

# Arbres et Arbres Binaires (§5)

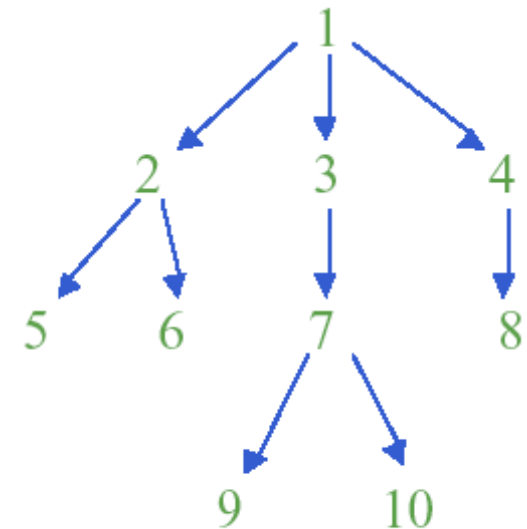


# Qu'est-ce qu'un arbre?

- Un arbre est un modèle abstrait d'une structure hiérarchique
- Un arbre est constitué de noeuds reliés par une relation parent-enfant

$A=(N,P)$

- $N$  ensemble de noeuds
- $P$  relation binaire "parents de"
- $r \in N$ , la racine



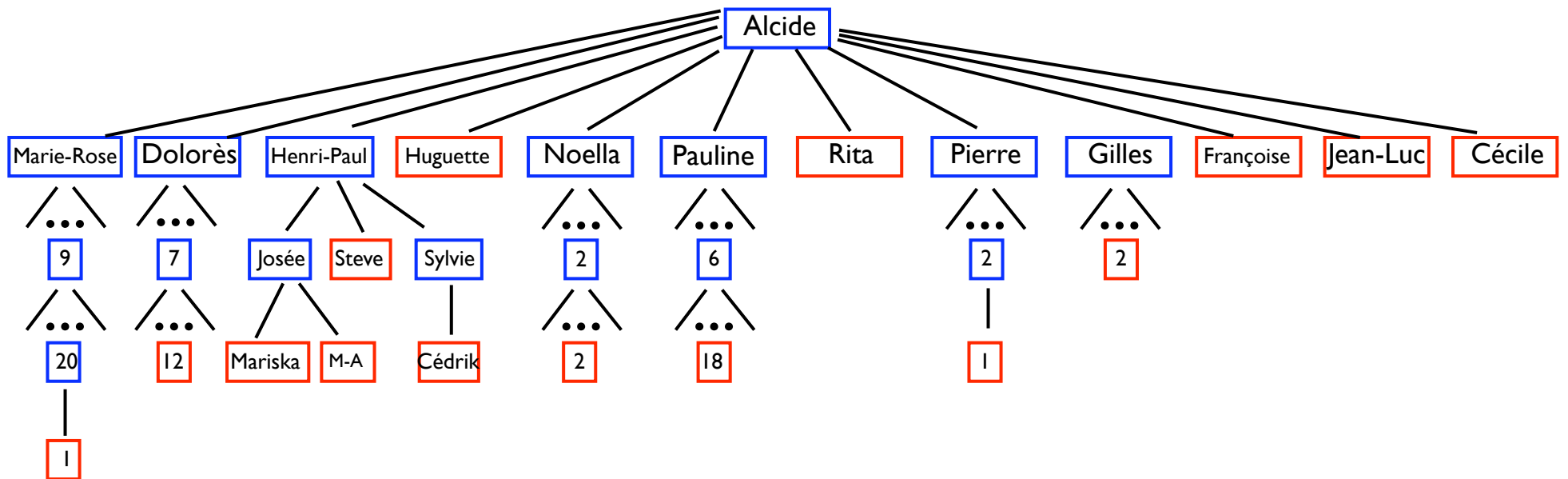
- $\forall x \in N, \exists$  un seul chemin de  $r$  vers  $x$

