Bioinformatique : Algorithmes d'analyse de séquences

Édition d'arborescences
Pré-requis et algorithme de Zhang & Shasha

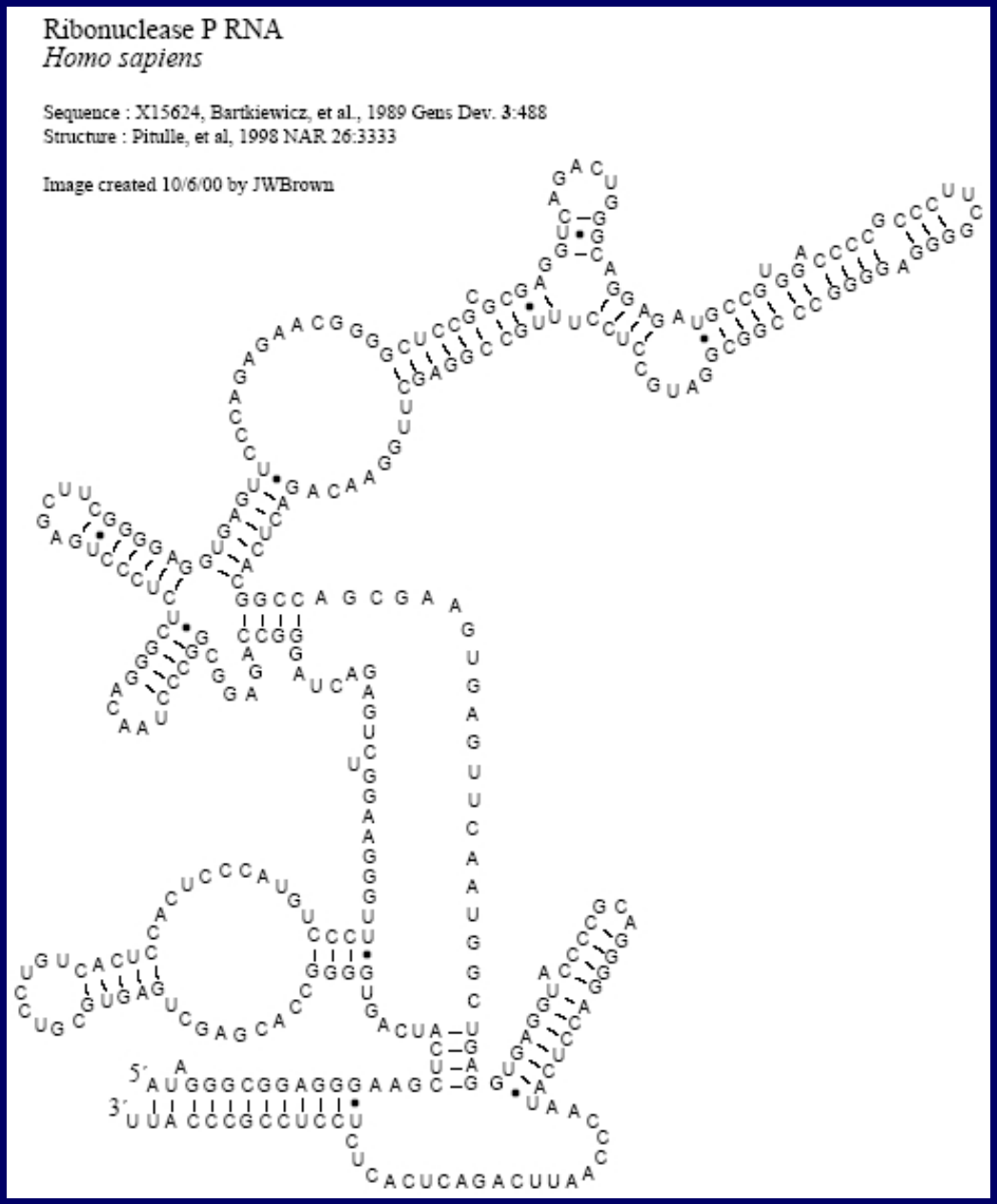
Sèverine Bérard

AMAP - Université Montpellier 2

Plan

- Introduction : modélisation
- Quelques rappels sur les arbres
- Édition et *mapping*
- L'algorithme de Zhang & Shasha
- Distance d'édition $\neq$ alignement
- Références

Structure secondaire d'ARN



- Dans le domaine de la comparaison de structure secondaire d'ARN, on modélise les structures secondaires par des arbres ordonnés étiquetés

- étiqueté = une étiquette est associée à chaque nœud

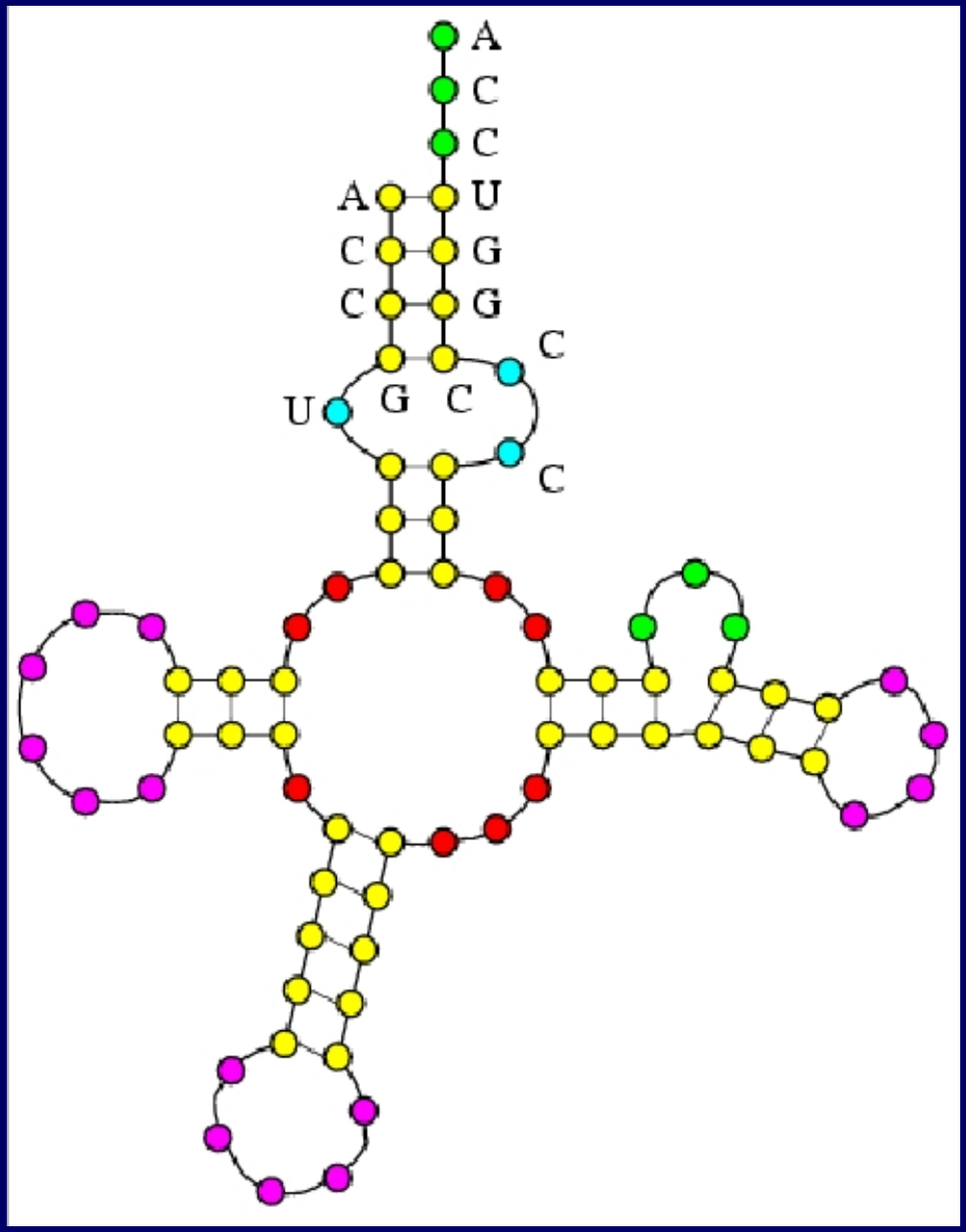
- ordonné = les arêtes incidentes à un nœud donné sont ordonnées

- Plusieurs niveaux de décomposition possible :

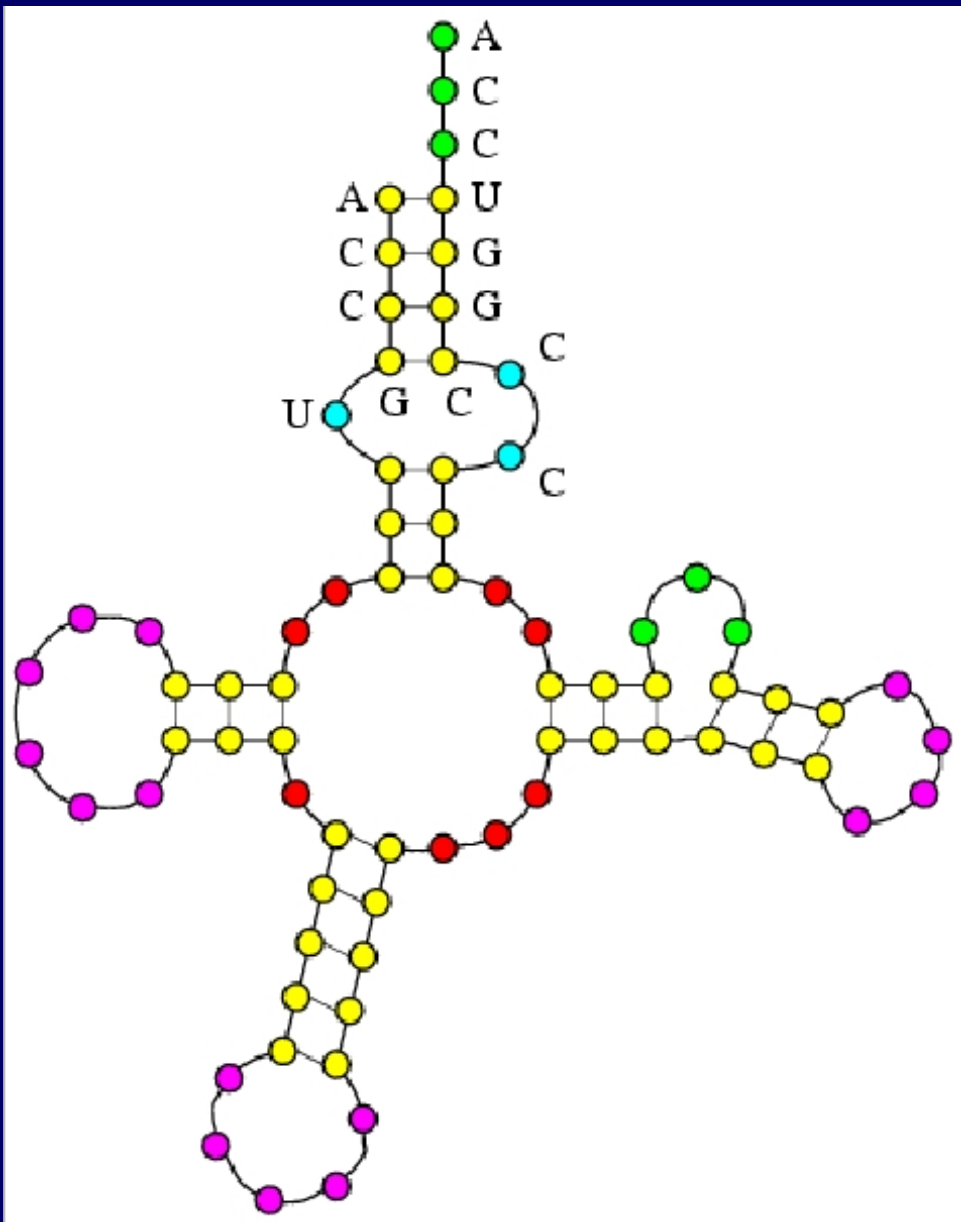
  → Grossier : nœuds = éléments structuraux (boucle, tige, renflement, . . .)

  → Fin : nœuds = acide nucléique ou paires (Watson-Crick, Wobble, . . .)

Exemple de représentation en arbre (1/3)



Exemple de représentation en arbre (2/3)



