Algorithmique pour l'évolution des interactions géniques

Sèverine Bérard

Institut des Sciences de l'Evolution-Montpellier

ISE-M – Université Montpellier 2

UNIVERSITÉ MONTPELLIER 2 SCIENCES ET TECHNIQUES

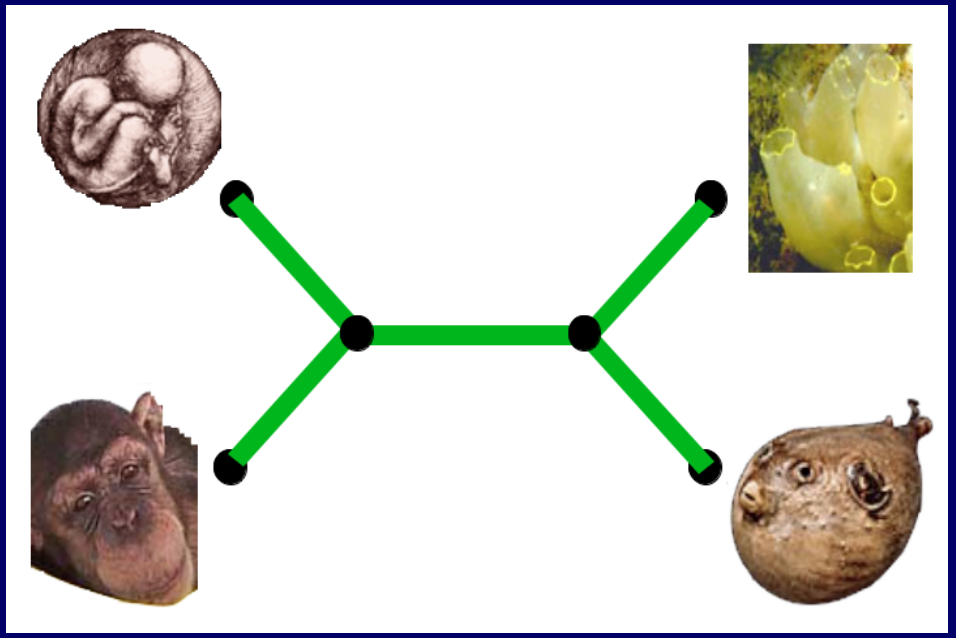
- Introduction
- Modèle
- Démarche
- Résultats
- Conclusion & perpectives

Inférence phylogénétique (1/3)

Homme	A**C**GTCGGTTGCCGAC
Chimpanzé	A**C**GTCGGTTGCCGAC
Fugu	A**T**GTCGGGTGCCGAC
Cione	A**A**GTCGCGTGCCGAC

$\approx$1980

$\Longrightarrow$

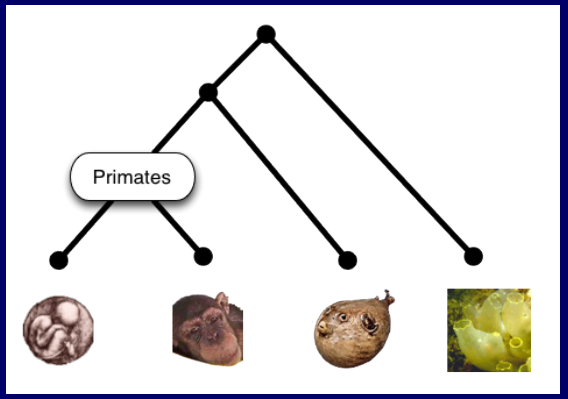


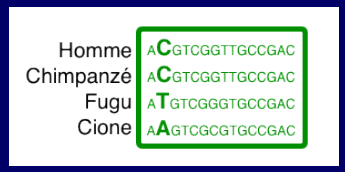
Événement : Substitutions

Inférence phylogénétique (2/3)

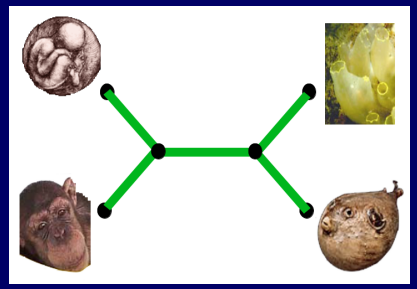
Données :

- Arbre des espèces
- Arbres de gènes racinés

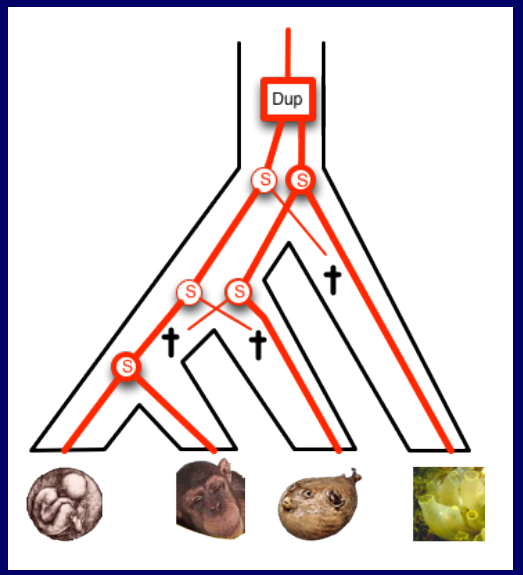




≈1980 ⟹



≈2000 ⟹

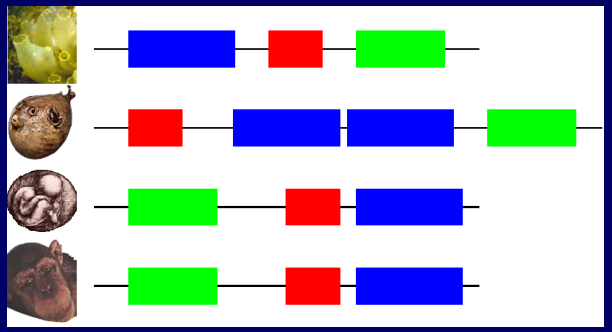


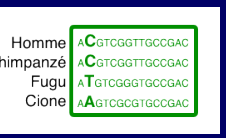
Événements : Substitutions

\+ duplications, pertes

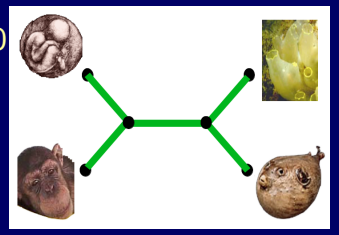
Inférence phylogénétique (3/3)

- Utilisation des adjacences de gènes
- Adjacence de gènes = 2 gènes voisins sur le génome

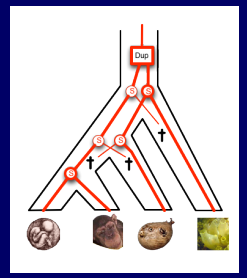




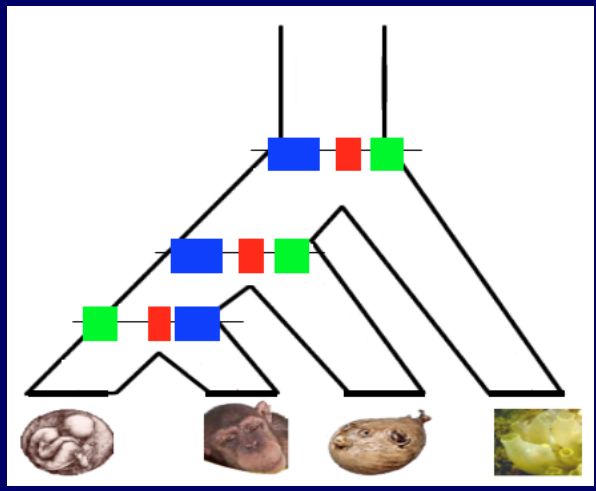
≈1980 ⇒



≈2000 ⇒



2012 ⟹



Événements :
Substitutions

+ duplications, pertes

+ réarrangements

- Stage de M2R Informatique Coralie Gallien $jan \rightarrow sept\ 2011$
  Encadrement : Sèverine Bérard et Éric Tannier

- Preuve, codage et écriture de l'article $sept\ 2011 \rightarrow juin\ 2012$

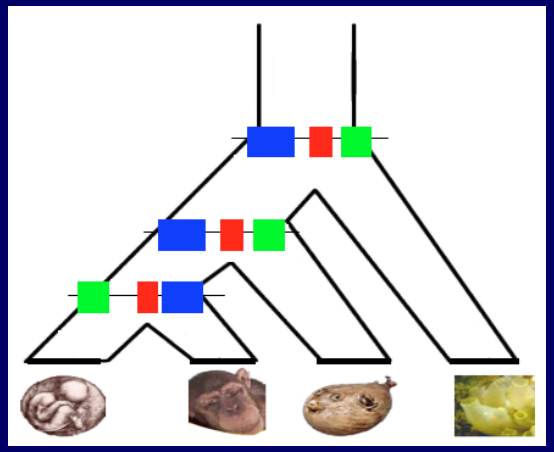
- Parution de l'article $sept\ 2012$

BIOINFORMATICS
Vol. 28 ECCB 2012, pages i382–i388
doi:10.1093/bioinformatics/bts374

Evolution of gene neighborhoods within reconciled phylogenies

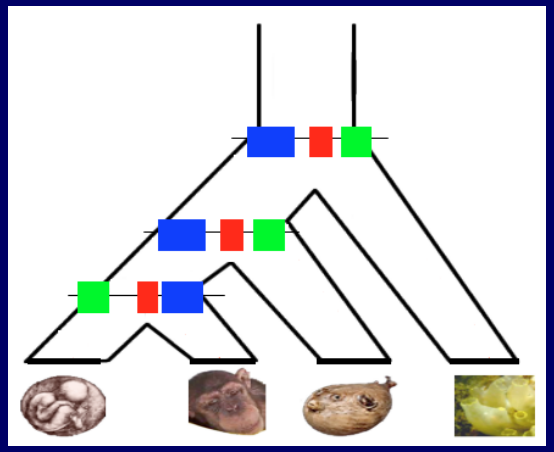
Sèverine Bérard[1,2,*], Coralie Gallien[1], Bastien Boussau[3,4], Gergely J. Szöllősi[3], Vincent Daubin[3] and Eric Tannier[3,5,*]

- Depuis, plusieurs améliorations du modèle



Objectifs

- Retracer l'histoire évolutive de relations entre gènes
- Adjacences : estimer les positions relatives des gènes ancestraux

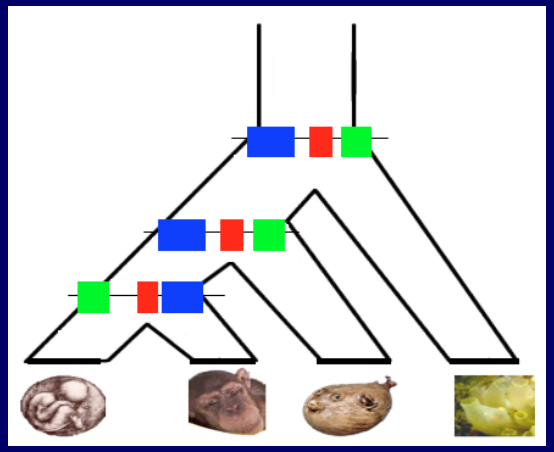


Objectifs

- Retracer l'histoire évolutive de relations entre gènes
- Adjacences : estimer les positions relatives des gènes ancestraux

Principes

- Parcimonie : min(#gains d'adjacence + #cassures d'adjacence)

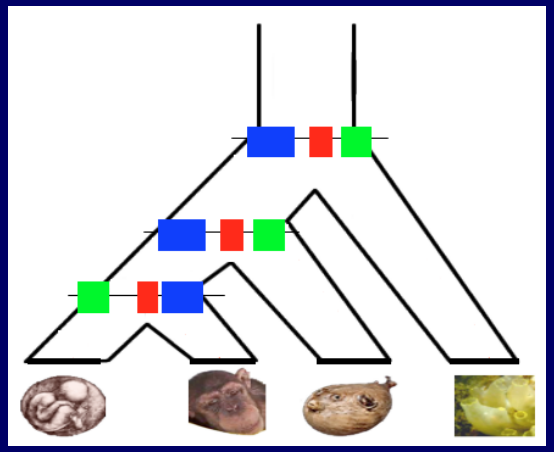


Objectifs

- Retracer l'histoire évolutive de relations entre gènes
- Adjacences : estimer les positions relatives des gènes ancestraux

Principes

- Parcimonie : min(#gains d'adjacence + #cassures d'adjacence)
- Algorithme exact



Objectifs

- Retracer l'histoire évolutive de relations entre gènes
- Adjacences : estimer les positions relatives des gènes ancestraux

Principes

- Parcimonie : min(#gains d'adjacence + #cassures d'adjacence)
- Algorithme exact
- Programmation dynamique, complexité polynomiale

- Introduction
- Modèle
- Démarche
- Résultats
- Conclusion & perpectives

- Adjacences : Deux gènes A_1 et A_2 sont adjacents s'ils se trouvent sur le même chromosome et qu'il n'y a pas de gène entre eux
  On note A_1A_2 ou A_2A_1 (symétrie)

- Adjacences : Deux gènes A_1 et A_2 sont adjacents s'ils se trouvent sur le même chromosome et qu'il n'y a pas de gène entre eux
  On note A_1A_2 ou A_2A_1 (symétrie)

- Arbre phylogénétique : graphe connexe non cyclique, orienté

- Adjacences : Deux gènes A_1 et A_2 sont adjacents s'ils se trouvent sur le même chromosome et qu'il n'y a pas de gène entre eux
  On note A_1A_2 ou A_2A_1 (symétrie)

- Arbre phylogénétique : graphe connexe non cyclique, orienté

  $\rightarrow$ Arbre d'espèces

  $\rightarrow$ Arbre de gènes

  $\rightarrow$ Arbre d'adjacences

- Adjacences : Deux gènes A_1 et A_2 sont adjacents s'ils se trouvent sur le même chromosome et qu'il n'y a pas de gène entre eux
  On note A_1A_2 ou A_2A_1 (symétrie)

- Arbre phylogénétique : graphe connexe non cyclique, orienté

  $\rightarrow$ Arbre d'espèces

  $\rightarrow$ Arbre de gènes

  $\rightarrow$ Arbre d'adjacences

- Forêt : ensemble d'arbres

- Adjacences : Deux gènes A_1 et A_2 sont adjacents s'ils se trouvent sur le même chromosome et qu'il n'y a pas de gène entre eux
  On note A_1A_2 ou A_2A_1 (symétrie)

- Arbre phylogénétique : graphe connexe non cyclique, orienté

  $\rightarrow$ Arbre d'espèces

  $\rightarrow$ Arbre de gènes

  $\rightarrow$ Arbre d'adjacences

- Forêt : ensemble d'arbres

- Pour un nœud n, $S(n)$ est son espèce, $E(n)$ l'événement associé

- Sur les espèces :

  $\rightarrow$ Spéciation ($Spec$) ●

- Sur les espèces :
  - → Spéciation ($Spec$) ●
- Sur les gènes :
  - → Duplication de gène ($GDup$) ■
  - → Perte de gène ($GLos$) ✖

Événements évolutifs

- Sur les espèces :
  - → Spéciation ($Spec$) ●
- Sur les gènes :
  - → Duplication de gène ($GDup$) ■
  - → Perte de gène ($GLos$) ✖
- Sur les adjacences
  - → Duplication d'adjacence ($ADup$) □
  - → Perte d'adjacence ($ALos$) ✖
  - → Création d'adjacence ($Gain$) *(toute racine d'un arbre d'adjacence)*
  - → Cassure d'adjacence ($Break$) ⚡

Arbre d'adjacences

Un arbre d'adjacences $\mathcal{A}$ est un arbre phylogénétique retraçant l'histoire évolutive d'adjacences

Arbre d'adjacences

Un arbre d'adjacences $\mathcal{A}$ est un arbre phylogénétique retraçant l'histoire évolutive d'adjacences

- Il est associé à 2 arbres de gènes, $\mathcal{G}_1$ et $\mathcal{G}_2$, et à une liste d'adjacences actuelles notée $\mathcal{L}$

Un arbre d'adjacences $\mathcal{A}$ est un arbre phylogénétique retraçant l'histoire évolutive d'adjacences

- Il est associé à 2 arbres de gènes, $\mathcal{G}_1$ et $\mathcal{G}_2$, et à une liste d'adjacences actuelles notée $\mathcal{L}$

- Tous ses nœuds sont étiquetés par des adjacences entre nœuds des arbres de gènes associés et/ou un événement évolutif
  $\rightarrow$ Évt feuilles : $\{GLos,\ ALos,\ Break,\ Extant\}$
  $\rightarrow$ Évt nœuds internes : $\{Spec,\ GDup,\ ADup\}$

Un arbre d'adjacences $\mathcal{A}$ est un arbre phylogénétique retraçant l'histoire évolutive d'adjacences

- Il est associé à 2 arbres de gènes, $\mathcal{G}_1$ et $\mathcal{G}_2$, et à une liste d'adjacences actuelles notée $\mathcal{L}$

- Tous ses nœuds sont étiquetés par des adjacences entre nœuds des arbres de gènes associés et/ou un événement évolutif
  $\rightarrow$ Évt feuilles : $\{GLos,\ ALos,\ Break,\ Extant\}$
  $\rightarrow$ Évt nœuds internes : $\{Spec,\ GDup,\ ADup\}$

- L'arbre respecte les règles de transmissions d'adjacences, et ses feuilles de type $Extant$ sont exactement les adjacences de $\mathcal{L}$

Un arbre d'adjacences $\mathcal{A}$ est un arbre phylogénétique retraçant l'histoire évolutive d'adjacences

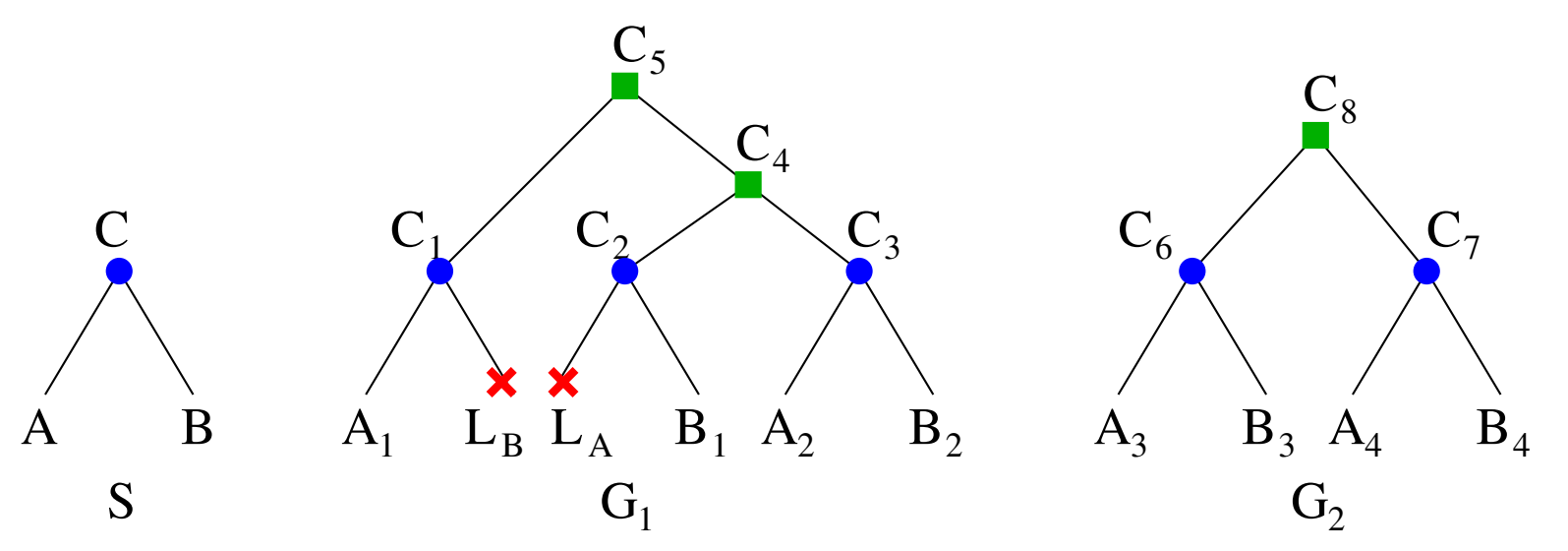
- Il est associé à 2 arbres de gènes, $\mathcal{G}_1$ et $\mathcal{G}_2$, et à une liste d'adjacences actuelles notée $\mathcal{L}$

- Tous ses nœuds sont étiquetés par des adjacences entre nœuds des arbres de gènes associés et/ou un événement évolutif
  $\rightarrow$ Évt feuilles : $\{GLos,\ ALos,\ Break,\ Extant\}$
  $\rightarrow$ Évt nœuds internes : $\{Spec,\ GDup,\ ADup\}$