$\Rightarrow \forall x \in N - \{r\}$   $x$  a exactement un parent

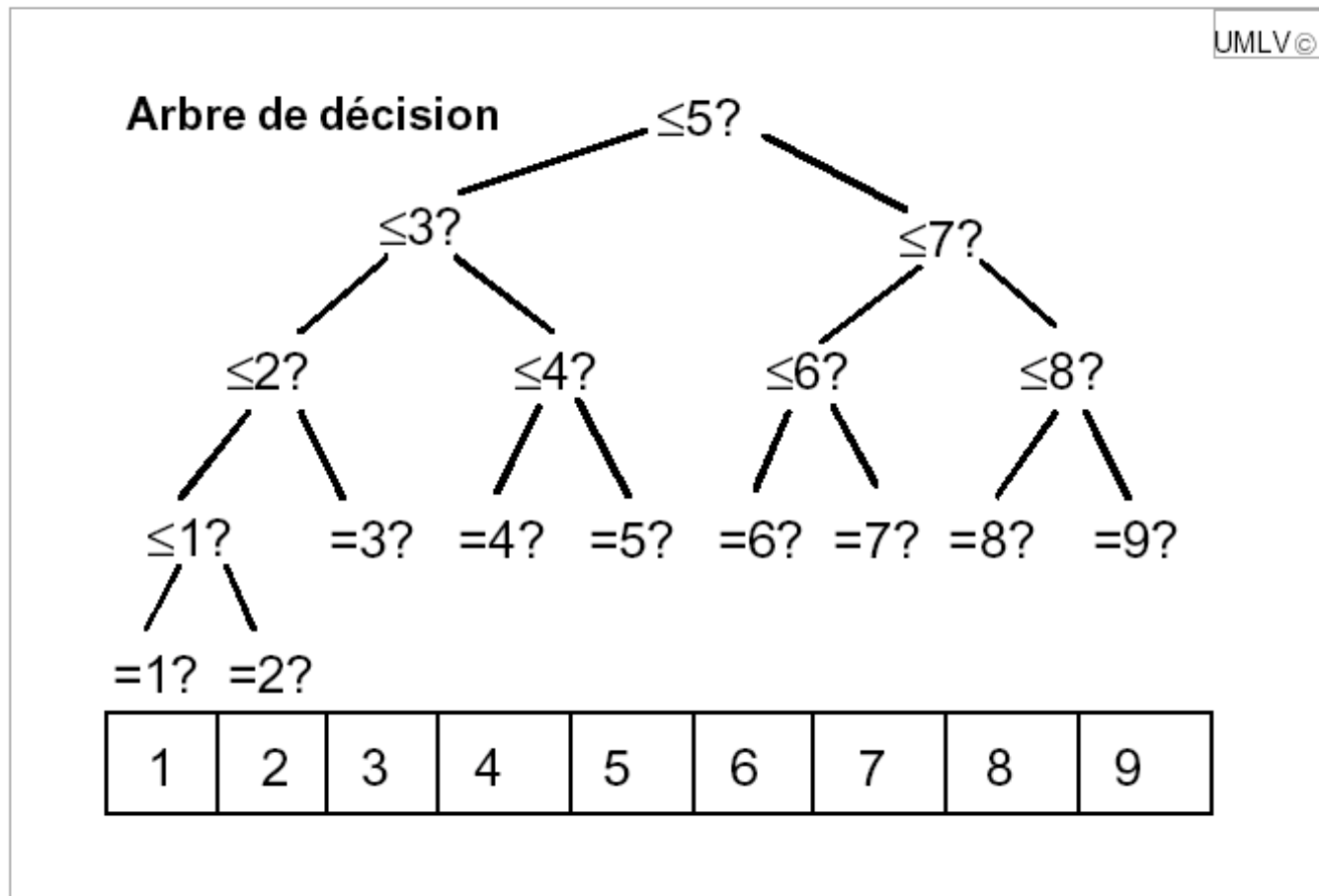
# Applications

- Organisation de fichiers
- Arbres généalogiques
- Livres
- Environnement de programmation

