse => Plus d'orthologies

Expériences

Strict => Moins d'orthologies



Satisfaisable ? NON (~90% familles)
S-Consistant ? NON (~96% familles)

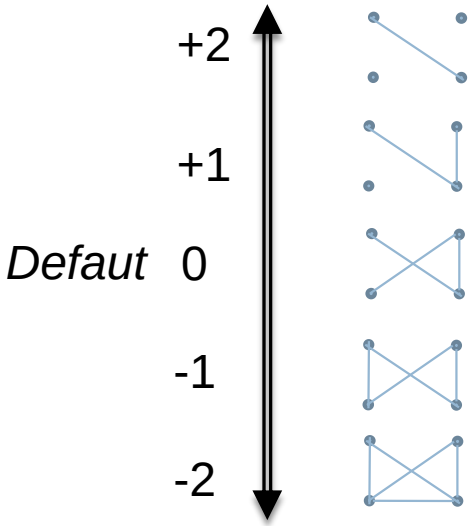
Lousse => Plus d'orthologies

Expériences

			NON Satisfaisable	NON S-Consistant
Strict => Moins d'orthologies				
	+2		80%	93%
	+1		82%	95%
Defaut	0		90%	96%
	-1		83%	95%
	-2		70%	89%
Lousse => Plus d'orthologies				

Expériences avec l'inconnu

Strict => Moins d'orthologies

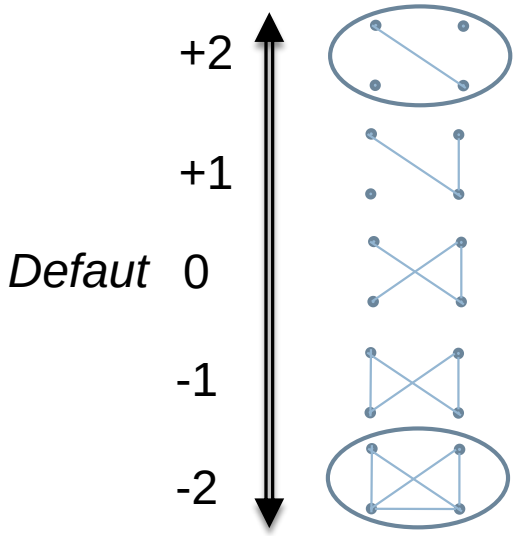


Lousse => Plus d'orthologies

OK, ce n'est presque jamais satisfaisable. Mais peut-on trouver des relations robustes?

Expériences avec l'inconnu

Strict => Moins d'orthologies



Lousse => Plus d'orthologies

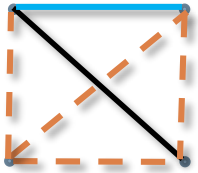
OK, ce n'est presque jamais satisfaisable. Mais peut-on trouver des relations robustes?

Expériences avec l'inconnu

+2

-2

Si strict et lousse sont en accord, on devrait pouvoir faire confiance. Le reste est posé comme "inconnu".

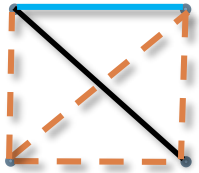


Expériences avec l'inconnu

+2

-2

Si strict et lousse sont en accord, on devrait pouvoir faire confiance. Le reste est posé comme "inconnu".



En combinant +2/-2, les relations partielles deviennent

satisfiable pour 98% des familles

S-consistent pour 65% des familles

Expériences avec l'inconnu

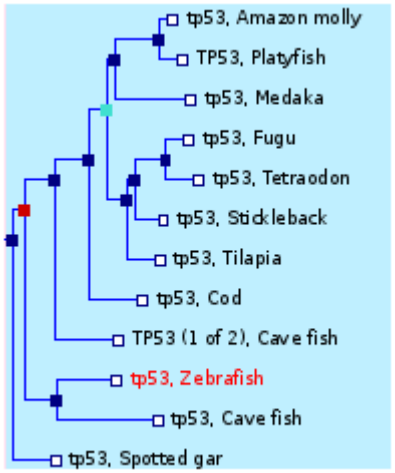
		NON Satisfaisable	NON S-Consistant
	-2/+2	1.9%	35.1%
	-2/+1	2.6%	35.1%
	-1/+1	4.2%	44.8%
	-1/+2	4.1%	40.8%

Superarbres de gènes et réconciliation

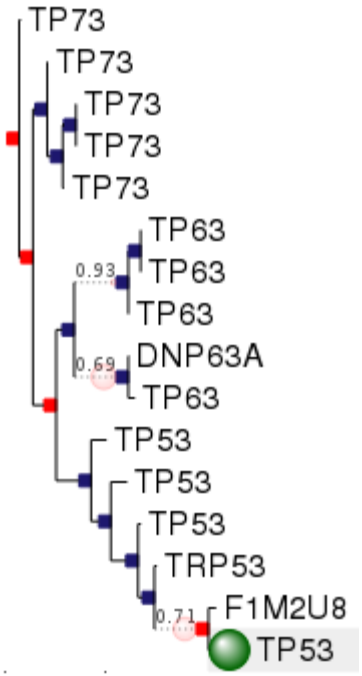
Reconstructing a SuperGeneTree minimizing reconciliation
M Lafond, A Ouangraoua, N El-Mabrouk
BMC bioinformatics 16 (2015)