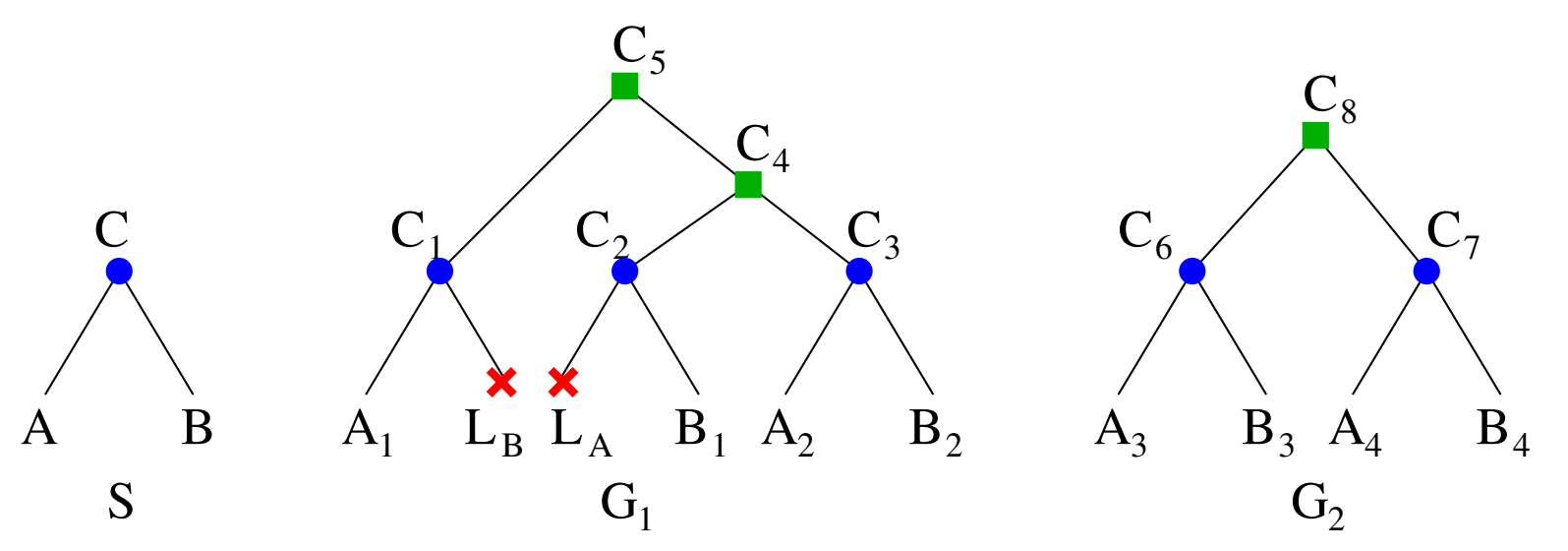
- L'arbre respecte les règles de transmissions d'adjacences, et ses feuilles de type $Extant$ sont exactement les adjacences de $\mathcal{L}$

- La racine correspond à la création d'une adjacence, ce nœud racine porte aussi une étiquette événement

Exemple d'arbres

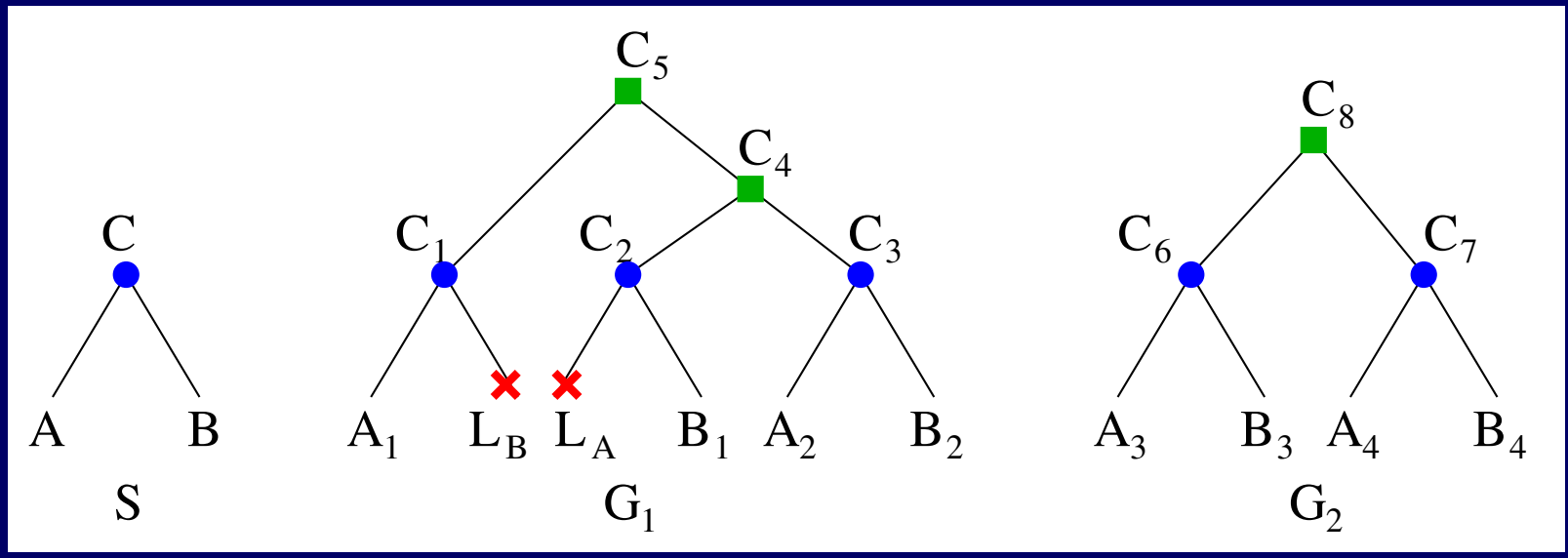


Exemple d'arbres

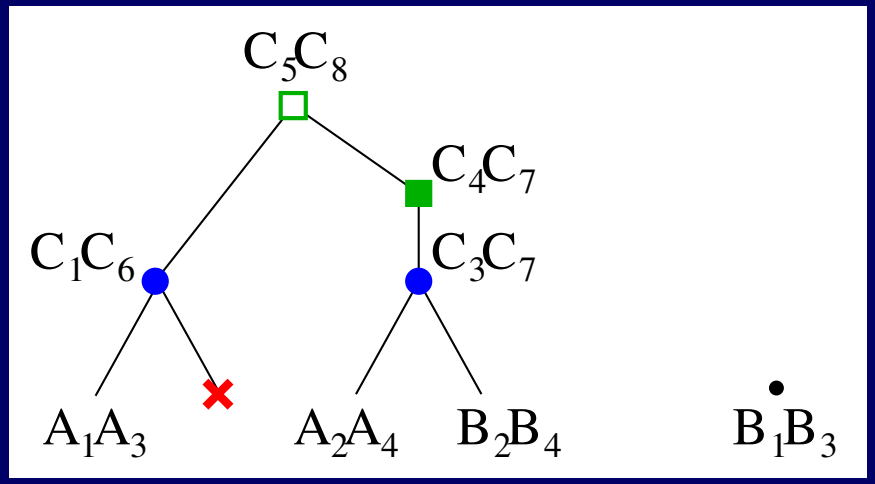


Liste d'adjacences associées $\mathcal{L} = \{A_1A_3, B_1B_3, A_2A_4, B_2B_4\}$

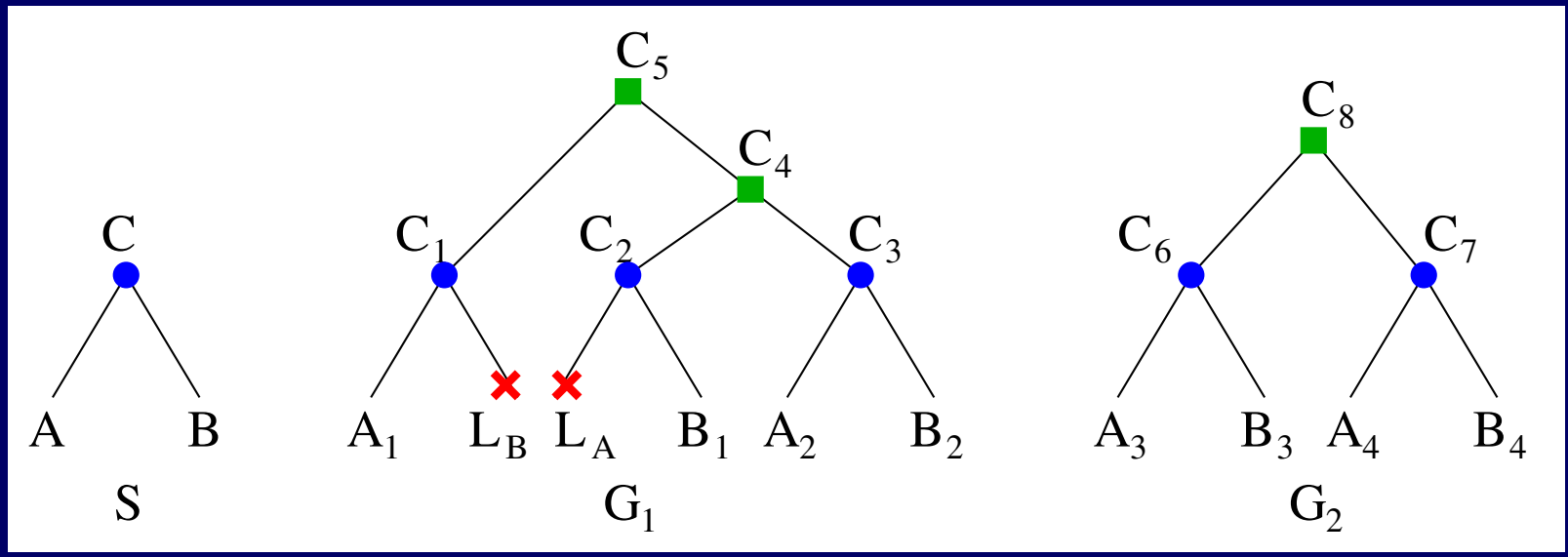
Exemple d'arbres



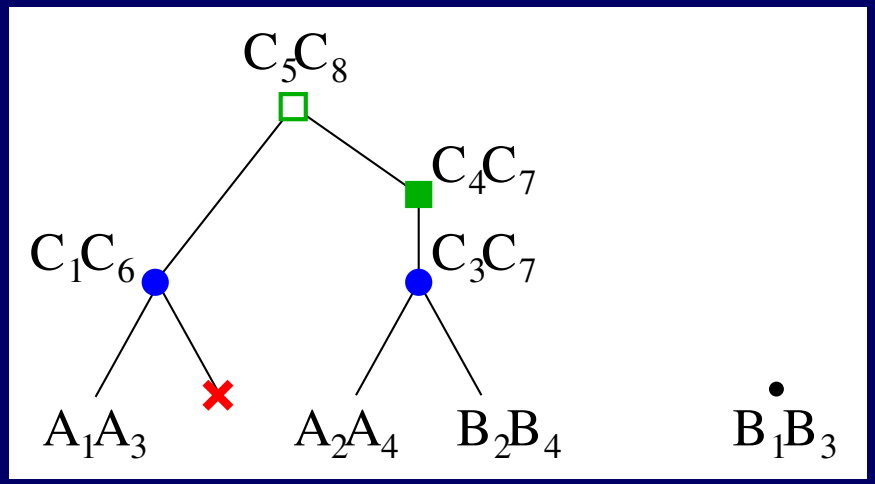
Liste d'adjacences associées $\mathcal{L} = \{A_1A_3, B_1B_3, A_2A_4, B_2B_4\}$



Exemple d'arbres



Liste d'adjacences associées $\mathcal{L} = \{A_1A_3, B_1B_3, A_2A_4, B_2B_4\}$



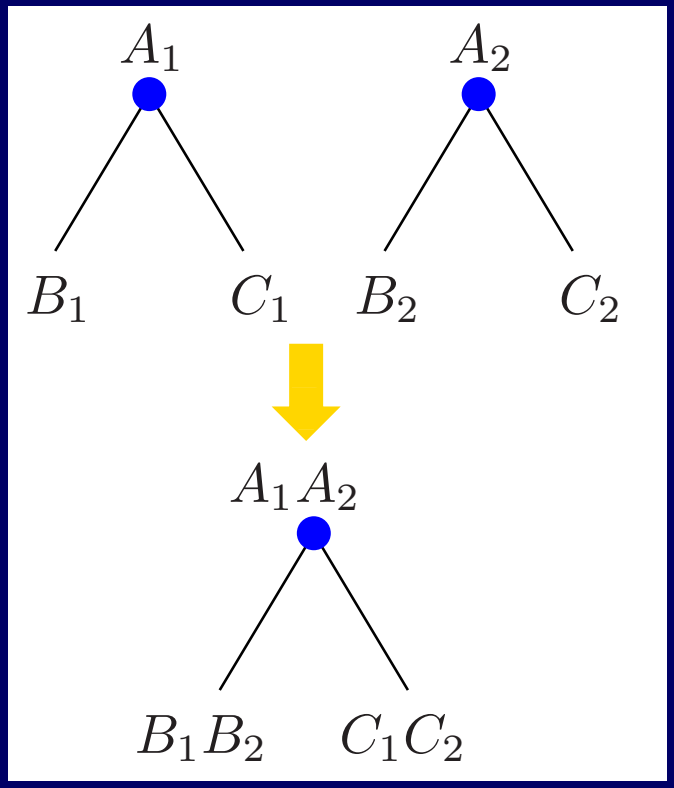
Remarque : Les arbres d'adjacences ne sont pas forcément binaires

Règles de transmissions d'adjacences (1/4)

Les nœuds d'un arbre d'adjacences $\mathcal{A}$ associé à 2 arbres de gènes $\mathcal{G}_1$ et $\mathcal{G}_2$ et à une liste d'adjacences $\mathcal{L}$ respectent les propriétés suivantes :

- A_1A_2 **Nœud de spéciation**
  Alors A_1 et A_2 doivent être 2 nœuds de spéciation dans leur arbre de gènes respectifs

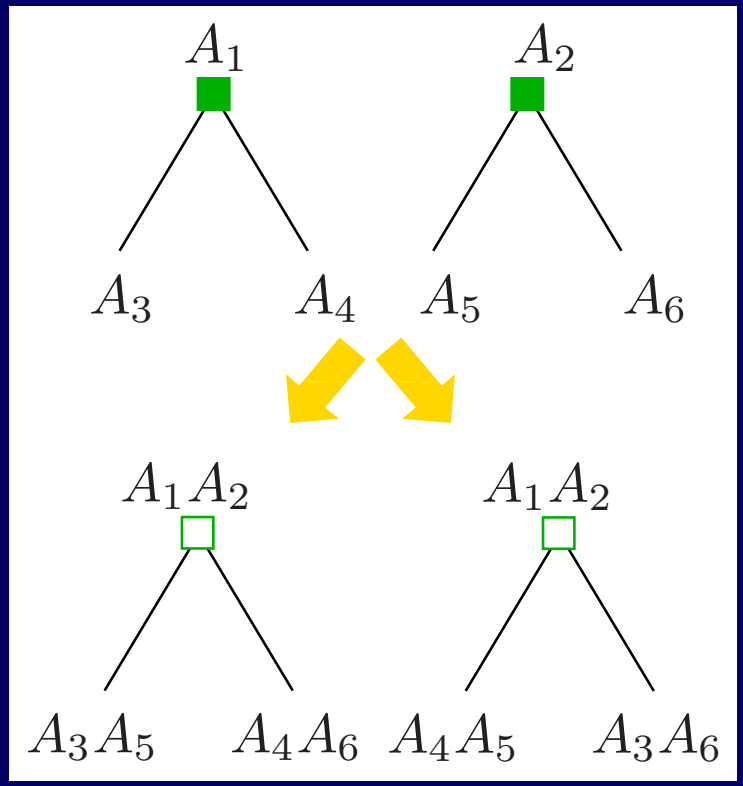
  Supposons que A_1 et A_2 aient 2 fils, respectivement B_1 et C_1 et B_2 et C_2 . Alors les fils de A_1A_2 dans $\mathcal{A}$ sont exactement B_1B_2 et C_1C_2



- A_1A_2 **Nœud de duplication d'adjacence**

Alors A_1 et A_2 doivent être 2 nœuds de duplication de gène

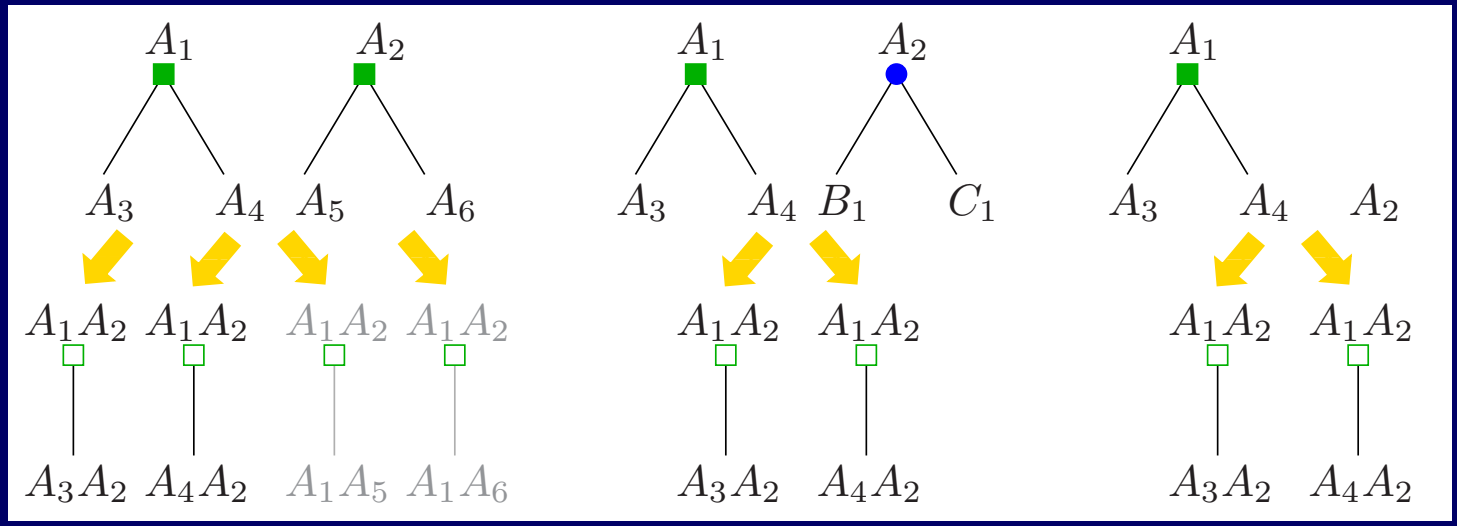
Si A_1 et A_2 ont 2 fils, respectivement A_3 et A_4 et A_5 et A_6 , alors A_1A_2 a exactement 2 fils dans $\mathcal{A}$ qui sont soit A_3A_5 et A_4A_6 soit A_4A_5 et A_3A_6



Règles de transmissions d'adjacences (3/4)

- A_1A_2 **Nœud de duplication de gène**
  Alors A_1 ou A_2 (ou les deux) doit être un nœud de duplication de gène dans son arbre de gènes

  Supposons que A_1 soit le nœud de duplication de gène avec 2 fils A_3 et A_4. Alors A_1A_2 a exactement 1 fils dans $\mathcal{A}$ qui est soit A_3A_2 soit A_4A_2



Règles de transmissions d'adjacences (4/4)

- $A_1 A_2$ **Nœud de création d'adjacence**
  Alors $A_1 A_2$ est la racine de $\mathcal{A}$ et A_1 et A_2 peuvent être de n'importe quel type *(sauf le couple $Spec/Extant$ qui n'existe pas)*

Règles de transmissions d'adjacences (4/4)

- $A_1 A_2$ **Nœud de création d'adjacence**
  Alors $A_1 A_2$ est la racine de $\mathcal{A}$ et A_1 et A_2 peuvent être de n'importe quel type *(sauf le couple $Spec/Extant$ qui n'existe pas)*

- $A_1 A_2$ **Feuille cassure d'adjacence**
  Alors A_1 et A_2 peuvent être de n'importe quel type

Règles de transmissions d'adjacences (4/4)

- A_1A_2 **Nœud de création d'adjacence**
  Alors A_1A_2 est la racine de $\mathcal{A}$ et A_1 et A_2 peuvent être de n'importe quel type *(sauf le couple $Spec/Extant$ qui n'existe pas)*

- A_1A_2 **Feuille cassure d'adjacence**
  Alors A_1 et A_2 peuvent être de n'importe quel type

- A_1A_2 **Feuille adjacence actuelle**
  Alors A_1 et A_2 doivent être 2 gènes actuels

Règles de transmissions d'adjacences (4/4)

- A_1A_2 **Nœud de création d'adjacence**
  Alors A_1A_2 est la racine de $\mathcal{A}$ et A_1 et A_2 peuvent être de n'importe quel type *(sauf le couple $Spec/Extant$ qui n'existe pas)*

- A_1A_2 **Feuille cassure d'adjacence**
  Alors A_1 et A_2 peuvent être de n'importe quel type

- A_1A_2 **Feuille adjacence actuelle**
  Alors A_1 et A_2 doivent être 2 gènes actuels

- A_1A_2 **Feuille perte d'adjacence**
  Alors A_1 et A_2 doivent être 2 perte de gènes

Règles de transmissions d'adjacences (4/4)

- A_1A_2 **Nœud de création d'adjacence**
  Alors A_1A_2 est la racine de $\mathcal{A}$ et A_1 et A_2 peuvent être de n'importe quel type *(sauf le couple $Spec/Extant$ qui n'existe pas)*

- A_1A_2 **Feuille cassure d'adjacence**
  Alors A_1 et A_2 peuvent être de n'importe quel type

- A_1A_2 **Feuille adjacence actuelle**
  Alors A_1 et A_2 doivent être 2 gènes actuels