M Multi–branch loop
= Boucle multiple

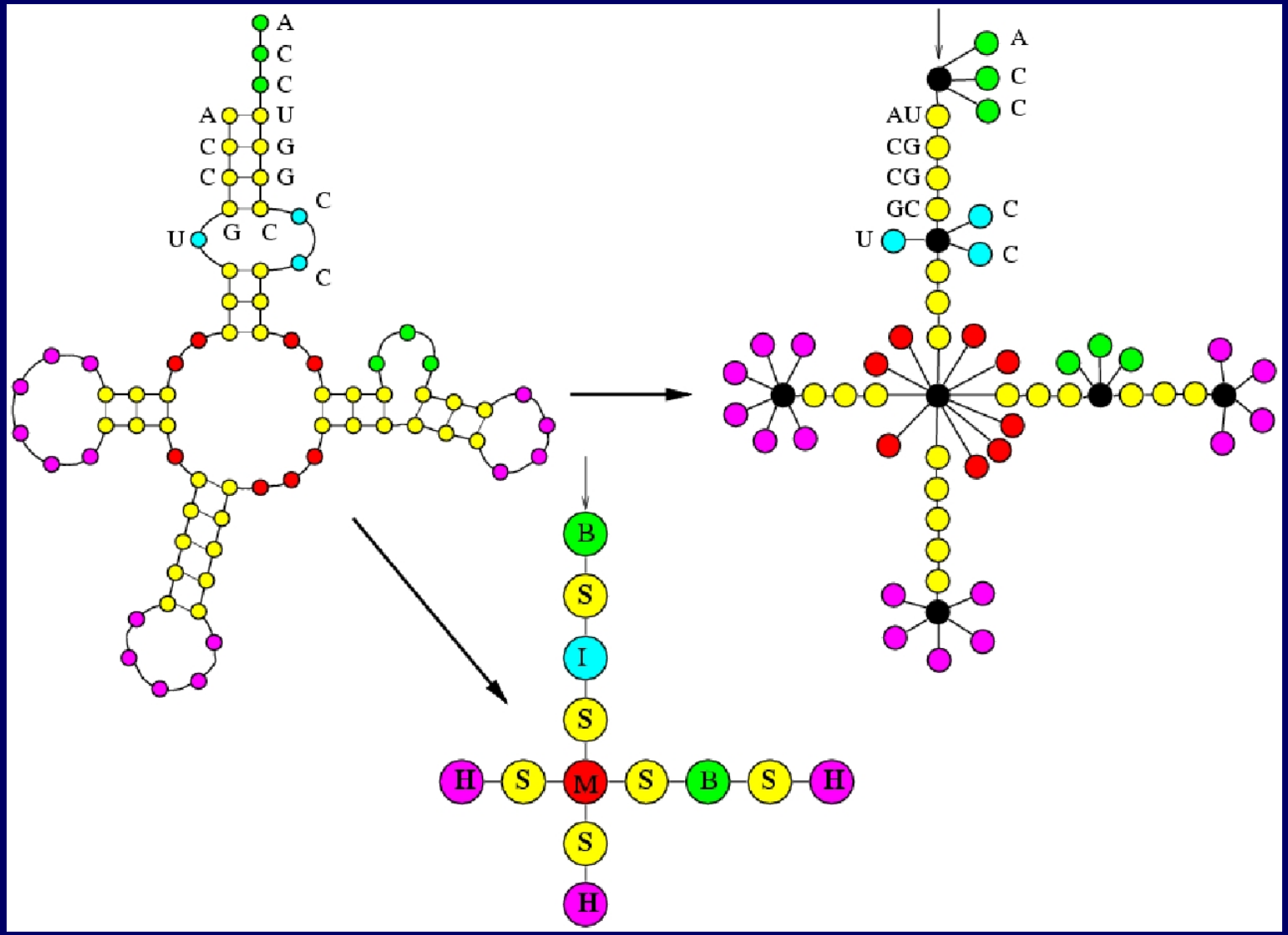
I Inside loop
= Boucle interne

H Hairpin = Boucle

S Stem = Tige

B Bulge = Renflement

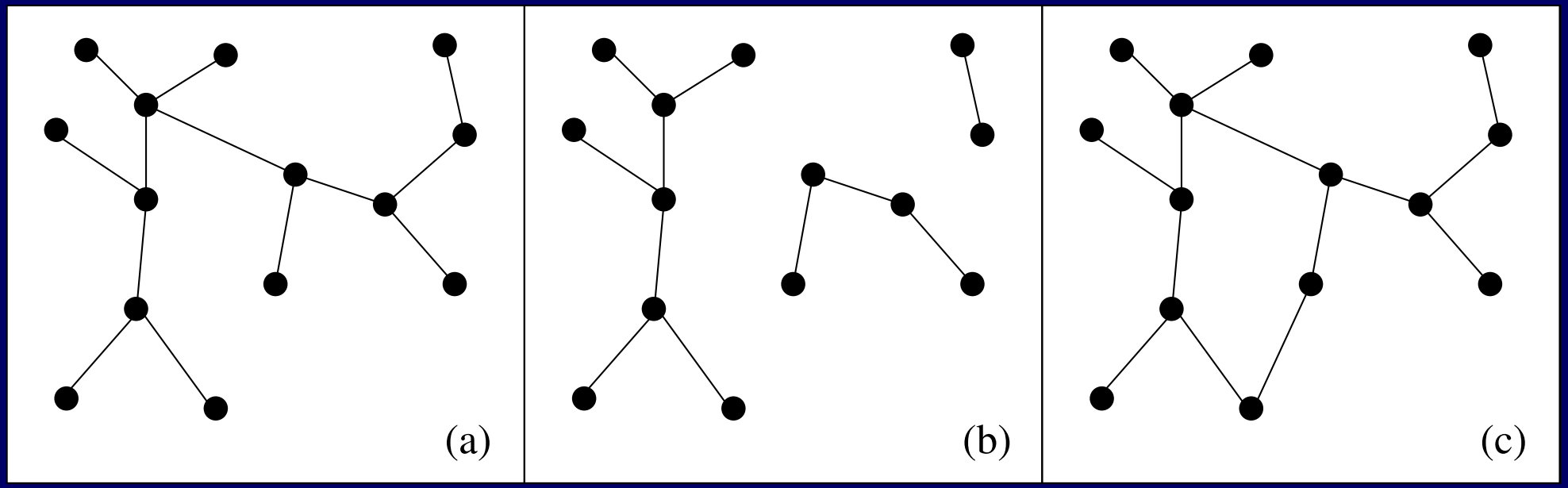
Exemple de représentation en arbre (3/3)



Définitions (1/5)

- Un arbre est un graphe non orienté, connexe et acyclique

- Si un graphe non orienté est acyclique mais pas forcément connexe, on dit que c'est une forêt



Définitions (2/5)

- Soit $G = (S, A)$ un graphe non orienté. Les affirmations suivantes sont équivalentes :

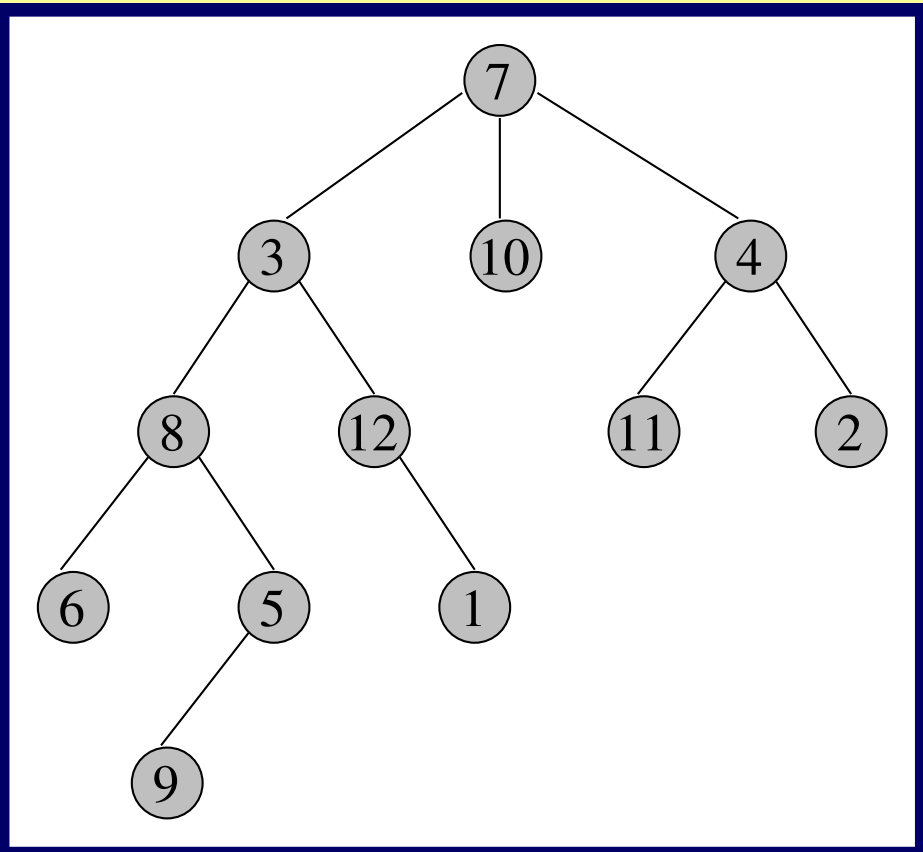
  1. G est un arbre
  2. Deux sommets quelconques de G sont reliés par une chaîne élémentaire* unique
  3. G est connexe mais si on enlève un sommet quelconque à S, le graphe résultant n'est plus connexe
  4. G est connexe et $|A| = |S| - 1$
  5. G est acyclique et $|A| = |S| - 1$
  6. G est acyclique mais si une arête quelconque est ajouté à A, le graphe résultant contient un cycle

* chaîne élémentaire = tous ses sommets sont distincts 2 à 2.

Définitions (3/5)

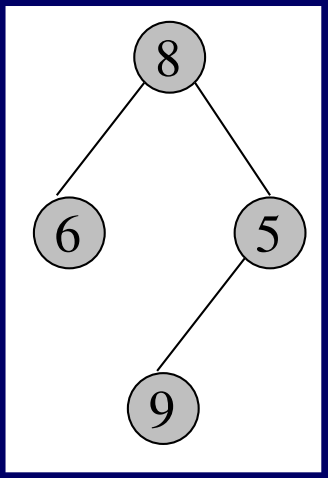
- Une arborescence est un arbre dans lequel l'un des sommets se distingue des autres
- Ce sommet particulier s'appelle racine de l'arborescence
- Notion d'ancêtre, de descendant, parent, enfant, nœud externe ou feuille, nœud interne
- La sous-arborescence de racine x est l'arborescence composée des descendants de x et ayant pour racine x
- Une arborescence ordonnée est une arborescence dans laquelle les enfants de chaque nœud sont ordonnés
- Cette notion d'ordre existe aussi sur les forêts : une forêt ordonnée est une suite ordonnée d'arborescences ordonnées

Définitions (4/5)

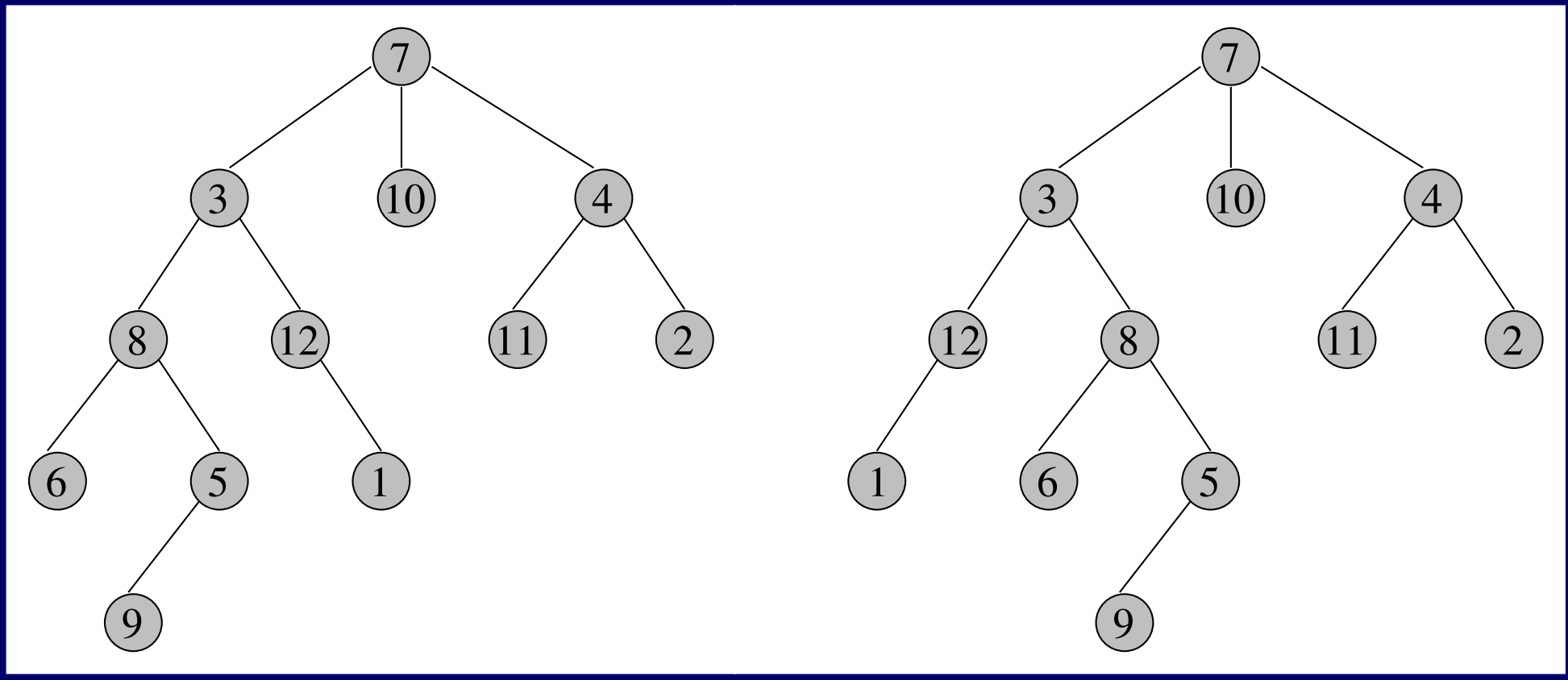


- Arborescence étiquetée de 12 sommets avec comme racine 7
- 3 est un ancêtre de 9, 1 est un descendant de 3,
- 3 est le parent de 8, 8 est un enfant de 3
- 8 et 12 sont frères
- 7 est le seul nœud sans parent

- 6, 9, 1, 11 et 2 sont les feuilles de l'arbre ci-dessus, tous les autres nœuds sont des nœuds internes
- La sous-arborescence de racine 8 est représentée ci-contre



Définitions (5/5)



- Ces deux arborescences sont différentes si on les regarde comme des arborescences ordonnées mais identiques si on les regarde comme des arborescences

Parcours d'arbres

- On explore un arbre en parcourant tous ses nœuds, éventuellement on applique un traitement sur chaque nœud (par ex. `imprimer étiquette`)
- On explore l'arbre en profondeur : on part de la racine, on descend par l'arête la plus à gauche, on marque au fur et à mesure les sommets explorés, on remonte qd on a atteint une feuille ou que tous les fils d'un sommet ont déjà été explorés. On répète le processus jusqu'à avoir exploré tous les sommets de l'arborescence
- L'ordre dans lequel on applique le traitement sur chaque nœud définit plusieurs parcours :

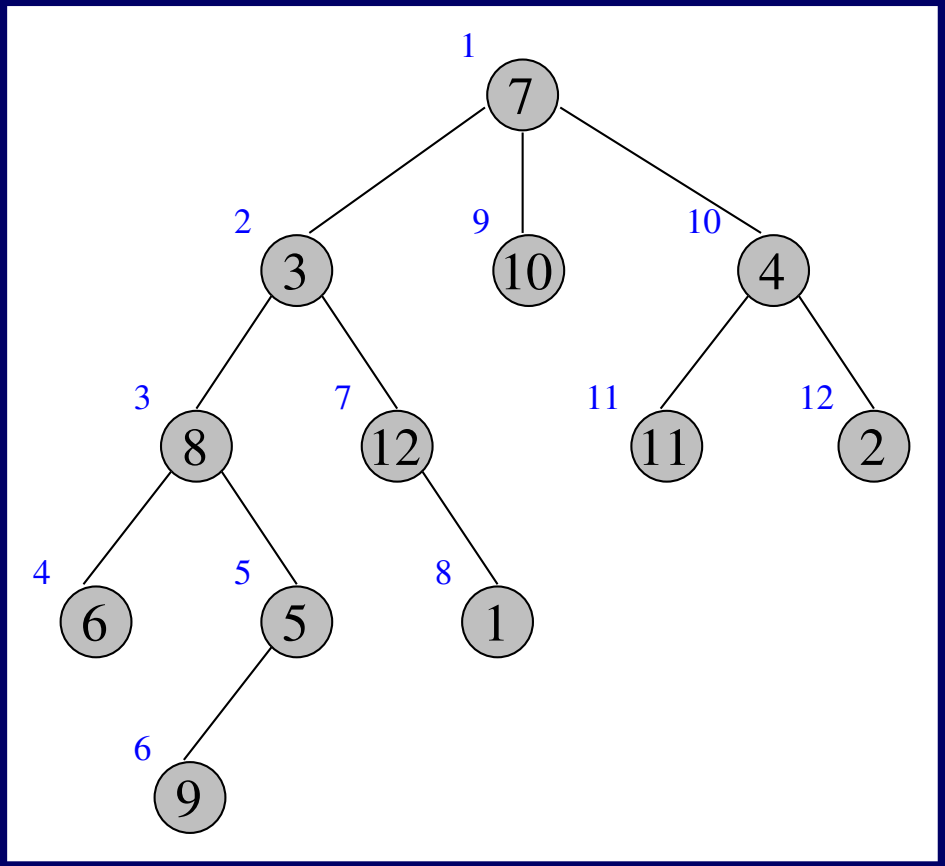
  → préfixe : traite la racine d'un sous-arbre avant les nœuds de ce sous-arbre

  → postfixe (aussi nommé suffixe) : traite la racine d'un sous-arbre après les nœuds de ce sous-arbre

  → infixe (pour les arbres binaires seulement)

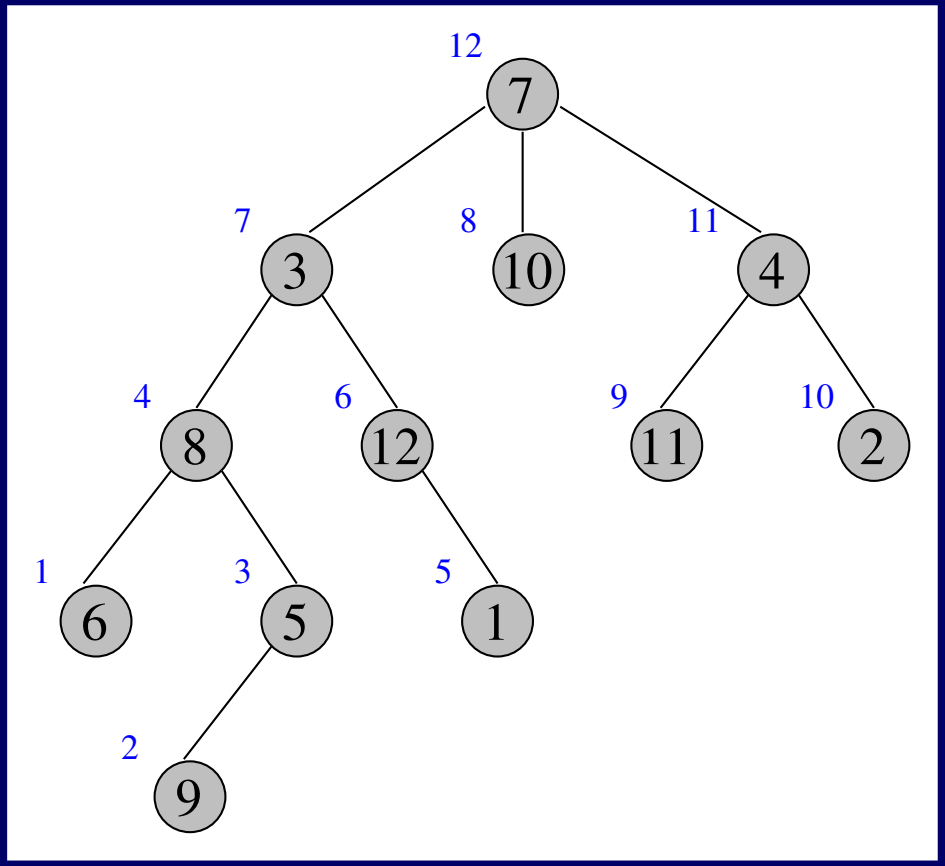
- Chacun de ses parcours définit une numérotation des sommets