TP53: plusieurs arbres de gènes

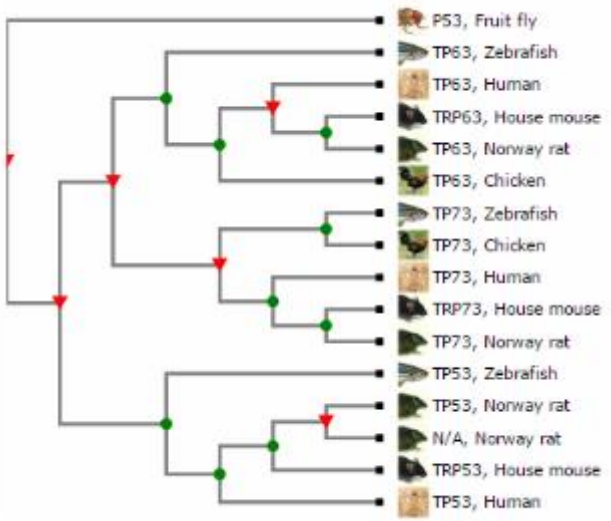
Ensembl

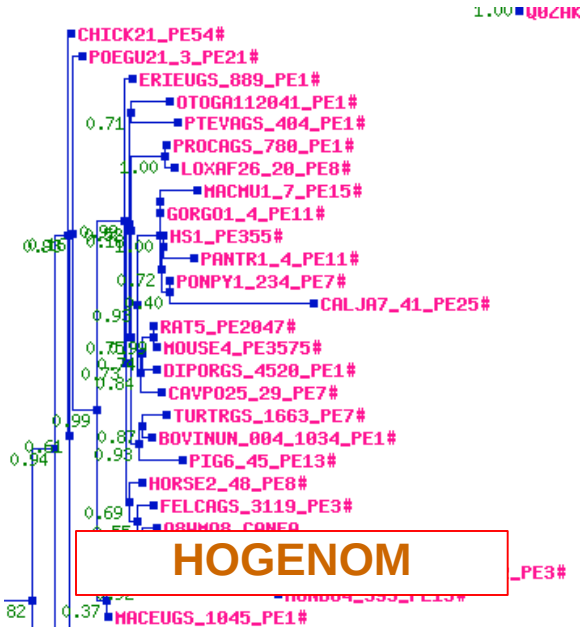


PhylomeDB



TreeFam





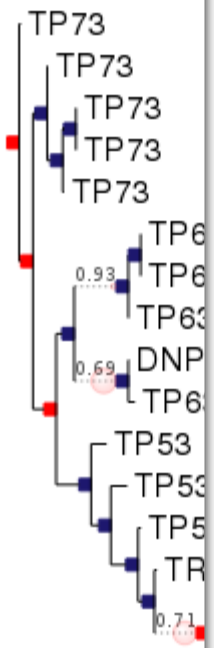
HOGENOM

TP53: plusieurs arbres de gènes

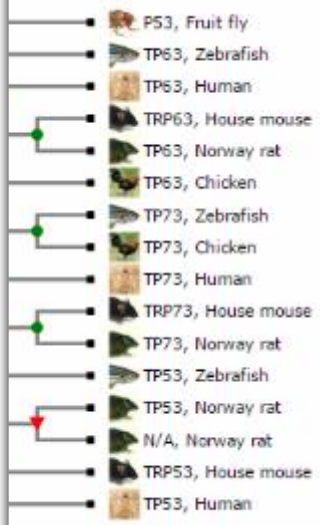
Ensembl

Phylome

Ensembl + PhylomeDB + TreeFam + HOGENOM + …

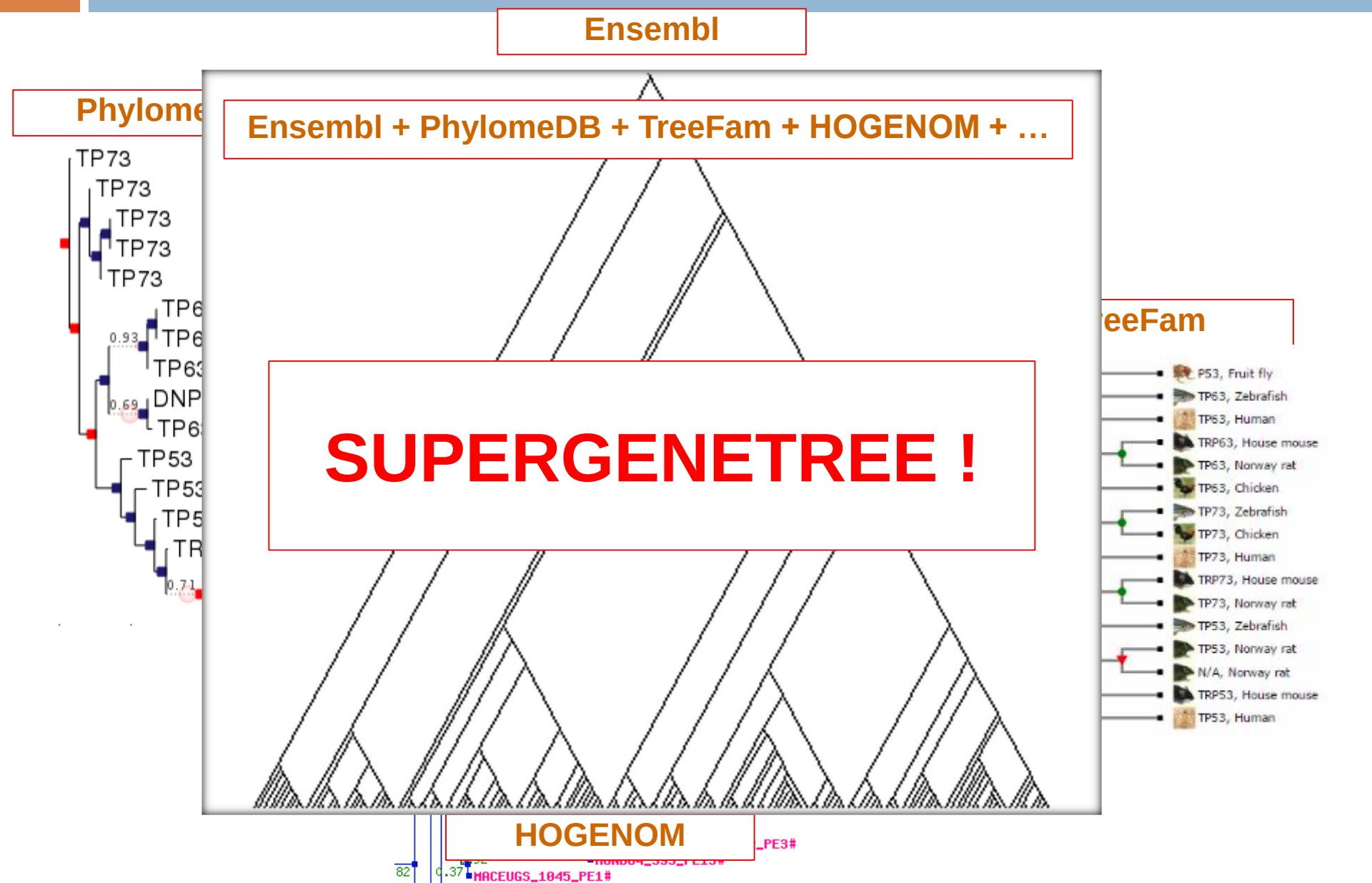


eeFam

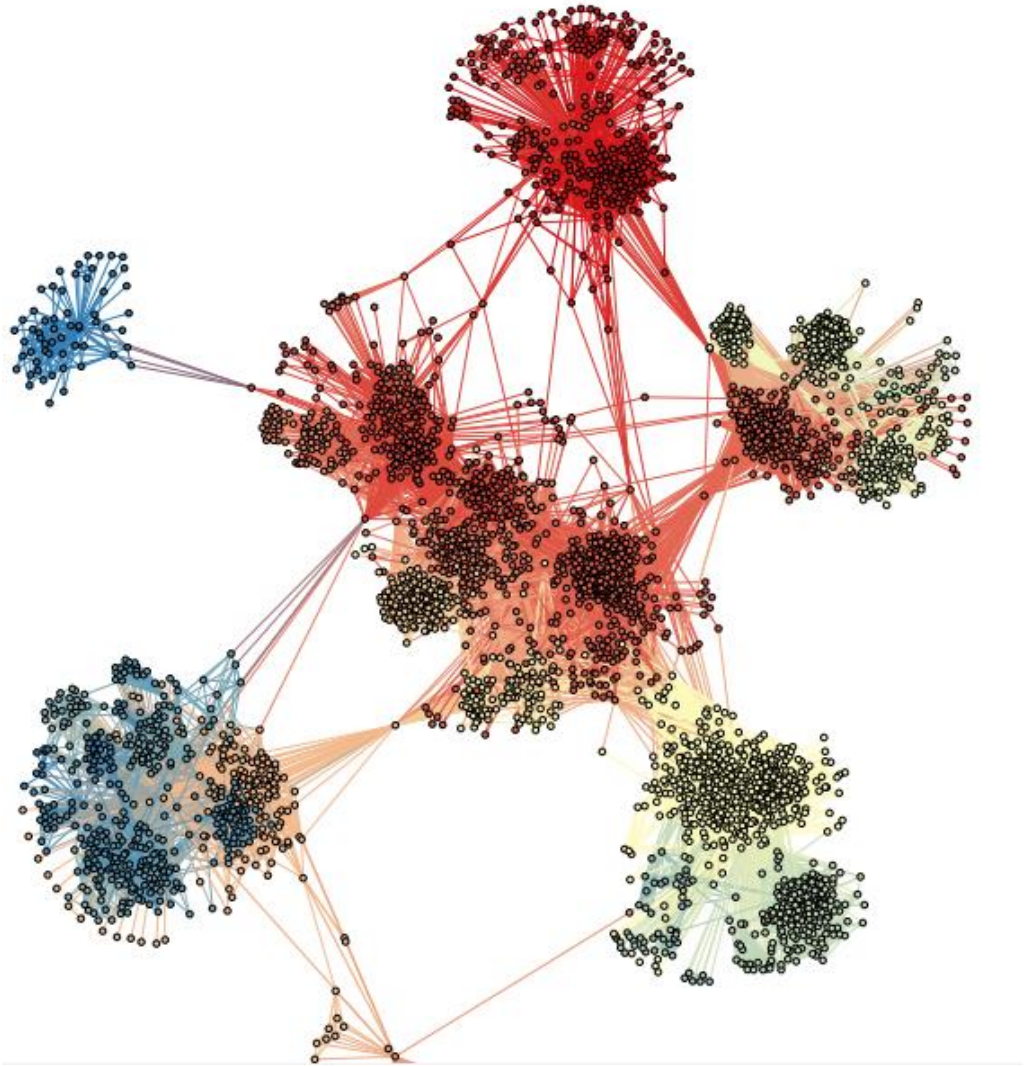


HOGENOM

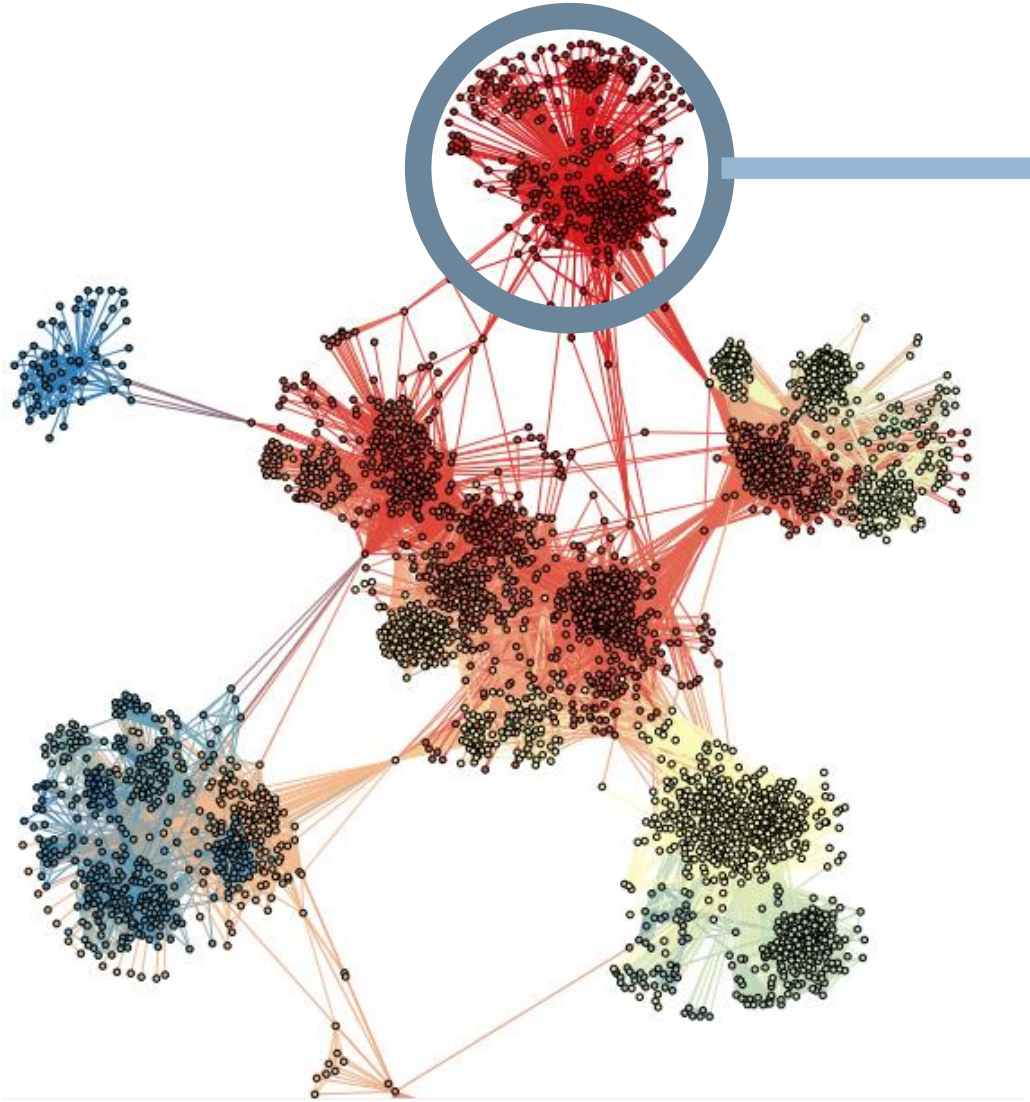
TP53: plusieurs arbres de gènes

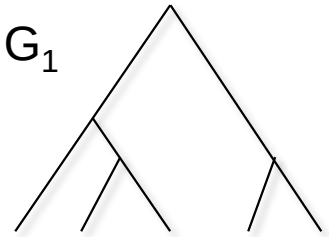


Clusters d'orthologues

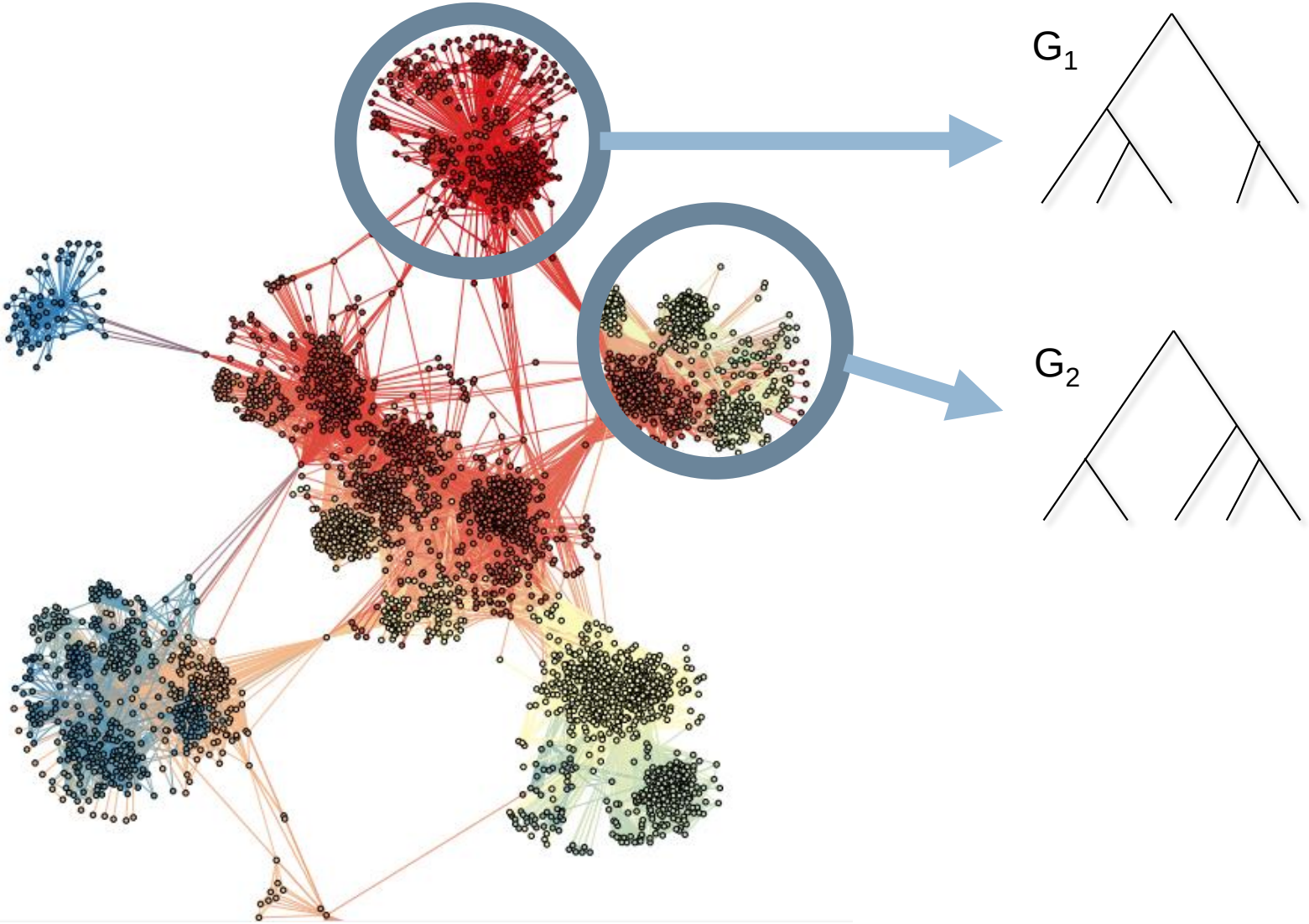


Clusters d'orthologues

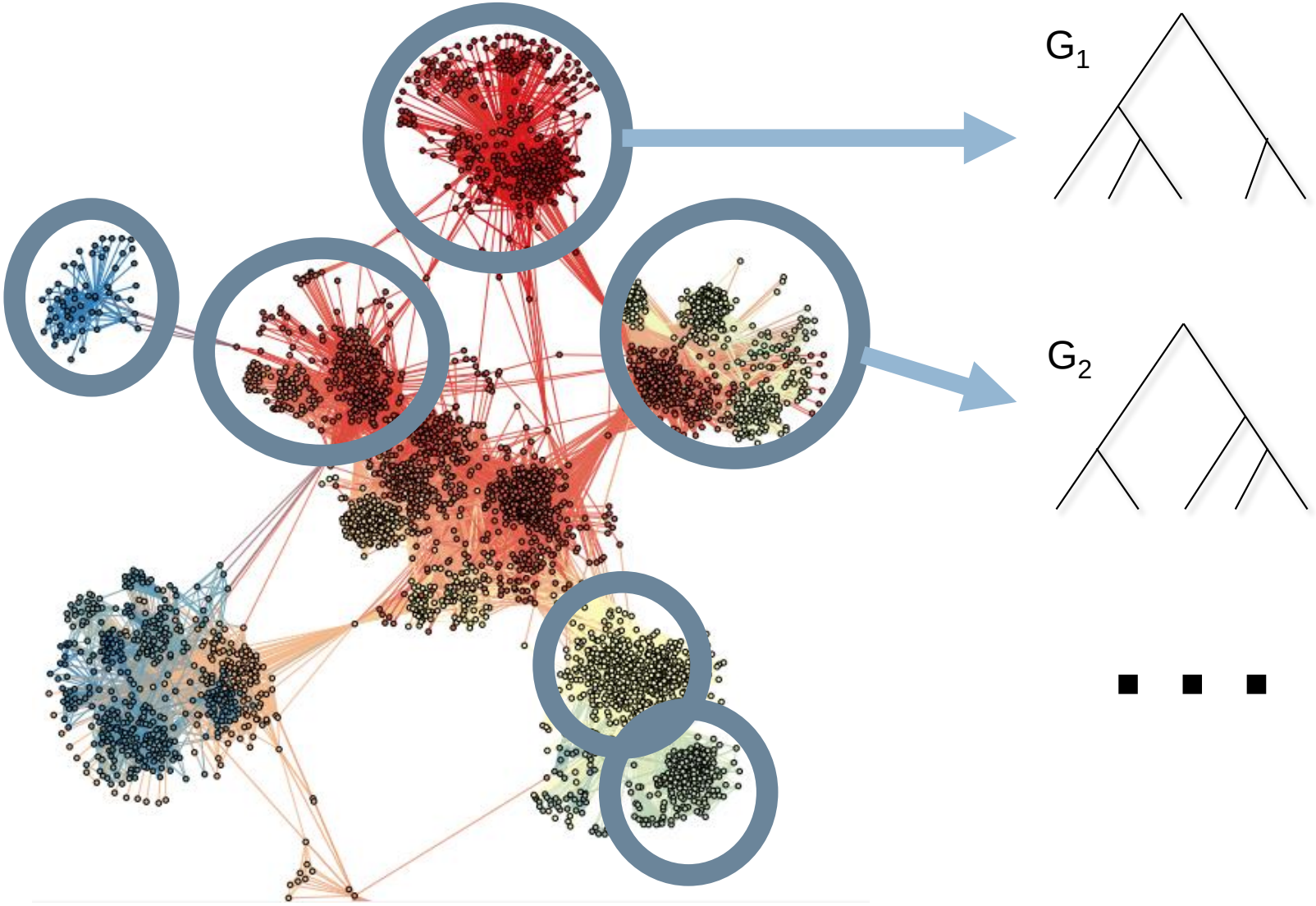




Clusters d'orthologues



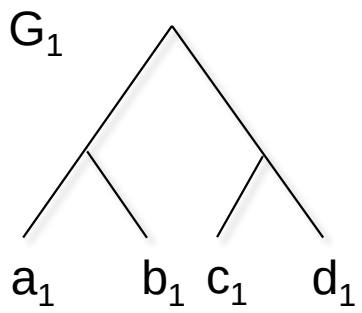
Clusters d'orthologues

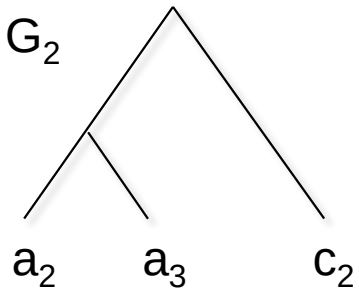