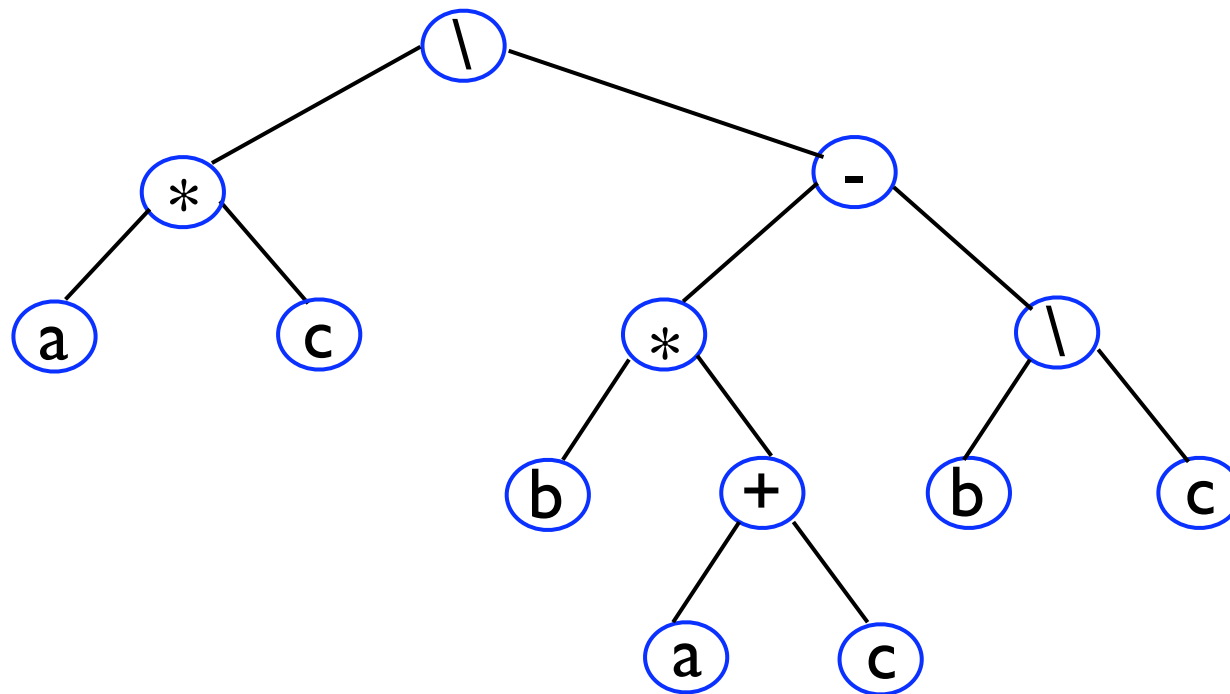
# Exemple 1: Arbre généalogique



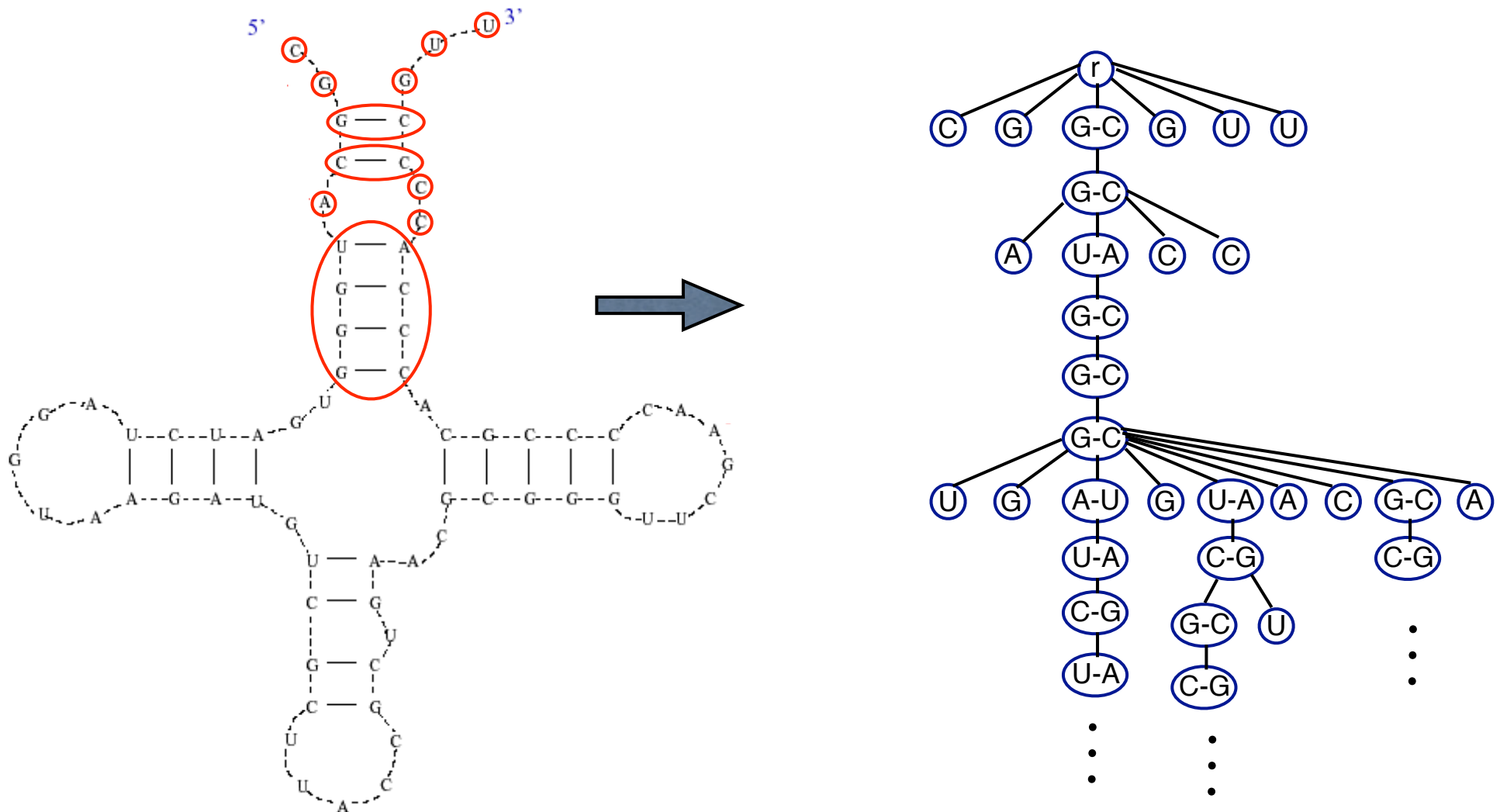
## Exemple2: Arbres de décision



## Exemple3: Arbres syntaxiques



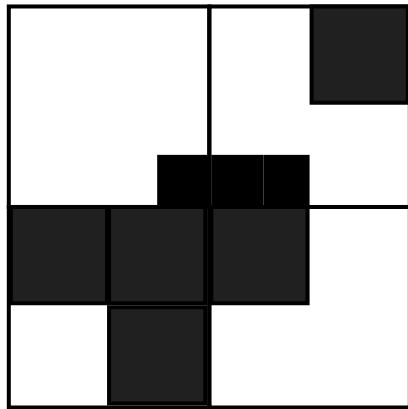
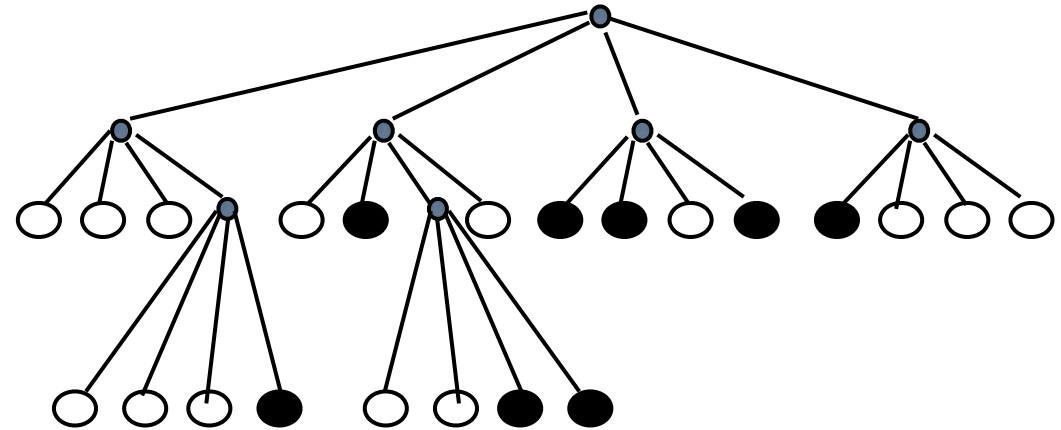
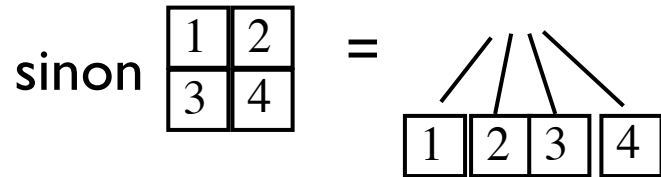
# Exemple4: Structures secondaires d'ARN



# Exemple5: Arbres quartiques

image toute noire = feuille ●

image toute blanche = feuille ○

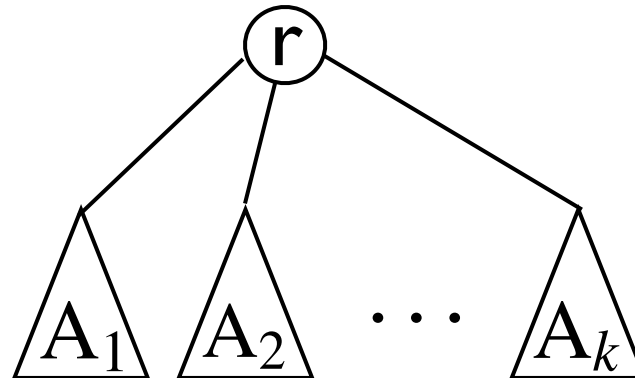




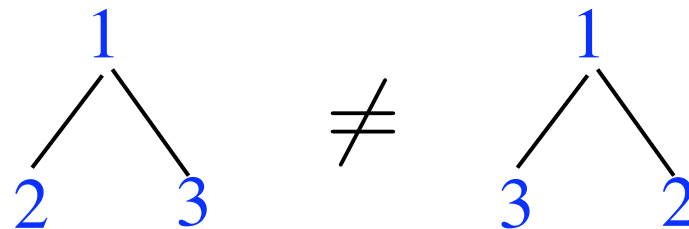
# Arbres: Définition récursive

$$\text{Arbre } A = \begin{cases} \Lambda & \text{arbre vide ou} \\ (r, \{A_1, \dots, A_k\}) & r \text{ élément, } A_1, \dots, A_k \text{ arbres} \end{cases}$$