- A_1A_2 **Feuille perte d'adjacence**
  Alors A_1 et A_2 doivent être 2 perte de gènes

- A_1A_2 **Feuille perte de gène**
  Alors A_1 ou bien A_2 doivent être une perte de gènes

Général Étant donné

- un arbre d'espèce,
- un ensemble d'arbres de gènes,
- une liste d'adjacences actuelles,

trouver l'histoire évolutive des adjacences minimisant le nombre de création et de cassure d'adjacences

Général Étant donné

- un arbre d'espèce,
- un ensemble d'arbres de gènes,
- une liste d'adjacences actuelles,

trouver l'histoire évolutive des adjacences minimisant le nombre de création et de cassure d'adjacences

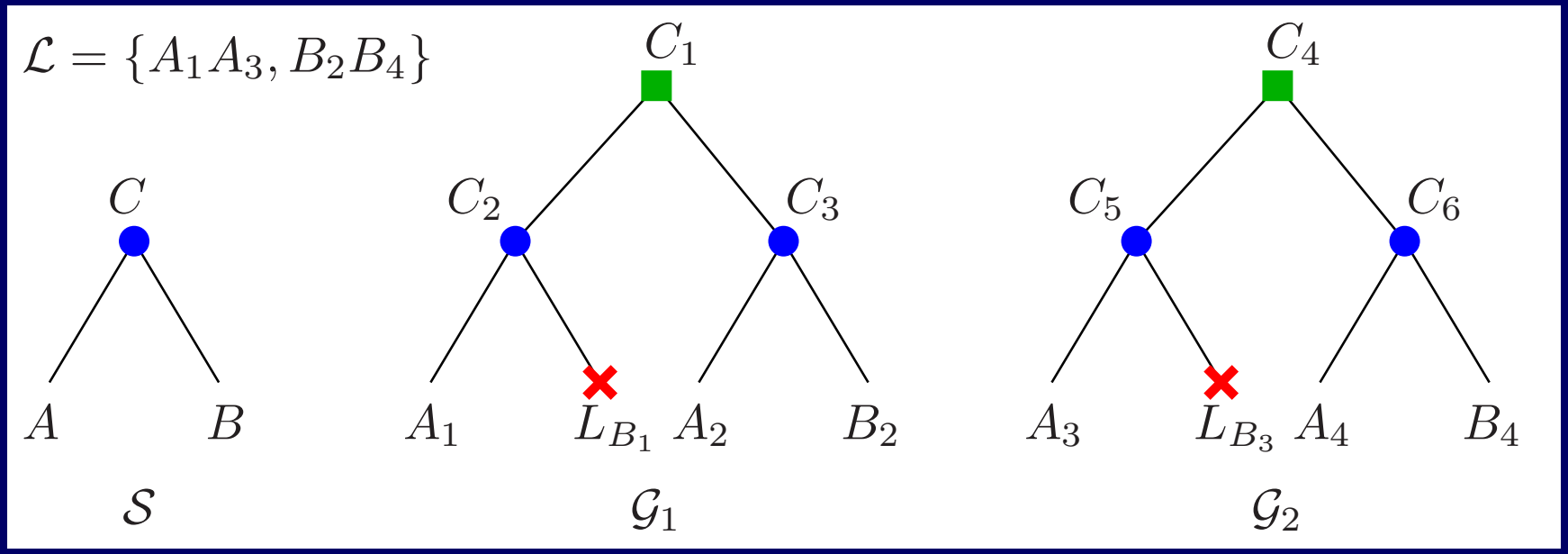
Sous-problème Étant donné

- un arbre d'espèce,
- 2 arbres de gènes réconciliés,
- une liste $\mathcal{L}$ d'adj. actuelles entre feuilles de ces arbres,

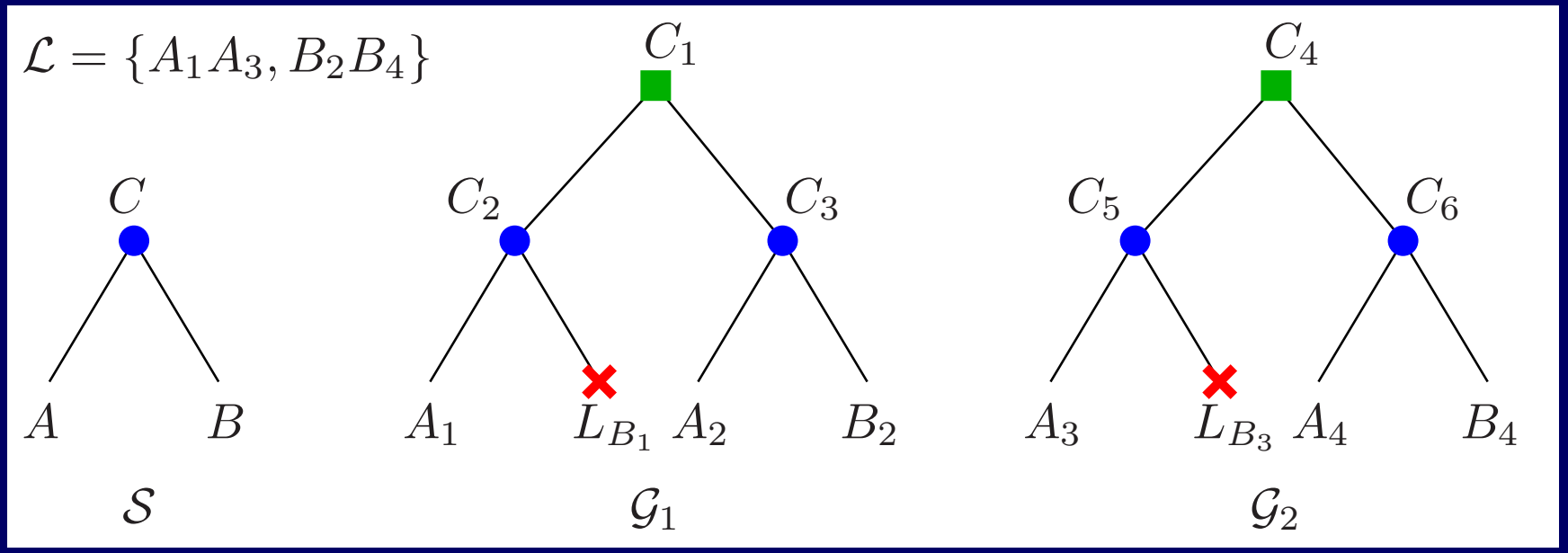
trouver l'histoire évolutive des adjacences de $\mathcal{L}$ minimisant le nombre de création et de cassure d'adjacences

$\Rightarrow$ Résolu par l'algorithme `DeCo`

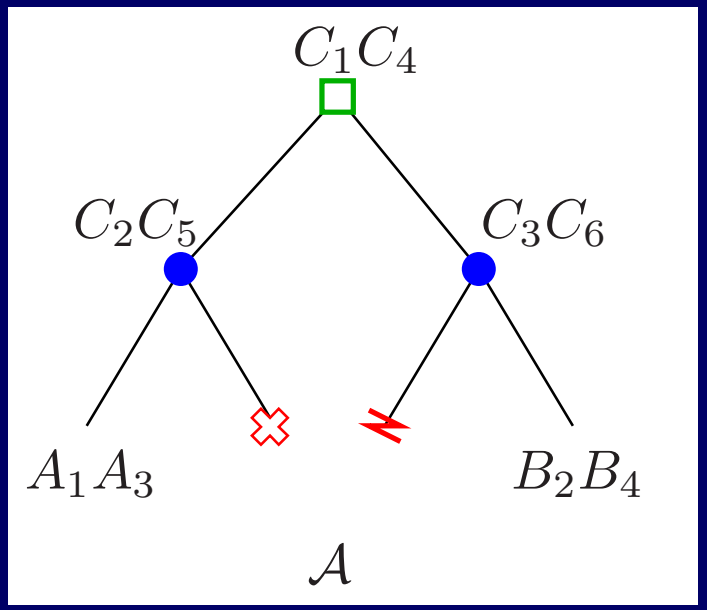
- Introduction
- Modèle
- Démarche
  - → Étape 1 : Réconciliation
  - → Étape 2 : Création des classes d'adjacences
  - → Étape 3 : Algorithme DeCo
  - → Étape 4 : Synthèse
- Résultats
- Conclusion & perpectives


$\mathcal{L} = \{A_1 A_3, B_2 B_4\}$
C
A
B
$\mathcal{S}$
C_1
C_2
C_3
A_1
L_{B_1}
A_2
B_2
$\mathcal{G}_1$
C_4
C_5
C_6
A_3
L_{B_3}
A_4
B_4
$\mathcal{G}_2$

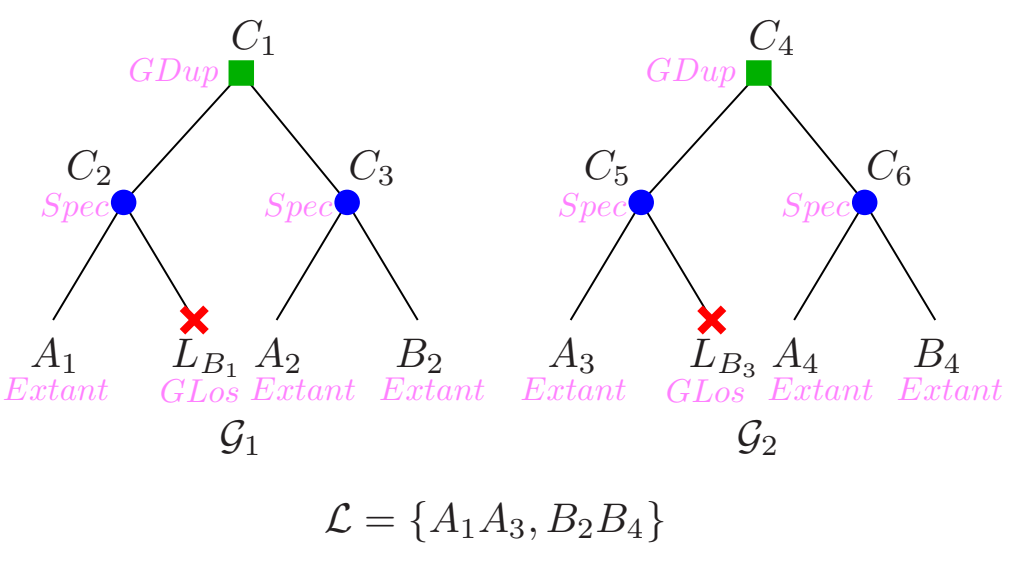
Fonctionnement de DeCo



1. Calcul d'une matrice de coût par programmation dynamique
2. Phase de *backtracking*



Calcul des coûts



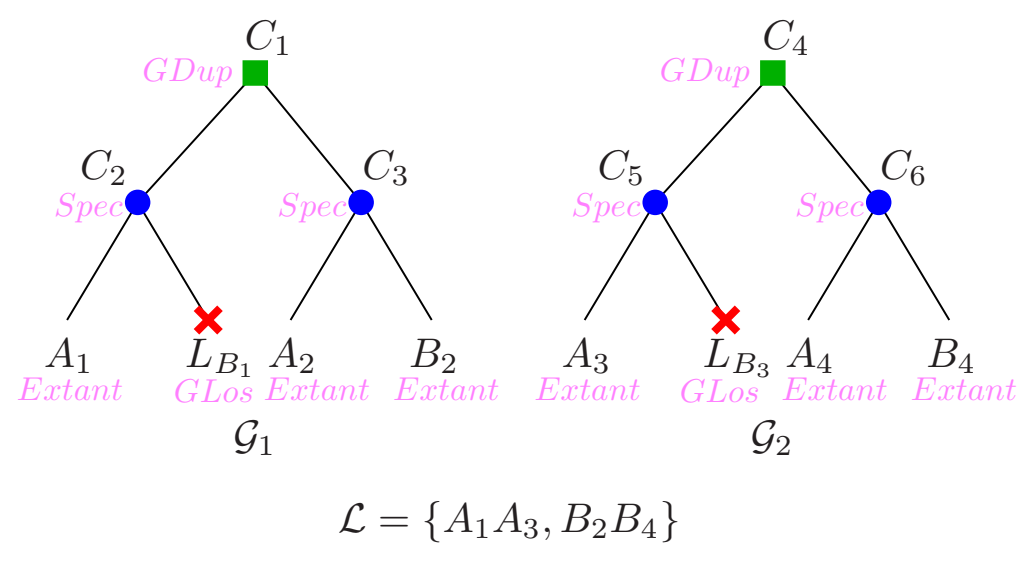
c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1							
L_{B_1}							
A_2							
B_2							
C_2							
C_3							
C_1							

- Matrice de programmation dynamique, coûts c_0 et c_1 :

$\rightarrow$ $c_0(n_1, n_2)$ coût minimum d'une histoire évolutive où n_1 et n_2 ne sont pas adjacents

$\rightarrow$ $c_1(n_1, n_2)$ coût minimum d'une histoire évolutive où n_1 et n_2 sont adjacents

Calcul des coûts



c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1		X		X	X	X	X
L_{B_1}	X		X		X	X	X
A_2		X		X	X	X	X
B_2	X		X		X	X	X
C_2	X	X	X	X			
C_3	X	X	X	X			
C_1	X	X	X	X			

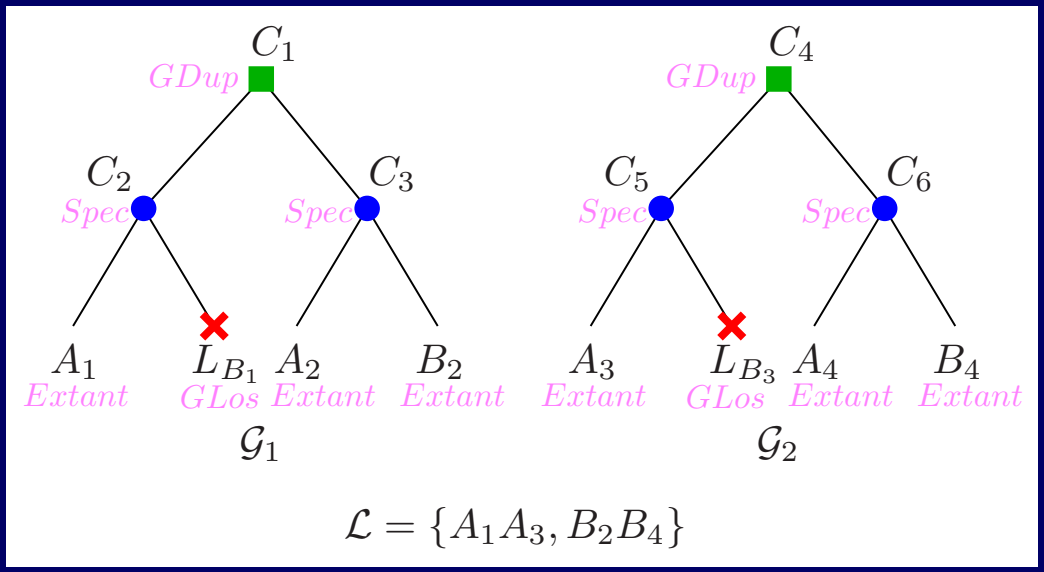
- Matrice de programmation dynamique, coûts c_0 et c_1 :

$\rightarrow$ $c_0(n_1, n_2)$ coût minimum d'une histoire évolutive où n_1 et n_2 ne sont pas adjacents

$\rightarrow$ $c_1(n_1, n_2)$ coût minimum d'une histoire évolutive où n_1 et n_2 sont adjacents

- Calcul des coûts entre nœuds de même espèce

Calcul des coûts



c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1		×		×	×	×	×
L_{B_1}	×		×		×	×	×
A_2		×		×	×	×	×
B_2	×		×		×	×	×
C_2	×	×	×	×			
C_3	×	×	×	×			
C_1	×	×	×	×			

- Matrice de programmation dynamique, coûts c_0 et c_1 :

→ $c_0(n_1, n_2)$ coût minimum d'une histoire évolutive où n_1 et n_2 ne sont pas adjacents

→ $c_1(n_1, n_2)$ coût minimum d'une histoire évolutive où n_1 et n_2 sont adjacents

- Calcul des coûts entre nœuds de même espèce

- Calcul des coûts selon les événements associés aux nœuds ⇒ plusieurs cas

À chaque cas est associé une formule de récurrence :

1. $\underline{E(n_1) = Extant \text{ et } E(n_2) = Extant}$
   Si $n_1 n_2 \in \mathcal{L}$ alors $c_1(n_1, n_2) = 0$ et $c_0(n_1, n_2) = \infty$
   sinon $c_1(n_1, n_2) = \infty$ et $c_0(n_1, n_2) = 0$

À chaque cas est associé une formule de récurrence :

1. $\underline{E(n_1) = Extant \text{ et } E(n_2) = Extant}$
   Si $n_1 n_2 \in \mathcal{L}$ alors $c_1(n_1, n_2) = 0$ et $c_0(n_1, n_2) = \infty$
   sinon $c_1(n_1, n_2) = \infty$ et $c_0(n_1, n_2) = 0$

2. $\underline{E(n_1) = GLos \text{ et } E(n_2) \neq GLos}$
   Alors $c_1(n_1, n_2) = 0$ et $c_0(n_1, n_2) = 0$

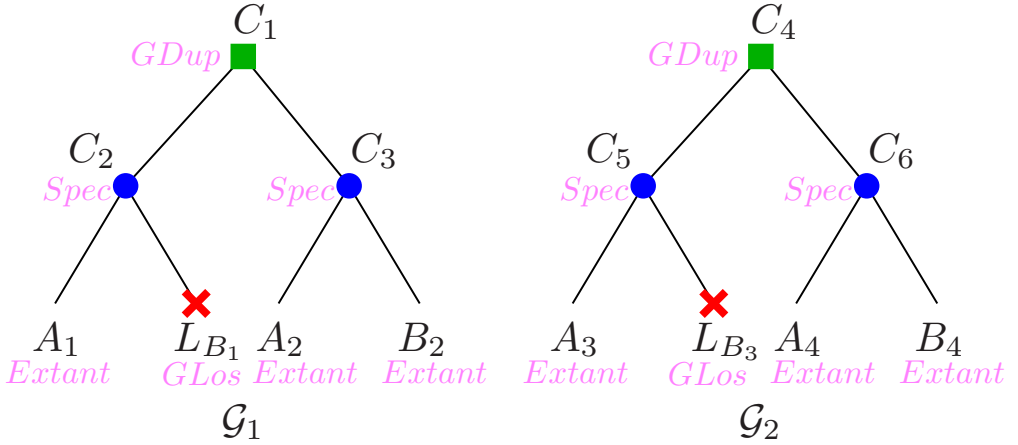
À chaque cas est associé une formule de récurrence :

1. $E(n_1) = Extant$ et $E(n_2) = Extant$
   Si $n_1 n_2 \in \mathcal{L}$ alors $c_1(n_1, n_2) = 0$ et $c_0(n_1, n_2) = \infty$
   sinon $c_1(n_1, n_2) = \infty$ et $c_0(n_1, n_2) = 0$

2. $E(n_1) = GLos$ et $E(n_2) \neq GLos$
   Alors $c_1(n_1, n_2) = 0$ et $c_0(n_1, n_2) = 0$

3. $E(n_1) = GLos$ et $E(n_2) = GLos$
   Alors $c_1(n_1, n_2) = 0$ et $c_0(n_1, n_2) = 0$

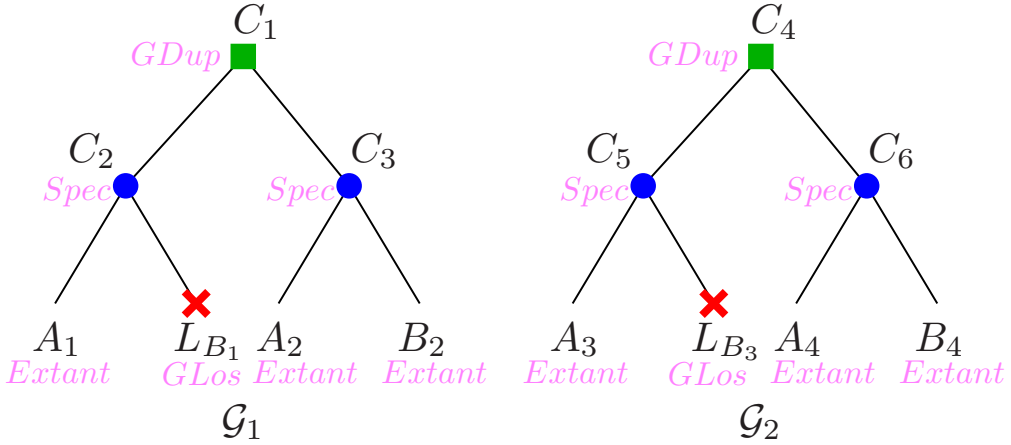
Exemple des cas des feuilles



c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1		X		X	X	X	X
L_{B_1}	X		X		X	X	X
A_2		X		X	X	X	X
B_2	X		X		X	X	X
C_2	X	X	X	X			
C_3	X	X	X	X			
C_1	X	X	X	X			