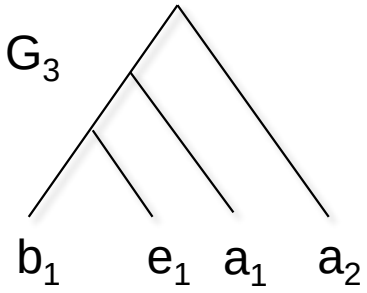
Clusters d'orthologues

Problème du superarbre

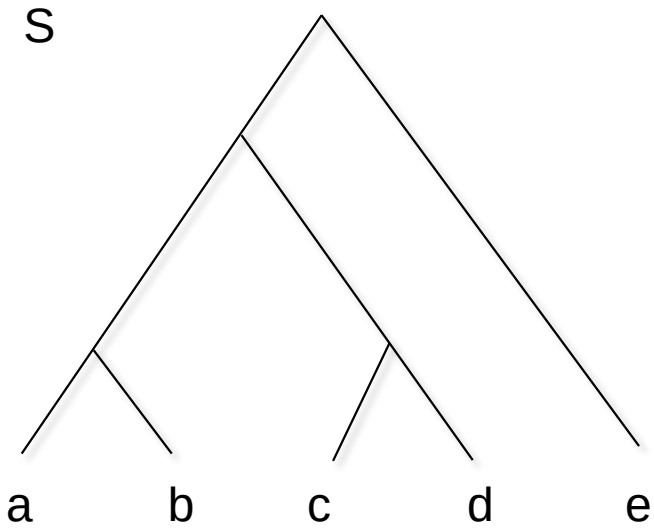
Arbres de gènes







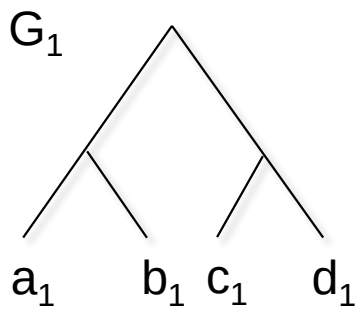
Arbre d'espèces

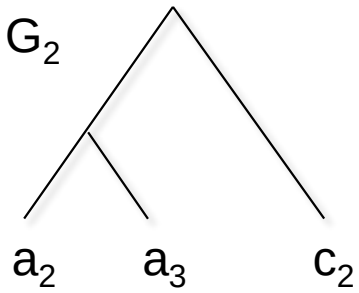


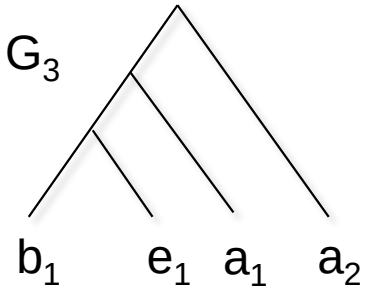
- **Étiquettes des gènes = espèces**
- **Copies multiples (paralogs)**
  - **e.g. a_1, a_2, a_3**

Problème du superarbre

Arbres de gènes



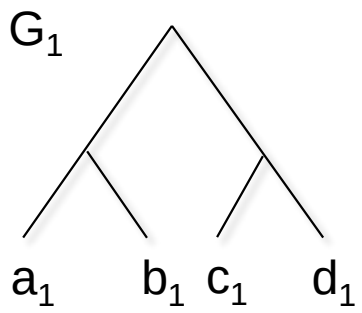


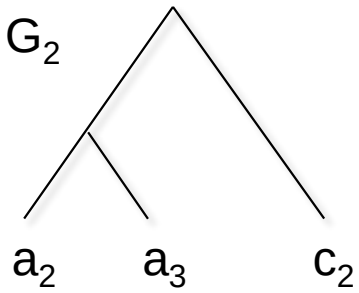


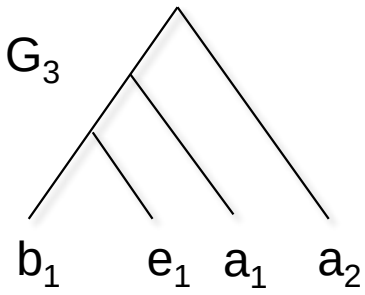
- **Objectif : trouver un superarbre de gènes qui les contient tous**

Problème du superarbre

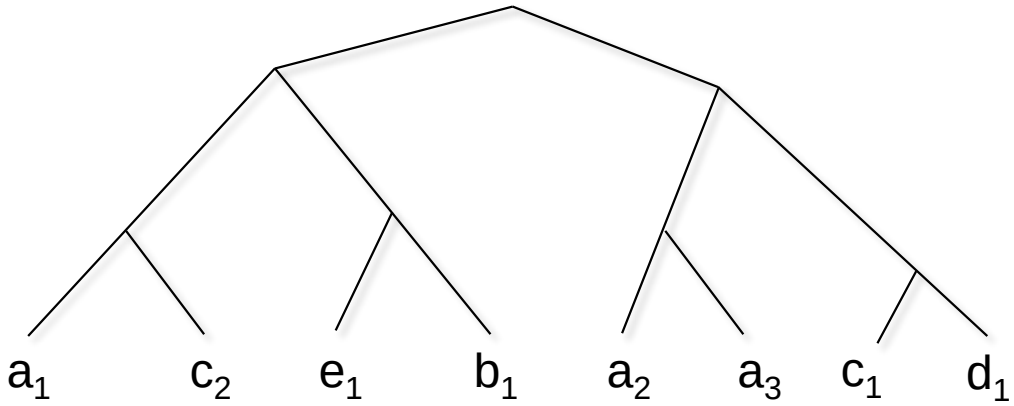
Arbres de gènes







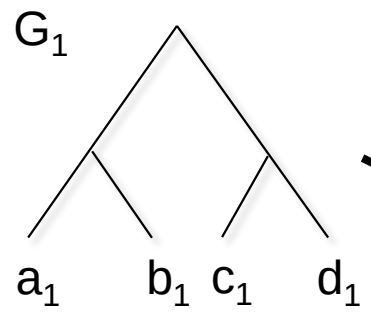
- **Objectif : trouver un superarbre de gènes qui les contient tous**