$A = \Lambda$  ou

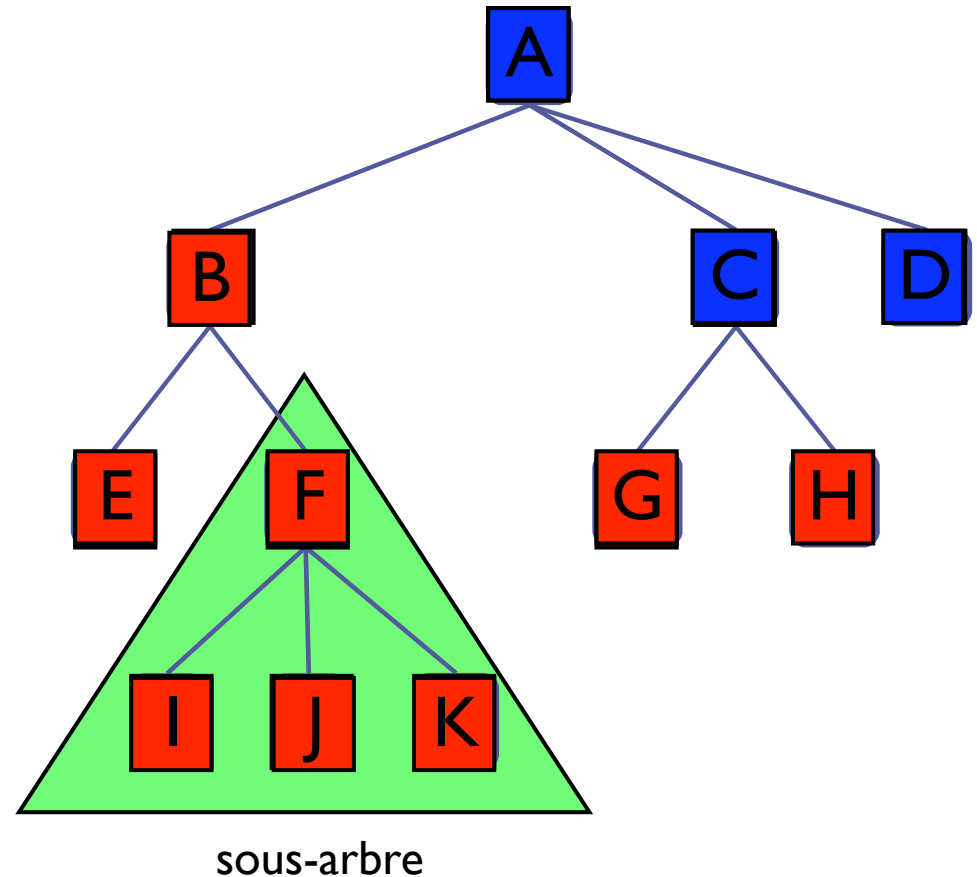


Si  $A$  ordonné (ou planaire)



# Terminologie

- **Racine**: le noeud sans parent
- **Noeuds internes**: noeuds ayant au moins un enfant
- **Noeuds externes (feuilles)**: noeuds sans enfant
- **Ancêtres d'un noeud**: parent, grand-parent, arrière grand-parent...
- **Descendants d'un noeud**: fils, petit-fils, arrière petit-fils...
- **Frères d'un noeud**: tous les autres noeuds ayant le même parent
- **Sous-arbre**: arbre constitué d'un noeud et de ses descendants

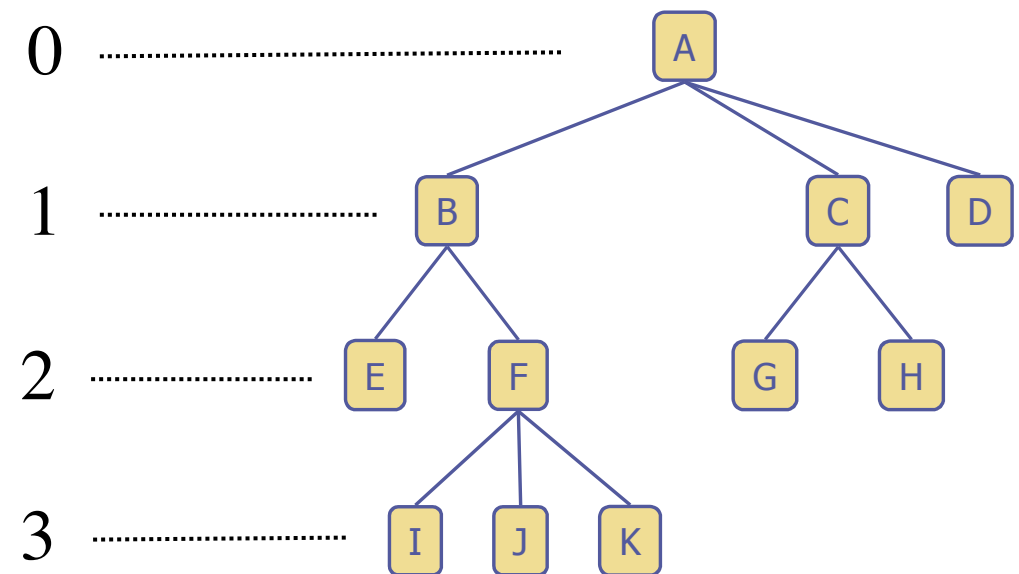


# Terminologie (suite)

- **Profondeur d'un noeud x**: le nombre d'ancêtres de x ou distance de x à la racine

- $\text{Profondeur}(x) = \begin{cases} 0 & \text{si } x \text{ est la racine de l'arbre} \\ 1 + \text{profondeur}(\text{parent}(x)) & \text{sinon} \end{cases}$

- **Hauteur d'un arbre**: profondeur maximale d'un noeud de l'arbre
- **Hauteur d'un noeud x**: distance de x à son plus lointain descendant qui est un noeud externe.