Numérotation préfixe



- Les sommets sont numérotés dans l'ordre dans lequel on les découvre selon un parcours en profondeur

Numérotation suffixe



- Les sommets sont numérotés lorsque c'est la dernière fois qu'on les voit dans le parcours
- Un sommet est donc traité lorsque tous ses descendants sont traités

Décomposition "branche-gauche"

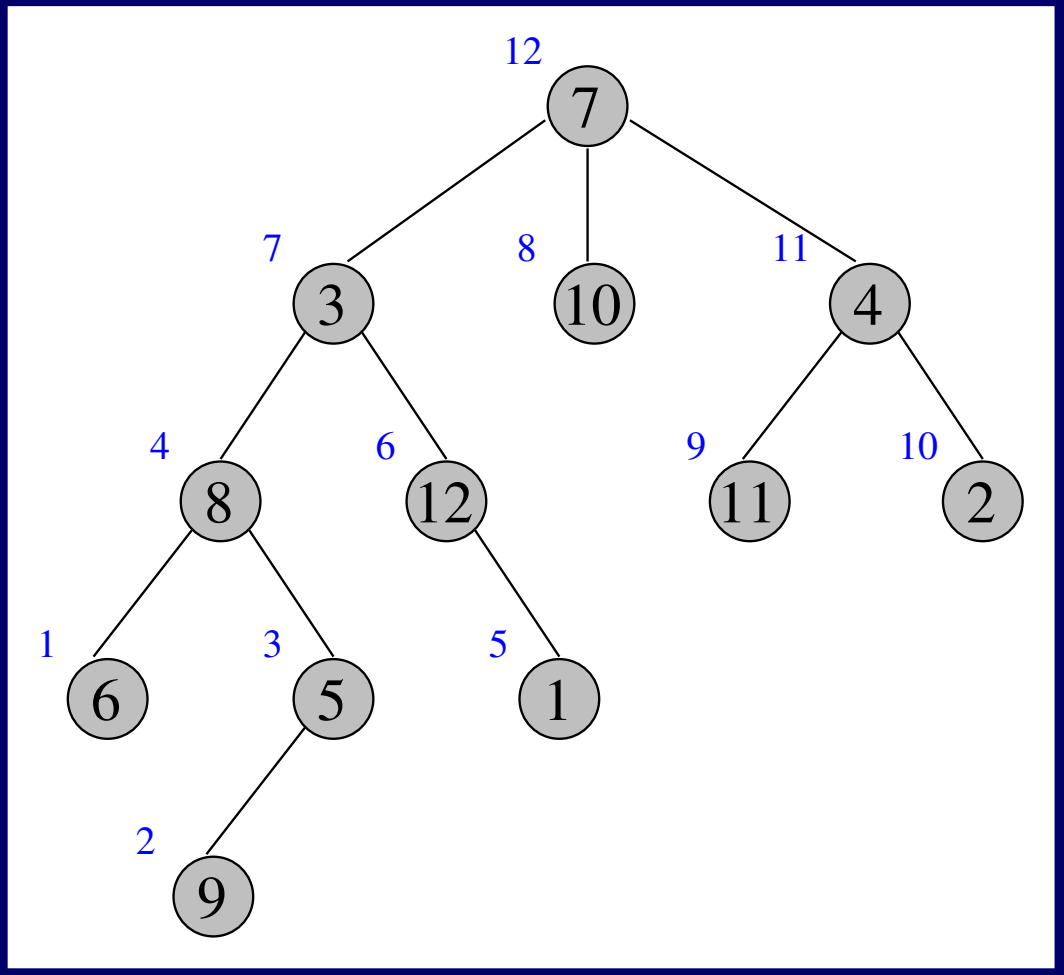
- Soit T une arborescence ordonnée étiquetée. Une décomposition "branche-gauche" de T est un ensemble de sous-arborescences de T dont les racines sont obtenues de la manière suivante :

  $\rightarrow$ Supprimer la 1^{re} feuille de T ainsi que tous ses ancêtres : le dernier sommet à être supprimé (la racine de T dans ce cas là) est la racine de la 1^{re} sous-arborescence de la décomposition

  $\rightarrow$ Répéter ce processus sur la forêt résultante, en commençant par sa première feuille, jusqu'à ce que la forêt soit ramenée à la forêt vide

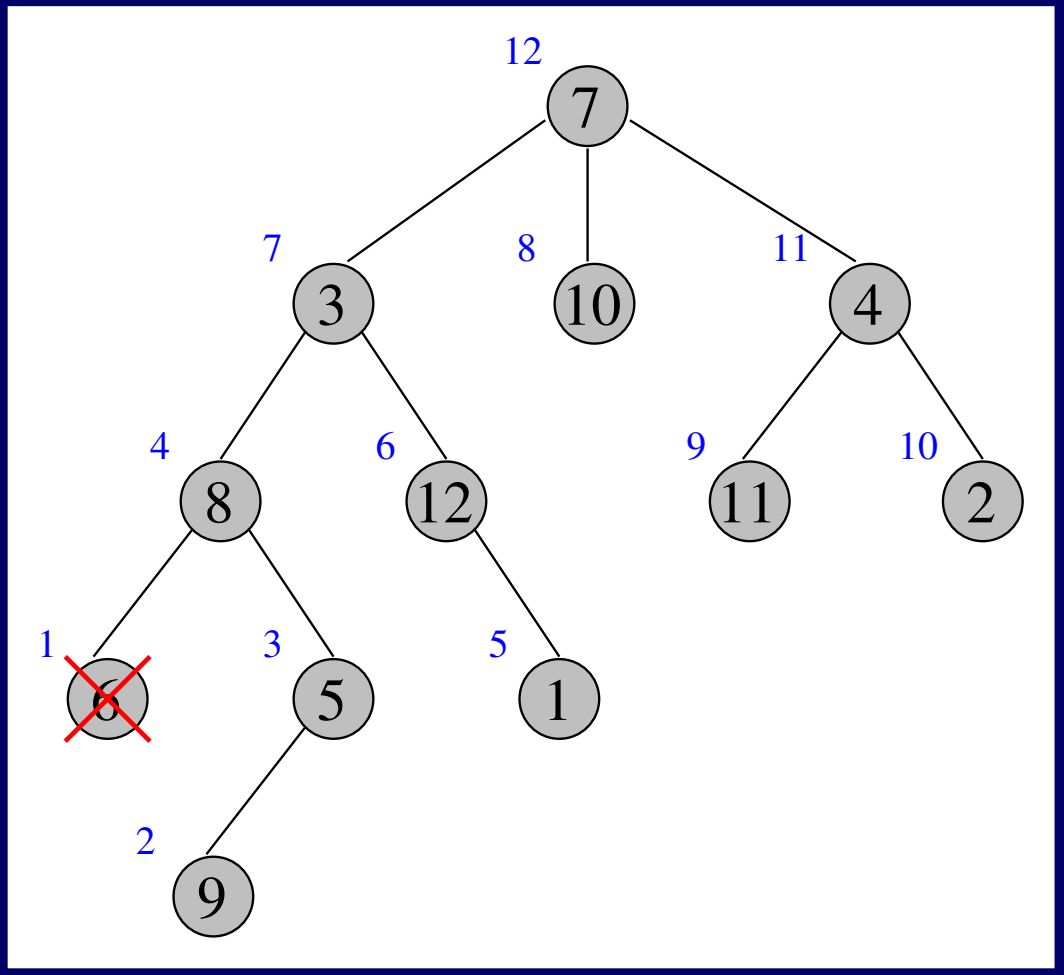
- Le nb de sous-arborescences obtenues est clairement égal au nb de feuilles de T

Exemple de décomposition branche-gauche



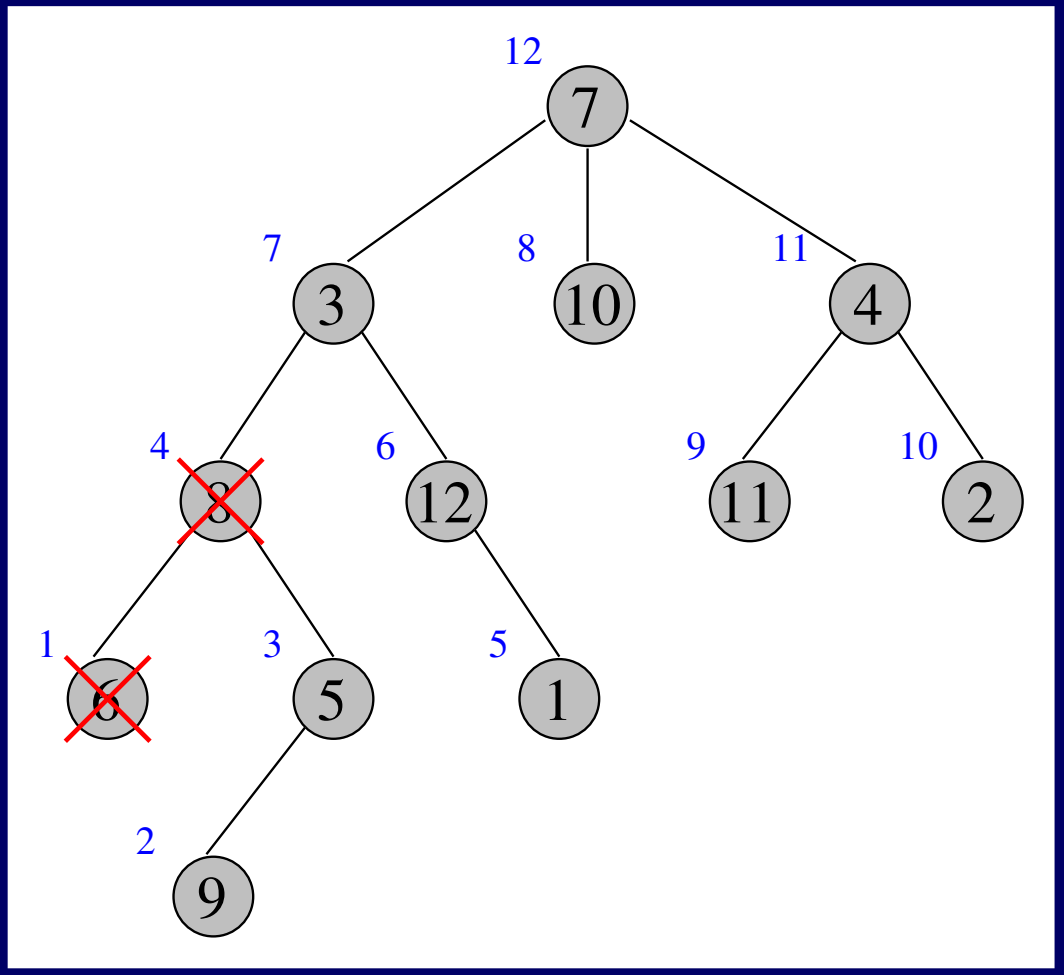
Racines des sous–arborescences de la décomposition :

Exemple de décomposition branche-gauche



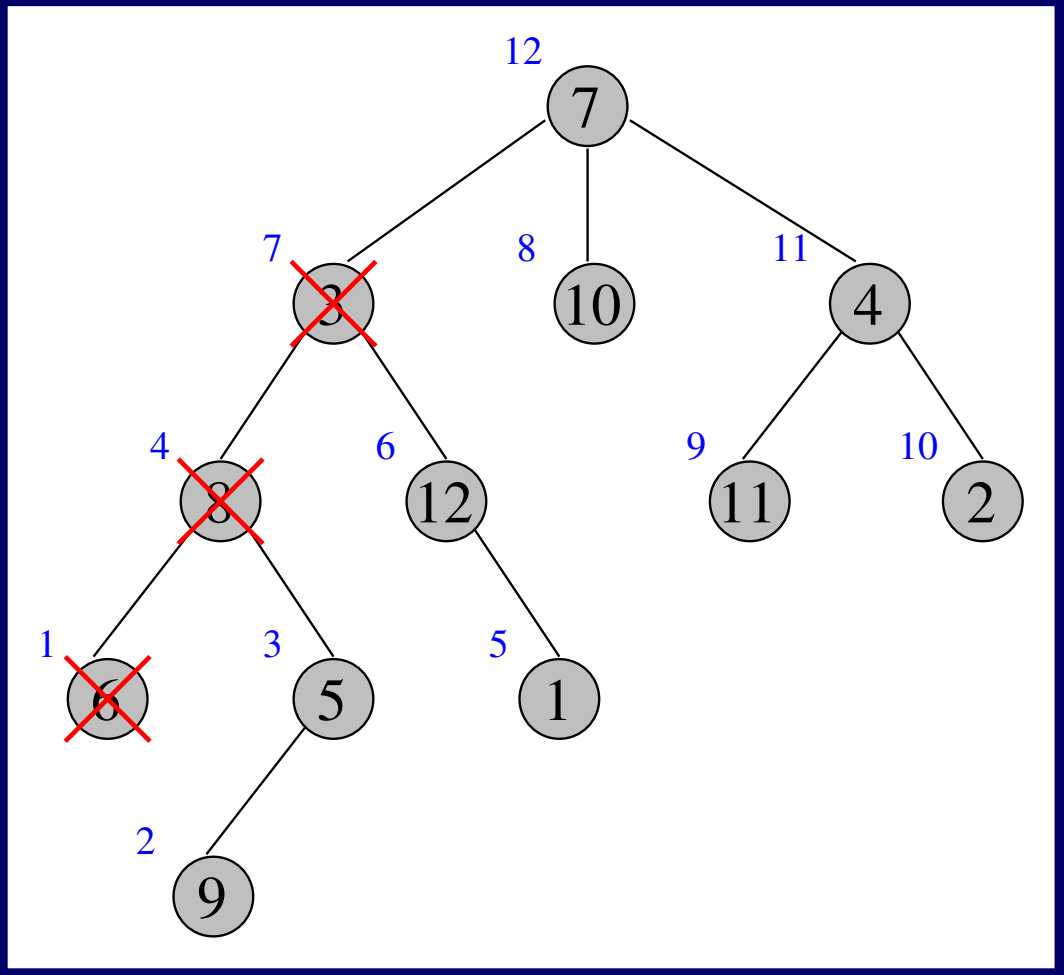
Racines des sous–arborescences de la décomposition :