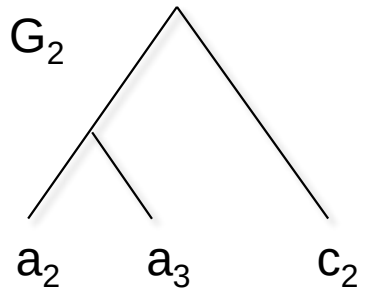


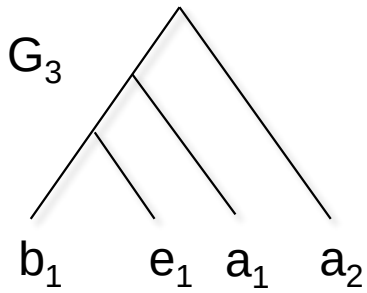
Problème du superarbre

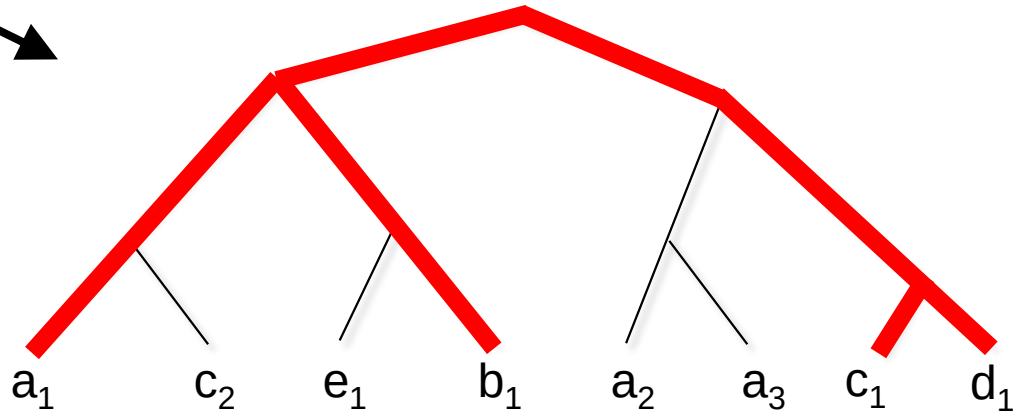
Arbres de gènes

- **Objectif : trouver un superarbre de gènes qui les contient tous**





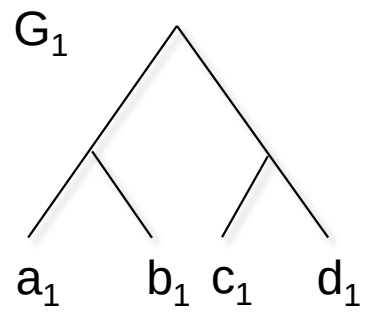


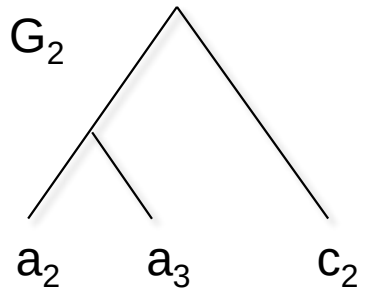


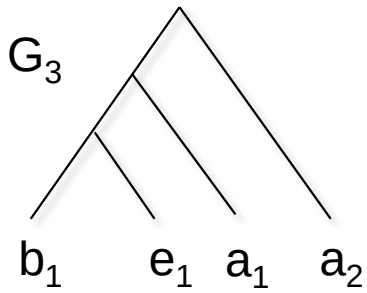
Problème du superarbre

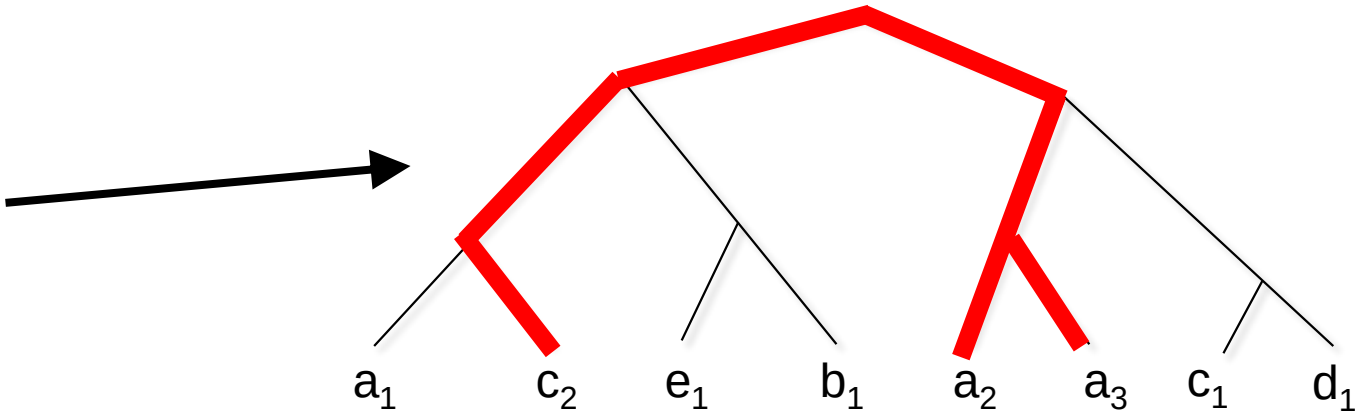
Arbres de gènes

- **Objectif : trouver un superarbre de gènes qui les contient tous**





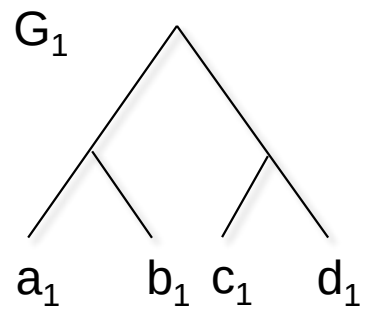


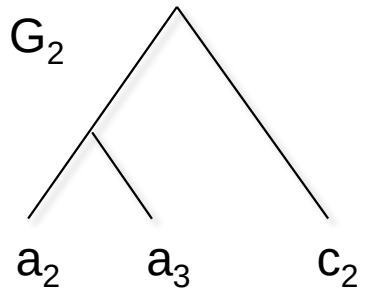


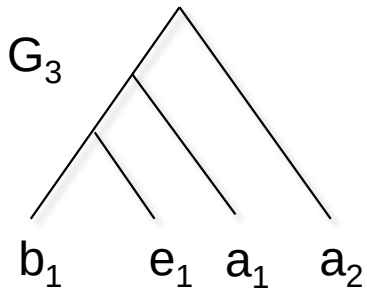
Problème du superarbre

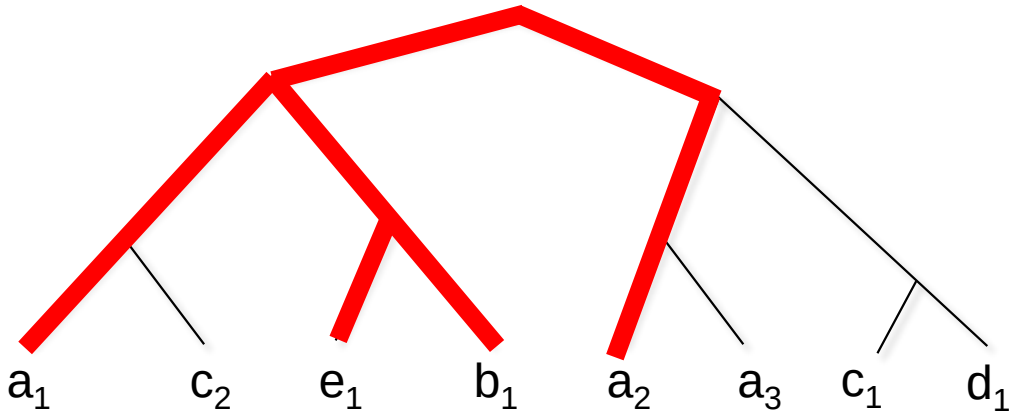
Arbres de gènes

- **Objectif : trouver un superarbre de gènes qui les contient tous**









Superarbres de gènes

- Nos arbres sont dits **compatibles** s'il existe un superarbre les contenant tous
- Trouver un superarbre (ou déterminer l'incompatibilité) est un vieux problème
  - Algorithme BUILD *(Aho & al., 1981)*
- Qu'est-ce qui est différent avec les superarbres de **gènes** ?

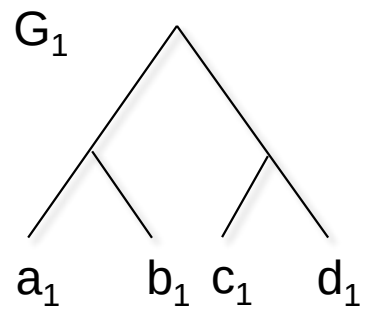
Superarbres de gènes

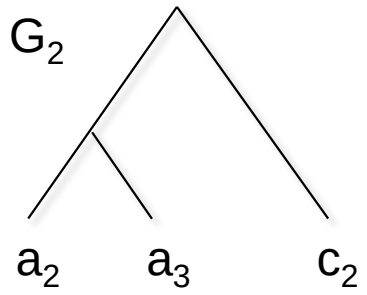
- Nos arbres sont dits **compatibles** s'il existe un superarbre les contenant tous
- Trouver un superarbre (ou déterminer l'incompatibilité) est un vieux problème
  - Algorithme BUILD *(Aho & al., 1981)*
- Qu'est-ce qui est différent avec les superarbres de **gènes** ?
  - Réconciliation

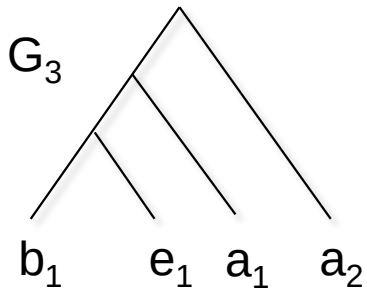
Superarbres de gènes

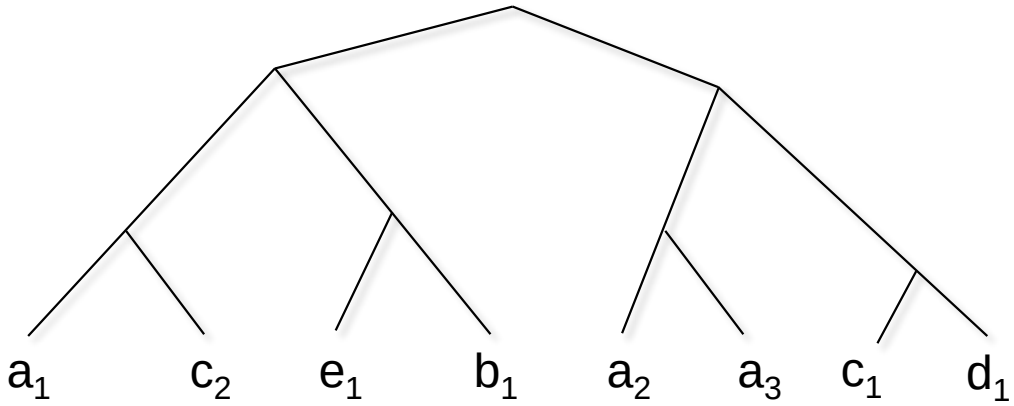
- Il peut exister un nombre exponentiel de superarbres.
  - Lequel est le "meilleur"?
- Nous avons exploré des façons de choisir un tel superarbre selon les informations de l'arbre d'espèces S
- Plus spécifiquement, nous utilisons la réconciliation avec S pour choisir un superarbre.