# Parcours

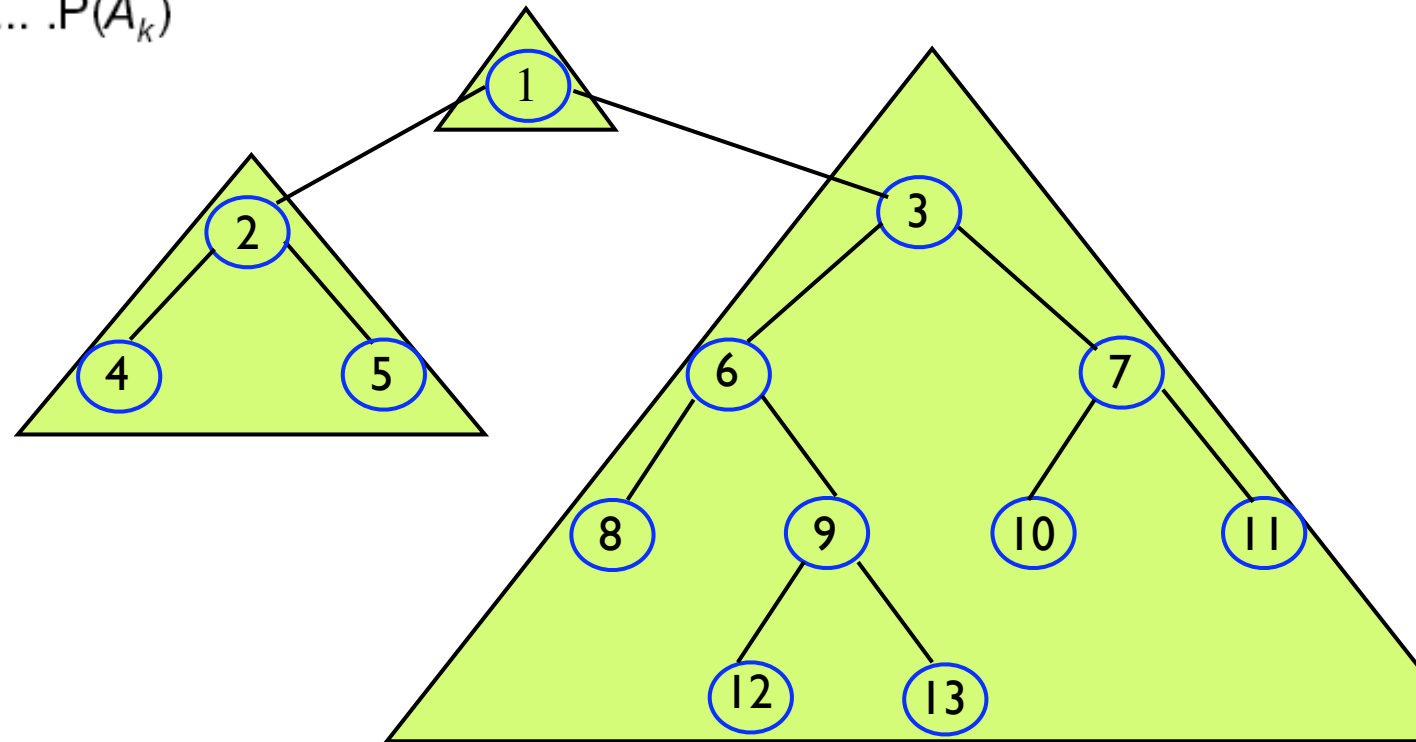
- Une fonction: arbre  $\longrightarrow$  liste de ses noeuds
- Utile pour l'exploration des arbres
- Deux types
  - **parcours en profondeur:**  
préfixe, suffixe, symétrique  
parcours branche après branche
  - **parcours en largeur:**  
hiérarchique  
parcours niveau par niveau