Exemple de décomposition branche-gauche



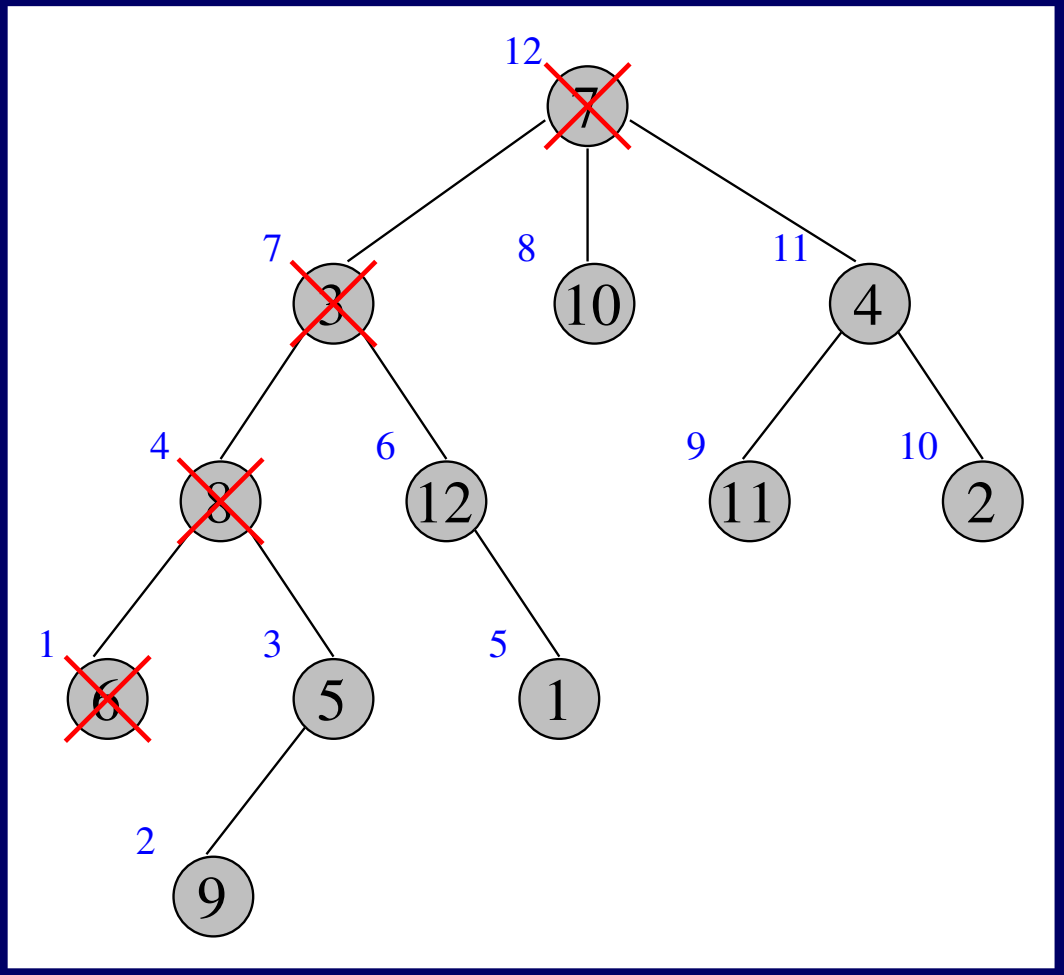
Racines des sous–arborescences de la décomposition :

Exemple de décomposition branche-gauche



Racines des sous–arborescences de la décomposition :

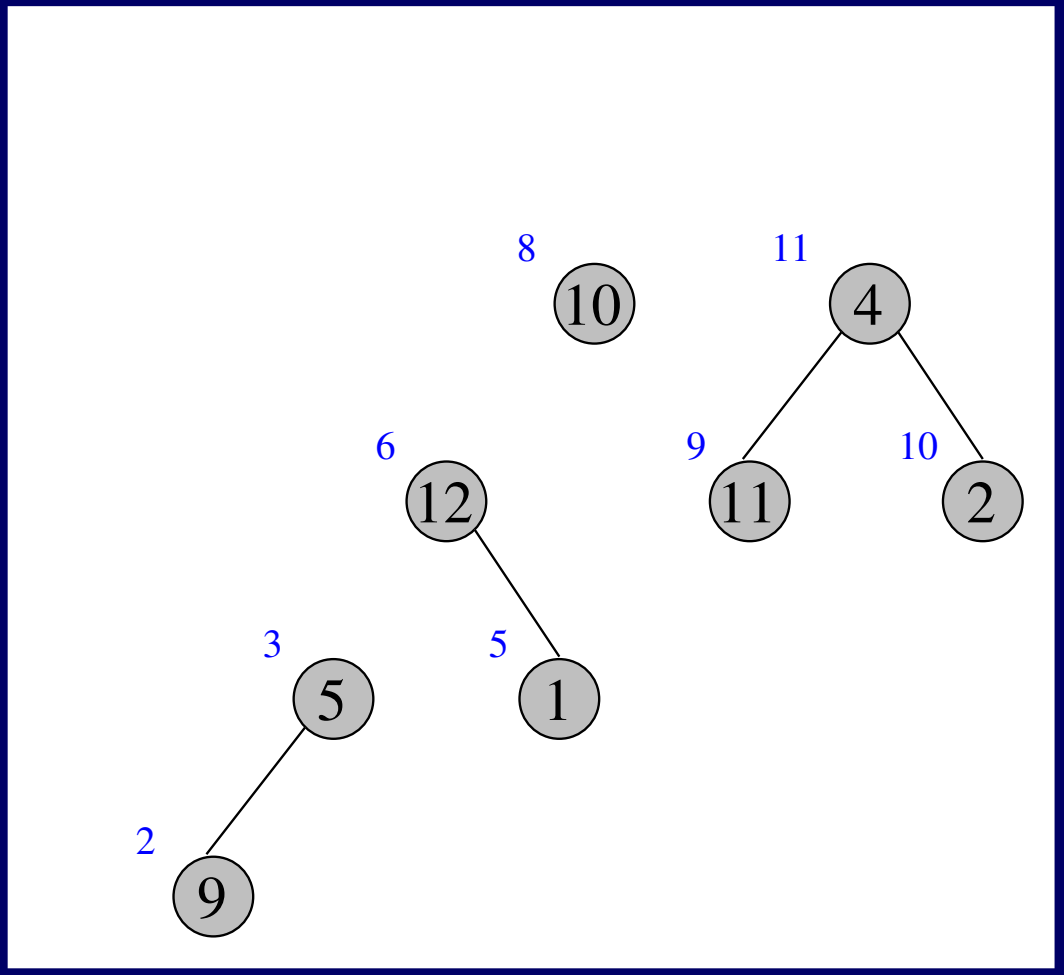
Exemple de décomposition branche-gauche



Racines des sous–arborescences de la décomposition :

7

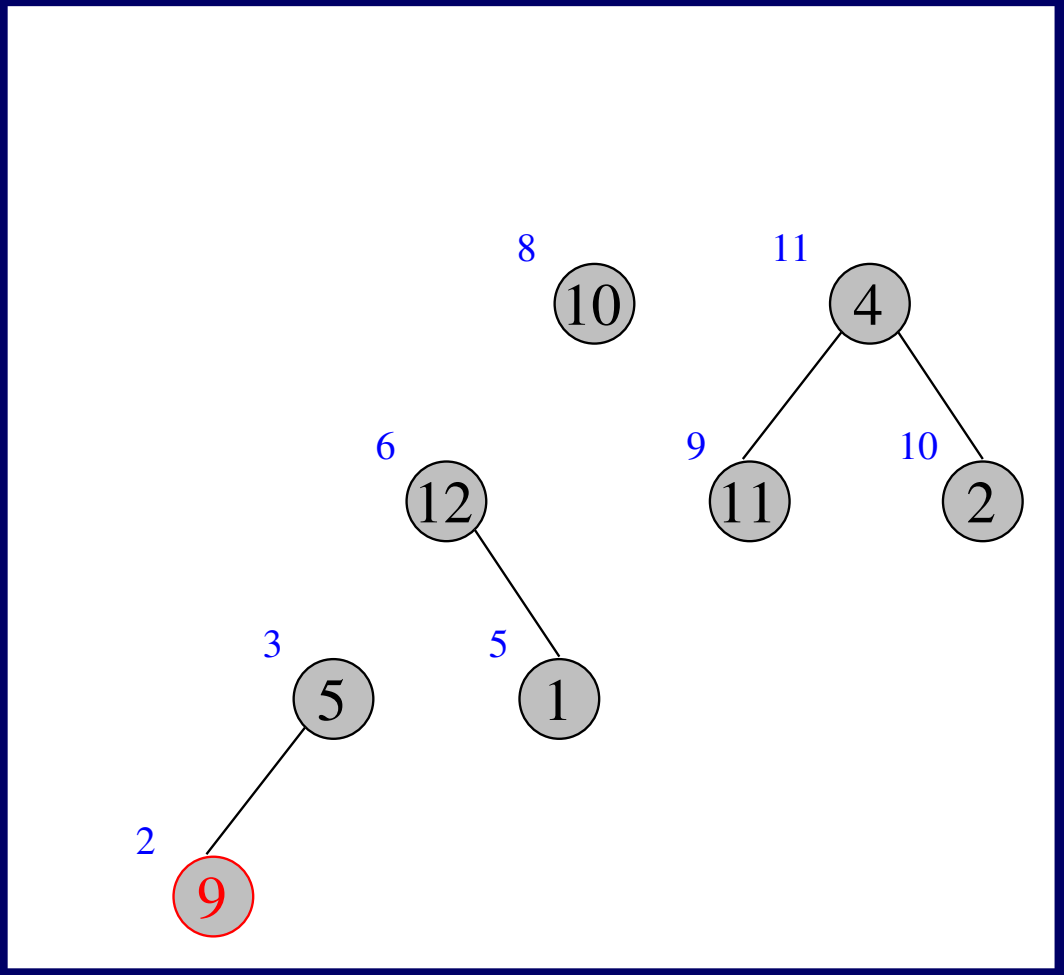
Exemple de décomposition branche-gauche



Racines des sous–arborescences de la décomposition :

7

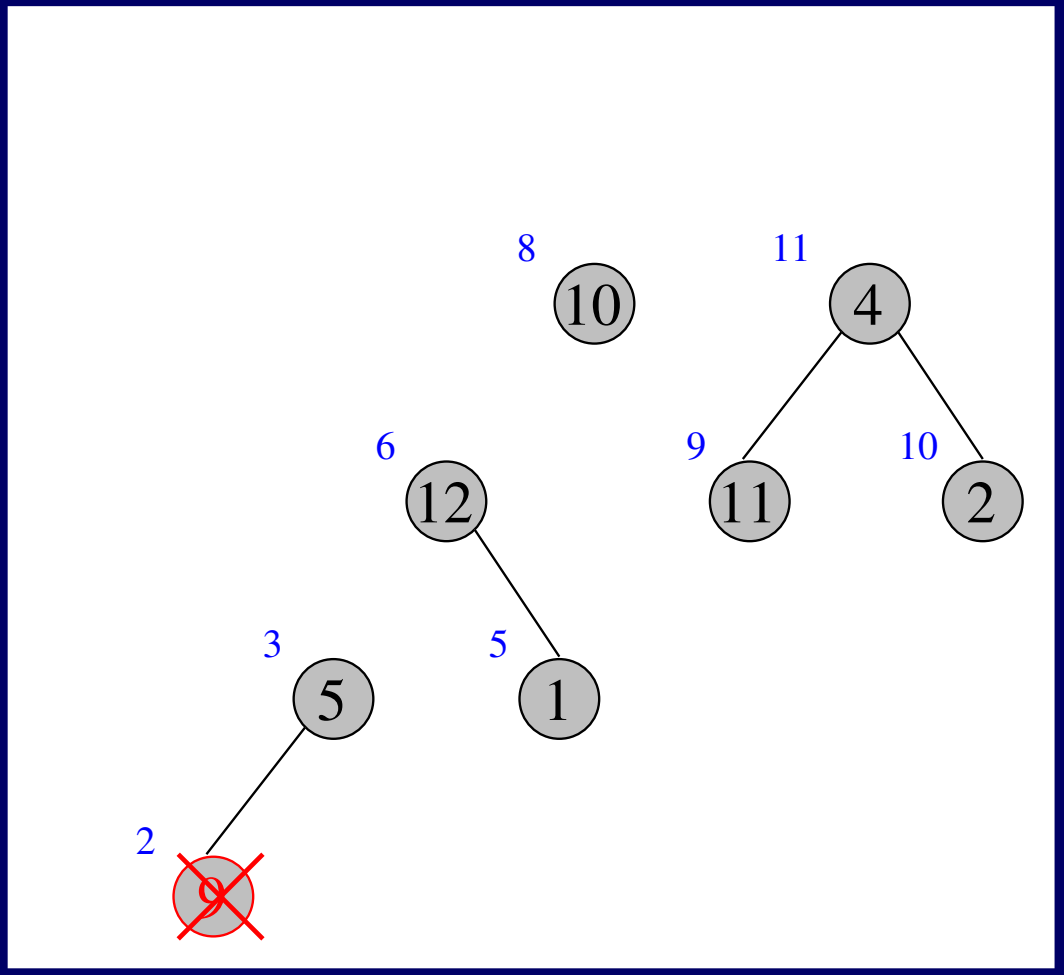
Exemple de décomposition branche-gauche



Racines des sous–arborescences de la décomposition :

7

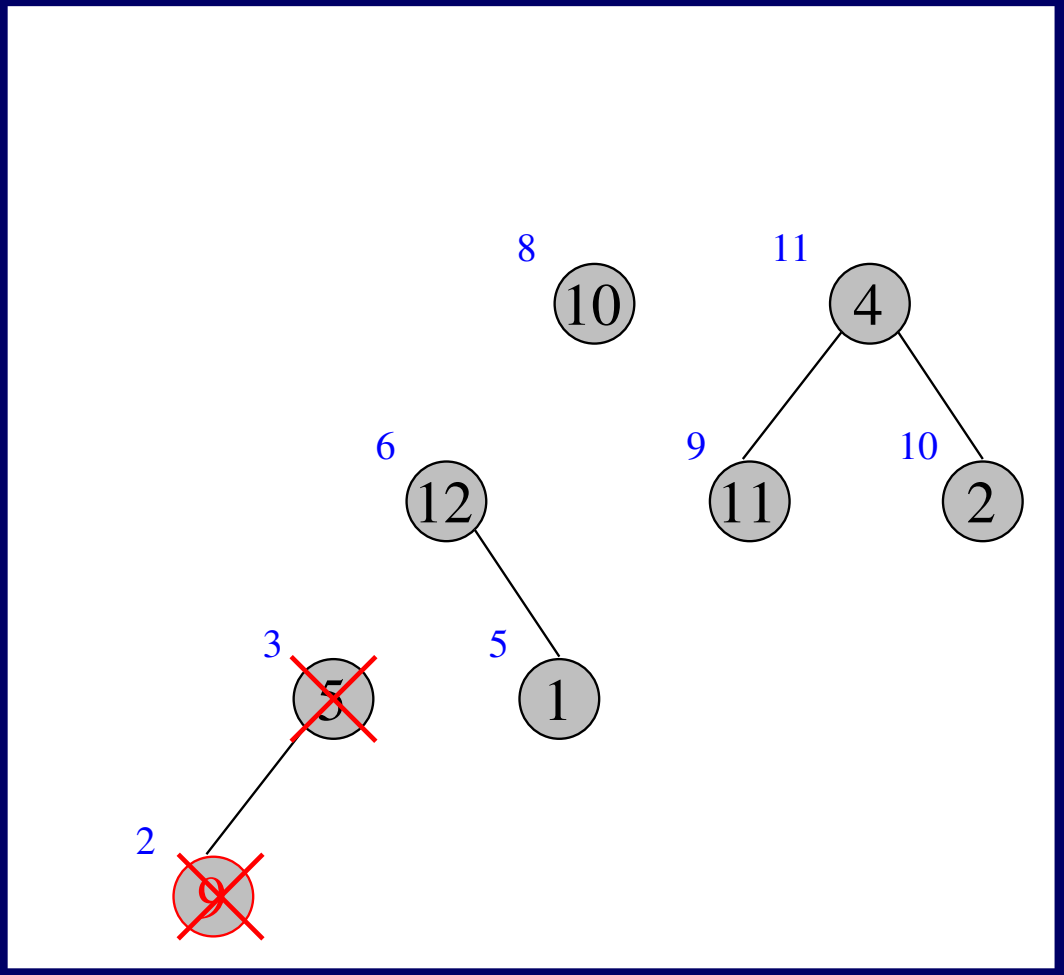
Exemple de décomposition branche-gauche



Racines des sous–arborescences de la décomposition :