Superarbre de gènes

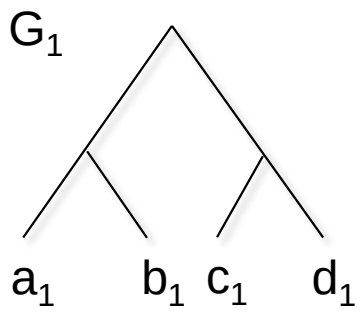


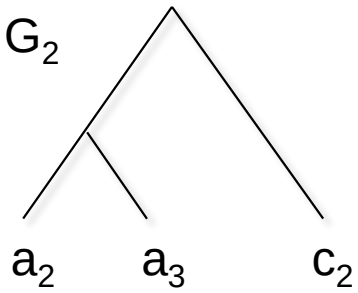


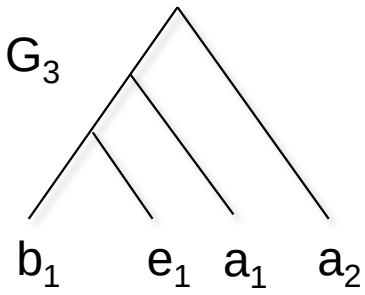


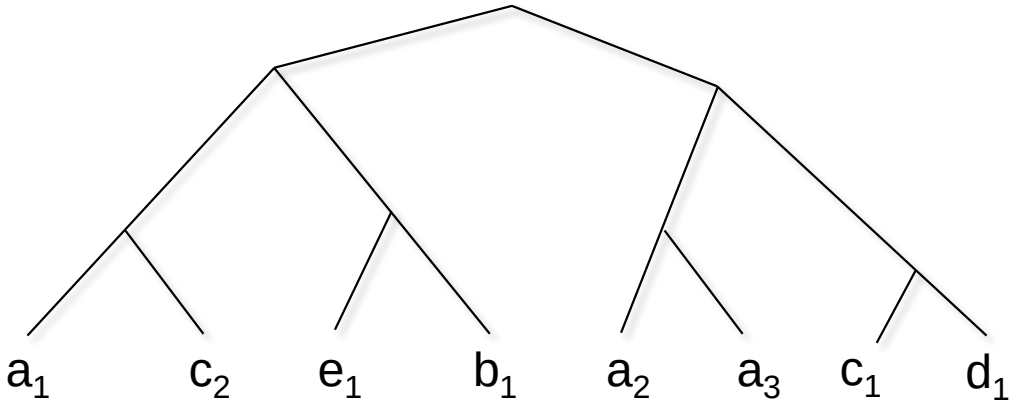


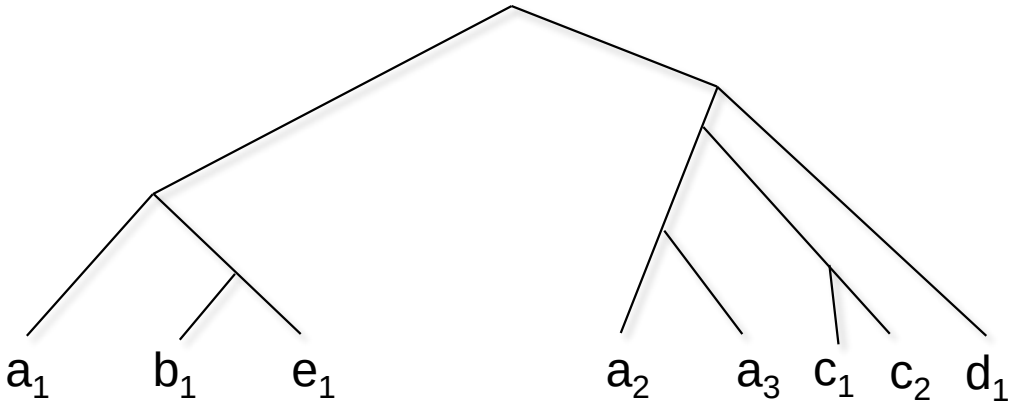
Superarbre de gènes



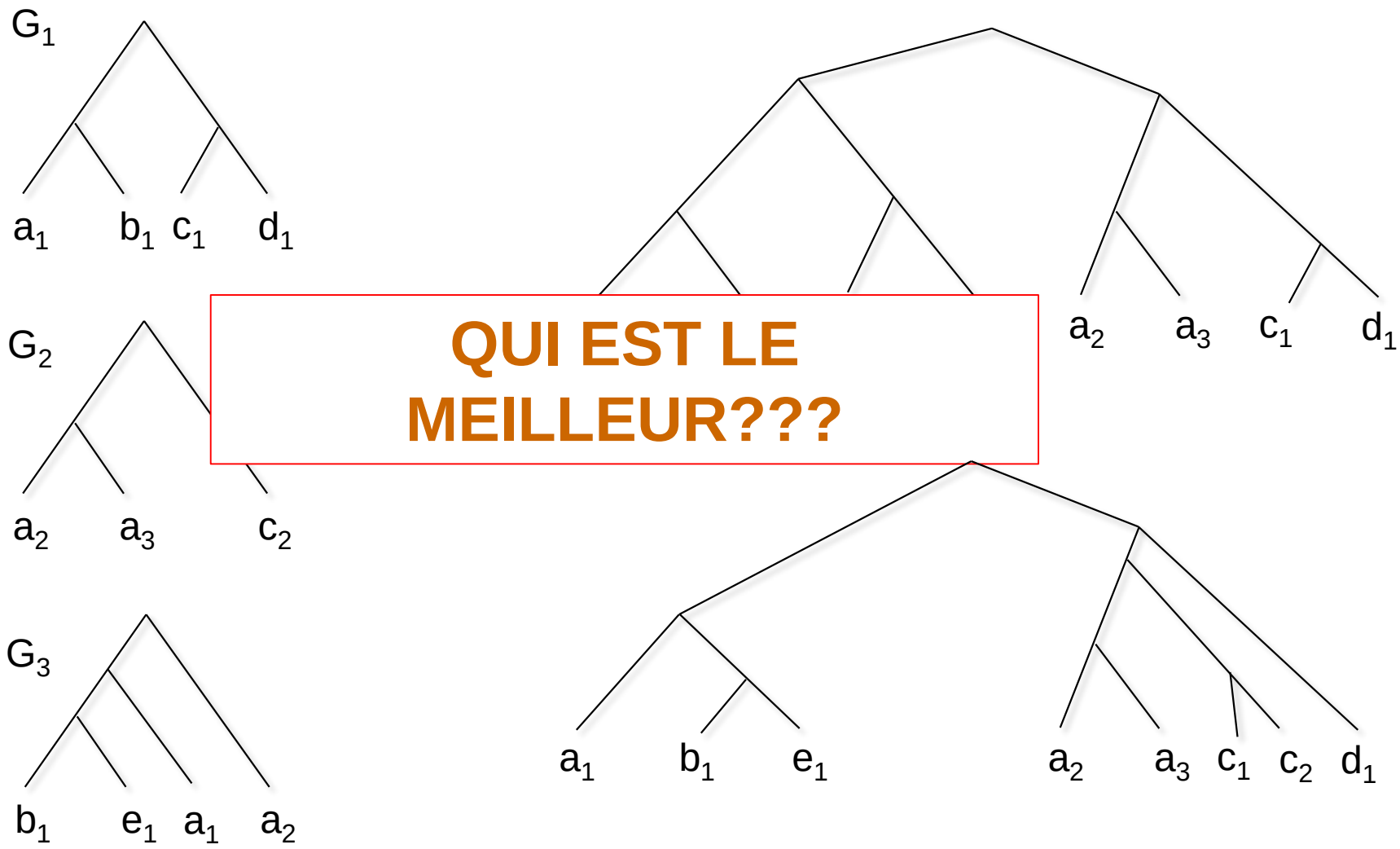




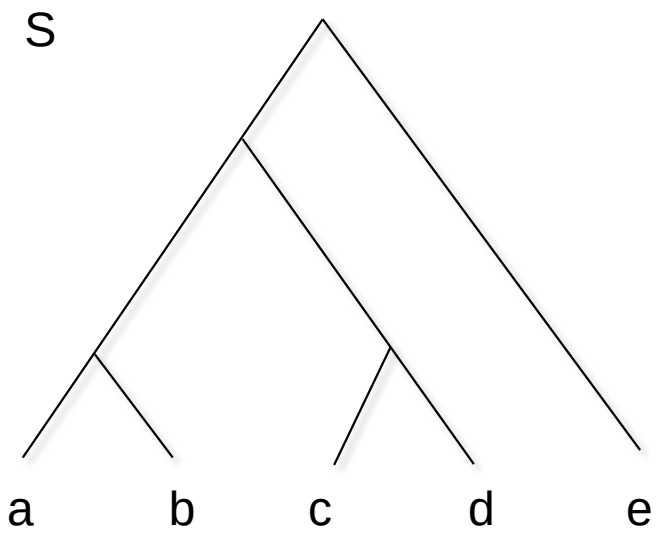


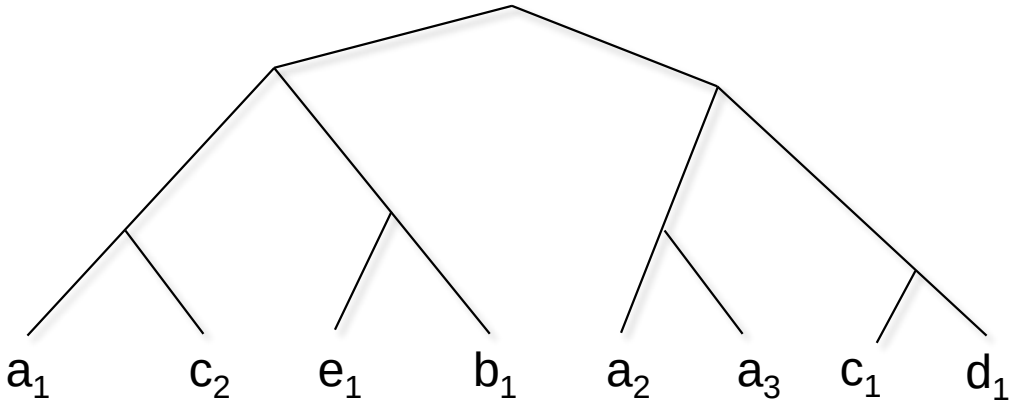


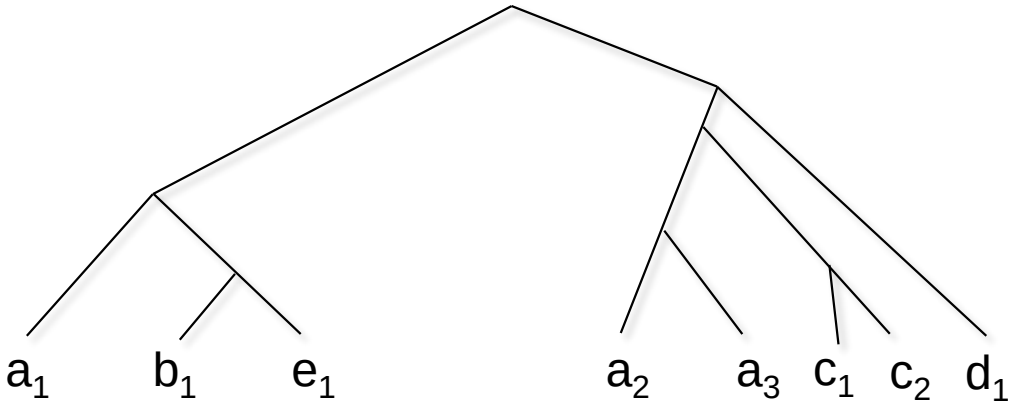
Superarbre de gènes



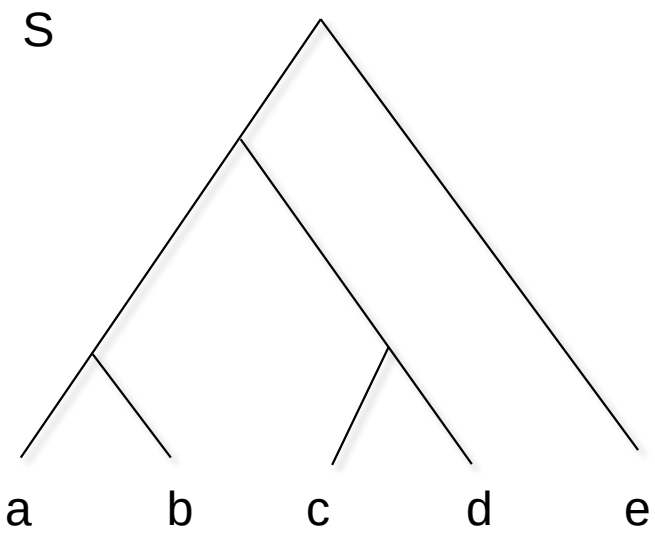
Superarbre de gènes

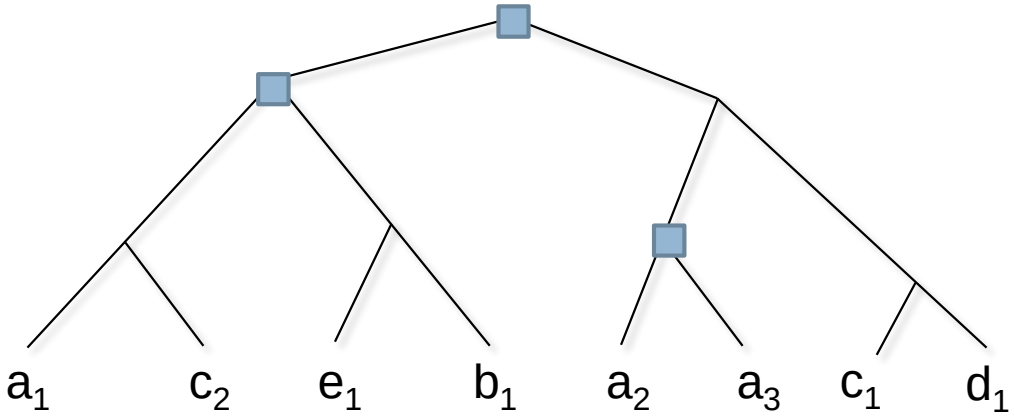


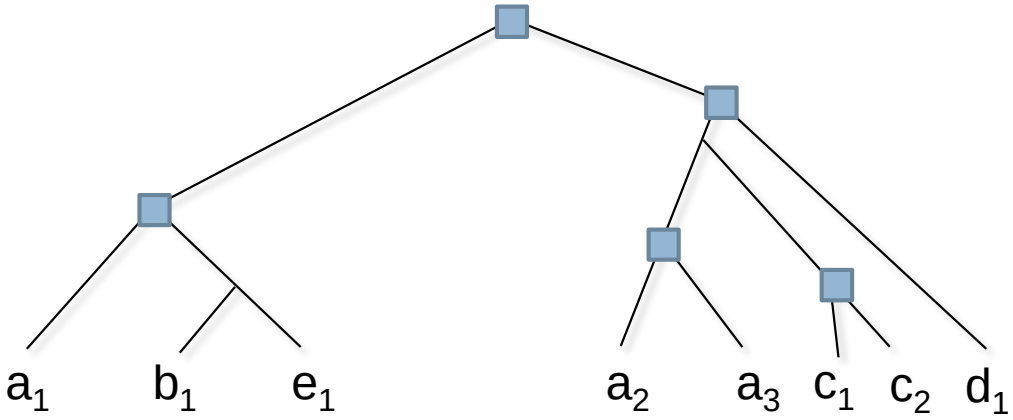




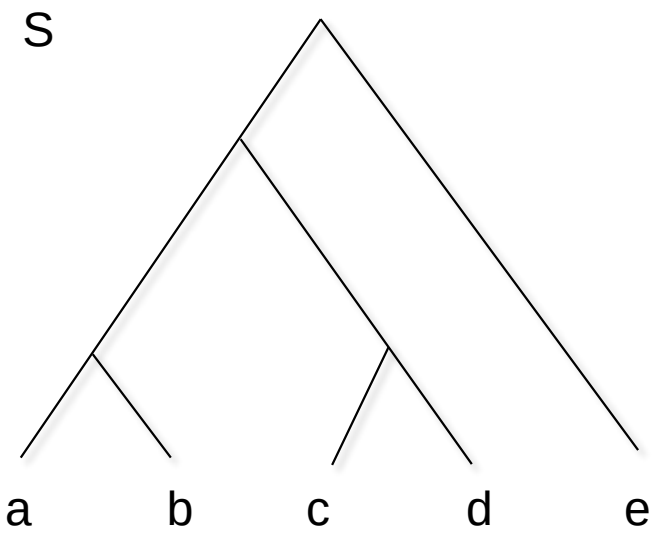
Superarbre de gènes

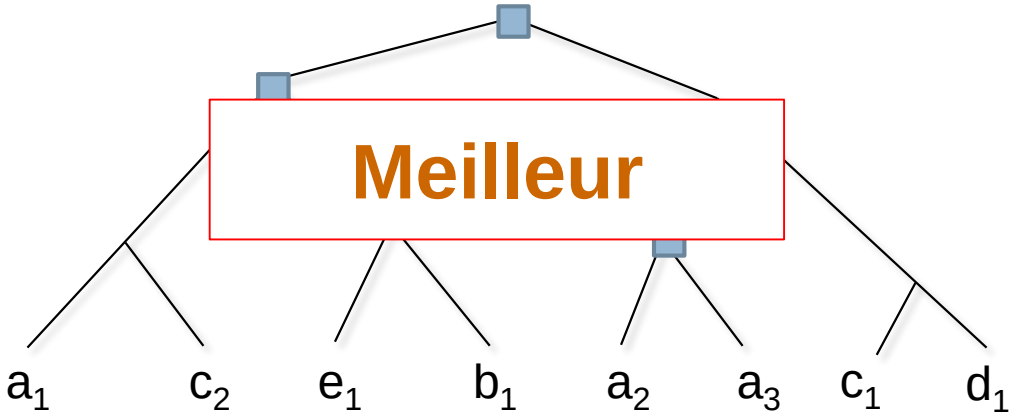


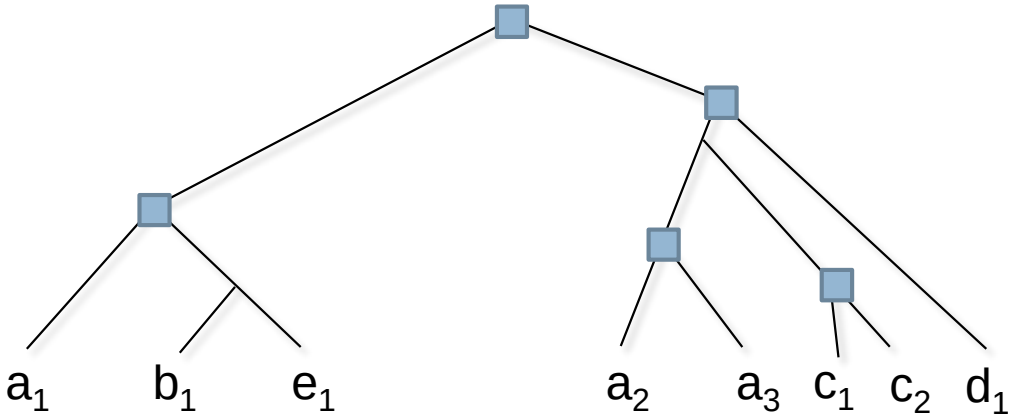




Superarbre de gènes







Problème du superabre de gènes

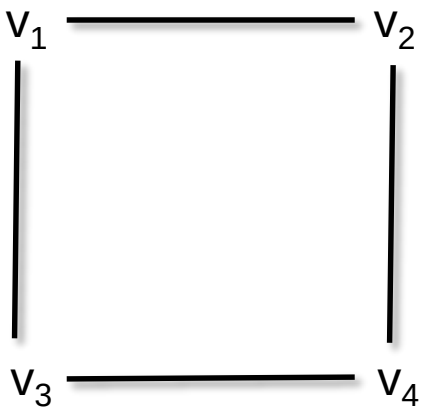
- **Entrée:** un ensemble d'arbres de gènes compatibles $G = \{G_1, \ldots, G_k\}$ et un arbre d'espèces S