- A_1-A_3 : cas 1. $Extant/Extant$, $A_1A_3 \in \mathcal{L}$

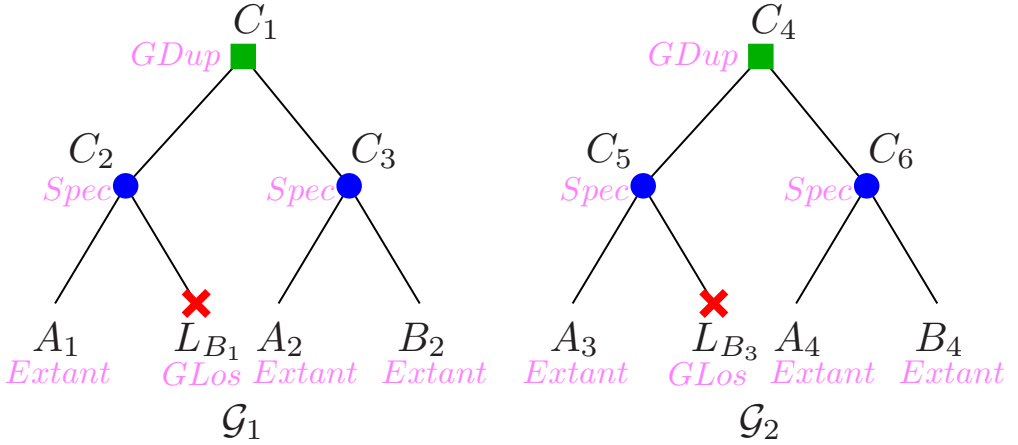
Exemple des cas des feuilles



c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	✗		✗	✗	✗	✗
L_{B_1}	✗		✗		✗	✗	✗
A_2		✗		✗	✗	✗	✗
B_2	✗		✗		✗	✗	✗
C_2	✗	✗	✗	✗			
C_3	✗	✗	✗	✗			
C_1	✗	✗	✗	✗			

- A_1-A_3 : cas 1. $Extant/Extant$, $A_1A_3 \in \mathcal{L}$

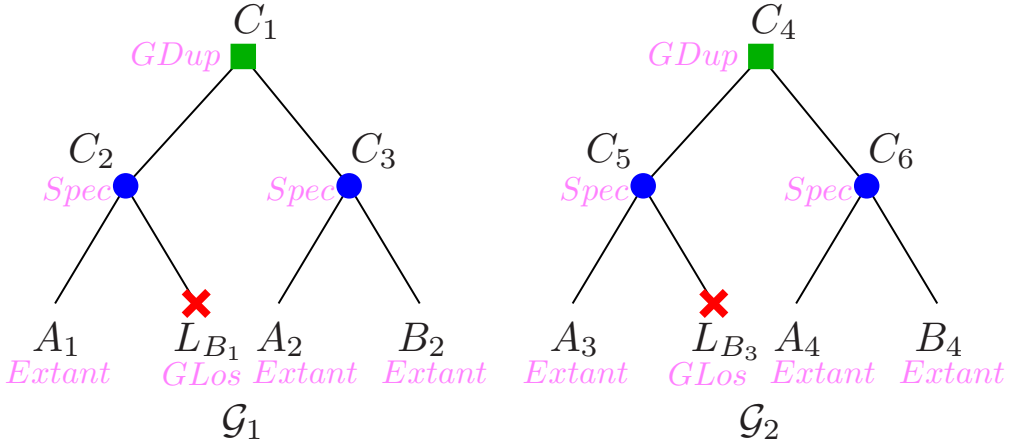
Exemple des cas des feuilles



c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	X		X	X	X	X
L_{B_1}	X		X		X	X	X
A_2		X		X	X	X	X
B_2	X		X		X	X	X
C_2	X	X	X	X			
C_3	X	X	X	X			
C_1	X	X	X	X			

- A_1-A_3 : cas 1. $Extant/Extant$, $A_1A_3 \in \mathcal{L}$
- A_1-A_4 : cas 1. $Extant/Extant$, $A_1A_4 \notin \mathcal{L}$

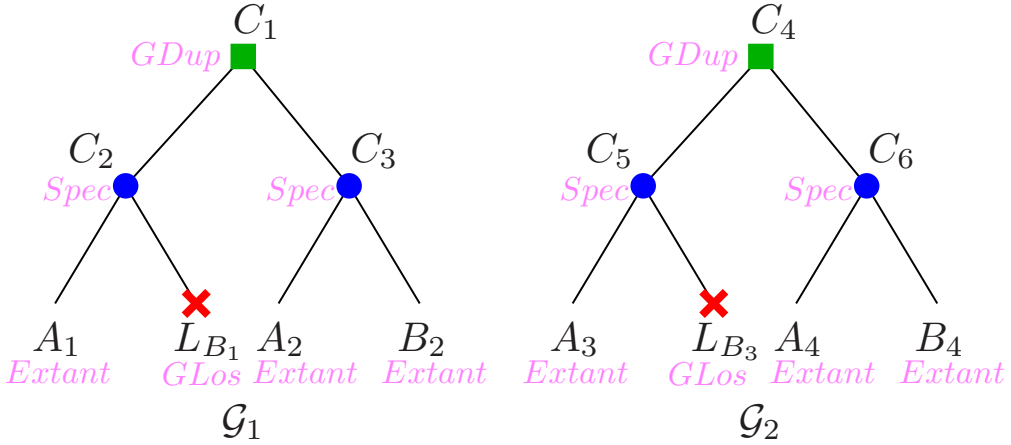
Exemple des cas des feuilles



c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	X	$0/\infty$	X	X	X	X
L_{B_1}	X		X		X	X	X
A_2		X		X	X	X	X
B_2	X		X		X	X	X
C_2	X	X	X	X			
C_3	X	X	X	X			
C_1	X	X	X	X			

- A_1-A_3 : cas 1. $Extant/Extant$, $A_1A_3 \in \mathcal{L}$
- A_1-A_4 : cas 1. $Extant/Extant$, $A_1A_4 \notin \mathcal{L}$

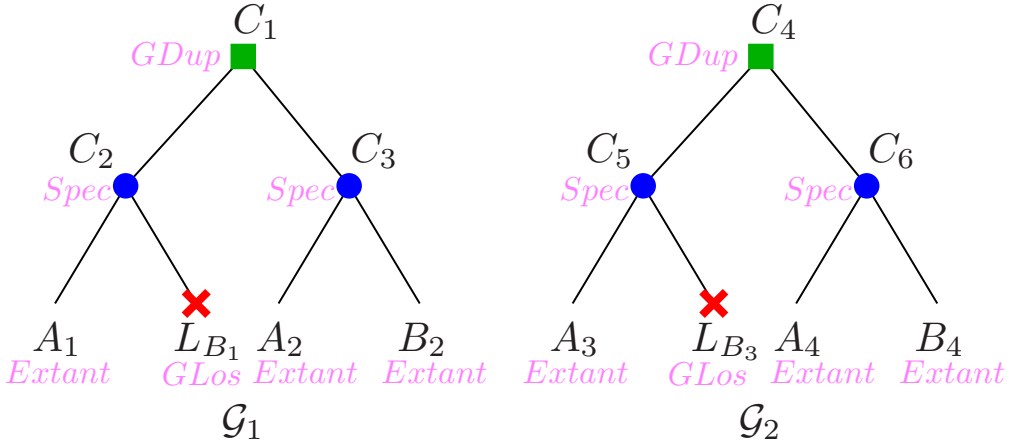
Exemple des cas des feuilles



c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	X	$0/\infty$	X	X	X	X
L_{B_1}	X		X		X	X	X
A_2		X		X	X	X	X
B_2	X		X		X	X	X
C_2	X	X	X	X			
C_3	X	X	X	X			
C_1	X	X	X	X			

- A_1-A_3 : cas 1. $Extant/Extant$, $A_1A_3 \in \mathcal{L}$
- A_1-A_4 : cas 1. $Extant/Extant$, $A_1A_4 \notin \mathcal{L}$
- L_{B_1}-L_{B_3} : cas 3. $GLos/GLos$

Exemple des cas des feuilles

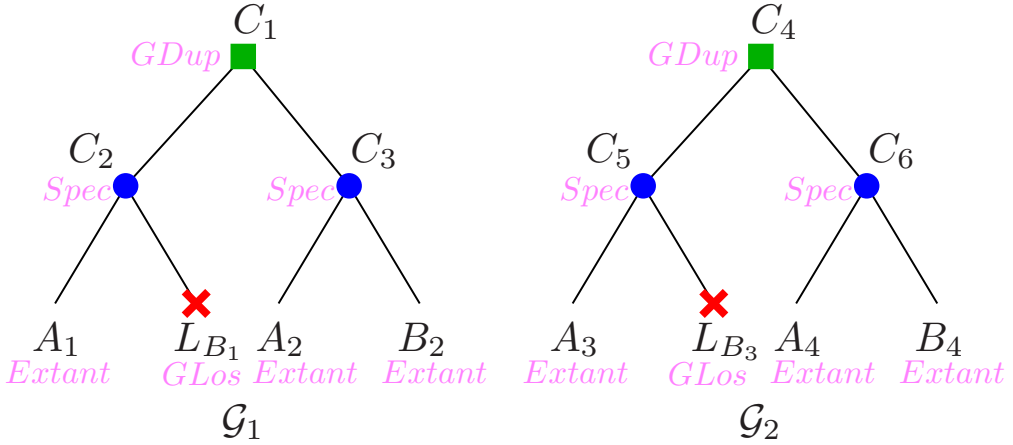


$$\mathcal{L} = \{A_1A_3, B_2B_4\}$$

c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	✗	$0/\infty$	✗	✗	✗	✗
L_{B_1}	✗	$0/0$	✗		✗	✗	✗
A_2		✗		✗	✗	✗	✗
B_2	✗		✗		✗	✗	✗
C_2	✗	✗	✗	✗			
C_3	✗	✗	✗	✗			
C_1	✗	✗	✗	✗			

- A_1-A_3 : cas 1. $Extant/Extant$, $A_1A_3 \in \mathcal{L}$
- A_1-A_4 : cas 1. $Extant/Extant$, $A_1A_4 \notin \mathcal{L}$
- L_{B_1}-L_{B_3} : cas 3. $GLos/GLos$

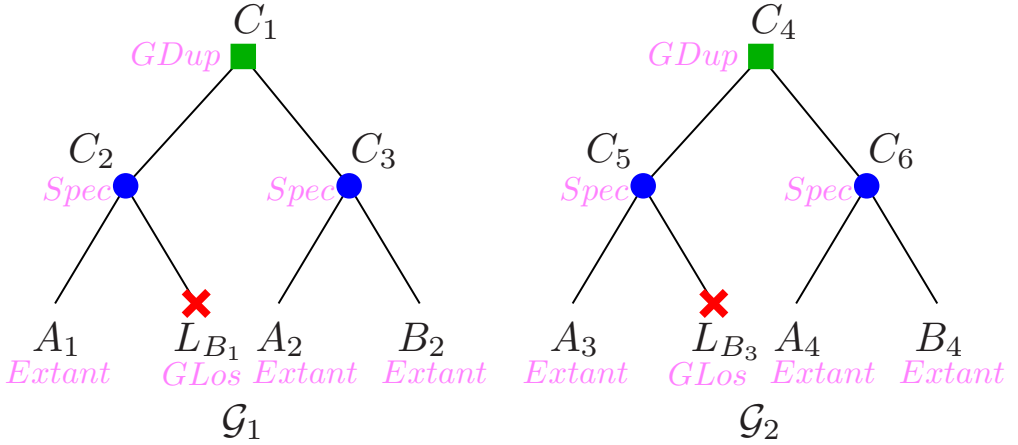
Exemple des cas des feuilles



c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	✗	$0/\infty$	✗	✗	✗	✗
L_{B_1}	✗	$0/0$	✗		✗	✗	✗
A_2		✗		✗	✗	✗	✗
B_2	✗		✗		✗	✗	✗
C_2	✗	✗	✗	✗			
C_3	✗	✗	✗	✗			
C_1	✗	✗	✗	✗			

- A_1-A_3 : cas 1. $Extant/Extant$, $A_1A_3 \in \mathcal{L}$
- A_1-A_4 : cas 1. $Extant/Extant$, $A_1A_4 \notin \mathcal{L}$
- L_{B_1}-L_{B_3} : cas 3. $GLos/GLos$
- L_{B_1}-B_4 : cas 2. $GLos/Any$

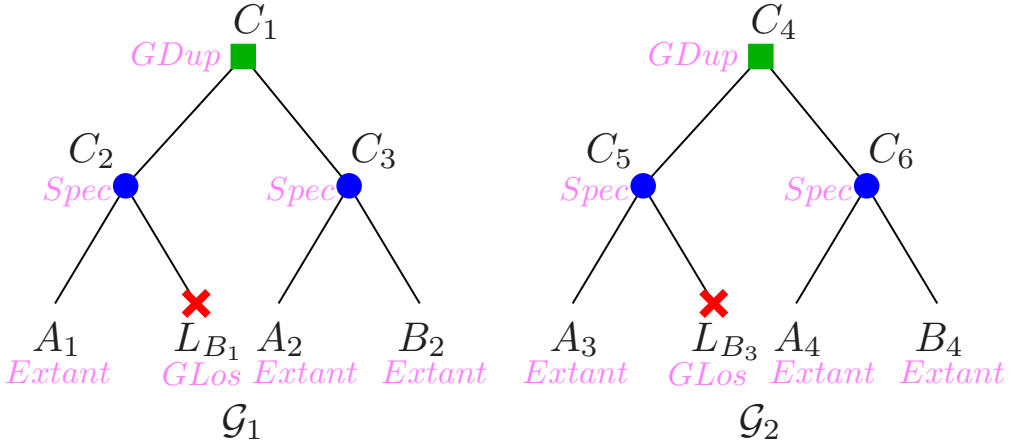
Exemple des cas des feuilles



c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	✗	$0/\infty$	✗	✗	✗	✗
L_{B_1}	✗	$0/0$	✗	$0/0$	✗	✗	✗
A_2		✗		✗	✗	✗	✗
B_2	✗		✗		✗	✗	✗
C_2	✗	✗	✗	✗			
C_3	✗	✗	✗	✗			
C_1	✗	✗	✗	✗			

- A_1-A_3 : cas 1. $Extant/Extant$, $A_1A_3 \in \mathcal{L}$
- A_1-A_4 : cas 1. $Extant/Extant$, $A_1A_4 \notin \mathcal{L}$
- L_{B_1}-L_{B_3} : cas 3. $GLos/GLos$
- L_{B_1}-B_4 : cas 2. $GLos/Any$

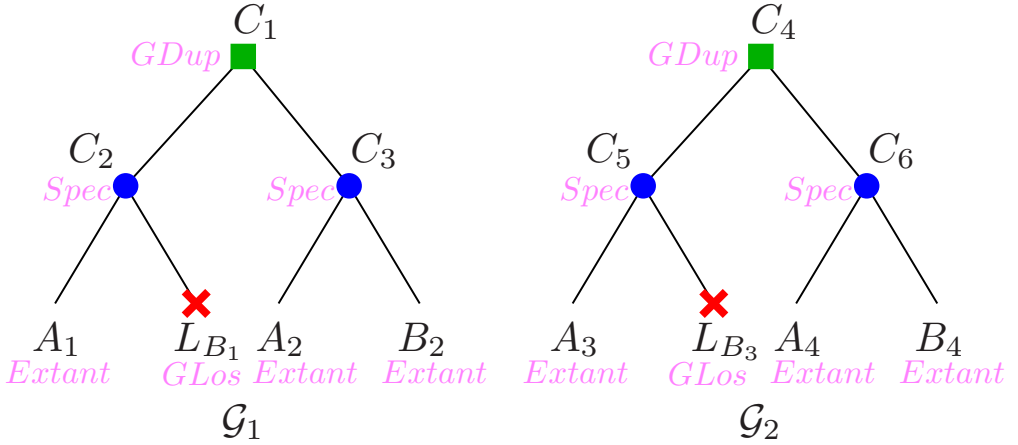
Exemple des cas des feuilles



c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	✗	$0/\infty$	✗	✗	✗	✗
L_{B_1}	✗	$0/0$	✗	$0/0$	✗	✗	✗
A_2		✗		✗	✗	✗	✗
B_2	✗		✗		✗	✗	✗
C_2	✗	✗	✗	✗			
C_3	✗	✗	✗	✗			
C_1	✗	✗	✗	✗			

- A_1-A_3 : cas 1. $Extant/Extant$, $A_1A_3 \in \mathcal{L}$
- A_1-A_4 : cas 1. $Extant/Extant$, $A_1A_4 \notin \mathcal{L}$
- L_{B_1}-L_{B_3} : cas 3. $GLos/GLos$
- L_{B_1}-B_4 : cas 2. $GLos/Any$
- ...

Exemple des cas des feuilles



c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	✗	$0/\infty$	✗	✗	✗	✗
L_{B_1}	✗	$0/0$	✗	$0/0$	✗	✗	✗
A_2	$0/\infty$	✗	$0/\infty$	✗	✗	✗	✗
B_2	✗	$0/0$	✗	$\infty/0$	✗	✗	✗
C_2	✗	✗	✗	✗			
C_3	✗	✗	✗	✗			
C_1	✗	✗	✗	✗			

- A_1-A_3 : cas 1. $Extant/Extant$, $A_1A_3 \in \mathcal{L}$
- A_1-A_4 : cas 1. $Extant/Extant$, $A_1A_4 \notin \mathcal{L}$
- L_{B_1}-L_{B_3} : cas 3. $GLos/GLos$
- L_{B_1}-B_4 : cas 2. $GLos/Any$
- ...

4. $E(n_1) \in \{Extant, Spec\}$ et $E(n_2) = GDup$

$$c_1(n_1, n_2) = \min \begin{cases} c_1(n_1, fg(n_2)) + c_0(n_1, fd(n_2)) \\ c_0(n_1, fg(n_2)) + c_1(n_1, fd(n_2)) \\ c_1(n_1, fg(n_2)) + c_1(n_1, fd(n_2)) + C(Gain) \\ c_0(n_1, fg(n_2)) + c_0(n_1, fd(n_2)) + C(Break) \end{cases}$$

4. $E(n_1) \in \{Extant, Spec\}$ et $E(n_2) = GDup$

$$c_1(n_1, n_2) = \min \begin{cases} c_1(n_1, fg(n_2)) + c_0(n_1, fd(n_2)) \\ c_0(n_1, fg(n_2)) + c_1(n_1, fd(n_2)) \\ c_1(n_1, fg(n_2)) + c_1(n_1, fd(n_2)) + C(Gain) \\ c_0(n_1, fg(n_2)) + c_0(n_1, fd(n_2)) + C(Break) \end{cases}$$

$$c_0(n_1, n_2) = \min \begin{cases} c_0(n_1, fg(n_2)) + c_0(n_1, fd(n_2)) \\ c_0(n_1, fg(n_2)) + c_1(n_1, fd(n_2)) + C(Gain) \\ c_1(n_1, fg(n_2)) + c_0(n_1, fd(n_2)) + C(Gain) \\ c_1(n_1, fg(n_2)) + c_1(n_1, fd(n_2)) + 2 * C(Gain) \end{cases}$$

5. $E(n_1) = Spec$ et $E(n_2) = Spec$

(on suppose que $S(fg(n_1)) = S(fg(n_2))$ et $S(fd(n_1)) = S(fd(n_2))$)

$$c_1(n_1, n_2) = \min \begin{cases} c_1(fg(n_1), fg(n_2)) + c_1(fd(n_1), fd(n_2)) \\ c_1(fg(n_1), fg(n_2)) + c_0(fd(n_1), fd(n_2)) + C(Break) \\ c_0(fg(n_1), fg(n_2)) + c_1(fd(n_1), fd(n_2)) + C(Break) \\ c_0(fg(n_1), fg(n_2)) + c_0(fd(n_1), fd(n_2)) + 2 * C(Break) \end{cases}$$

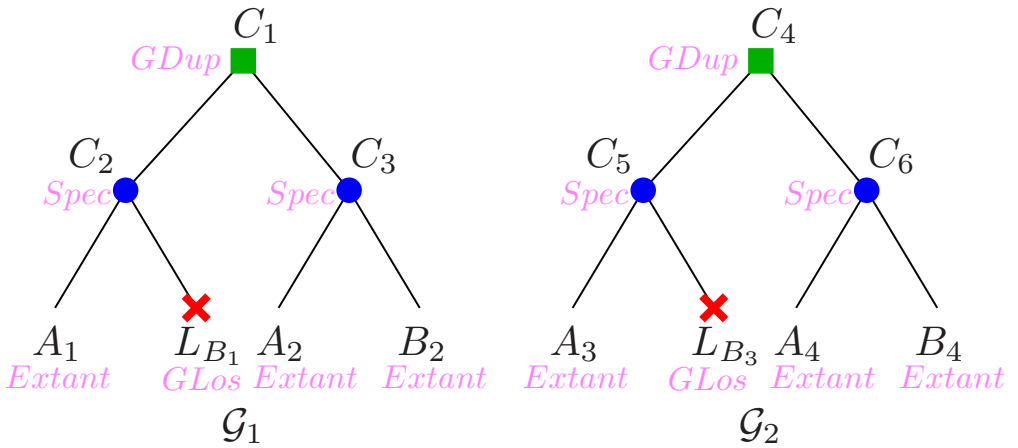
5. $E(n_1) = Spec$ et $E(n_2) = Spec$

(on suppose que $S(fg(n_1)) = S(fg(n_2))$ et $S(fd(n_1)) = S(fd(n_2))$)

$$c_1(n_1, n_2) = \min \begin{cases} c_1(fg(n_1), fg(n_2)) + c_1(fd(n_1), fd(n_2)) \\ c_1(fg(n_1), fg(n_2)) + c_0(fd(n_1), fd(n_2)) + C(Break) \\ c_0(fg(n_1), fg(n_2)) + c_1(fd(n_1), fd(n_2)) + C(Break) \\ c_0(fg(n_1), fg(n_2)) + c_0(fd(n_1), fd(n_2)) + 2 * C(Break) \end{cases}$$

$$c_0(n_1, n_2) = \min \begin{cases} c_0(fg(n_1), fg(n_2)) + c_0(fd(n_1), fd(n_2)) \\ c_1(fg(n_1), fg(n_2)) + c_0(fd(n_1), fd(n_2)) + C(Gain) \\ c_0(fg(n_1), fg(n_2)) + c_1(fd(n_1), fd(n_2)) + C(Gain) \\ c_1(fg(n_1), fg(n_2)) + c_1(fd(n_1), fd(n_2)) + 2 * C(Gain) \end{cases}$$