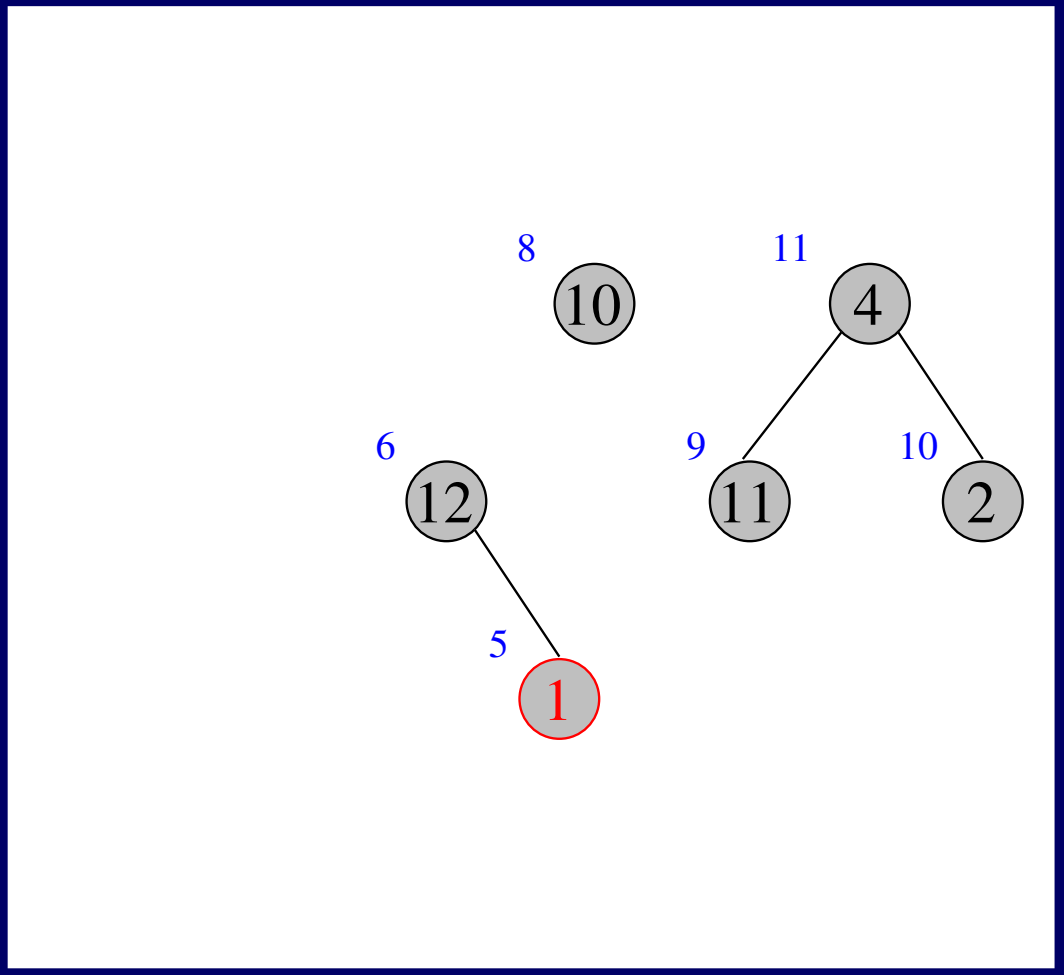
7

Exemple de décomposition branche-gauche



Racines des sous–arborescences de la décomposition :

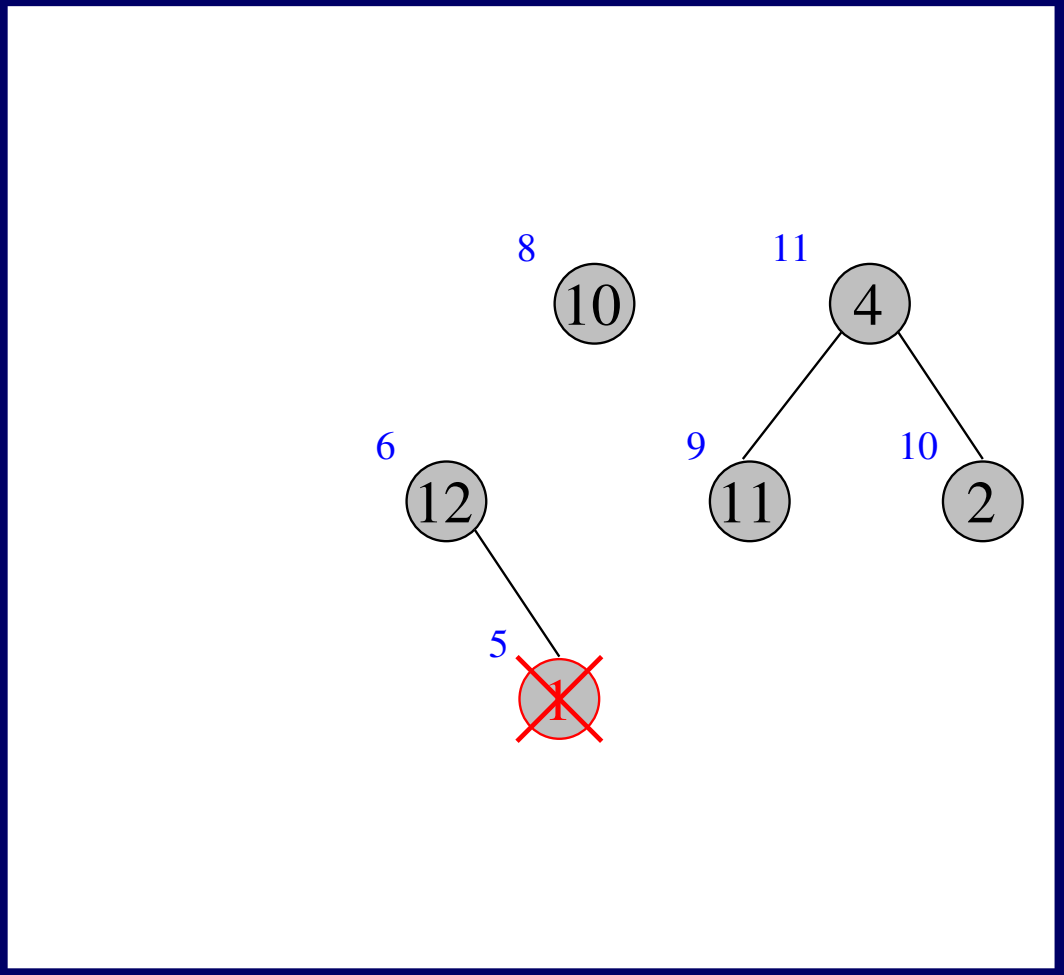
Exemple de décomposition branche-gauche



Racines des sous–arborescences de la décomposition :

7 5

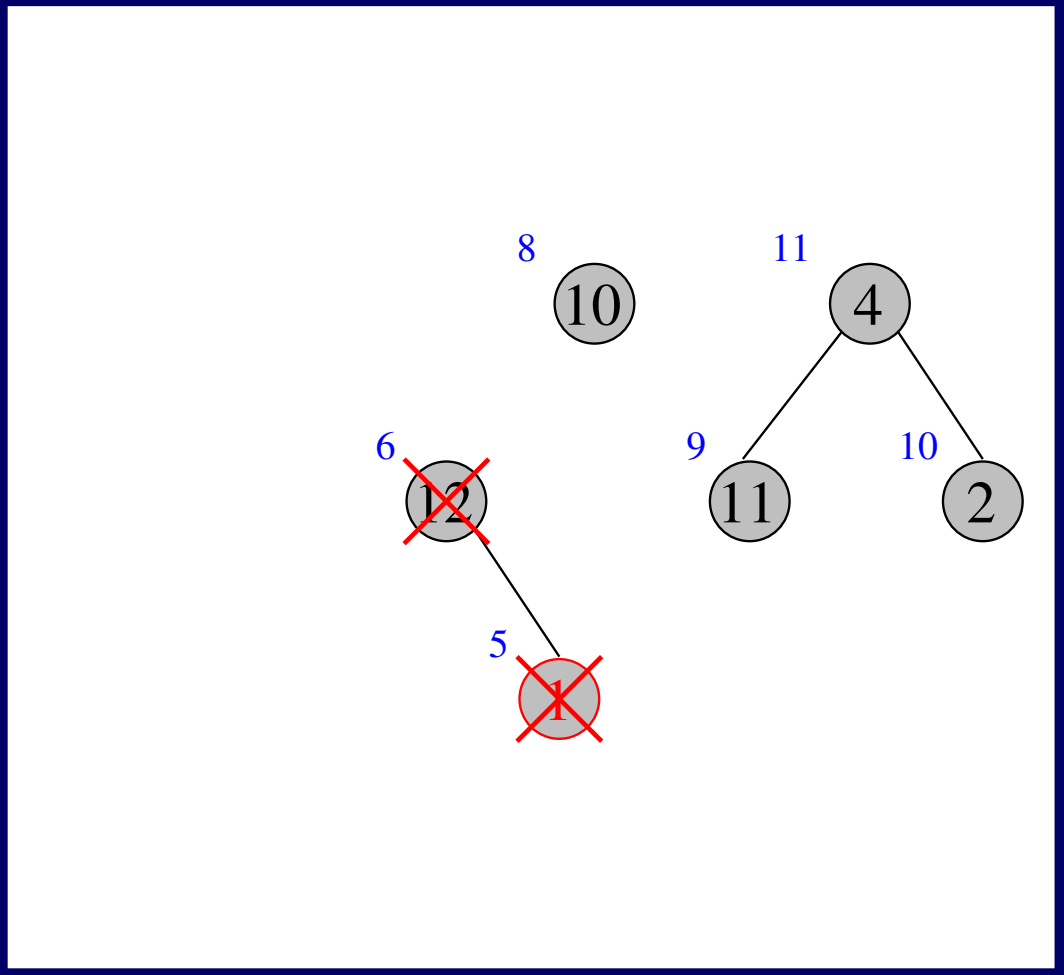
Exemple de décomposition branche-gauche



Racines des sous–arborescences de la décomposition :

7 5

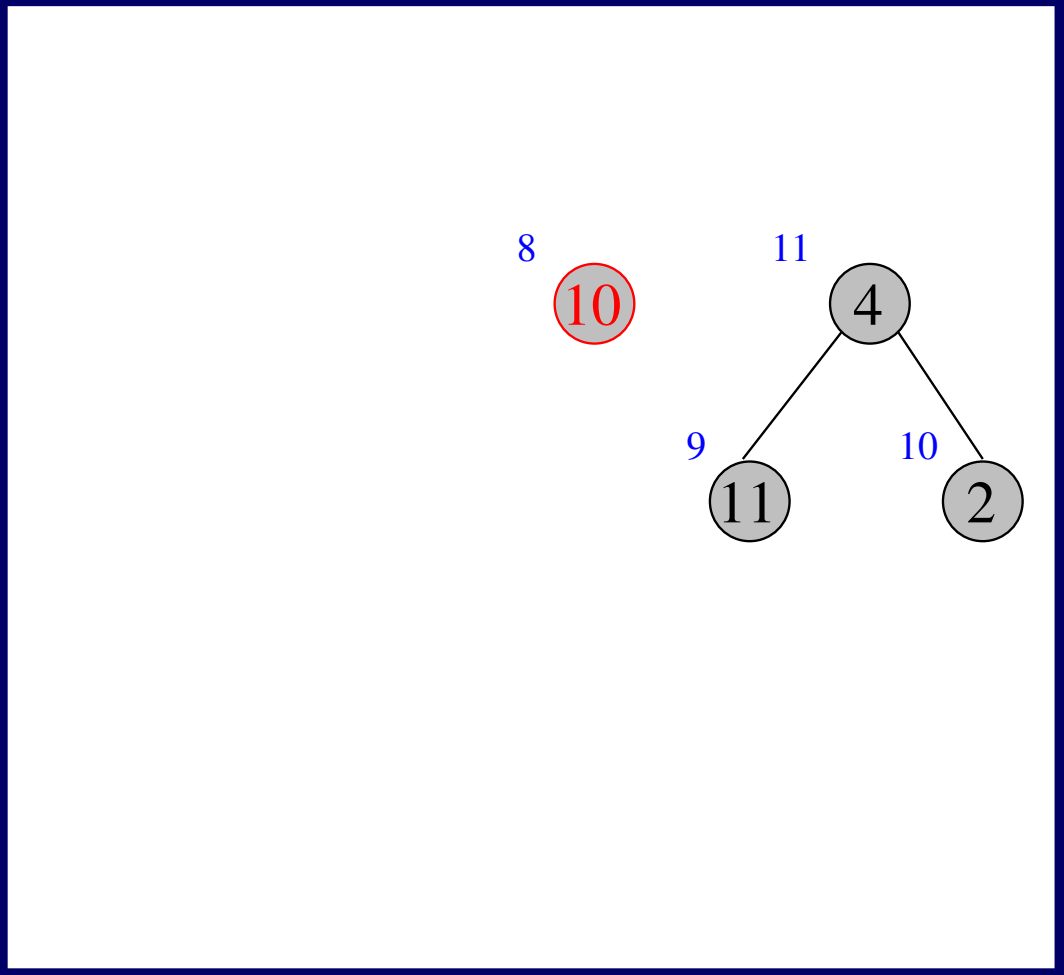
Exemple de décomposition branche-gauche



Racines des sous–arborescences de la décomposition :

7 5 12

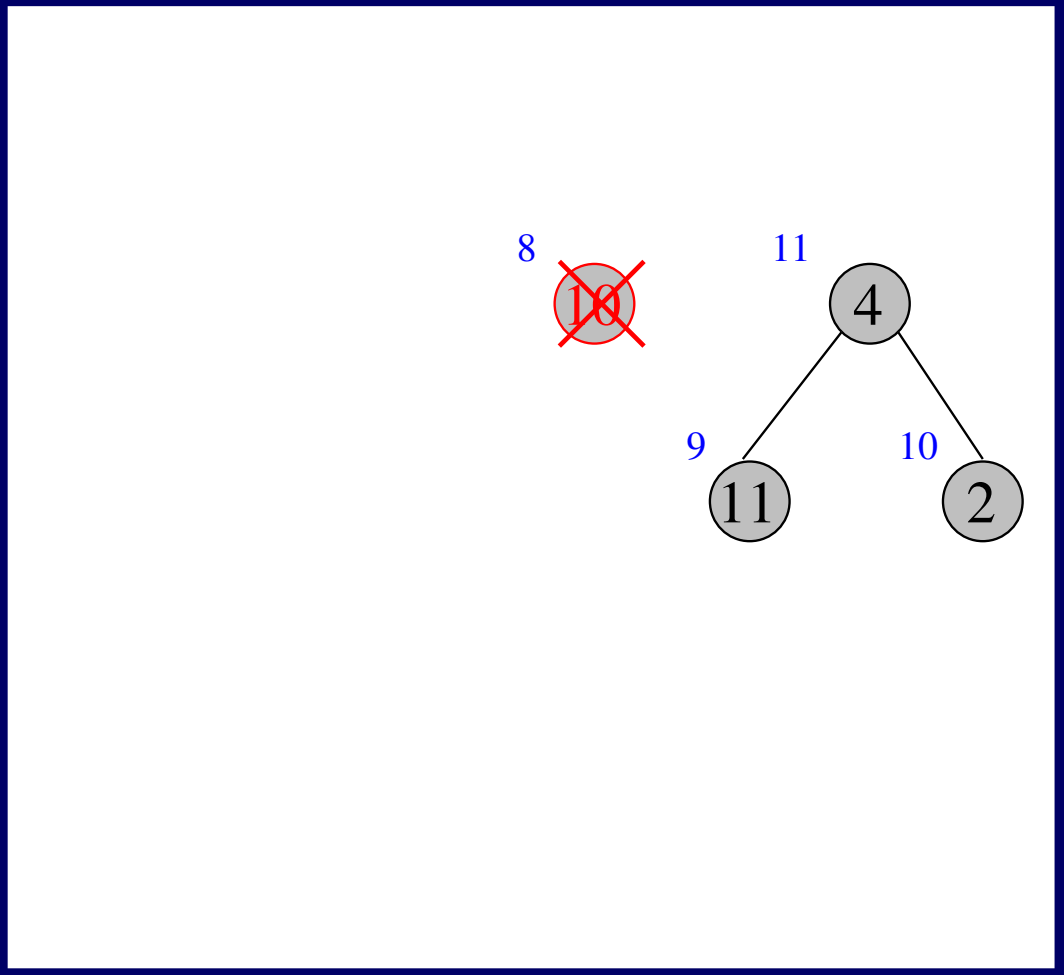
Exemple de décomposition branche-gauche



Racines des sous–arborescences de la décomposition :