- **Sortie:** un superarbre de gènes G* qui:
  - contient tous les arbres de G
  - minimise #dups(G*, S)

Problème du superabre de gènes

- **Entrée:** un ensemble d'arbres de gènes compatibles $G = \{G_1, \ldots, G_k\}$ et un arbre d'espèces S
- **Sortie:** un superarbre de gènes G* qui:
  - contient tous les arbres de G
  - minimise #dups(G*, S)

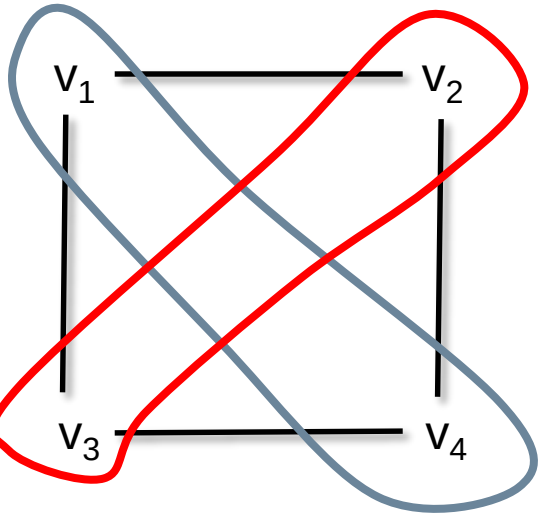
- NP-complet
- Difficile à approximer à un facteur $n^{1-\varepsilon}$ près

Idée de la réduction (en surface)



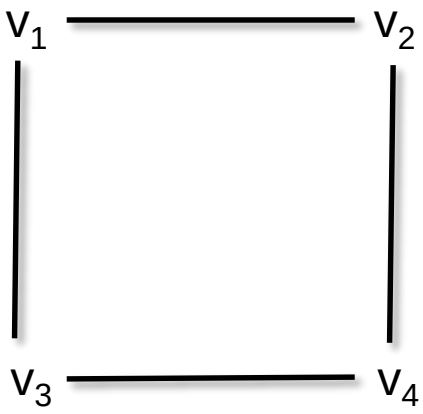
Réduction du problème de "vertex coloring": trouver le min. # de couleurs pour colorier V(G) t.q. les sommets adjacents sont de couleur différente

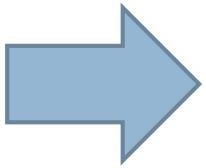
Idée de la réduction (en surface)

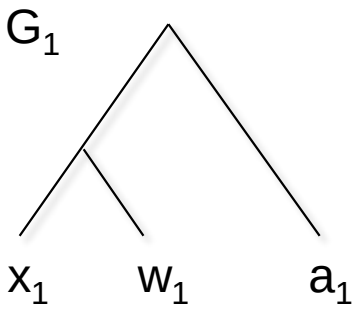


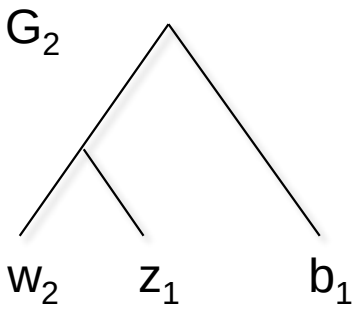
Réduction du problème de "vertex coloring": trouver le min. # de couleurs pour colorier V(G) t.q. les sommets adjacents sont de couleur différente