Exemple des cas *Spec/Spec* et *Spec/GDup*

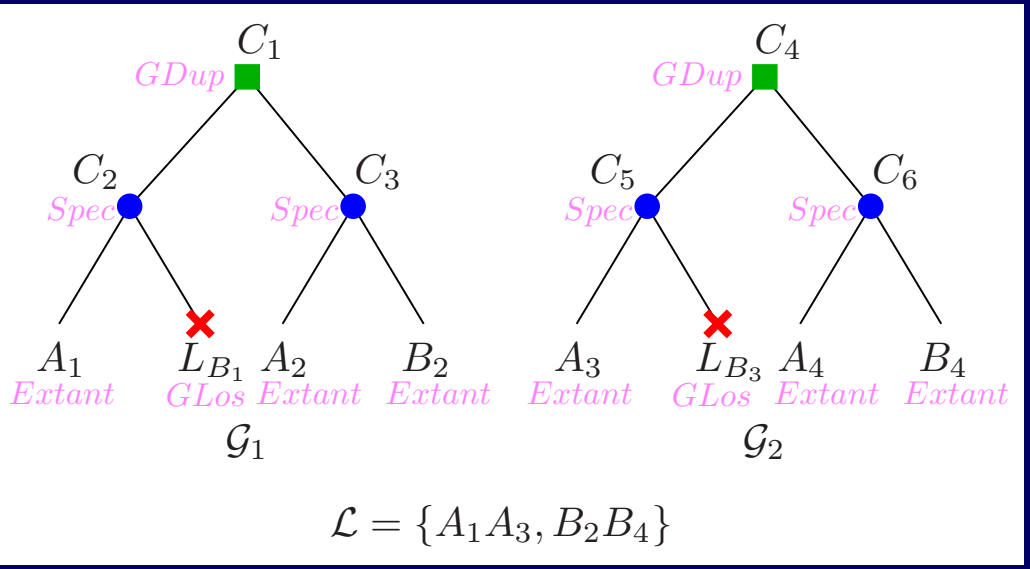


c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	X	$0/\infty$	X	X	X	X
L_{B_1}	X	$0/0$	X	$0/0$	X	X	X
A_2	$0/\infty$	X	$0/\infty$	X	X	X	X
B_2	X	$0/0$	X	$\infty/0$	X	X	X
C_2	X	X	X	X			
C_3	X	X	X	X			
C_1	X	X	X	X			

- C_2-C_5 : cas 5. *Spec/Spec*

$$c_1(C_2, C_5) = \min \begin{cases} c_1(A_1, A_3) + c_1(L_{B_1}, L_{B_3}) \\ c_1(A_1, A_3) + c_0(L_{B_1}, L_{B_3}) + C(Break) \\ c_0(A_1, A_3) + c_1(L_{B_1}, L_{B_3}) + C(Break) \\ c_0(A_1, A_3) + c_0(L_{B_1}, L_{B_3}) + 2 * C(Break) \end{cases}$$

Exemple des cas *Spec/Spec* et *Spec/GDup*

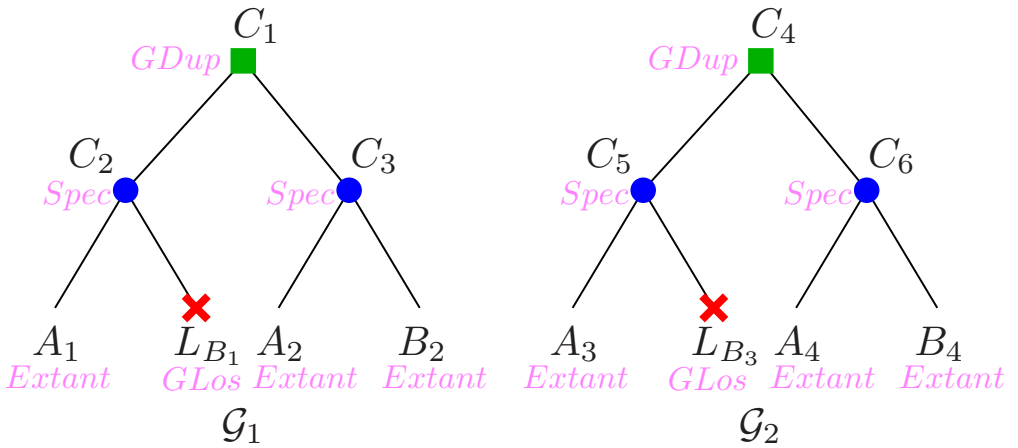


c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	X	$0/\infty$	X	X	X	X
L_{B_1}	X	$0/0$	X	$0/0$	X	X	X
A_2	$0/\infty$	X	$0/\infty$	X	X	X	X
B_2	X	$0/0$	X	$\infty/0$	X	X	X
C_2	X	X	X	X			
C_3	X	X	X	X			
C_1	X	X	X	X			

- C_2-C_5 : cas 5. *Spec/Spec*

$$c_1(C_2, C_5) = \min \begin{cases} 0 & + & 0 & & \\ 0 & + & 0 & + & 1 \\ \infty & + & 0 & + & 1 \\ \infty & + & 0 & + & 2 \end{cases}$$

Exemple des cas *Spec/Spec* et *Spec/GDup*

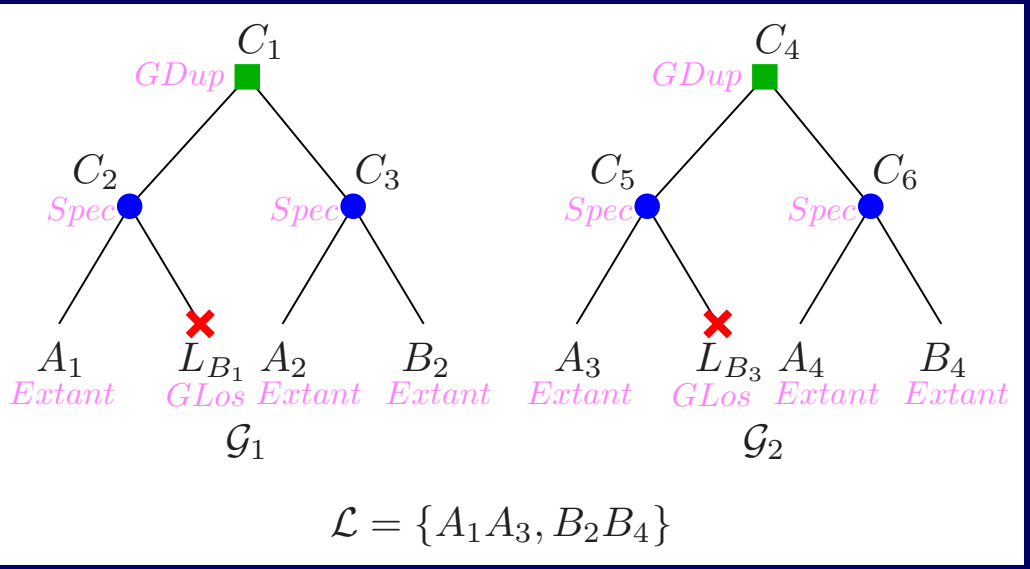


c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	X	$0/\infty$	X	X	X	X
L_{B_1}	X	$0/0$	X	$0/0$	X	X	X
A_2	$0/\infty$	X	$0/\infty$	X	X	X	X
B_2	X	$0/0$	X	$\infty/0$	X	X	X
C_2	X	X	X	X	$/0$		
C_3	X	X	X	X			
C_1	X	X	X	X			

- C_2-C_5 : cas 5. *Spec/Spec*

$$c_1(C_2, C_5) = \min \begin{cases} 0 + 0 \\ 0 + 0 + 1 \\ \infty + 0 + 1 \\ \infty + 0 + 2 \end{cases}$$

Exemple des cas *Spec/Spec* et *Spec/GDup*

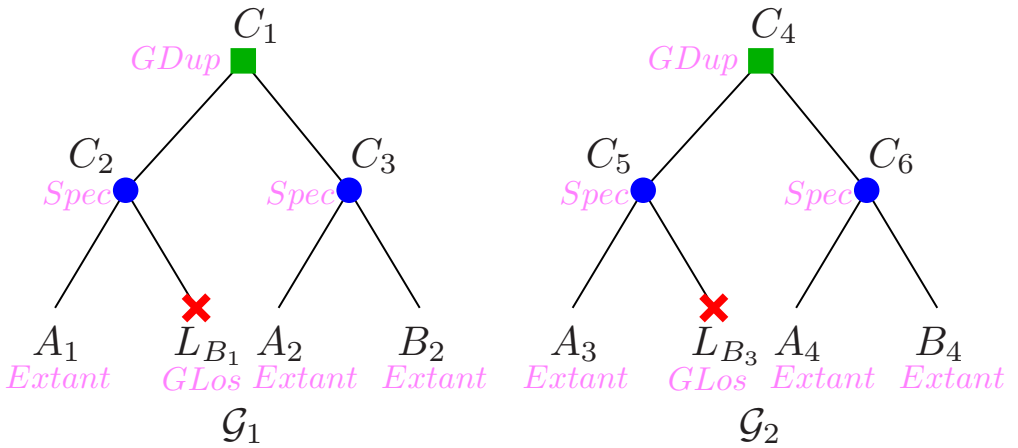


c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	X	$0/\infty$	X	X	X	X
L_{B_1}	X	$0/0$	X	$0/0$	X	X	X
A_2	$0/\infty$	X	$0/\infty$	X	X	X	X
B_2	X	$0/0$	X	$\infty/0$	X	X	X
C_2	X	X	X	X	$/0$		
C_3	X	X	X	X			
C_1	X	X	X	X			

- C_2-C_5 : cas 5. *Spec/Spec*

$$c_0(C_2, C_5) = \min \begin{cases} c_0(A_1, A_3) + c_0(L_{B_1}, L_{B_3}) \\ c_1(A_1, A_3) + c_0(L_{B_1}, L_{B_3}) + C(Gain) \\ c_0(A_1, A_3) + c_1(L_{B_1}, L_{B_3}) + C(Gain) \\ c_1(A_1, A_3) + c_1(L_{B_1}, L_{B_3}) + 2 * C(Gain) \end{cases}$$

Exemple des cas *Spec/Spec* et *Spec/GDup*

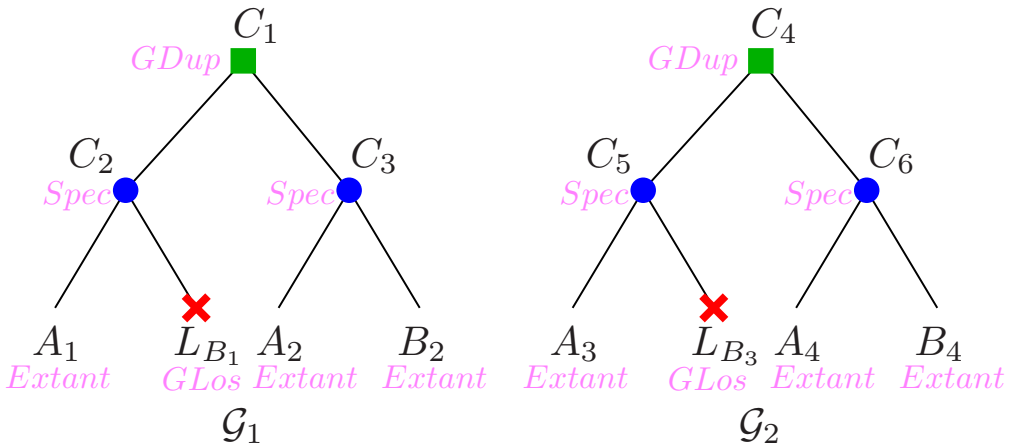


c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	X	$0/\infty$	X	X	X	X
L_{B_1}	X	$0/0$	X	$0/0$	X	X	X
A_2	$0/\infty$	X	$0/\infty$	X	X	X	X
B_2	X	$0/0$	X	$\infty/0$	X	X	X
C_2	X	X	X	X	$/0$		
C_3	X	X	X	X			
C_1	X	X	X	X			

- C_2-C_5 : cas 5. *Spec/Spec*

$$c_0(C_2, C_5) = \min \begin{cases} \infty & + & 0 & & \\ 0 & + & 0 & + & 1 \\ \infty & + & 0 & + & 1 \\ 0 & + & 0 & + & 2 \end{cases}$$

Exemple des cas *Spec/Spec* et *Spec/GDup*

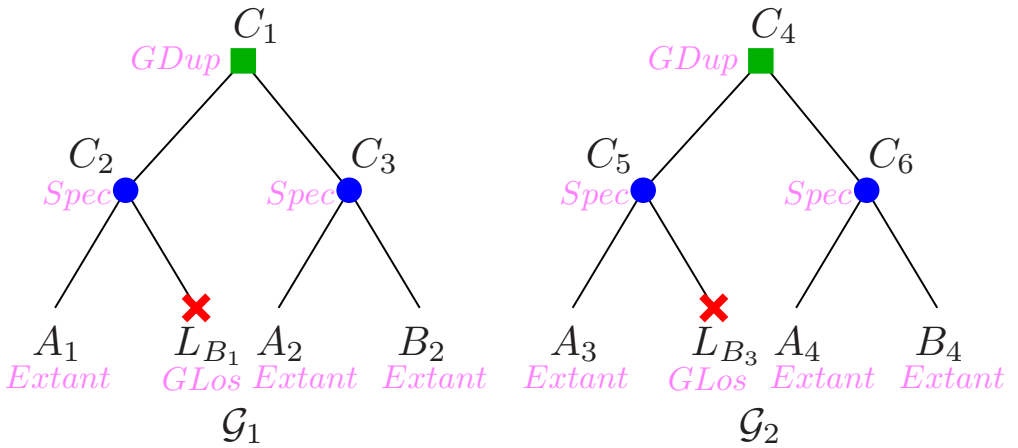


c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	X	$0/\infty$	X	X	X	X
L_{B_1}	X	$0/0$	X	$0/0$	X	X	X
A_2	$0/\infty$	X	$0/\infty$	X	X	X	X
B_2	X	$0/0$	X	$\infty/0$	X	X	X
C_2	X	X	X	X	$1/0$		
C_3	X	X	X	X			
C_1	X	X	X	X			

- C_2-C_5 : cas 5. *Spec/Spec*

$$c_0(C_2, C_5) = \min \begin{cases} \infty & + & 0 & & \\ 0 & + & 0 & + & 1 \\ \infty & + & 0 & + & 1 \\ 0 & + & 0 & + & 2 \end{cases}$$

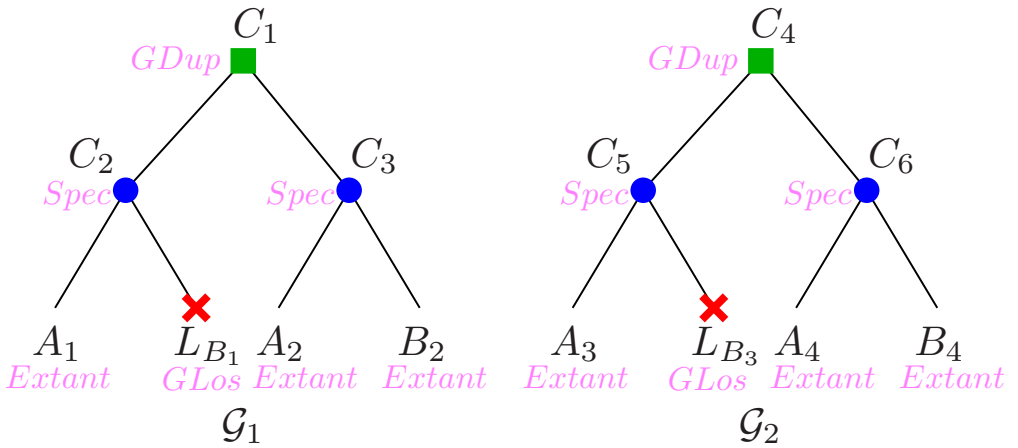
Exemple des cas *Spec/Spec* et *Spec/GDup*



c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	X	$0/\infty$	X	X	X	X
L_{B_1}	X	$0/0$	X	$0/0$	X	X	X
A_2	$0/\infty$	X	$0/\infty$	X	X	X	X
B_2	X	$0/0$	X	$\infty/0$	X	X	X
C_2	X	X	X	X	$1/0$		
C_3	X	X	X	X			
C_1	X	X	X	X			

- C_2-C_6 : cas 5. *Spec/Spec*

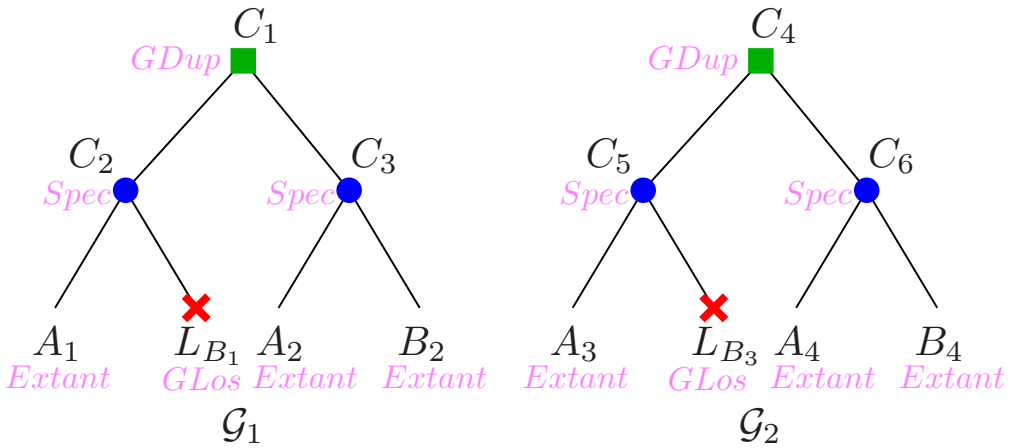
Exemple des cas *Spec/Spec* et *Spec/GDup*



c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	X	$0/\infty$	X	X	X	X
L_{B_1}	X	$0/0$	X	$0/0$	X	X	X
A_2	$0/\infty$	X	$0/\infty$	X	X	X	X
B_2	X	$0/0$	X	$\infty/0$	X	X	X
C_2	X	X	X	X	$1/0$	$0/1$	
C_3	X	X	X	X			
C_1	X	X	X	X			

- C_2-C_6 : cas 5. *Spec/Spec*

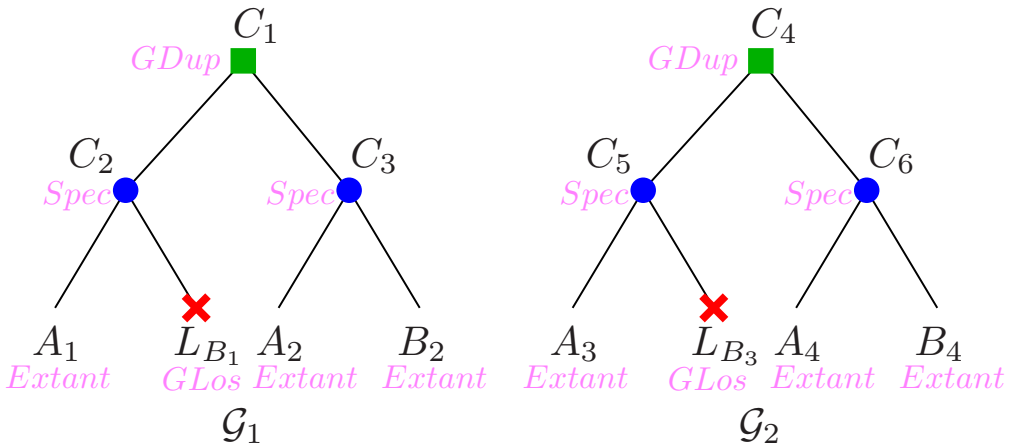
Exemple des cas *Spec/Spec* et *Spec/GDup*



c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	X	$0/\infty$	X	X	X	X
L_{B_1}	X	$0/0$	X	$0/0$	X	X	X
A_2	$0/\infty$	X	$0/\infty$	X	X	X	X
B_2	X	$0/0$	X	$\infty/0$	X	X	X
C_2	X	X	X	X	$1/0$	$0/1$	
C_3	X	X	X	X			
C_1	X	X	X	X			