# Parcours préfixe

Arbre non vide  $A = (r, A_1, A_2, \dots, A_k)$

**Parcours préfixe**

$$P(A) = (r).P(A_1). \dots .P(A_k)$$

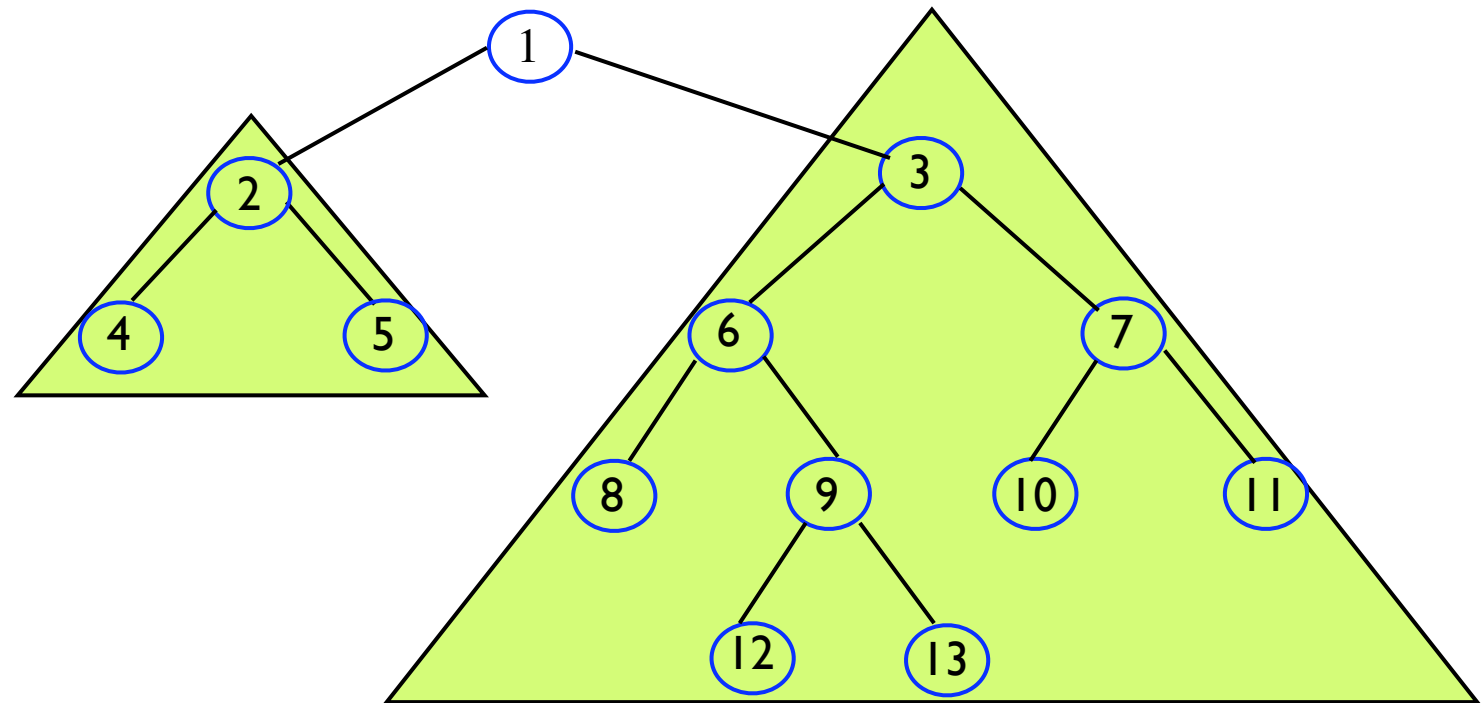


**(1, 2, 4, 5, 3, 6, 8, 9, 12, 13, 7, 10, 11)**

# Parcours suffixe

**Parcours suffixe:** un noeud est visité après ses descendants

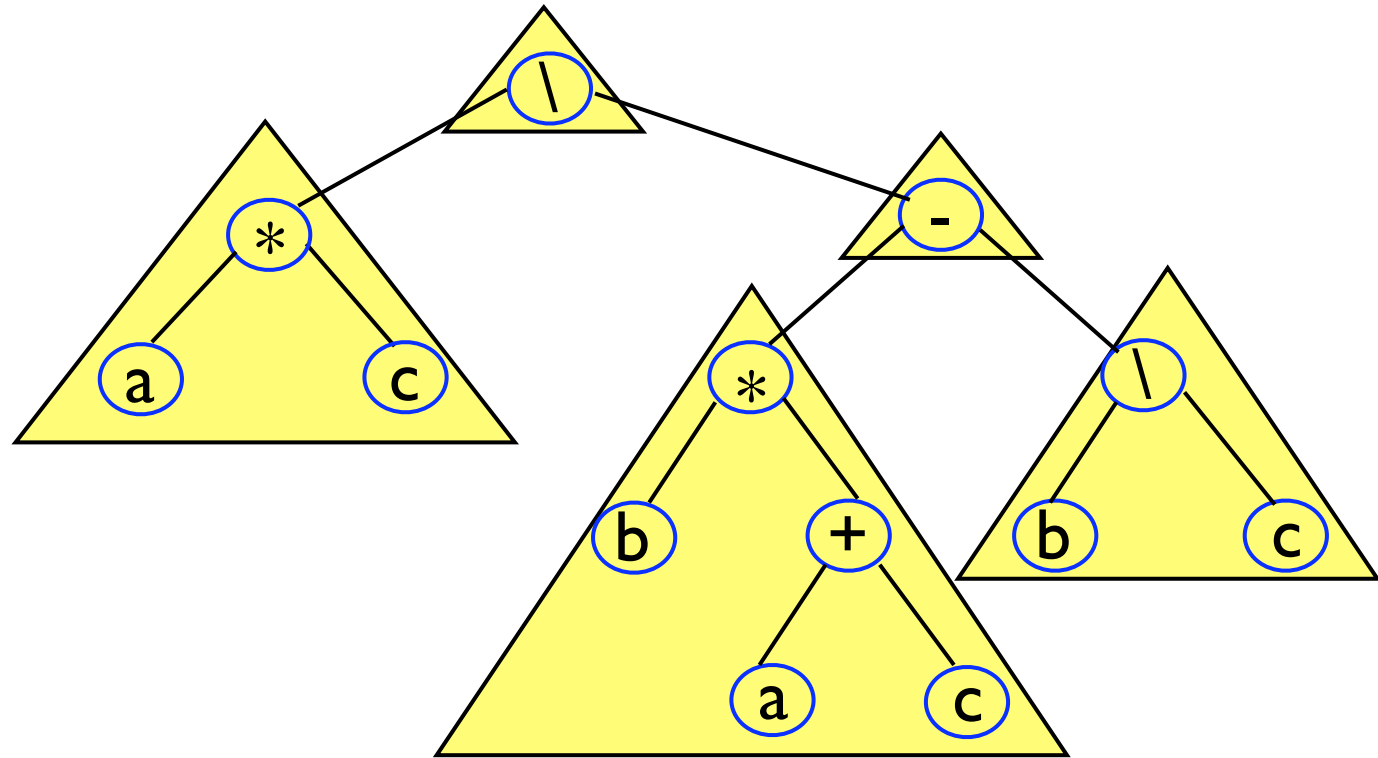
$$S(A) = S(A_1) \cdot S(A_2) \dots \cdot S(A_k) \cdot r$$



(4, 5, 2, 8, 12, 13, 9, 6, 10, 11, 7, 3, 1)

# Parcours symétrique (arbre binaire)

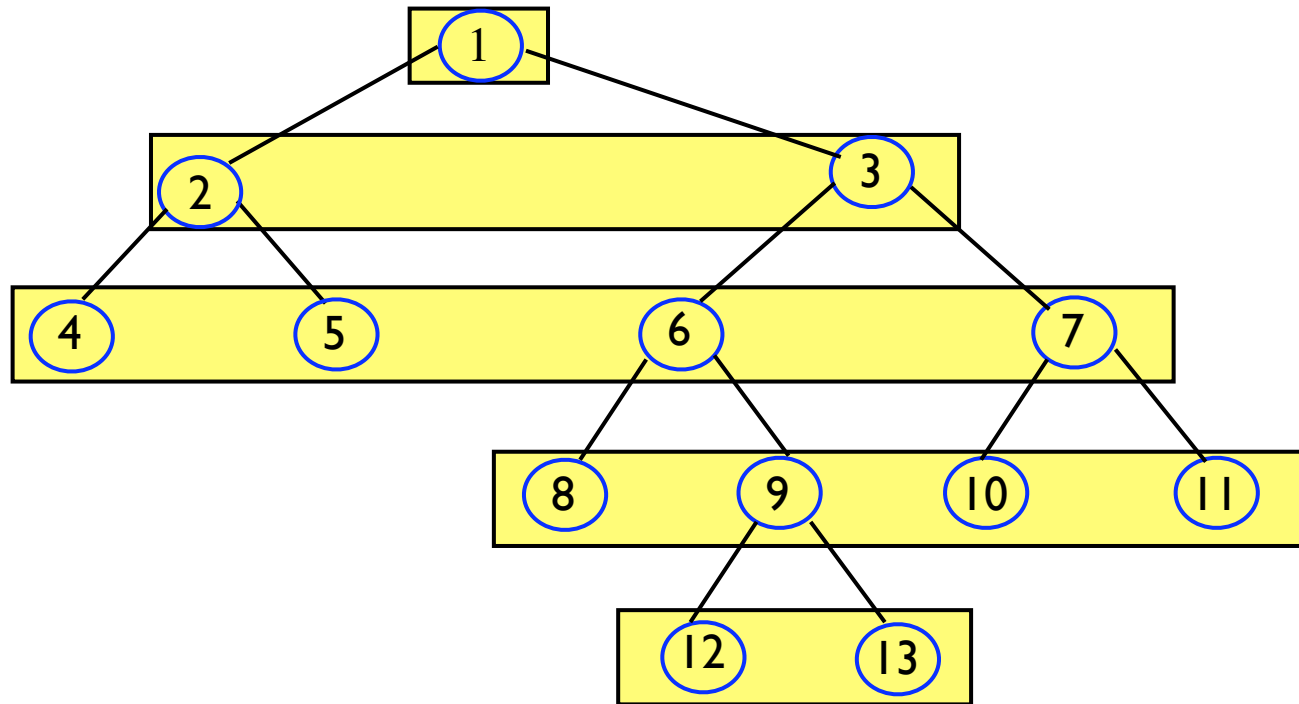
**Parcours symétrique:** un noeud est visité après son sous-arbre gauche et avant son sous-arbre droit



$$(a * c) \ [b * (a + c) - (b \ c)]$$

# Parcours hiérarchique

**Parcours hiérarchique:** on visite les noeuds niveau par niveau, de gauche à droite, en commençant par la racine



(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13)