7 5 12

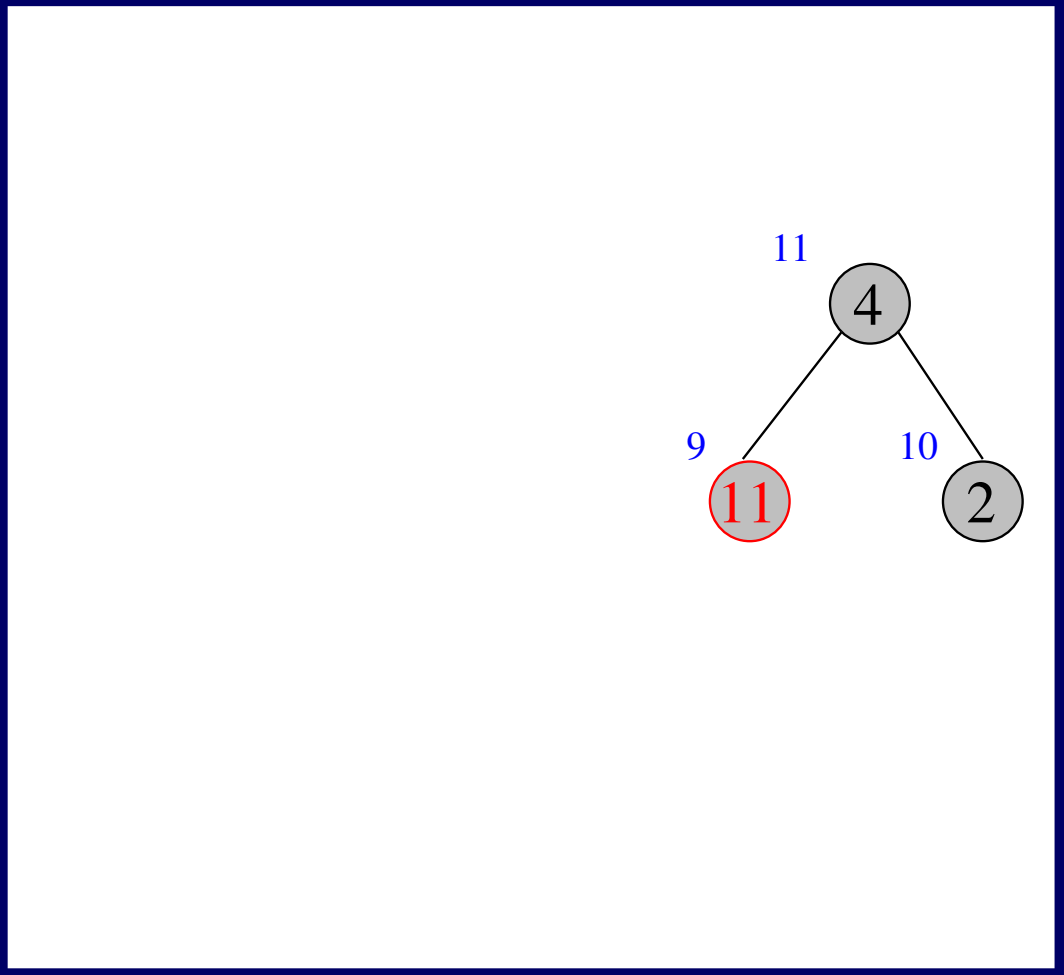
Exemple de décomposition branche-gauche



Racines des sous–arborescences de la décomposition :

7 5 12 10

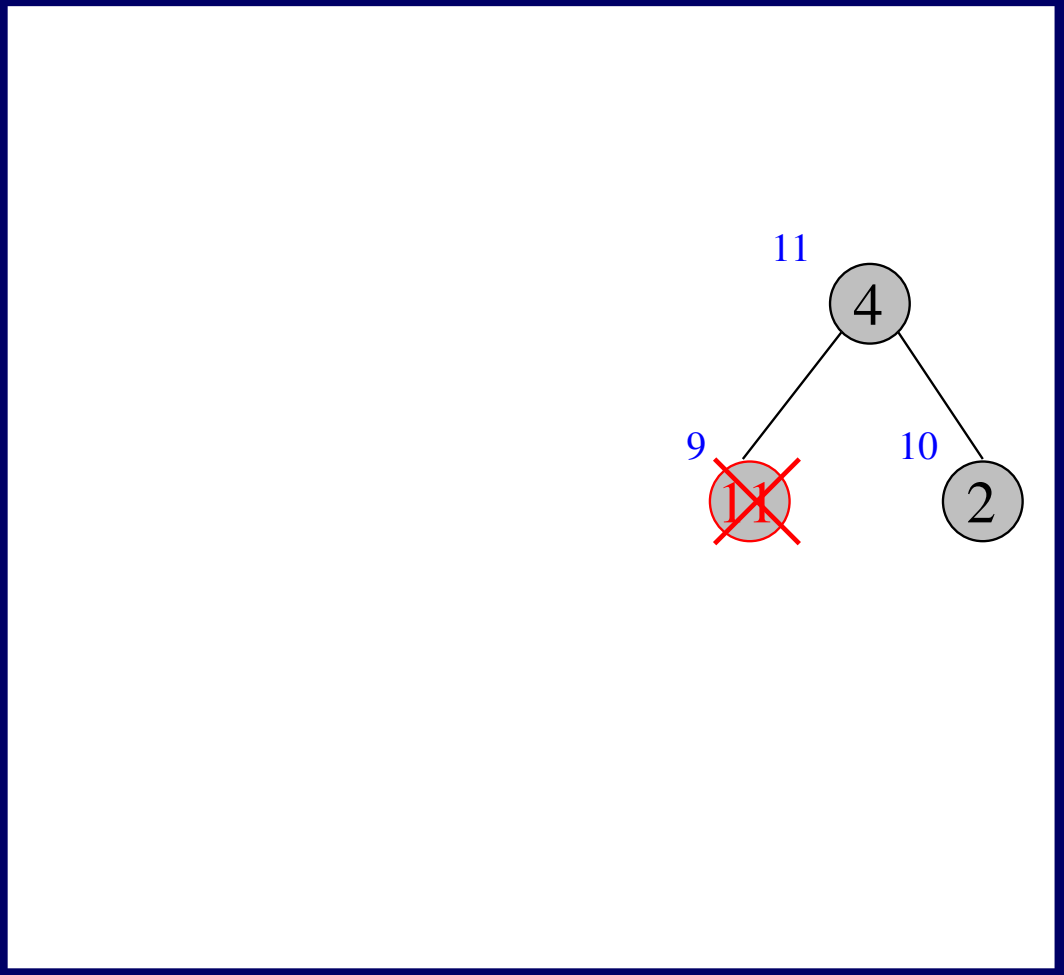
Exemple de décomposition branche-gauche



Racines des sous–arborescences de la décomposition :

7 5 12 10

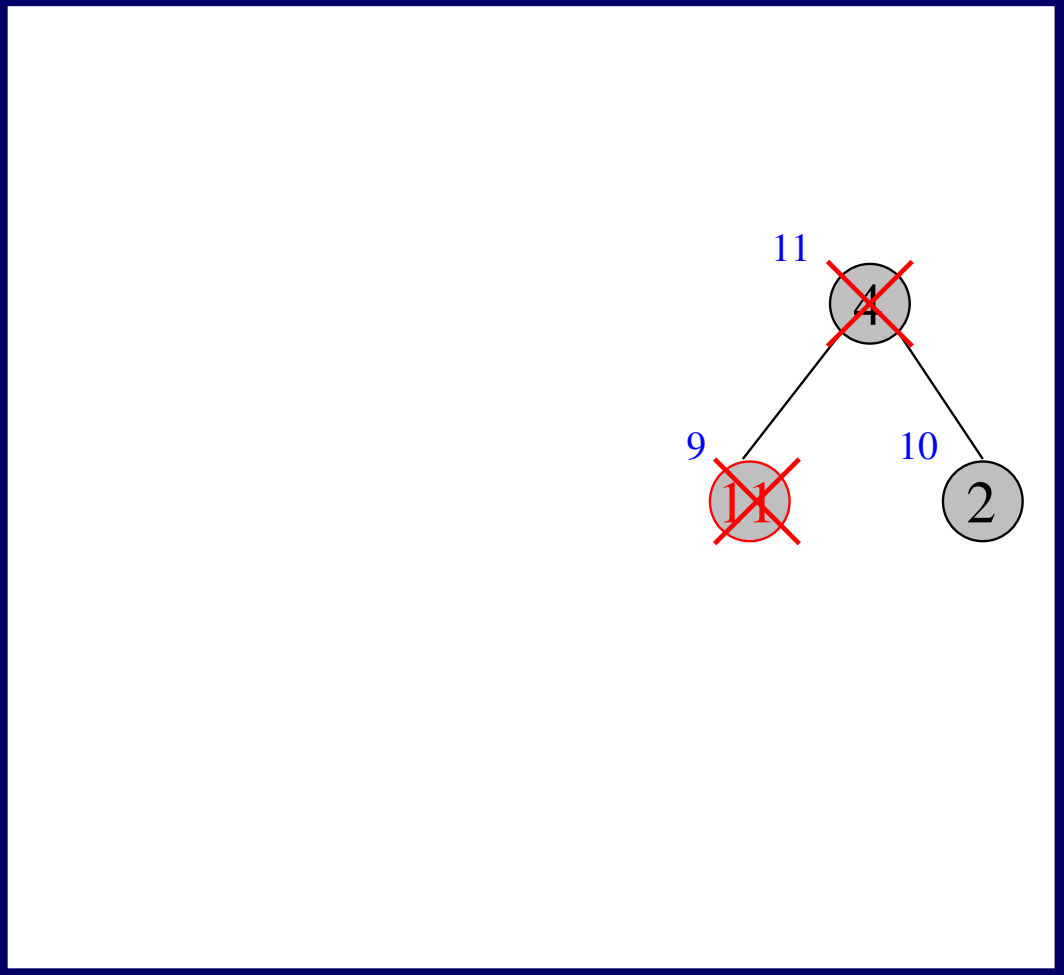
Exemple de décomposition branche-gauche



Racines des sous–arborescences de la décomposition :

7 5 12 10

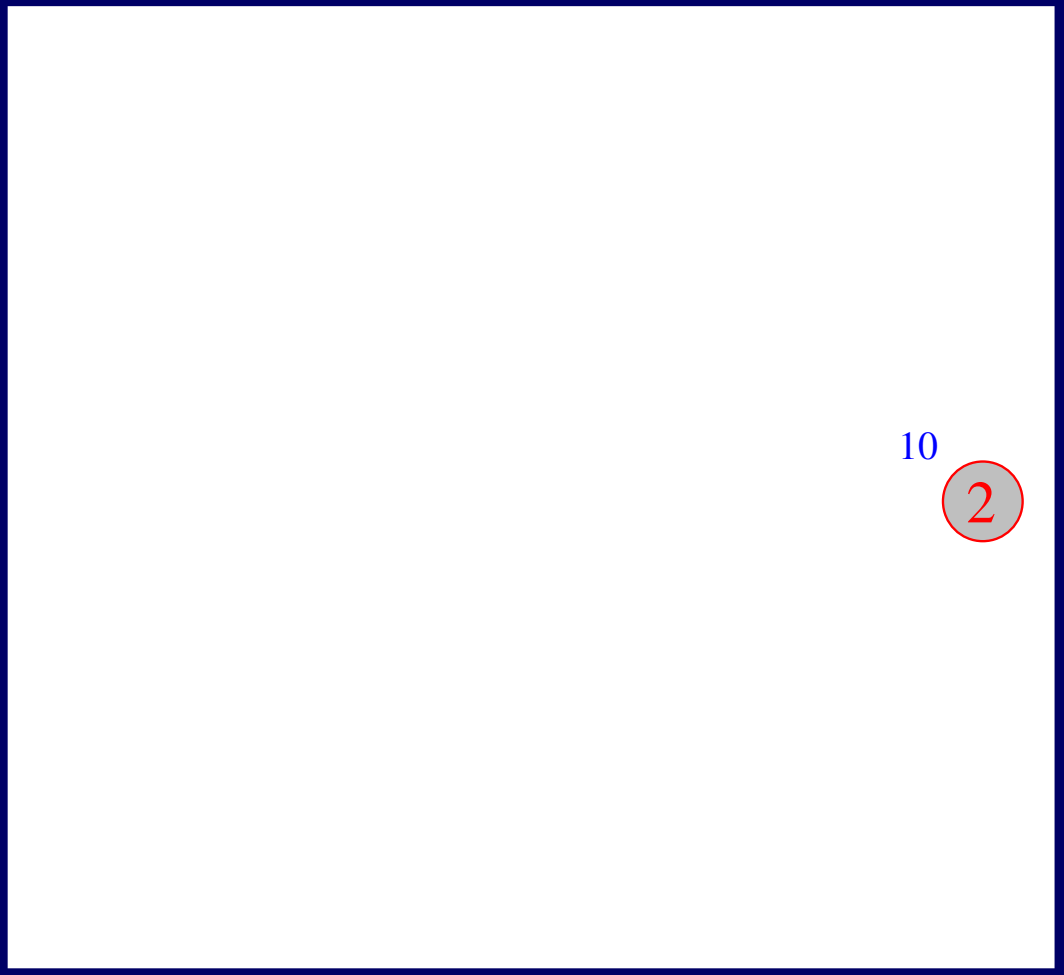
Exemple de décomposition branche-gauche



Racines des sous–arborescences de la décomposition :

7 5 12 10 4

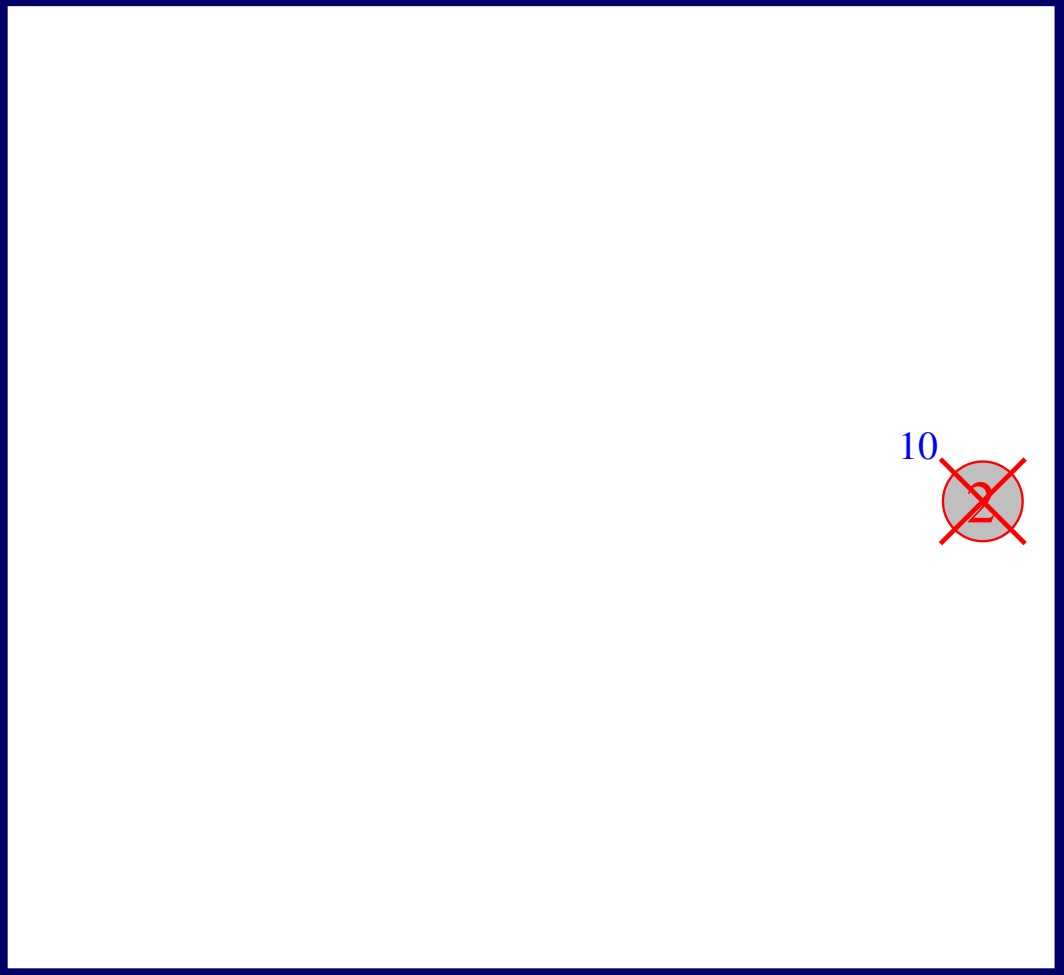
Exemple de décomposition branche-gauche



Racines des sous–arborescences de la décomposition :