- C_2-C_6 : cas 5. *Spec/Spec*
- C_2-C_4 : cas 4. *Spec/GDup*

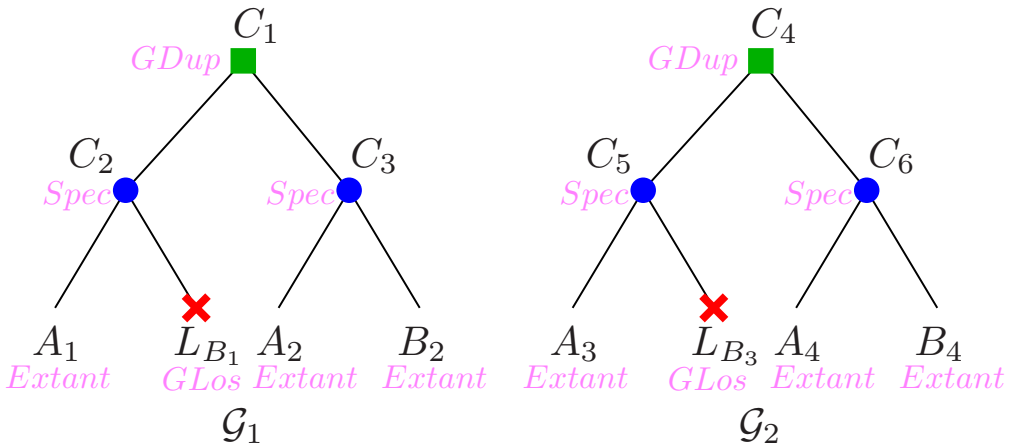
Exemple des cas *Spec/Spec* et *Spec/GDup*



c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	X	$0/\infty$	X	X	X	X
L_{B_1}	X	$0/0$	X	$0/0$	X	X	X
A_2	$0/\infty$	X	$0/\infty$	X	X	X	X
B_2	X	$0/0$	X	$\infty/0$	X	X	X
C_2	X	X	X	X	$1/0$	$0/1$	$1/0$
C_3	X	X	X	X			
C_1	X	X	X	X			

- C_2-C_6 : cas 5. *Spec/Spec*
- C_2-C_4 : cas 4. *Spec/GDup*

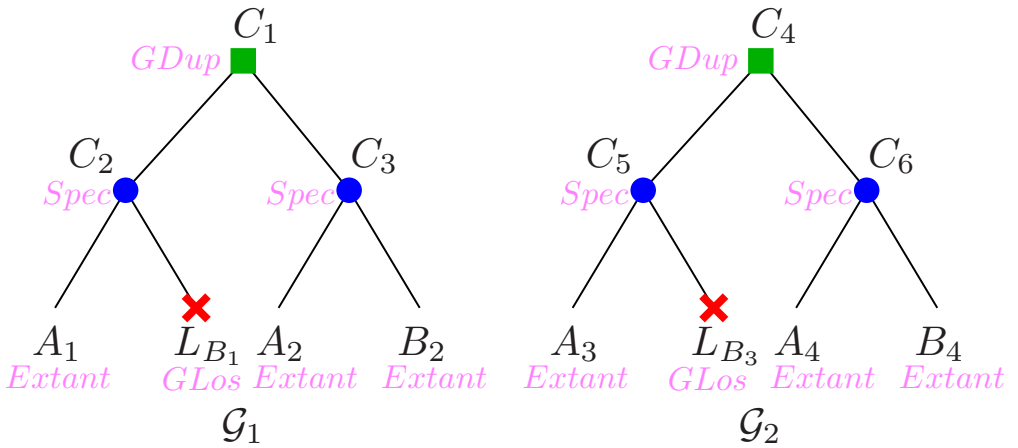
Exemple des cas *Spec/Spec* et *Spec/GDup*



c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	X	$0/\infty$	X	X	X	X
L_{B_1}	X	$0/0$	X	$0/0$	X	X	X
A_2	$0/\infty$	X	$0/\infty$	X	X	X	X
B_2	X	$0/0$	X	$\infty/0$	X	X	X
C_2	X	X	X	X	$1/0$	$0/1$	$1/0$
C_3	X	X	X	X			
C_1	X	X	X	X			

- C_2-C_6 : cas 5. *Spec/Spec*
- C_2-C_4 : cas 4. *Spec/GDup*
- C_3-C_5 & C_3-C_6 : cas 5. *Spec/Spec*

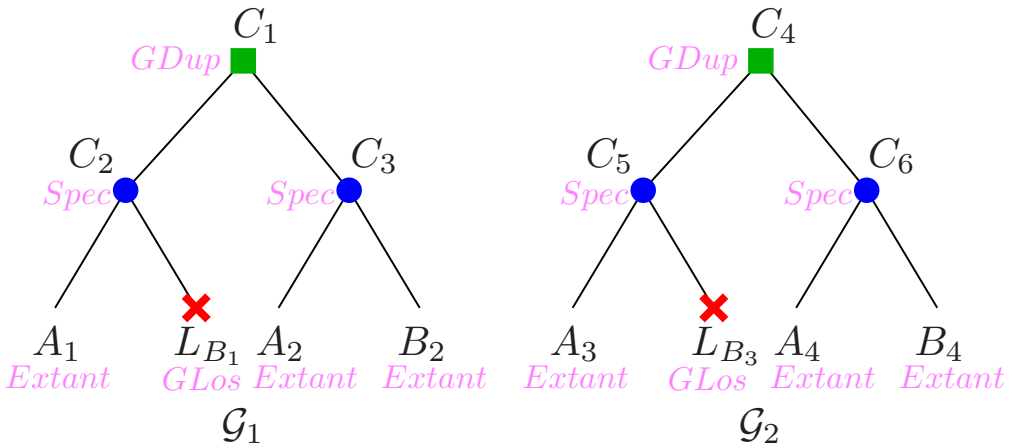
Exemple des cas *Spec/Spec* et *Spec/GDup*



c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	X	$0/\infty$	X	X	X	X
L_{B_1}	X	$0/0$	X	$0/0$	X	X	X
A_2	$0/\infty$	X	$0/\infty$	X	X	X	X
B_2	X	$0/0$	X	$\infty/0$	X	X	X
C_2	X	X	X	X	$1/0$	$0/1$	$1/0$
C_3	X	X	X	X	$0/1$	$1/1$	
C_1	X	X	X	X			

- C_2-C_6 : cas 5. *Spec/Spec*
- C_2-C_4 : cas 4. *Spec/GDup*
- C_3-C_5 & C_3-C_6 : cas 5. *Spec/Spec*

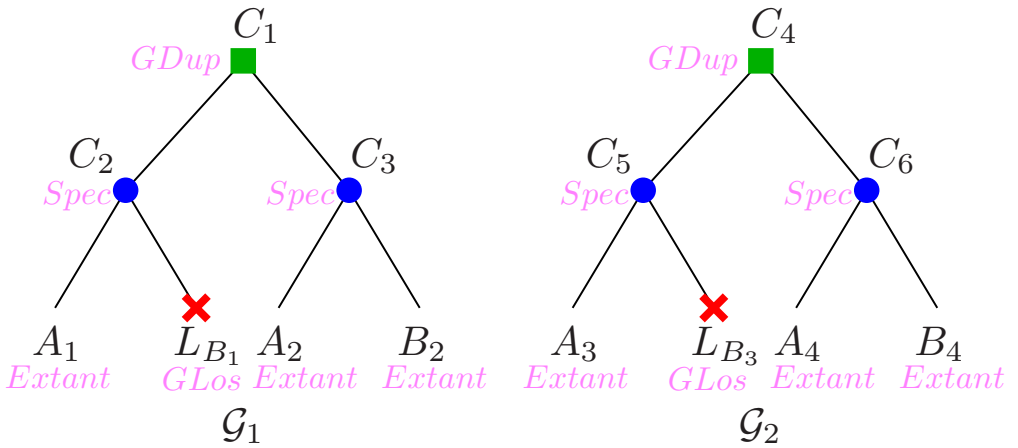
Exemple des cas *Spec/Spec* et *Spec/GDup*



c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	X	$0/\infty$	X	X	X	X
L_{B_1}	X	$0/0$	X	$0/0$	X	X	X
A_2	$0/\infty$	X	$0/\infty$	X	X	X	X
B_2	X	$0/0$	X	$\infty/0$	X	X	X
C_2	X	X	X	X	$1/0$	$0/1$	$1/0$
C_3	X	X	X	X	$0/1$	$1/1$	
C_1	X	X	X	X			

- C_2-C_6 : cas 5. *Spec/Spec*
- C_2-C_4 : cas 4. *Spec/GDup*
- C_3-C_5 & C_3-C_6 : cas 5. *Spec/Spec*
- C_3-C_4, C_1-C_5 & C_1-C_6 : cas 4. *Spec/GDup*

Exemple des cas *Spec/Spec* et *Spec/GDup*



c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	X	$0/\infty$	X	X	X	X
L_{B_1}	X	$0/0$	X	$0/0$	X	X	X
A_2	$0/\infty$	X	$0/\infty$	X	X	X	X
B_2	X	$0/0$	X	$\infty/0$	X	X	X
C_2	X	X	X	X	$1/0$	$0/1$	$1/0$
C_3	X	X	X	X	$0/1$	$1/1$	$1/1$
C_1	X	X	X	X	$1/0$	$1/1$	

- C_2-C_6 : cas 5. *Spec/Spec*
- C_2-C_4 : cas 4. *Spec/GDup*
- C_3-C_5 & C_3-C_6 : cas 5. *Spec/Spec*
- C_3-C_4, C_1-C_5 & C_1-C_6 : cas 4. *Spec/GDup*

Le dernier cas

Remarque : On n'examine pas le cas $E(n_1) = Extant$ et $E(n_2) = Spec$ car $S(n_1) \neq S(n_2)$

6. $\underline{E(n_1) = GDup \text{ and } E(n_2) = GDup}$

$$c_1(n_1, n_2) = \min(D1, D2, D12)$$

Le dernier cas

Remarque : On n'examine pas le cas $E(n_1) = Extant$ et $E(n_2) = Spec$ car $S(n_1) \neq S(n_2)$

6. $E(n_1) = GDup$ and $E(n_2) = GDup$

$$c_1(n_1, n_2) = \min(D1, D2, D12)$$

D1 : cas où n_1 se duplique avant n_2

$$D1 = \min \begin{cases} c_1(fg(n_1), n_2) + c_0(fd(n_1), n_2) \\ c_0(fg(n_1), n_2) + c_1(fd(n_1), n_2) \\ c_1(fg(n_1), n_2) + c_1(fd(n_1), n_2) + C(Gain) \\ c_0(fg(n_1), n_2) + c_0(fd(n_1), n_2) + C(Break) \end{cases}$$

Le dernier cas

Remarque : On n'examine pas le cas $E(n_1) = Extant$ et $E(n_2) = Spec$ car $S(n_1) \neq S(n_2)$

6. $E(n_1) = GDup$ and $E(n_2) = GDup$

$$c_1(n_1, n_2) = \min(D1, D2, D12)$$

D1 : cas où n_1 se duplique avant n_2

$$D1 = \min \begin{cases} c_1(fg(n_1), n_2) + c_0(fd(n_1), n_2) \\ c_0(fg(n_1), n_2) + c_1(fd(n_1), n_2) \\ c_1(fg(n_1), n_2) + c_1(fd(n_1), n_2) + C(Gain) \\ c_0(fg(n_1), n_2) + c_0(fd(n_1), n_2) + C(Break) \end{cases}$$

D2 : cas où n_2 se duplique avant n_1

$$D2 = \min \begin{cases} c_1(n_1, fg(n_2)) + c_0(n_1, fd(n_2)) \\ c_0(n_1, fg(n_2)) + c_1(n_1, fd(n_2)) \\ c_1(n_1, fg(n_2)) + c_1(n_1, fd(n_2)) + C(Gain) \\ c_0(n_1, fg(n_2)) + c_0(n_1, fd(n_2)) + C(Break) \end{cases}$$

Le dernier cas (suite)

D12 : cas où n_1 et n_2 se dupliquent en même temps (16 cas)

$$D12 = \min \begin{cases}
c_1(fg(n_1), fg(n_2)) + c_1(fd(n_1), fd(n_2)) + c_0(fg(n_1), fd(n_2)) + c_0(fd(n_1), fg(n_2)) \\
c_1(fg(n_1), fg(n_2)) + c_1(fd(n_1), fd(n_2)) + c_0(fg(n_1), fd(n_2)) + c_1(fd(n_1), fg(n_2)) + C(Gain) \\
c_1(fg(n_1), fg(n_2)) + c_1(fd(n_1), fd(n_2)) + c_1(fg(n_1), fd(n_2)) + c_0(fd(n_1), fg(n_2)) + C(Gain) \\
c_1(fg(n_1), fg(n_2)) + c_1(fd(n_1), fd(n_2)) + c_1(fg(n_1), fd(n_2)) + c_1(fd(n_1), fg(n_2)) + 2 * C(Gain) \\
c_1(fg(n_1), fg(n_2)) + c_0(fd(n_1), fd(n_2)) + c_0(fg(n_1), fd(n_2)) + c_0(fd(n_1), fg(n_2)) + C(Break) \\
c_1(fg(n_1), fg(n_2)) + c_0(fd(n_1), fd(n_2)) + c_0(fg(n_1), fd(n_2)) + c_1(fd(n_1), fg(n_2)) + C(Gain) + C(Break) \\
c_1(fg(n_1), fg(n_2)) + c_0(fd(n_1), fd(n_2)) + c_1(fg(n_1), fd(n_2)) + c_0(fd(n_1), fg(n_2)) + C(Gain) + C(Break) \\
c_0(fg(n_1), fg(n_2)) + c_1(fd(n_1), fd(n_2)) + c_0(fg(n_1), fd(n_2)) + c_0(fd(n_1), fg(n_2)) + C(Break) \\
c_0(fg(n_1), fg(n_2)) + c_1(fd(n_1), fd(n_2)) + c_0(fg(n_1), fd(n_2)) + c_1(fd(n_1), fg(n_2)) + C(Gain) + C(Break) \\
c_0(fg(n_1), fg(n_2)) + c_1(fd(n_1), fd(n_2)) + c_1(fg(n_1), fd(n_2)) + c_0(fd(n_1), fg(n_2)) + C(Gain) + C(Break) \\
c_0(fg(n_1), fg(n_2)) + c_0(fd(n_1), fd(n_2)) + c_1(fg(n_1), fd(n_2)) + c_1(fd(n_1), fg(n_2)) \\
c_0(fg(n_1), fg(n_2)) + c_1(fd(n_1), fd(n_2)) + c_1(fg(n_1), fd(n_2)) + c_1(fd(n_1), fg(n_2)) + C(Gain) \\
c_1(fg(n_1), fg(n_2)) + c_0(fd(n_1), fd(n_2)) + c_1(fg(n_1), fd(n_2)) + c_1(fd(n_1), fg(n_2)) + C(Gain) \\
c_0(fg(n_1), fg(n_2)) + c_0(fd(n_1), fd(n_2)) + c_1(fg(n_1), fd(n_2)) + c_0(fd(n_1), fg(n_2)) + C(Break) \\
c_0(fg(n_1), fg(n_2)) + c_0(fd(n_1), fd(n_2)) + c_0(fg(n_1), fd(n_2)) + c_1(fd(n_1), fg(n_2)) + C(Break) \\
c_0(fg(n_1), fg(n_2)) + c_0(fd(n_1), fd(n_2)) + c_0(fg(n_1), fd(n_2)) + c_0(fd(n_1), fg(n_2)) + 2 * C(Break)
\end{cases}$$

Le dernier cas (suite)

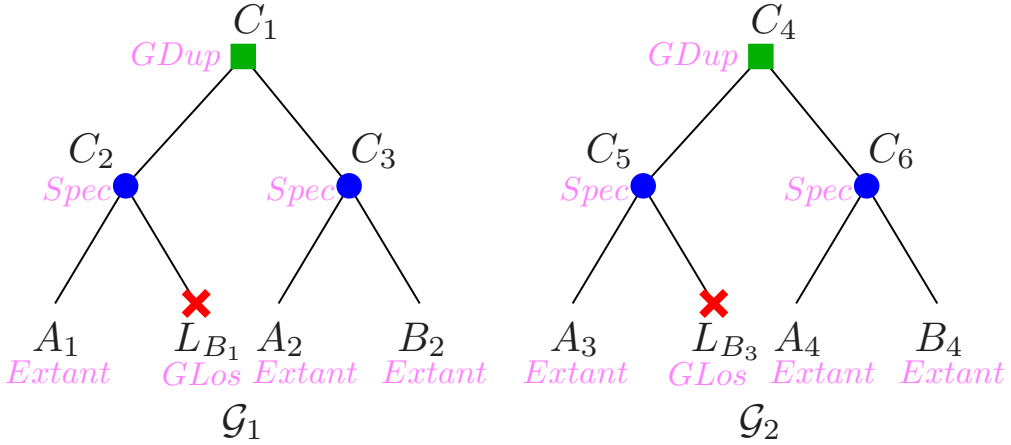
D12 : cas où n_1 et n_2 se dupliquent en même temps (16 cas)

$$D12 = \min \begin{cases}
c_1(fg(n_1), fg(n_2)) + c_1(fd(n_1), fd(n_2)) + c_0(fg(n_1), fd(n_2)) + c_0(fd(n_1), fg(n_2)) \\
c_1(fg(n_1), fg(n_2)) + c_1(fd(n_1), fd(n_2)) + c_0(fg(n_1), fd(n_2)) + c_1(fd(n_1), fg(n_2)) + C(Gain) \\
c_1(fg(n_1), fg(n_2)) + c_1(fd(n_1), fd(n_2)) + c_1(fg(n_1), fd(n_2)) + c_0(fd(n_1), fg(n_2)) + C(Gain) \\
c_1(fg(n_1), fg(n_2)) + c_1(fd(n_1), fd(n_2)) + c_1(fg(n_1), fd(n_2)) + c_1(fd(n_1), fg(n_2)) + 2 * C(Gain) \\
c_1(fg(n_1), fg(n_2)) + c_0(fd(n_1), fd(n_2)) + c_0(fg(n_1), fd(n_2)) + c_0(fd(n_1), fg(n_2)) + C(Break) \\
c_1(fg(n_1), fg(n_2)) + c_0(fd(n_1), fd(n_2)) + c_0(fg(n_1), fd(n_2)) + c_1(fd(n_1), fg(n_2)) + C(Gain) + C(Break) \\
c_1(fg(n_1), fg(n_2)) + c_0(fd(n_1), fd(n_2)) + c_1(fg(n_1), fd(n_2)) + c_0(fd(n_1), fg(n_2)) + C(Gain) + C(Break) \\
c_0(fg(n_1), fg(n_2)) + c_1(fd(n_1), fd(n_2)) + c_0(fg(n_1), fd(n_2)) + c_0(fd(n_1), fg(n_2)) + C(Break) \\
c_0(fg(n_1), fg(n_2)) + c_1(fd(n_1), fd(n_2)) + c_0(fg(n_1), fd(n_2)) + c_1(fd(n_1), fg(n_2)) + C(Gain) + C(Break) \\
c_0(fg(n_1), fg(n_2)) + c_1(fd(n_1), fd(n_2)) + c_1(fg(n_1), fd(n_2)) + c_0(fd(n_1), fg(n_2)) + C(Gain) + C(Break) \\
c_0(fg(n_1), fg(n_2)) + c_0(fd(n_1), fd(n_2)) + c_1(fg(n_1), fd(n_2)) + c_1(fd(n_1), fg(n_2)) \\
c_0(fg(n_1), fg(n_2)) + c_1(fd(n_1), fd(n_2)) + c_1(fg(n_1), fd(n_2)) + c_1(fd(n_1), fg(n_2)) + C(Gain) \\
c_1(fg(n_1), fg(n_2)) + c_0(fd(n_1), fd(n_2)) + c_1(fg(n_1), fd(n_2)) + c_1(fd(n_1), fg(n_2)) + C(Gain) \\
c_0(fg(n_1), fg(n_2)) + c_0(fd(n_1), fd(n_2)) + c_1(fg(n_1), fd(n_2)) + c_0(fd(n_1), fg(n_2)) + C(Break) \\
c_0(fg(n_1), fg(n_2)) + c_0(fd(n_1), fd(n_2)) + c_0(fg(n_1), fd(n_2)) + c_1(fd(n_1), fg(n_2)) + C(Break) \\
c_0(fg(n_1), fg(n_2)) + c_0(fd(n_1), fd(n_2)) + c_0(fg(n_1), fd(n_2)) + c_0(fd(n_1), fg(n_2)) + 2 * C(Break)
\end{cases}$$