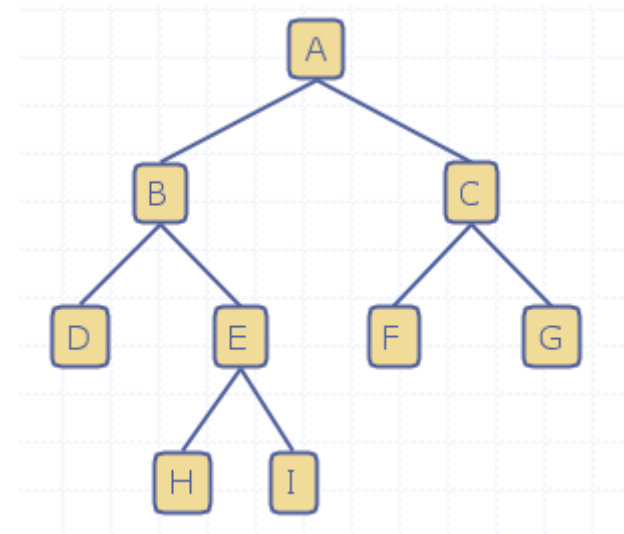


# Arbres binaires (§5.5)

- Un arbre binaire est un arbre ayant les propriétés suivantes:
  - ▣ Un noeud interne à au plus deux fils (exactement deux si l'arbre est complet)
  - ▣ Les fils d'un noeud forment une paire ordonnée
- On appelle les fils d'un noeud interne fils droit et fils gauche

- Applications:

- ▣ Processus de décisions
- ▣ Expressions arithmétiques
- ▣ Recherche



# Hauteur d'un arbre binaire

## Mesures des arbres binaires

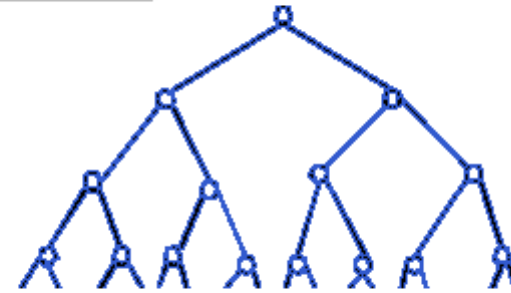
UMLV©

Arbre binaire, hauteur  $h$  et  $n$  nœuds

**Arbre plein**

$2^i$  nœuds au niveau  $i$

$n = 2^{h+1} - 1$



**Arbre filiforme**

1 nœud par niveau

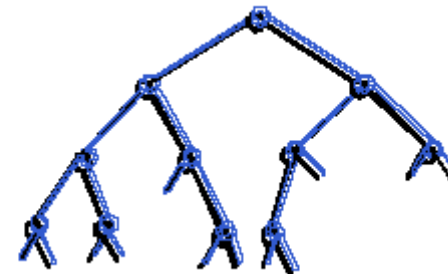
$n = h + 1$



**Arbre binaire**

$h + 1 \leq n \leq 2^{h+1} - 1$

$\log_2(n + 1) - 1 \leq h \leq n - 1$



# TAD Arbre

## ○ Principales opérations:

- ❑ ajouter(**noeud**): ajouter un noeud à l'arbre
- ❑ objet **enlever()**: enlever un noeud de l'arbre et retourner l'objet dans le noeud
- ❑ noeud **parent(p)**: retourne le noeud parent du noeud p
- ❑ noeud **racine()**: retourne la racine de l'arbre
- ❑ ens **fil(s(p))**: retourne l'ensemble des noeuds fils de p

## ○ Opérations booléenne:

- ❑ booléen **estRacine(p)**: retourne vrai si p est la racine
- ❑ booléen **estExterne(p)**: retourne vrai si p est une feuille
- ❑ booléen **estInterne(p)**: retourne vrai si p est un noeud interne
- ❑ ...

# TAD Arbre Binaire

- Le TAD arbre binaire étend le TDA arbre i.e. qu'il hérite de toutes les opérations du TAD arbre.
- Opérations additionnelles:
  - `filGauche(p)`
  - `filDroit(p)`
  - boolean `aFrèreGauche(p)`
  - boolean `aFrèreDroit(p)`

# Implémentation: liste

UMLV©

Implémentation

Représentation de la relation  $P$   
table des parents

**Avantages**  
représentation simple  
parcours faciles vers la racine  
économique en mémoire

**Inconvénients**  
accès difficiles aux nœuds  
depuis la racine

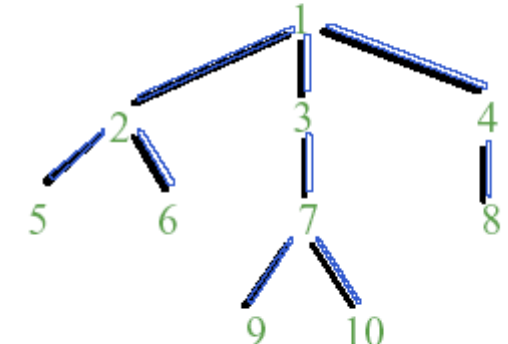
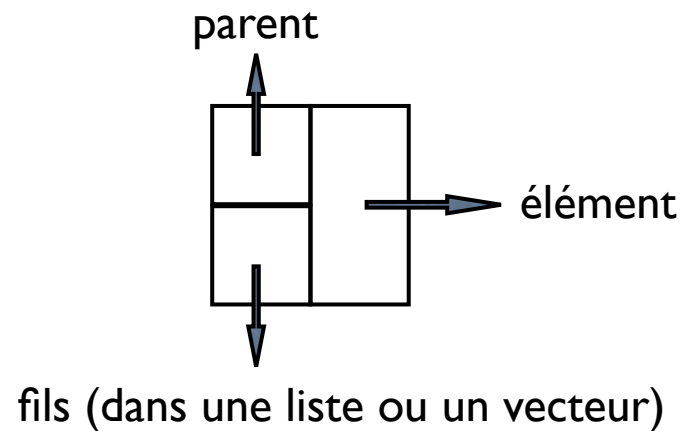