7 5 12 10 4

Exemple de décomposition branche-gauche



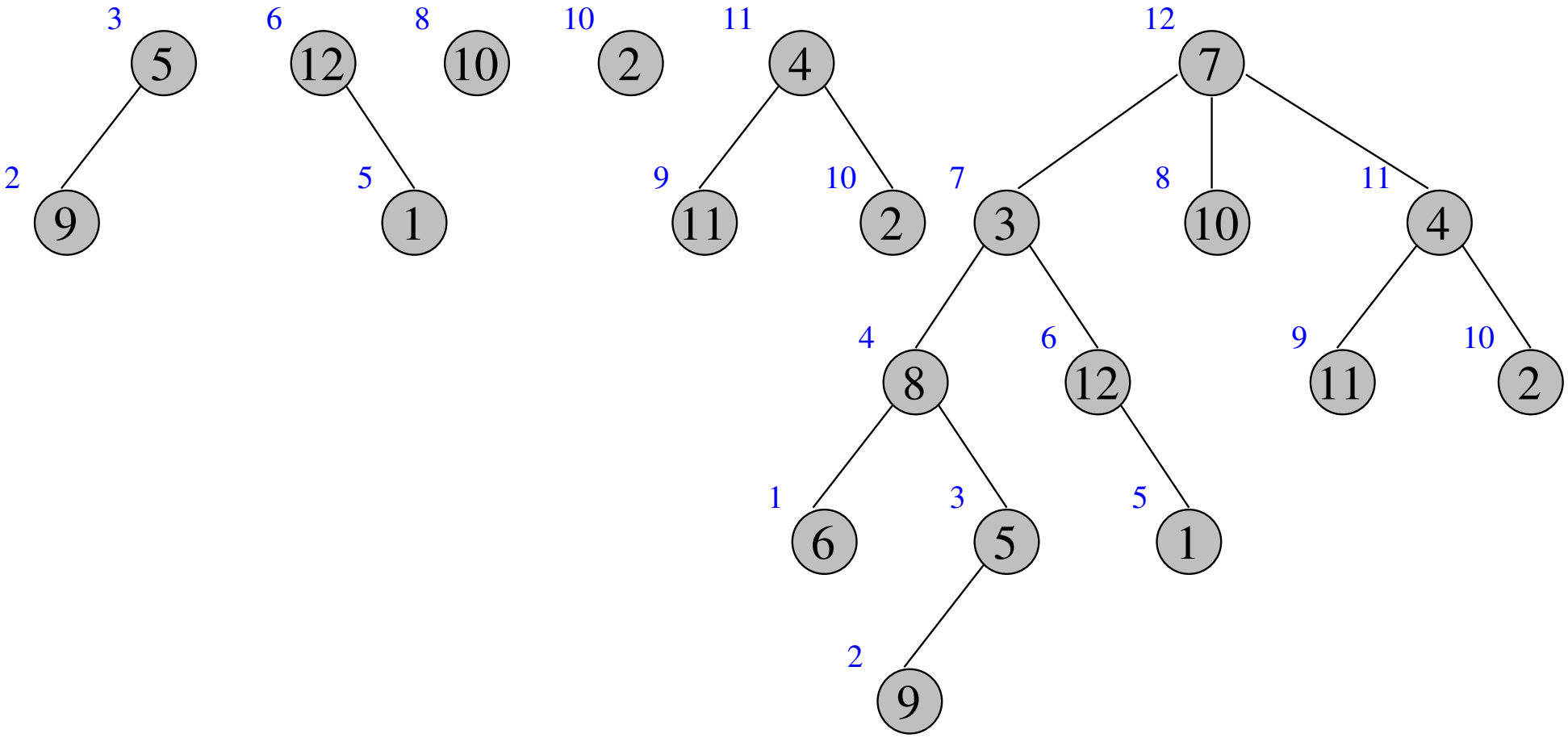
Racines des sous–arborescences de la décomposition :

7 5 12 10 4 2

Exemple de décomposition branche-gauche

Racines des sous–arborescences de la décomposition :

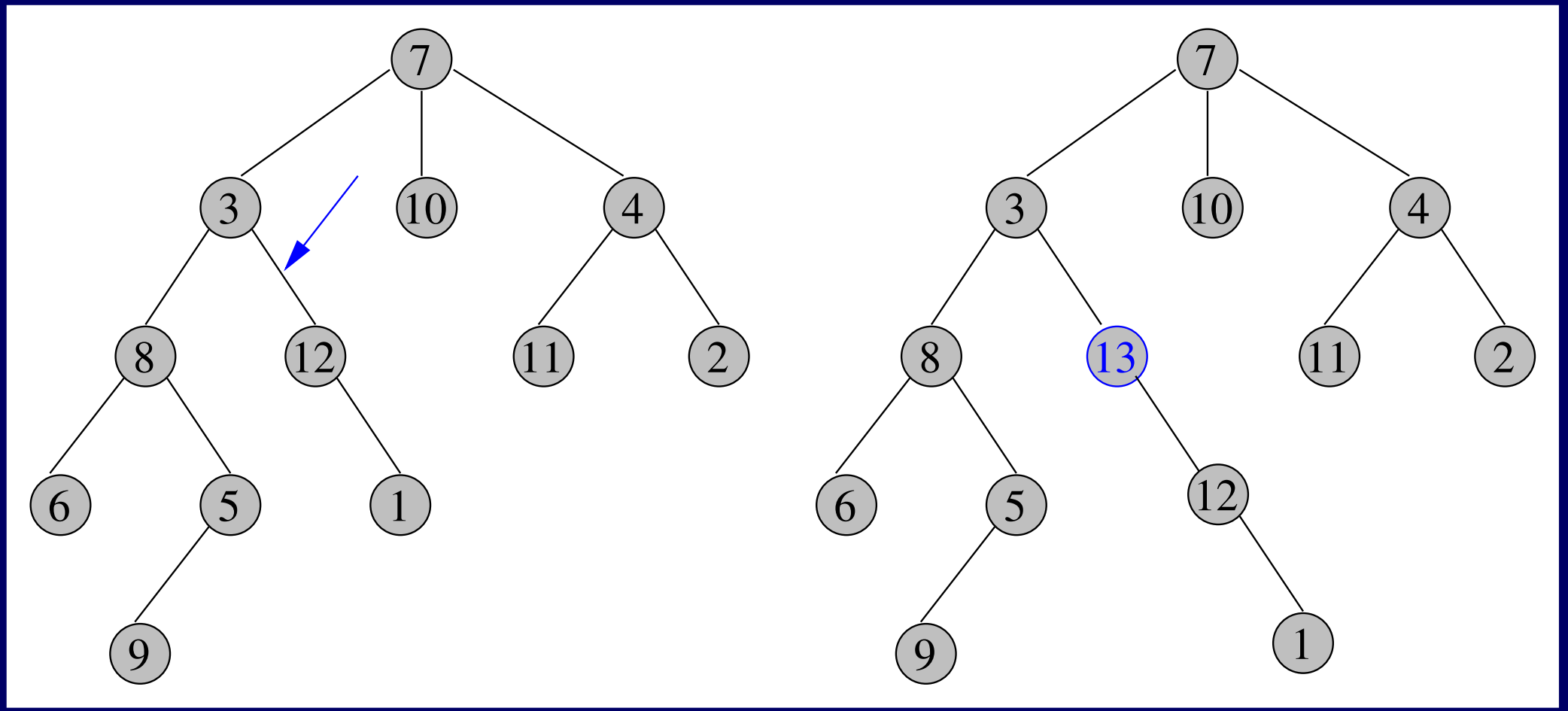
Sous–arborescences ordonnées selon la numérotation préfixe de leur racine :



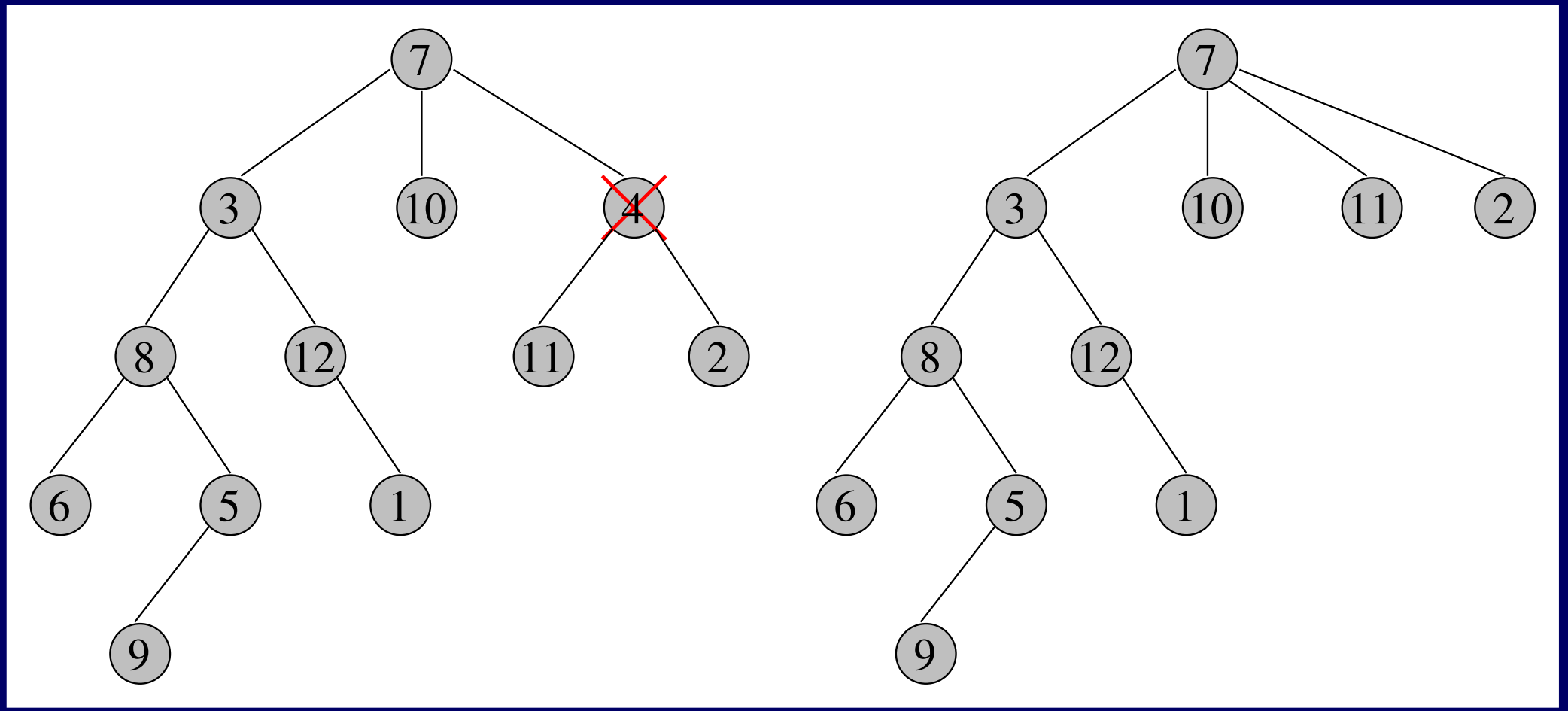
Les opérations d'édition

- Les 3 opérations classiques :
  - → Insertion d'un nœud
  - → Délétion d'un nœud
  - → Substitution de l'étiquette d'un nœud
- Opérations spécifiques pour certains domaines
- Ex : comparaison de structures secondaires d'ARN
  - → Opérations sur les arcs représentants les appariements entre bases
  - → Fusion/fissions de nœuds
- Pour un ensemble de 8 opérations données, le problème d'édition est NP-complet [Jiang *et al*, 02]
- Dans la suite on ne considère que les 3 opérations classiques

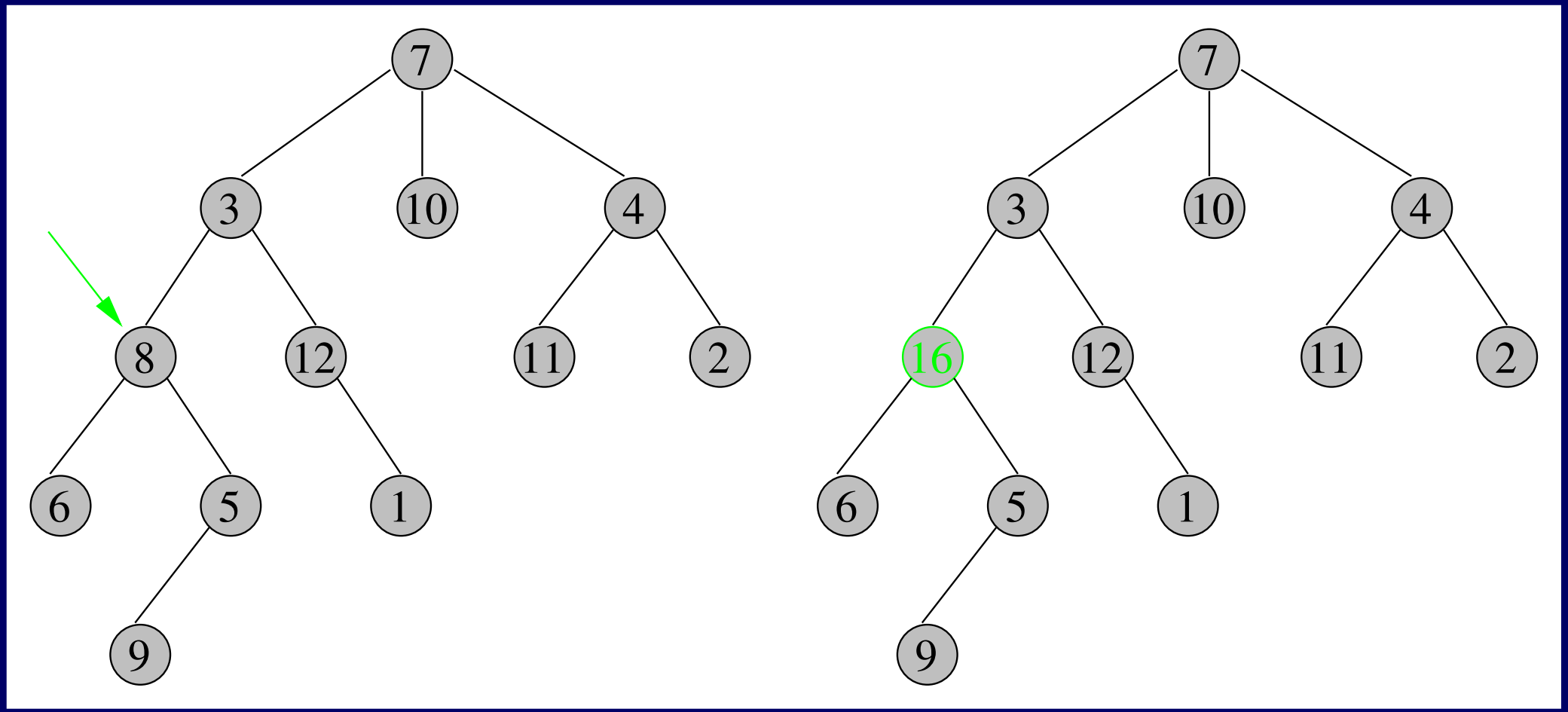
Insertion



Délétion



Substitution



Script d'édition

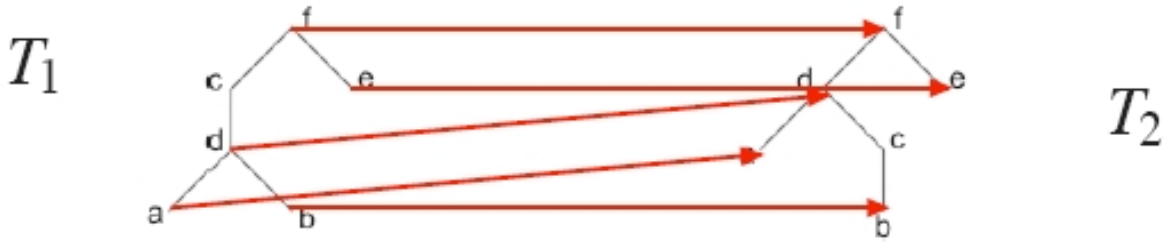
- Un script d'édition entre deux forêts F et G est une suite d'opérations d'édition transformant F en G

- Si on assigne un coût à chacune de ces opérations, la distance entre 2 forêts est définie comme le coût minimal d'un script d'édition pour passer d'une forêt à l'autre

- Problème d'édition entre arborescences :

Étant donné 2 arbres (ou 2 forêts) calculer leur distance et exhiber un script d'édition qui y correspond

Mapping entre 2 arbres



Un “mapping” entre deux arbres définit une séquence d’opérations d’édition permettant de passer de l’un à l’autre.