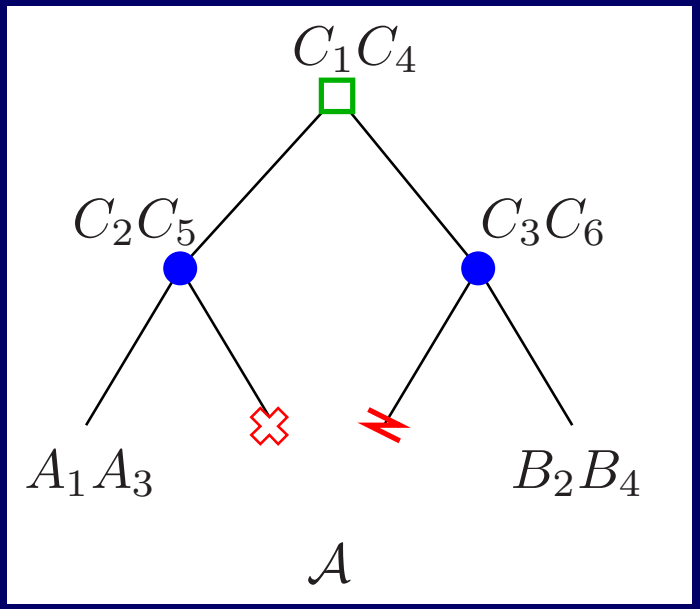
Coût c_0 pour le cas $GDup$-$GDup$:

$$c_0(n_1, n_2) \sim \min(D1_0, D2_0)$$

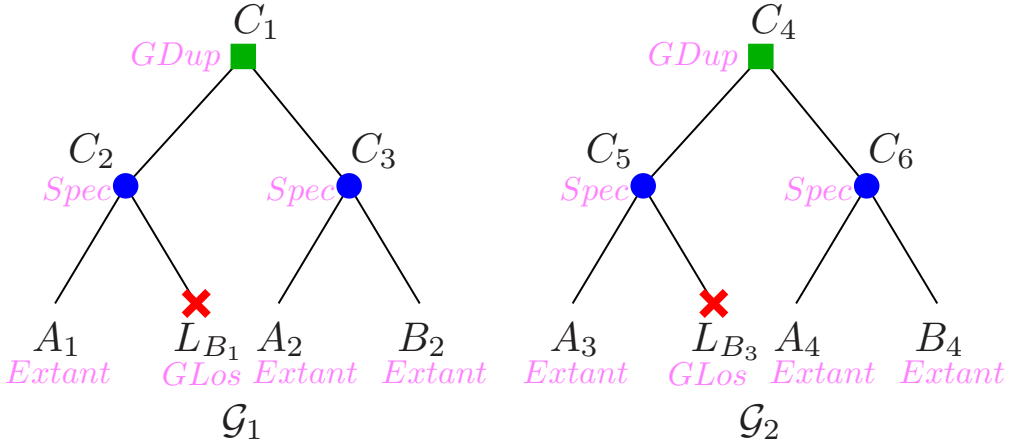
Fin de l'exemple : le backtracking



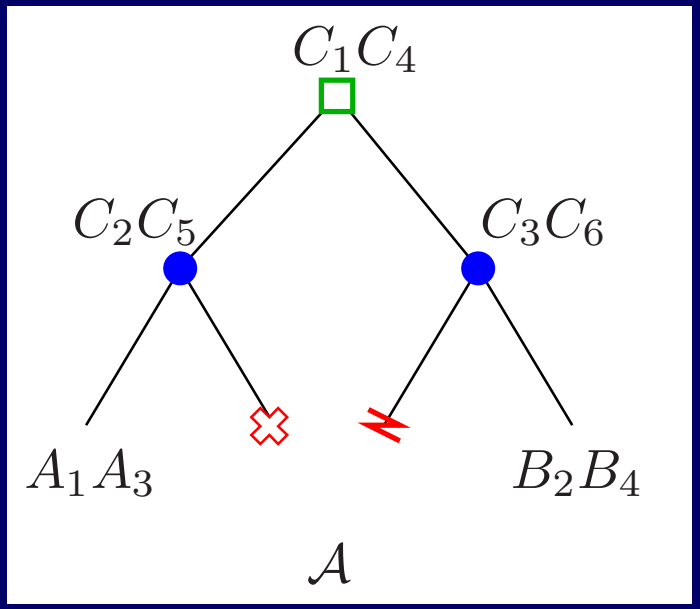
c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	X	$0/\infty$	X	X	X	X
L_{B_1}	X	$0/0$	X	$0/0$	X	X	X
A_2	$0/\infty$	X	$0/\infty$	X	X	X	X
B_2	X	$0/0$	X	$\infty/0$	X	X	X
C_2	X	X	X	X	$1/0$	$0/1$	$1/0$
C_3	X	X	X	X	$0/1$	$1/1$	$1/1$
C_1	X	X	X	X	$1/0$	$1/1$	$2/1$



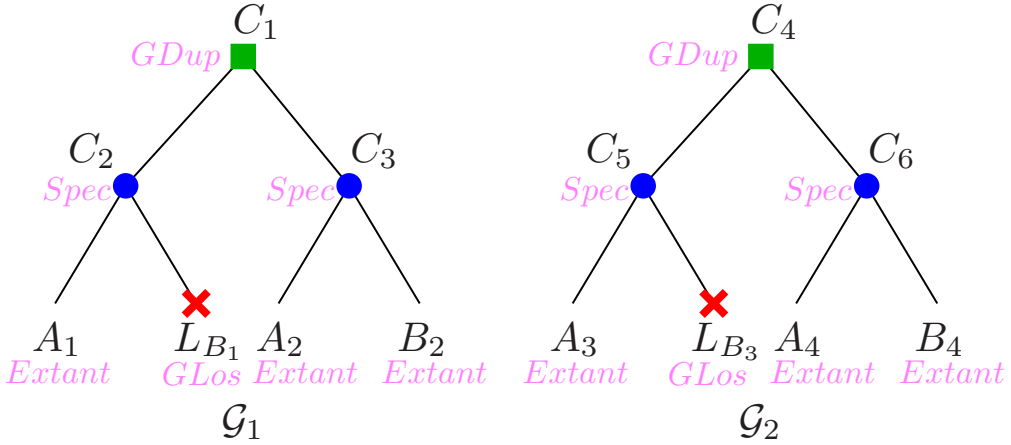
Fin de l'exemple : le backtracking



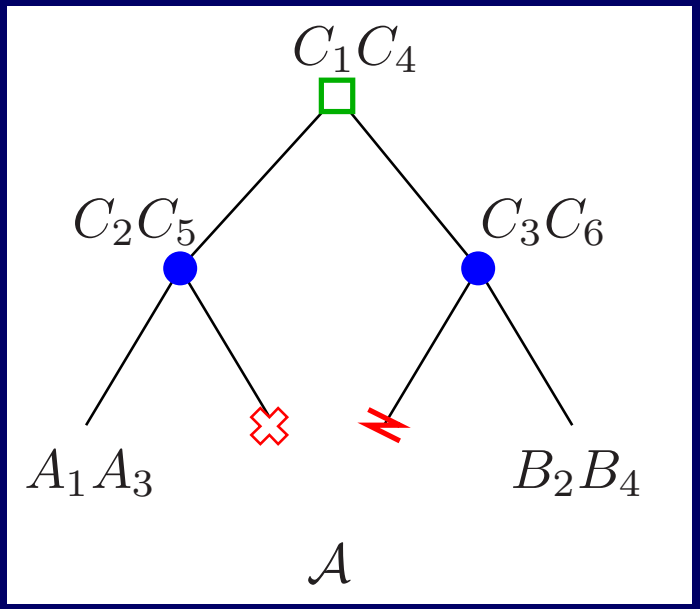
c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	✗	$0/\infty$	✗	✗	✗	✗
L_{B_1}	✗	$0/0$	✗	$0/0$	✗	✗	✗
A_2	$0/\infty$	✗	$0/\infty$	✗	✗	✗	✗
B_2	✗	$0/0$	✗	$\infty/0$	✗	✗	✗
C_2	✗	✗	✗	✗	$1/0$	$0/1$	$1/0$
C_3	✗	✗	✗	✗	$0/1$	$1/1$	$1/1$
C_1	✗	✗	✗	✗	$1/0$	$1/1$	$2/1$



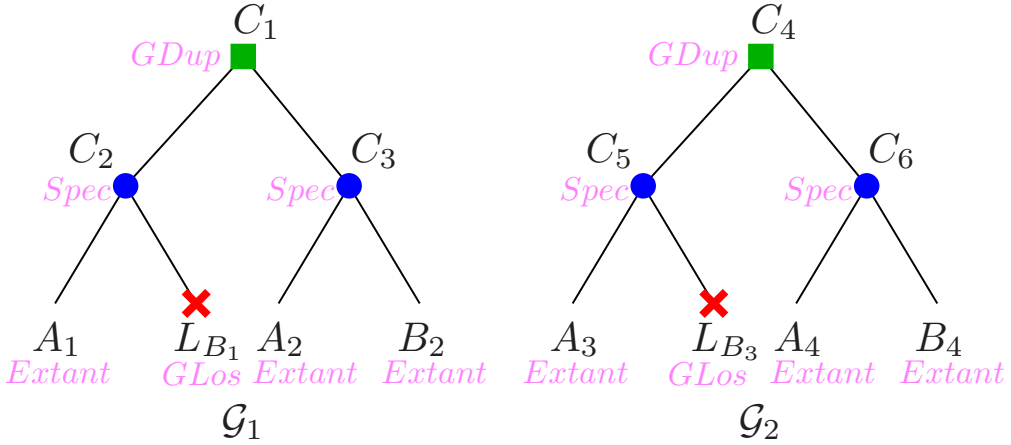
Fin de l'exemple : le backtracking



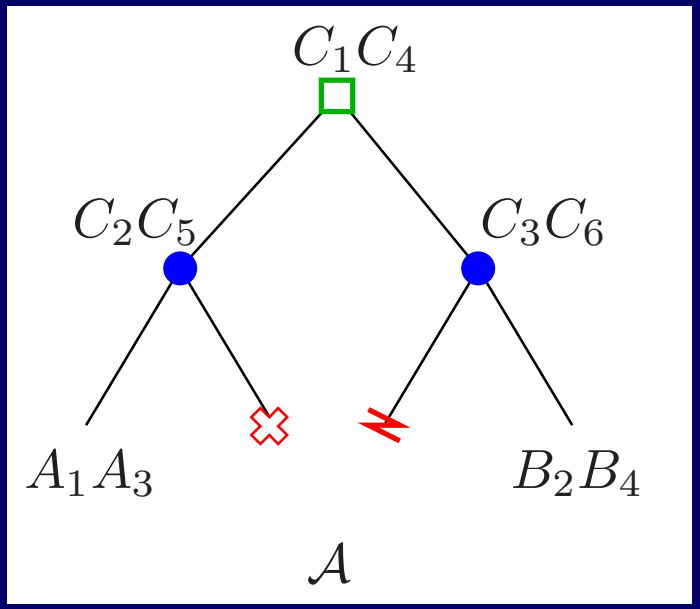
c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	X	$0/\infty$	X	X	X	X
L_{B_1}	X	$0/0$	X	$0/0$	X	X	X
A_2	$0/\infty$	X	$0/\infty$	X	X	X	X
B_2	X	$0/0$	X	$\infty/0$	X	X	X
C_2	X	X	X	X	$1/0$	$0/1$	$1/0$
C_3	X	X	X	X	$0/1$	$1/1$	$1/1$
C_1	X	X	X	X	$1/0$	$1/1$	$2/1$



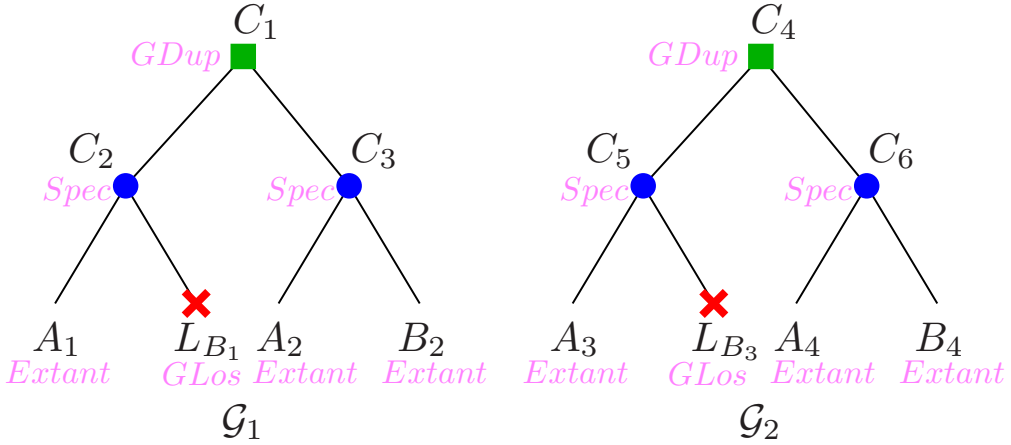
Fin de l'exemple : le backtracking



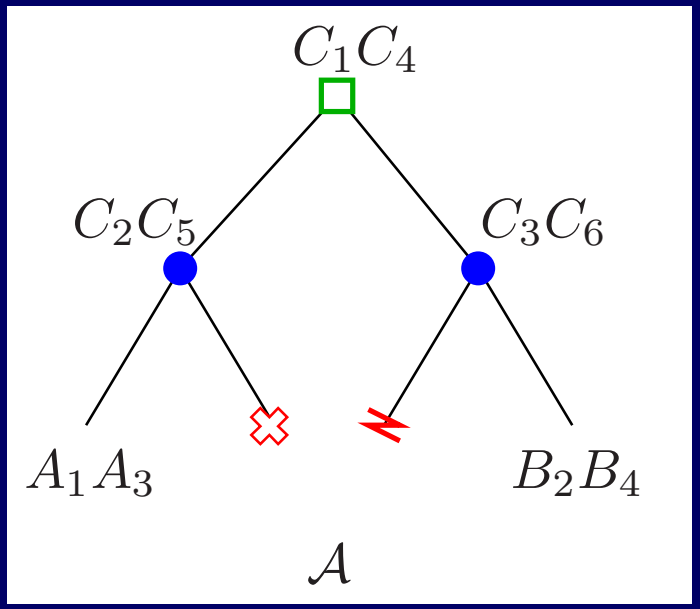
c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	X	$0/\infty$	X	X	X	X
L_{B_1}	X	$0/0$	X	$0/0$	X	X	X
A_2	$0/\infty$	X	$0/\infty$	X	X	X	X
B_2	X	$0/0$	X	$\infty/0$	X	X	X
C_2	X	X	X	X	$1/0$	$0/1$	$1/0$
C_3	X	X	X	X	$0/1$	$1/1$	$1/1$
C_1	X	X	X	X	$1/0$	$1/1$	$2/1$



Fin de l'exemple : le backtracking



c_0/c_1	A_3	L_{B_3}	A_4	B_4	C_5	C_6	C_4
A_1	$\infty/0$	X	$0/\infty$	X	X	X	X
L_{B_1}	X	$0/0$	X	$0/0$	X	X	X
A_2	$0/\infty$	X	$0/\infty$	X	X	X	X
B_2	X	$0/0$	X	$\infty/0$	X	X	X
C_2	X	X	X	X	$1/0$	$0/1$	$1/0$
C_3	X	X	X	X	$0/1$	$1/1$	$1/1$
C_1	X	X	X	X	$1/0$	$1/1$	$2/1$



Caractéristiques de DeCo

- DeCo gère les pertes et les duplications de gènes (sans besoin de les ordonner), ainsi que des familles de gènes sans restriction d'un seul représentant pas espèce

Caractéristiques de DeCo

- DeCo gère les pertes et les duplications de gènes (sans besoin de les ordonner), ainsi que des familles de gènes sans restriction d'un seul représentant pas espèce
- Algorithme exact : on minimise le nombre de cassures et de créations d'adjacence

Caractéristiques de DeCo

- DeCo gère les pertes et les duplications de gènes (sans besoin de les ordonner), ainsi que des familles de gènes sans restriction d'un seul représentant pas espèce
- Algorithme exact : on minimise le nombre de cassures et de créations d'adjacence
- Généralisation des algorithmes de parcimonie de Fitch-Sankoff sur un alphabet binaire

Caractéristiques de DeCo

- DeCo gère les pertes et les duplications de gènes (sans besoin de les ordonner), ainsi que des familles de gènes sans restriction d'un seul représentant pas espèce
- Algorithme exact : on minimise le nombre de cassures et de créations d'adjacence
- Généralisation des algorithmes de parcimonie de Fitch-Sankoff sur un alphabet binaire
- Hypothèse : les adjacences évoluent indépendemment
  $\Rightarrow$ utilisation du principe de programmation dynamique

Caractéristiques de DeCo

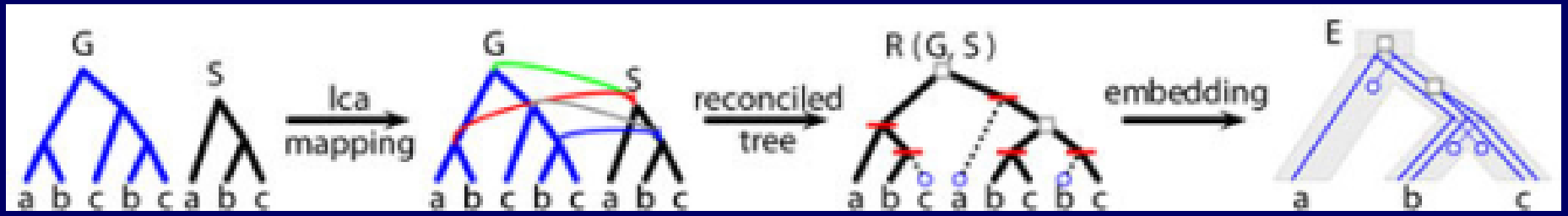
- DeCo gère les pertes et les duplications de gènes (sans besoin de les ordonner), ainsi que des familles de gènes sans restriction d'un seul représentant pas espèce
- Algorithme exact : on minimise le nombre de cassures et de créations d'adjacence
- Généralisation des algorithmes de parcimonie de Fitch-Sankoff sur un alphabet binaire
- Hypothèse : les adjacences évoluent indépendemment
  $\Rightarrow$ utilisation du principe de programmation dynamique
- Complexité en temps et en espace en $O(n^2)$ où n est le nombre de nœuds d'un arbre de gènes

Caractéristiques de DeCo

- DeCo gère les pertes et les duplications de gènes (sans besoin de les ordonner), ainsi que des familles de gènes sans restriction d'un seul représentant pas espèce
- Algorithme exact : on minimise le nombre de cassures et de créations d'adjacence
- Généralisation des algorithmes de parcimonie de Fitch-Sankoff sur un alphabet binaire
- Hypothèse : les adjacences évoluent indépendemment
  $\Rightarrow$ utilisation du principe de programmation dynamique
- Complexité en temps et en espace en $O(n^2)$ où n est le nombre de nœuds d'un arbre de gènes
- Inclus dans un "pipeline" pour traiter le problème général

Caractéristique de la réconciliation

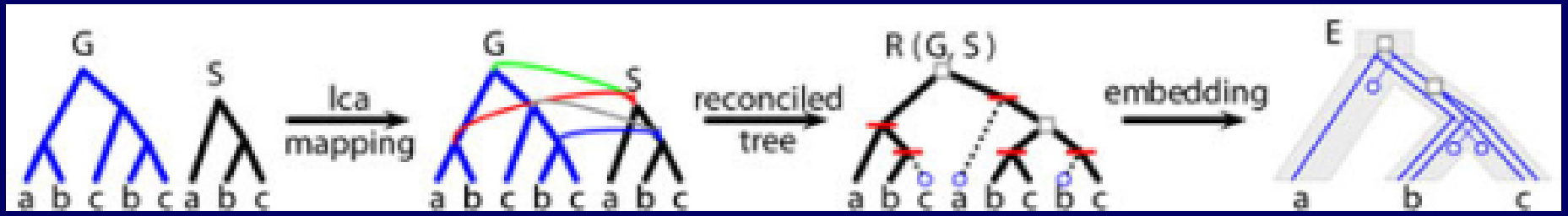
- Réconciliation LCA (Least Common Ancestor)



- Solution unique
- Événements géniques pris en compte : duplications et pertes

Caractéristique de la réconciliation

- Réconciliation LCA (Least Common Ancestor)



- Solution unique
- Événements géniques pris en compte : duplications et pertes
- Constitue le "pré-traitement" : tous les arbres de gènes du jeu de donnée sont réconciliés avec l'arbre d'espèce $\mathcal{S}$
  $\rightarrow$ certains sont éliminés (non binaire par exemple)
  $\rightarrow$ d'autres effeuillés (gènes d'espèces $\notin \mathcal{S}$)

Les classes d'adjacences

- On groupe ensemble les adjacences qui peuvent avoir une origine évolutive commune

Les classes d'adjacences

- On groupe ensemble les adjacences qui peuvent avoir une origine évolutive commune

- Analogie avec les familles de gènes :
  $\rightarrow$ une classe produit une forêt d'arbres d'adjacences
  $\rightarrow$ un arbre d'adjacences contient des adjacences homologues
  $\Rightarrow$ famille d'adjacences

Les classes d'adjacences

- On groupe ensemble les adjacences qui peuvent avoir une origine évolutive commune

- Analogie avec les familles de gènes :
  $\rightarrow$ une classe produit une forêt d'arbres d'adjacences
  $\rightarrow$ un arbre d'adjacences contient des adjacences homologues
  $\Rightarrow$ famille d'adjacences

- Chaque classe est associée à exactement 2 arbres de gènes, ceux-ci peuvent-être des sous-arbres d'arbres passés en entrée

Les classes d'adjacences

- On groupe ensemble les adjacences qui peuvent avoir une origine évolutive commune

- Analogie avec les familles de gènes :
  $\rightarrow$ une classe produit une forêt d'arbres d'adjacences
  $\rightarrow$ un arbre d'adjacences contient des adjacences homologues
  $\Rightarrow$ famille d'adjacences

- Chaque classe est associée à exactement 2 arbres de gènes, ceux-ci peuvent-être des sous-arbres d'arbres passés en entrée