table des parents  $P$

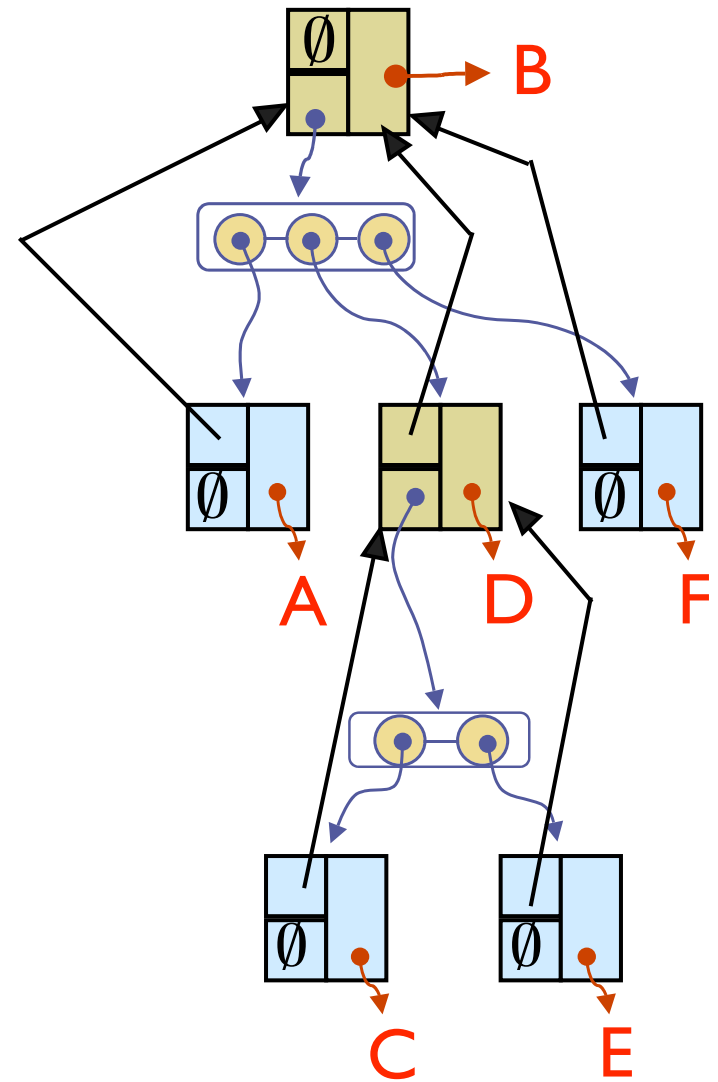
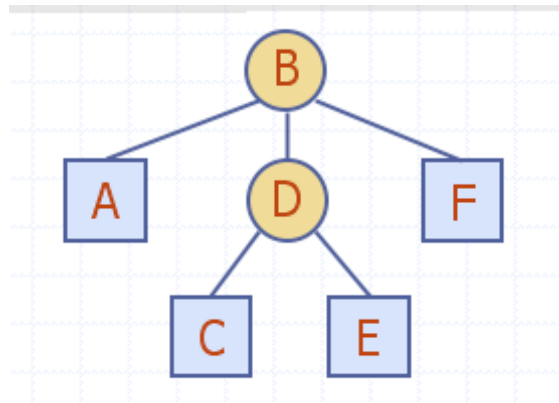
-	1	1	1	2	2	3	4	7	7
1	2	3	4	5	6	7	8	9	10

# Implémentation: structure chaînée

- Un noeud de la structure chaînée représentant le noeud d'un arbre est de cette forme



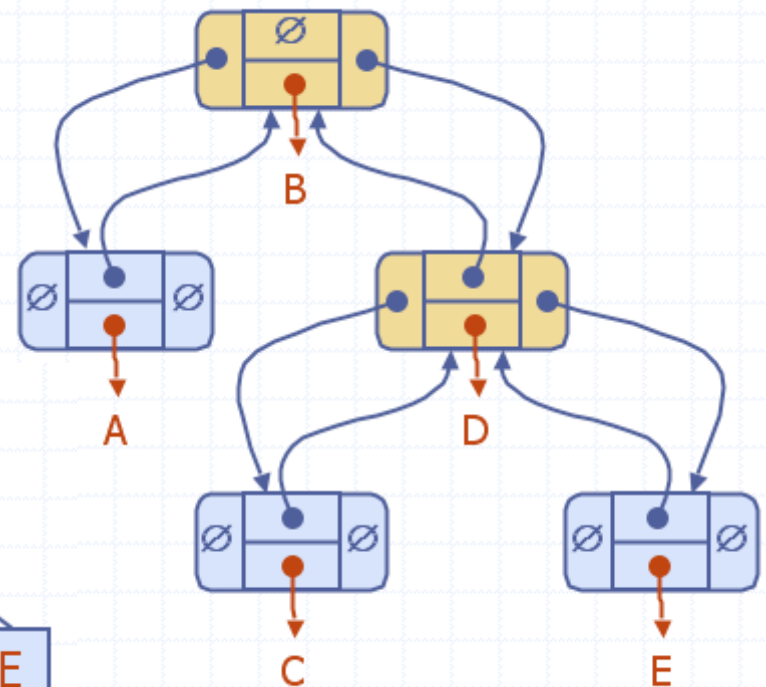
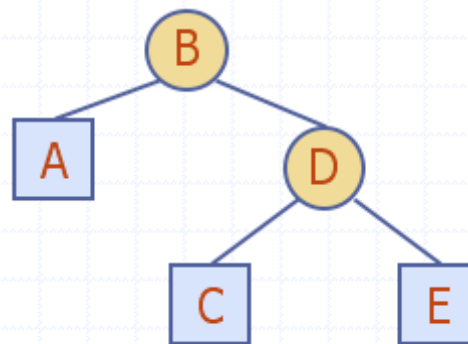
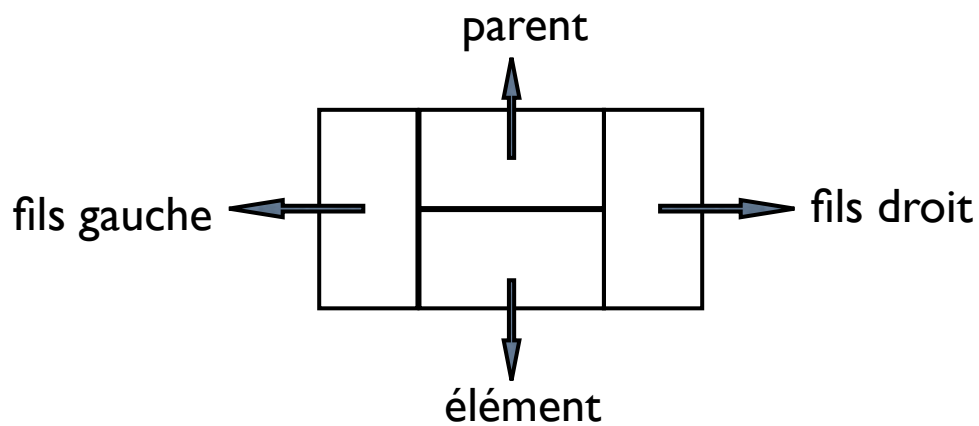
# Exemple



© 2004, Goodrich, Tamassia

# structure chaînée (arbres binaires)

- Le noeud de la structure chaînée représentant le noeud d'un arbre binaire est de cette forme



© 2004, Goodrich, Tamassia