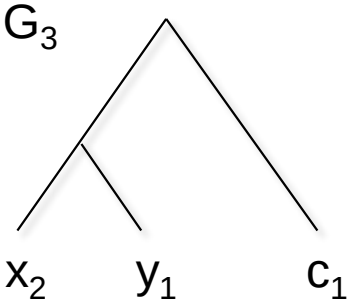
Idée de la réduction (en surface)

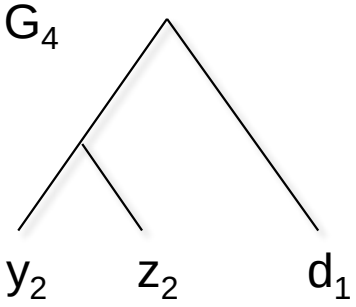






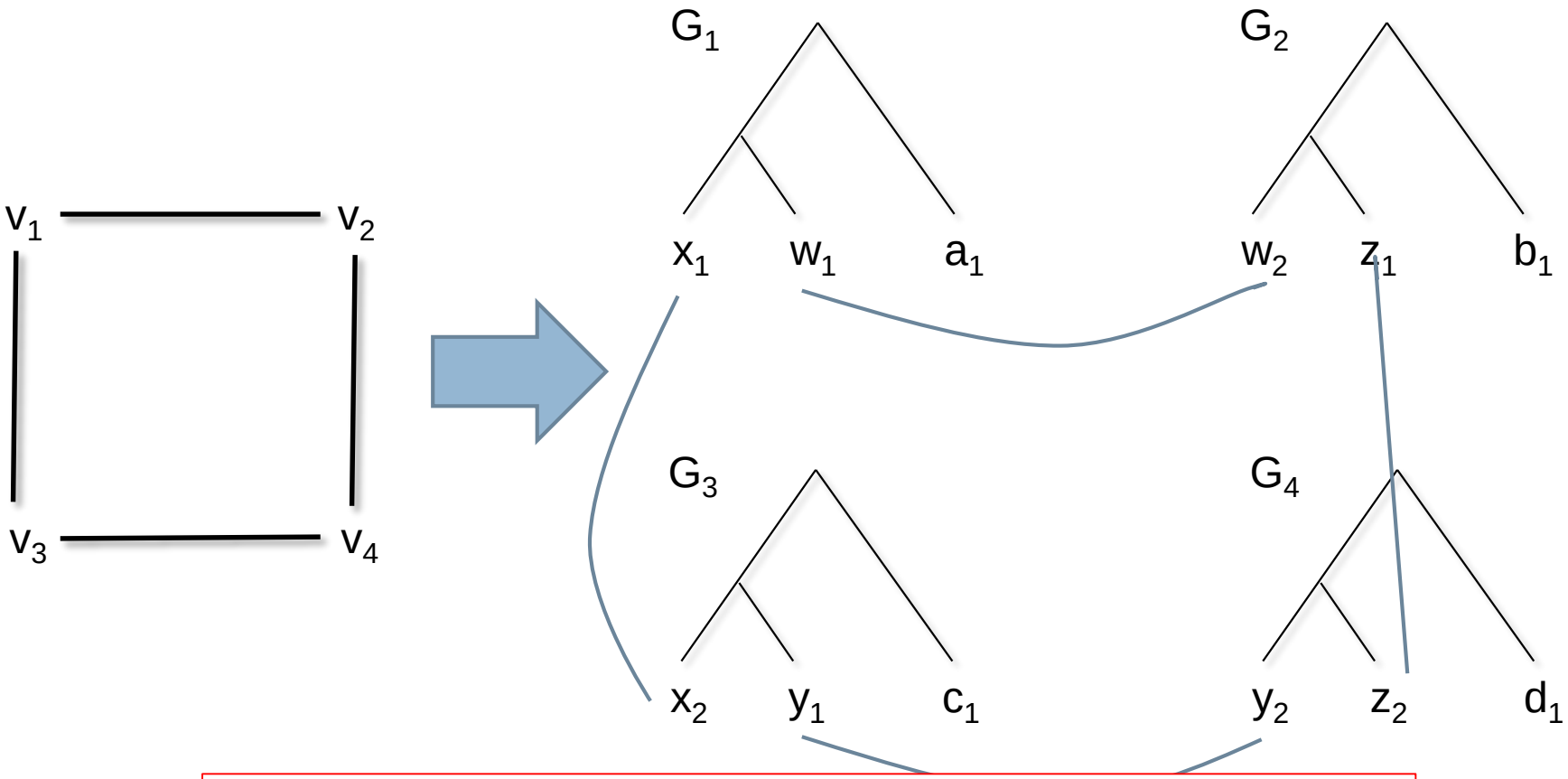






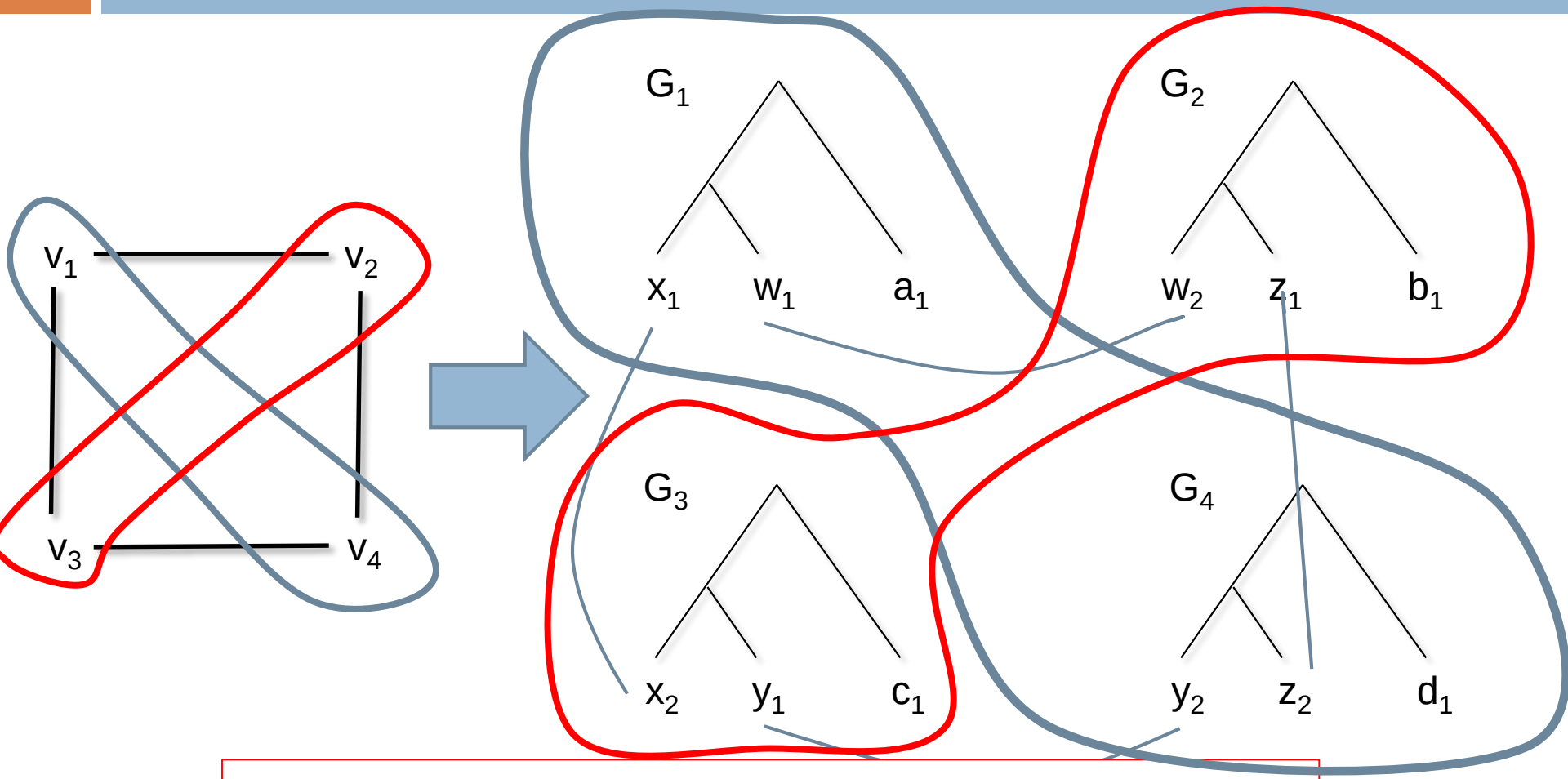
On génère un arbre par sommet tel que v_i, v_j sont adjacents SSI G_i, G_j partagent une étiquette.

Idée de la réduction (en surface)



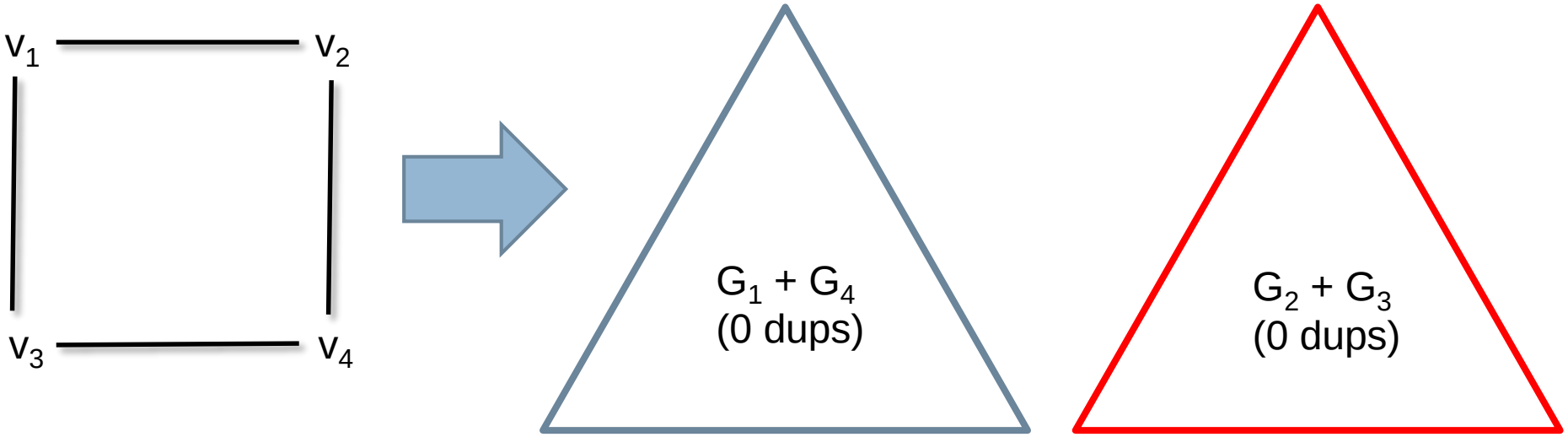
On génère un arbre par sommet tel que v_i, v_j sont adjacents SSI G_i, G_j partagent une étiquette.

Idée de la réduction (en surface)



Un coloriage est équivalent à partitionner les arbres en sous-arbres d'espèces disjointes.