- Les classes d'adjacences forment des classes d'équivalence (au sens mathématique)

Relation d'équivalence entre adjacences

Deux adjacences AB et CD sont dans la même classe d'équivalence si

- A et C sont dans le même arbre de gènes, noté $\mathcal{G}_1$, ainsi que B et D, dans un arbre noté $\mathcal{G}_2$

Relation d'équivalence entre adjacences

Deux adjacences AB et CD sont dans la même classe d'équivalence si

- A et C sont dans le même arbre de gènes, noté $\mathcal{G}_1$, ainsi que B et D, dans un arbre noté $\mathcal{G}_2$

$\rightarrow$ si A et B sont dans le même arbre de gènes ($\mathcal{G}_1 = \mathcal{G}_2$), alors $LCA(A, B) = LCA(C, D) = n_d$ (et c'est un nœud de duplication)

Relation d'équivalence entre adjacences

Deux adjacences AB et CD sont dans la même classe d'équivalence si

- A et C sont dans le même arbre de gènes, noté $\mathcal{G}_1$, ainsi que B et D, dans un arbre noté $\mathcal{G}_2$

$\rightarrow$ si A et B sont dans le même arbre de gènes ($\mathcal{G}_1 = \mathcal{G}_2$), alors $LCA(A, B) = LCA(C, D) = n_d$ (et c'est un nœud de duplication)

$\rightarrow$ sinon, il existe 2 nœuds $n_1 \in \mathcal{G}_1, n_2 \in \mathcal{G}_2$ tels que

— $S(n_1) = S(n_2)$

— A et C sont des descendants de n_1

— B et D sont des descendants de n_2

Relation d'équivalence entre adjacences

Deux adjacences AB et CD sont dans la même classe d'équivalence si

- A et C sont dans le même arbre de gènes, noté $\mathcal{G}_1$, ainsi que B et D, dans un arbre noté $\mathcal{G}_2$

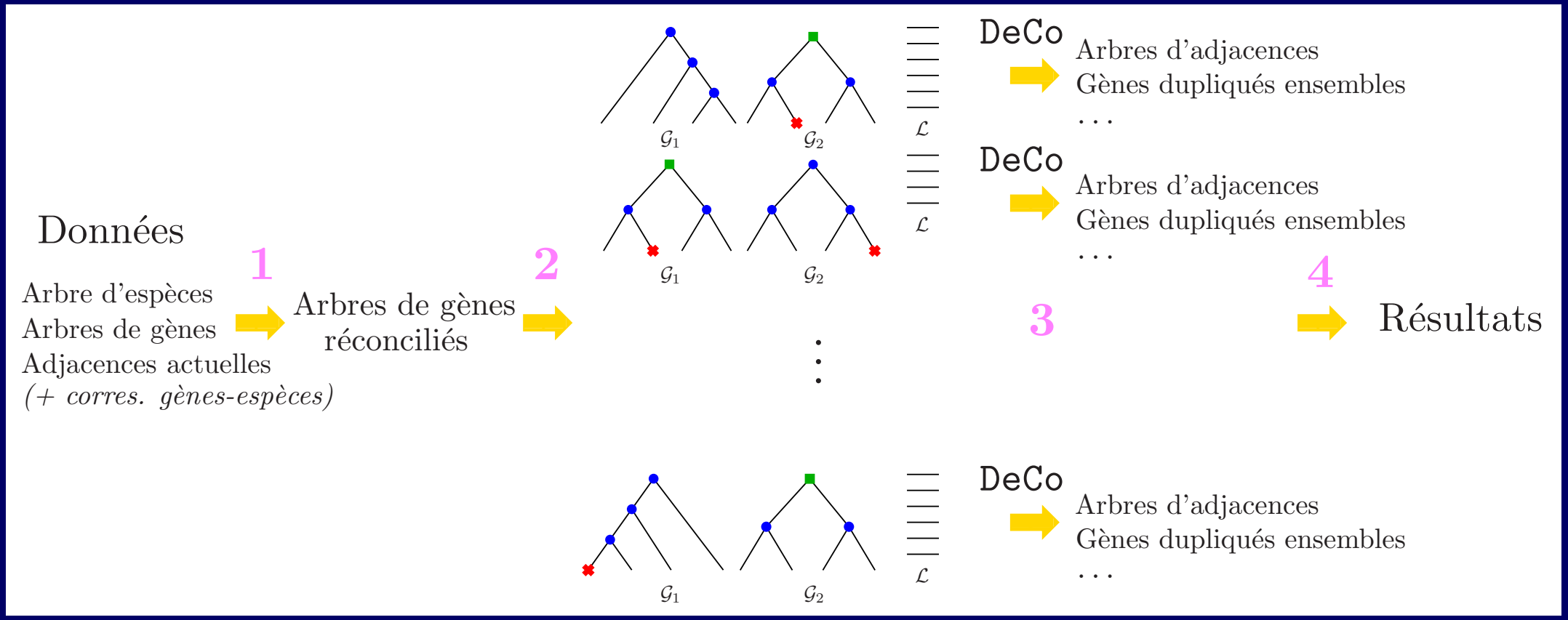
$\rightarrow$ si A et B sont dans le même arbre de gènes ($\mathcal{G}_1 = \mathcal{G}_2$), alors $LCA(A, B) = LCA(C, D) = n_d$ (et c'est un nœud de duplication)

$\rightarrow$ sinon, il existe 2 nœuds $n_1 \in \mathcal{G}_1, n_2 \in \mathcal{G}_2$ tels que
- $S(n_1) = S(n_2)$
- A et C sont des descendants de n_1
- B et D sont des descendants de n_2

Les 2 arbres de gènes associés à cette classe sont soit

- les 2 sous-arbres de $\mathcal{G}_1 = \mathcal{G}_2$ enracinés aux fils de n_d
- le sous-arbre de $\mathcal{G}_1$ enraciné à n_1 et le sous-arbre de $\mathcal{G}_2$ enraciné à n_2

Pipeline DeCo



1. Réconciliation
2. Création des classes
3. DeCo
4. Synthèse

Programmé en C++/Bio++, disponible sur http://pbil.univ-lyon1.fr/software/DeCo/

- À l'issue de tous les appels à `DeCo` on récupère plusieurs informations :

  → Les adjacences ancestrales par espèces

  → Les gènes qui se sont dupliqués ou perdus ensemble

  → Les arbres d'adjacences

  → Les degrés des gènes ancestraux (nombre de voisins)

  → Taille des arbres d'adjacences, nb d'arbres /classes, . . .

- Introduction
- Modèle
- Démarche
- Résultats
- Conclusion & perpectives

- Deux jeux de données principaux : mammifères et plantes

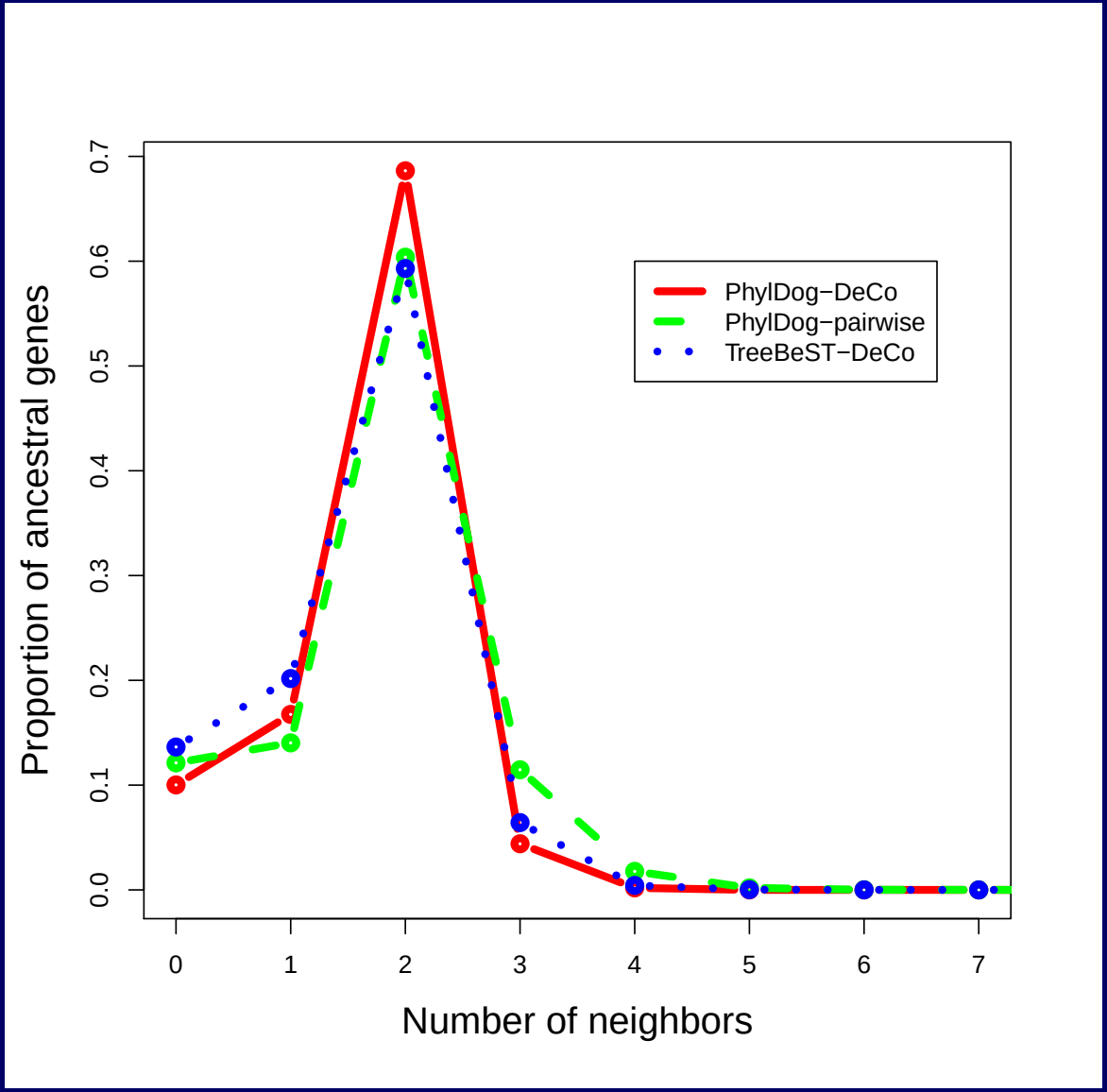
  → Mammifères : 5039 arbres de gènes (Ensembl 57)
  11 espèces dont les génomes étaient assemblés,
  $\sim 107\,000$ gènes

  → Plantes (angiospermes) : 35 182 arbres (EnsemblPlant 12)
  9 espèces dont les génomes étaient assemblés

- Gros jeux de données plantes $\sim 50\,000$ arbres, $\sim 615\,000$ gènes

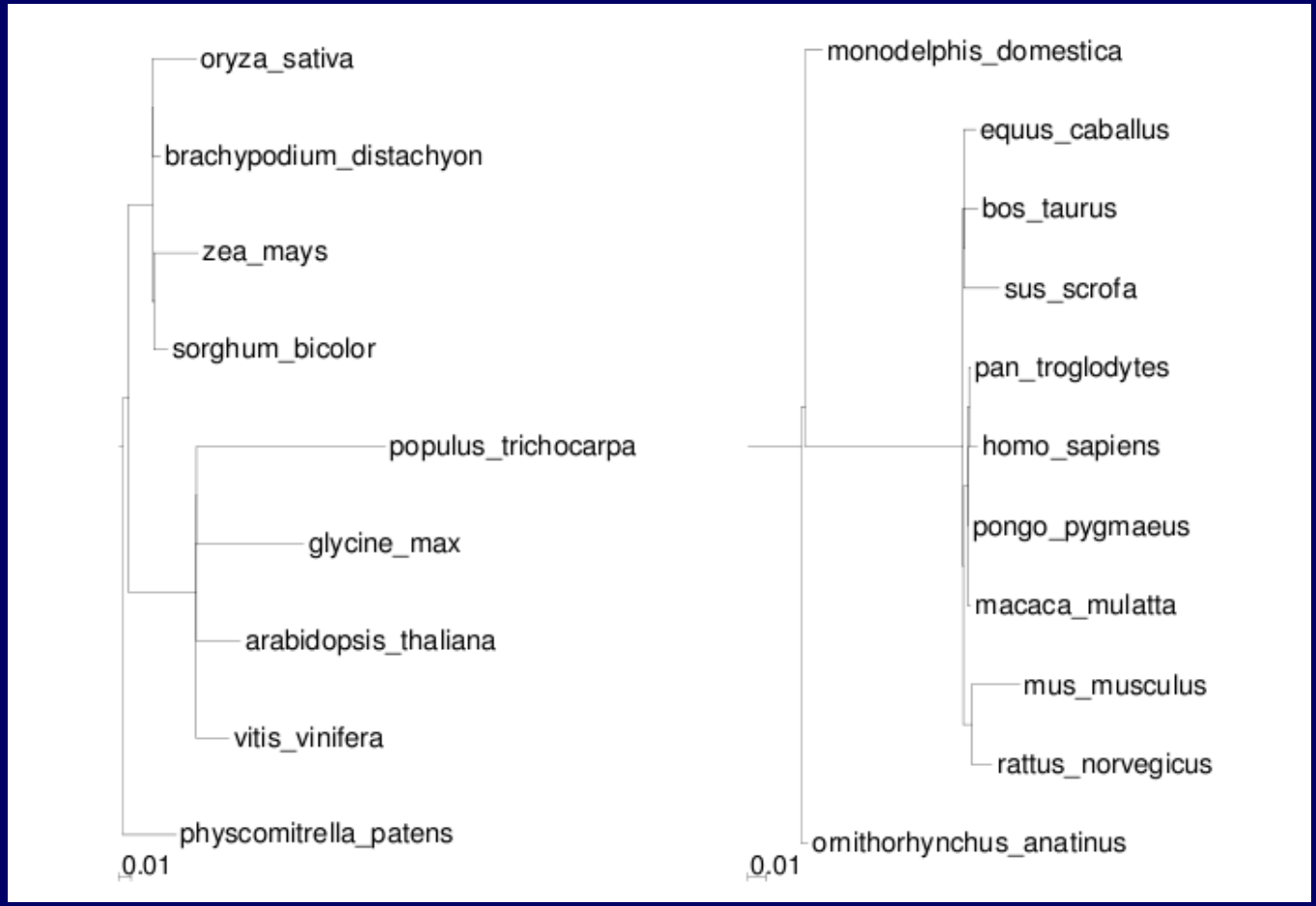
- Temps de calcul moyen d'une dizaine de minutes (réconciliation comprise)

Linéarité : nombres de voisins



Pour avoir de meilleurs génomes ancestraux il faut à la fois de bons arbres et de bonnes méthodes d'inférences d'adjacences

Phylogénies en fonction du nb de dupli en bloc



- Clades des angiospermes et des mammifères : temps de divergence et nb d'espèces assemblées similaires
- Longueur de branches = # $ADup$ / # nb gènes ancestraux sur cette branche
- En moyenne les lg de branches sont 3 fois plus longues chez les plantes

- Introduction
- Modèle
- Démarche
- Résultats
- Conclusion & perpectives

Première méthode pour reconstruire l'évolution de relations entres gènes

Première méthode pour reconstruire l'évolution de relations entres gènes

- Modèle minimaliste mais algorithme exact et rapide

Première méthode pour reconstruire l'évolution de relations entres gènes

- Modèle minimaliste mais algorithme exact et rapide
- Appliquée ici à la relation d'adjacence $\Rightarrow$ donne accès à de l'information sur les génomes ancestraux

Première méthode pour reconstruire l'évolution de relations entres gènes

- Modèle minimaliste mais algorithme exact et rapide
- Appliquée ici à la relation d'adjacence $\Rightarrow$ donne accès à de l'information sur les génomes ancestraux
- Peut facilement être étendu à d'autres type de relations : interactions protéines-protéines, présence dans une même voie métabolique, co-expression, . . .

- Stage de M2R BCD Pierre-Antoine Jean $mars \rightarrow sept\ 2013$
  Algorithmique pour l'évolution des interactions géniques
  Encadrement : *Vincent Berry* et *Annie Chateau* (*Éric Tannier*, *SB*)

- Stage de M2R BCD Pierre-Antoine Jean $mars \rightarrow sept\ 2013$
  Algorithmique pour l'évolution des interactions géniques
  Encadrement : *Vincent Berry* et *Annie Chateau* (*Éric Tannier*, *SB*)

- `DeCoLT` Lateral gene transfer, rearrangement, reconciliation *Murray Patterson*, *Gergely Szöllősi*, *Vincent Daubin* et *Eric Tannier*. BMC Bioinformatics 2013, 14(Suppl 15):S4

- Stage de M2R BCD Pierre-Antoine Jean $mars \rightarrow sept\ 2013$
  Algorithmique pour l'évolution des interactions géniques
  Encadrement : *Vincent Berry* et *Annie Chateau* (*Éric Tannier*, *SB*)

- `DeCoLT` Lateral gene transfer, rearrangement, reconciliation *Murray Patterson*, *Gergely Szöllősi*, *Vincent Daubin* et *Eric Tannier*. BMC Bioinformatics 2013, 14(Suppl 15):S4

- Thèse Magali Semeria (LBBE - Lyon) $sept\ 2012 \rightarrow 2015$
  Modèle probabiliste pour l'évolution des relations entre gènes
  Encadrement : *Laurent Guéguen* et *Éric Tannier*

Travaux connexes

- Stage de M2R BCD Pierre-Antoine Jean $mars \rightarrow sept\ 2013$
  Algorithmique pour l'évolution des interactions géniques
  Encadrement : *Vincent Berry* et *Annie Chateau* (*Éric Tannier*, *SB*)

- DeCoLT Lateral gene transfer, rearrangement, reconciliation *Murray Patterson*, *Gergely Szöllősi*, *Vincent Daubin* et *Eric Tannier*. BMC Bioinformatics 2013, 14(Suppl 15):S4

- Thèse Magali Semeria (LBBE - Lyon) $sept\ 2012 \rightarrow 2015$
  Modèle probabiliste pour l'évolution des relations entre gènes
  Encadrement : *Laurent Guéguen* et *Éric Tannier*

- Stages en cours à l'UM2 $2013 - 14$
  $\rightarrow$ Groupe M1 IPS : découper le code en 4 exécutables distincts
  $\rightarrow$ M2 BCD : "Reconstruction de l'histoire évolutive des génomes : réconciliations et reconstructions de relations ancestrales"

Perspectives

- Prise en compte de l'état d'assemblage des génomes pour moins pénaliser les cassures d'adjacences aux feuilles des génomes mal/peu assemblés

Perspectives

- Prise en compte de l'état d'assemblage des génomes pour moins pénaliser les cassures d'adjacences aux feuilles des génomes mal/peu assemblés
- Intégration à la fonction objectif de `DeCo` des duplications et pertes en bloc

- Prise en compte de l'état d'assemblage des génomes pour moins pénaliser les cassures d'adjacences aux feuilles des génomes mal/peu assemblés
- Intégration à la fonction objectif de `DeCo` des duplications et pertes en bloc
- Respect de la contrainte de linéarité du génome
  - $\rightarrow$ Définition d'une mesure de linéarité fondée sur les relations d'adjacences entre couples de gènes
  - $\rightarrow$ Converger vers la linéarité des génomes ancestraux par le biais d'aller-retour entre génomes reconstruits et arbres de gènes

Perspectives

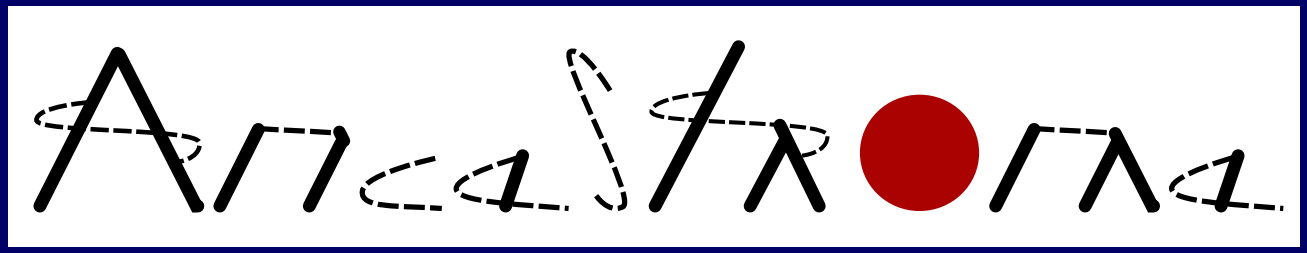
- Prise en compte de l'état d'assemblage des génomes pour moins pénaliser les cassures d'adjacences aux feuilles des génomes mal/peu assemblés

- Intégration à la fonction objectif de `DeCo` des duplications et pertes en bloc

- Respect de la contrainte de linéarité du génome
  - $\rightarrow$ Définition d'une mesure de linéarité fondée sur les relations d'adjacences entre couples de gènes
  - $\rightarrow$ Converger vers la linéarité des génomes ancestraux par le biais d'aller-retour entre génomes reconstruits et arbres de gènes

- Considérer l'histoire évolutive conjointe non plus de deux familles de gènes mais d'un très grand nombre de familles de gènes

Crédits



- Certaines parties de cette présentations sont issues des rapports de stage de Coralie Gallien et Pierre-Antoine Jean
- Image réconciliation issue de http://openi.nlm.nih.gov/