- Pour passer de T_1 à T_2:

1) Pour tous les couples faisant partie de la fonction de mappage, faire les substitutions, si nécessaire.
2) Supprimer tous les éléments de T_1 qui ne sont pas “mappés”
3) Insérer tous les éléments de T_2 qui ne sont pas “mappés”

Attention! Pour que cela soit possible on doit imposer certaines conditions sur les couples appartenant au “mapping”

[S. Hammel]

Formellement

- 2 arbres T_1 et T_2, $V(T_1)$ est l'ensemble des sommets de T_1

- Un triplet (M, T_1, T_2) est un *mapping* de T_1 vers T_2 si $M \subseteq V(T_1) \times V(T_2)$ et que pour chaque paire (v_1, w_1), $(v_2, w_2) \in M$:

  $\rightarrow$ $v_1 = v_2$ ssi $w_1 = w_2$
  ono-to-one condition

  $\rightarrow$ v_1 est un ancêtre de v_2 ssi w_1 est un ancêtre de w_2
  ancestor condition

  $\rightarrow$ v_1 est sur la gauche de v_2 ssi w_1 est sur la gauche de w_2
  sibling condition

[Bill, TCS 2005]

Soit $\gamma(M)$ le nombres de substitutions, suppressions et insertions correspondant à un mapping M. Alors

La distance d'édition entre deux arbres T_1 et T_2 est

$$dist(T_1, T_2) = min\{\gamma(M) | \, M \text{ mapping de } T_1 \text{ vers } T_2\}$$

L'algorithme de Zhang et Shasha utilise un étiquettage des sommets selon l'ordre suffixe et calcule

$$dist(T_1[i], T_2[j])$$

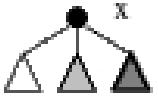
pour $1 \leq i \leq N_1$ et $1 \leq j \leq N_2$

Pour calculer ces valeurs de distances entre arbres, l'algorithme va se servir de distance entre forêts:

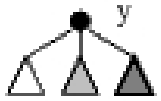
$$forestdist(T_1[1..i], T_2[1..j])$$

[S. Hammel]

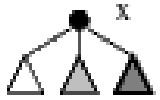
Récurrence (1/2)

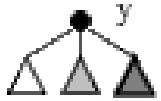
Theorème 5.1 *Soient les deux arborescences ordonnées étiquetées* $T_1 =$ 

et

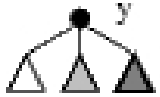
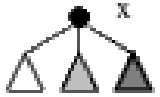
$T_2 =$ 

. *La distance d'édition entre les arborescences* T_1 *et* T_2 *est définie comme suit :*

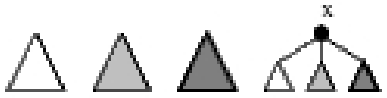
$treedist($ 

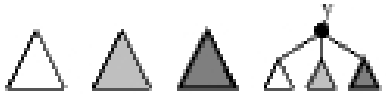
$,$ 

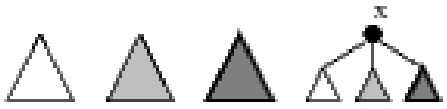
$) =$

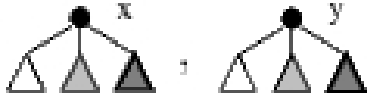
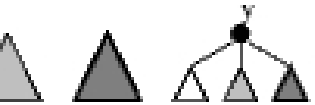
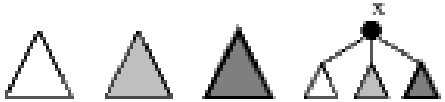
$$min \begin{cases} forestdist(\triangle\triangle\triangle , \triangle\triangle\triangle) + \gamma(\bullet^{x} , \bullet^{y}) \\ forestdist(\triangle\triangle\triangle , \text{  }) + \gamma(\bullet^{x} , \theta) \\ forestdist(\text{  } , \triangle\triangle\triangle) + \gamma(\theta, \bullet^{y}) \end{cases}$$

[Thèse L. Tichit]

Theorème 5.2 *Soient les forêts ordonnées* $F_1 =$  *et*

$F_2 =$  *. La distance entre les forêts ordonnées* F_1 *et* F_2 *est définie comme suit :*

$forestdist($  $) =$

$min \begin{cases} forestdist($  $) + treedist($  $) \\ forestdist($  $) + \gamma($  $) \\ forestdist($  $) + \gamma($  $) \end{cases}$

[Thèse L. Tichit]

Dans cet algorithme les sommets des arborescences sont ordonnés selon l'ordre suffixe gauche-droite.

- $T[i..j]$ représente la forêt ordonnée de T induite par les sommets numérotés de i à j inclus.
- $t[i_1], \ldots, t[i_{n_i}]$ représentent les sommets fils de $t[i]$.
- $F[i_r, i_s]$, telle que $1 \leq r \leq s \leq n_i$, représente la forêt ordonnée constituée des sous-arborescences $T[i_r], \ldots, T[i_s]$.
- $l(i)$ représente le numéro dans l'ordre suffixe gauche-droite de la première feuille (celle la plus à gauche) du sous-arbre enraciné en $t[i]$.

On a donc $F[i_1, i_{n_i}] = F[i]$ et $F[i_p, i_p] = T[i_p] \neq F[i_p]$.

Dans le cas où $t[i]$ est une feuille, alors $l(i) = i$. Avec cette notation on a donc $T[i] = T[l(i)..i]$ et $F[i] = T[l(i)..i-1]$.

On peut de plus noter que $F[l(i)..i] = T[l(i)..i] = T[i]$.

En ce qui concerne le calcul de la distance d'édition proprement dit

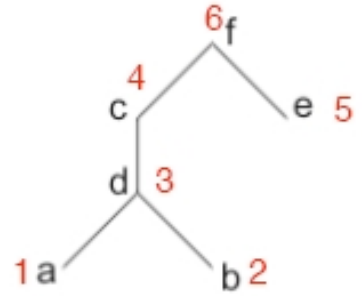
- La distance entre deux arborescences ordonnées étiquetées $T_1[i]$ et $T_2[j]$ sera notée $treedist(i, j)$.
- La distance d'édition entre deux sous-forêts ordonnées $T_1[i'..i]$ et $T_2[j'..j]$ sera notée $forestdist(T_1[i'..i], T_2[j'..j])$ ou plus simplement $forestdist(i'..i, j'..j)$ si le contexte ne permet pas d'ambiguïté.

[Thèse L. Tichit]

Définitions nécessaires (1/2)

1) $\ell(i)$: Soit T[i] le sommet de l'arbre T portant l'étiquette i selon l'ordre suffixe. $\ell(i)$ est l'étiquette de la feuille la plus à gauche dans le sous-arbre de racine T[i] de l'arbre:

T_1



i	1	2	3	4	5	6
$\ell(i)$	1	2	1	1	5	1

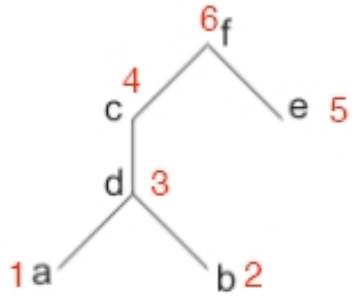
Remarque: Chaque fois que la valeur de $\ell(i)$ est 1, $T_1[1..i]$ est un arbre et non une forêt.

[S. Hammel]

2) LR_keyroots(T):

$$LR\_keyroots(T) = \{k \mid \text{il n'existe pas de } k' > k \text{ tel que } \ell(k) = \ell(k')\}$$

T_1



i	1	2	3	4	5	6
$\ell(i)$	1	2	1	1	5	1

$$\Rightarrow \quad LR\_keyroots(T_1) = \{2,5,6\}$$

[S. Hammel]

Algorithme (1/2)

Algorithme : Edit(T_1,T_2)

Début
  Prétraitement :
    Calcul de $l()$
    Calcul de $LR\_keyroots(T_1)$
    Calcul de $LR\_keyroots(T_2)$
  Pour $s := 1$ **à** $|LR\_keyroots(T_1)|$ **faire**
    Pour $t := 1$ **à** $|LR\_keyroots(T_2)|$ **faire**
      $i := LR\_keyroots(T_1)[s]$;
      $j := LR\_keyroots(T_2)[t]$;
      Calcul de $treedist(i, j)$;
Fin

Sortie : $treedist(T_1[i], T_2[j])$ où $1 \leq i \leq |T_1|$ et $1 \leq j \leq |T_2|$.

[Thèse L. Tichit]

Algorithme (2/2)

Algorithme : $treedist(i, j)$

Début
  $forestdist(\theta, \theta) = 0$;
  Pour $i_1 := l(i)$ **à** i **faire**
    $forestdist(T_1[l(i)..i_1], \theta) = forestdist(T_1[l(i)..i_1 - 1], \theta) + \gamma(t_1[i_1], -)$
  Pour $j_1 := l(j)$ **à** j **faire**
    $forestdist(\theta, T_2[l(j)..j_1]) = forestdist(\theta, T_1[l(j)..j_1 - 1]) + \gamma(-, t_2[j_1])$
  Pour $i_1 := l(i)$ **à** i **faire**
    Pour $j_1 := l(j)$ **à** j **faire**
      Si $l(i_1) = l(i)$ et $l(j_1) = l(j)$ **alors**
        Calculer $forestdist(T_1[l(i)..i_1], T_2[l(j)..j_1])$
          comme indiqué dans le Lemme 5.3 (1).
        $treedist(i_1, j_1) = forestdist(T_1[l(i)..i_1], T_2[l(j)..j_1])$
        /* Stocker les valeurs de $treedist$ dans un tableau permanent */
      Sinon
        Calculer $forestdist(T_1[l(i)..i_1], T_2[l(j)..j_1])$
          comme indiqué dans le Lemme 5.3 (2).
Fin

Sortie : $treedist(T_1[s], T_2[t])$ où $s \in desc(i)$ et $t \in desc(j)$ avec $l(s) = l(i)$ et $l(t) = l(j)$.

[Thèse L. Tichit]

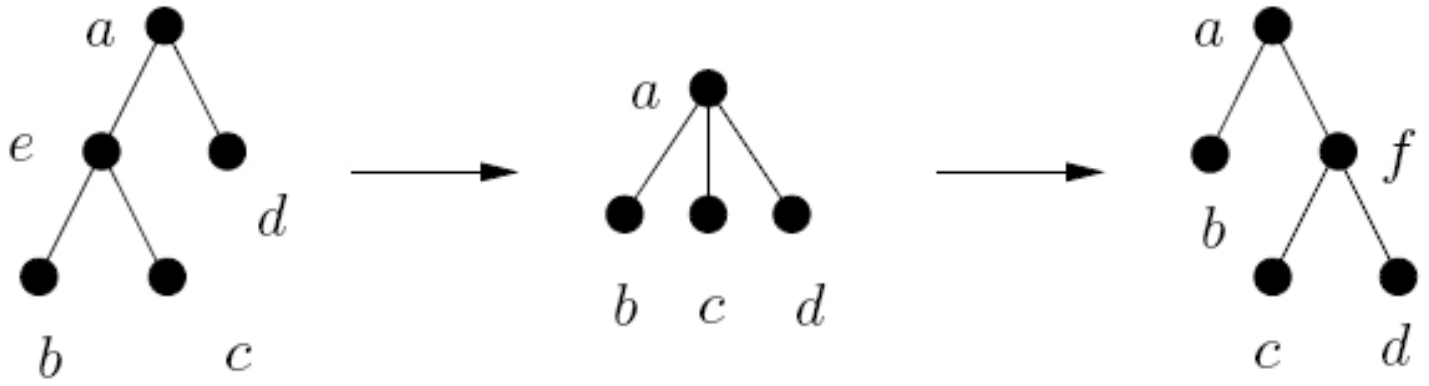
Comparaison édition *vs* alignement

- [Jiang *et al*, 95] introduit une nouvelle approche pour la comparaison de structures secondaires en appliquant à la lettre la notion d'alignement, càd en ajoutant des espaces
- Sur les séquences, les définitions de distance d'édition et d'alignement coïncident
- Ce n'est plus le cas en général sur les arbres
- L'alignement définit une superstructure alors que la distance définit un sous-arbre commun
- Vu d'un script d'édition, cela revient à imposer que toutes les insertions aient lieu avant les délétions, ce qui autorise moins de remaniement interne

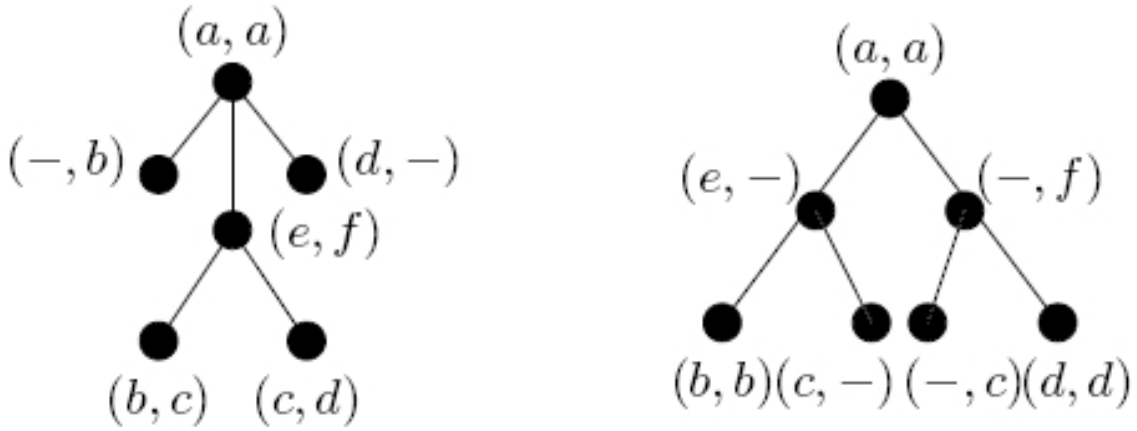
[HDR H. Touzet]

Exemple

1. Distance d'édition



2. Alignement



[HDR H. Touzet] *(ré-étiqueté)*

- Introduction : modélisation
- Quelques rappels sur les arbres
- Édition et *mapping*
- L'algorithme de Zhang & Shasha
- Distance d'édition $\neq$ alignement
- Références

Références

- Thèse de Laurent Tichit "Algorithmes des structures biologiques : l'édition d'arborescences pour la comparaison de structures secondaires d'ARN" (2003)

- HDR d'Hélène Touzet "Structures combinatoires pour l'analyse de génomes" (2004)

- Partie modélisation structure topologique des plantes : Godin, C., Caraglio, Y., 1998. **A multiscale model of plant topological structures**. *Journal of Theoretical Biology*, 191:1-46.

- Illustrations :

  → Transp. 4, 5 et 6 : matériel d'illustration provenant de L. Tichit

  → Transp. 8, 9, 10, 11 et 12 : définitions et propriétés des arbres d'après "Introduction à l'algorithmique", Cormen, Leiserson, Rivest et Stein. 2^{e} édition, 1994, Dunod.