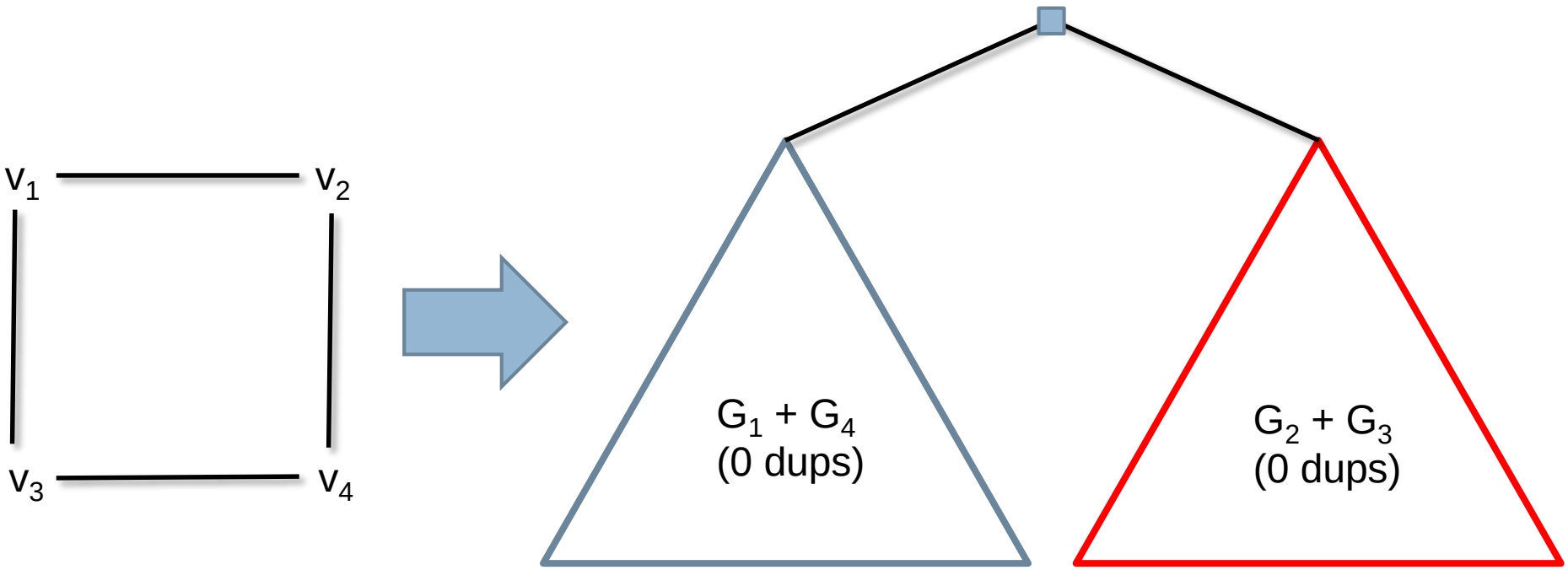
Idée de la réduction (en surface)



Un coloriage est équivalent à partitionner les arbres en sous-arbres d'espèces disjointes.

Chaque partie nécéssite 0 duplications.

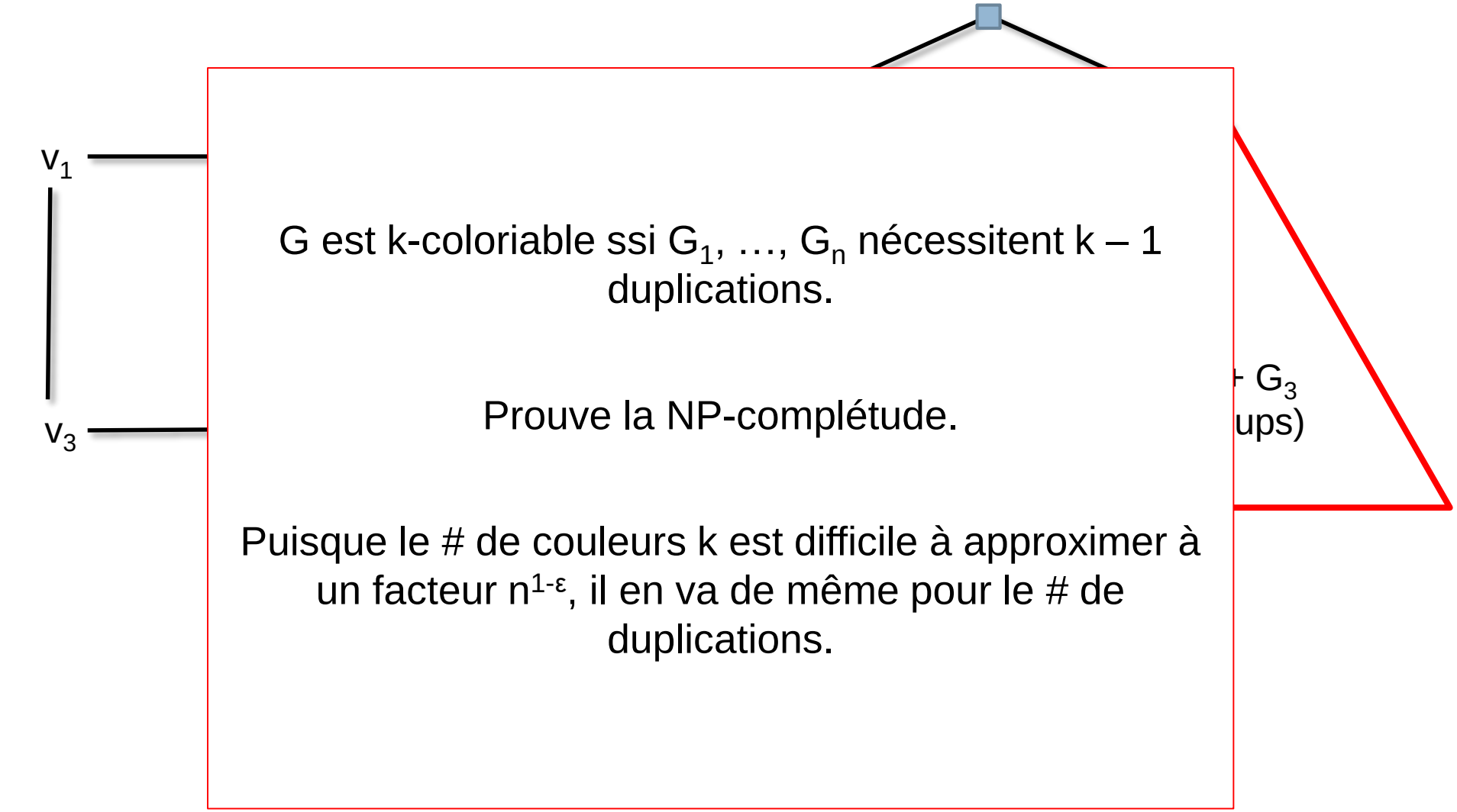
Idée de la réduction (en surface)



Un coloriage est équivalent à partitionner les arbres en sous-arbres d'espèces disjointes.

Chaque partie nécéssite 0 duplications.

Idée de la réduction (en surface)



G est k-coloriable ssi G_1, …, G_n nécessitent k – 1 duplications.

Prouve la NP-complétude.

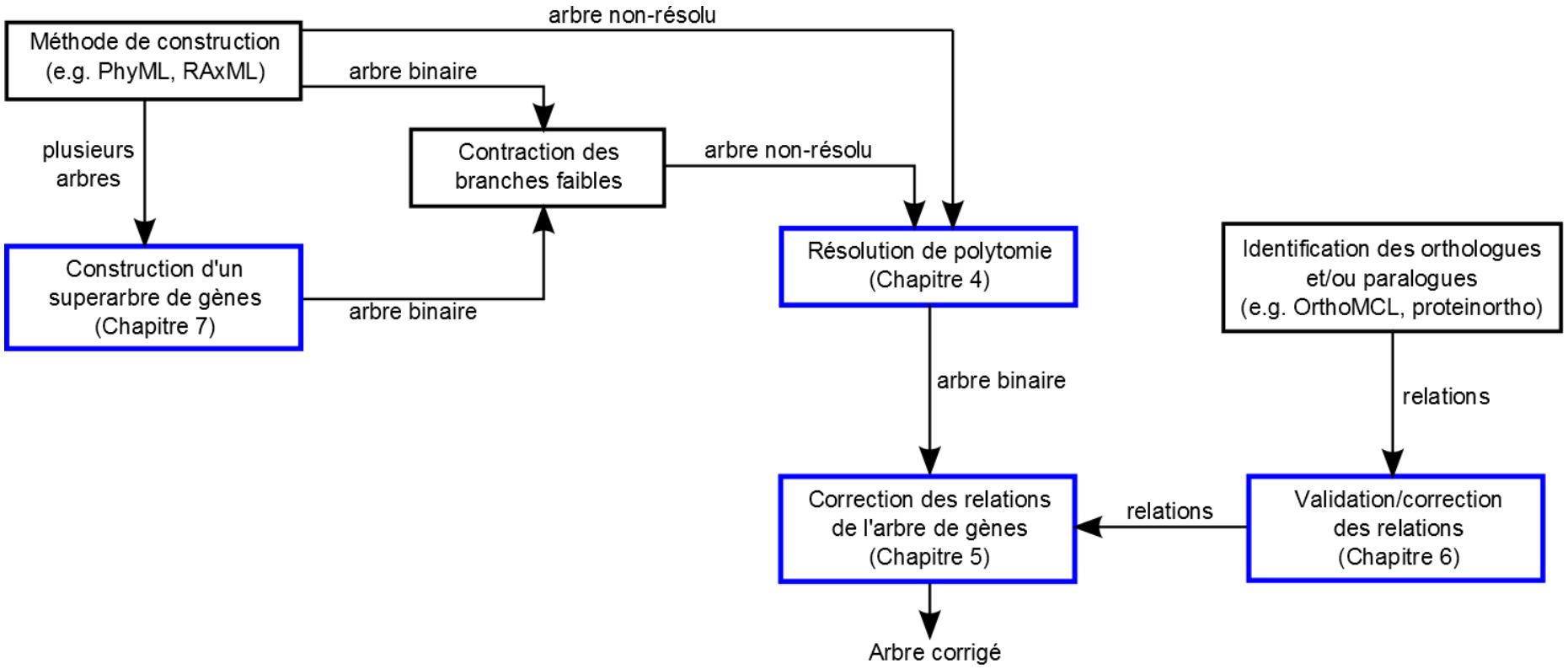
Puisque le # de couleurs k est difficile à approximer à un facteur $n^{1-\varepsilon}$, il en va de même pour le # de duplications.

Algorithmes exacts

- Construction d'un superarbre par programmation dynamique sur les 2^n sous-ensemble des n gènes
  - Nécessite un temps total de $O(4^n)$
- Énumération de tous les superarbres possibles, calcul de la réconciliation sur chacun
  - Temps $\Omega(n * n^{n/2})$
- Nouvel algorithme (non-publié)
  - Prog. dynamique sur les vecteurs de nœuds (1 nœud par arbre)
  - Temps $O(4^k n^k)$, où k = nb d'arbres

Conclusion

Pipeline de correction d'arbres



Travaux futurs

- Toutes les méthodes supposent que l'on a accès à un **arbre d'espèces S exact**
  - Comment modéliser des erreurs sur S (e.g. S non-binaire) ?
- Étendre les méthodes de la thèse à d'autres types d'événements, e.g. transferts horizontaux, coalescence profonde, …
- **Correction** de relations d'orthologie et paralogie (et non seulement la validation).

Travaux futurs

- Thèse axée sur les résultats théoriques.
- Comment les méthodes présentées peuvent-elles devenir pratiques?
  - Résolution de polytomies: logiciel ProfileNJ (avec E. Noutahi).
  - Correction de relations: comment inférer de bonnes relations d'orthologie/paralogie?
  - Superarbres: algorithme FPT? Restriction à des jeux de données biologiquement plausibles?

Remerciements

- Nadia El-Mabrouk
- Tous les membres du LBIT !
- Chercheurs collaborateurs:
  - Cédric Chauve, Riccardo Dondi, Aïda Ouangraoua, Céline Scornovacca, Krister M. Swenson, Éric Tannier
- Copine, famille et amis !
- Supporteurs financiers: FRQNT, CRSNG, DIRO, FESP, Hydro-Québec