Université Félix Houphouët-Boigny
(UFHB)

Cours d'algorithme avancée
L3-Info

Enseignant :

Dr GBAME Gbede Sylvain

Assistant, enseignant chercheur à l'UFHB

gbamegbedesylvain@gmail.com

MODULE : Algorithme avancée

PLAN

Chapitre 7 : Partie 2

Les Structures de données

7.6 Les listes chainée

6.1 Définition

Une liste chaînée désigne une structure de données dynamique représentant une collection d'éléments de même type. Les éléments d'une liste chainée appelés maillon sont éparpillés dans la mémoire et reliés entre eux par des pointeurs. Une liste chainée est donc une collection de maillons reliés les uns à la suite des autres.

7.6 Les listes chainée

6.2 Déclaration d'une liste chainée

La déclaration d'une liste chainée passe par la déclaration d'un maillon en utilisant une structure de données enregistrement.

```
Type
Maillon = enregistrement
    Valeur : type ;
    Maillon *suivant ;
Fin enregistrement
Maillon *liste ;
```

7.6 Les listes chainée

6.3 Initialisation d'une liste chainée

A l'initialisation, une liste chainée ne pointe sur aucun maillon.

Liste = Null ;

6.4 Les primitives d'une liste chaînée

6.4.1 Test de liste vide

Une liste chainée est vide lorsqu'elle n'a aucun maillon.

```
Estvide (liste : L)
    Si L = Null alors
            Retourner vrai
    Sinon
            Retourner faux
Fin
```

7.6 Les listes chainée

6.4.2 Insertion d'un maillon

Pour insérer un nouveau maillon, on :
– Créer le nouveau maillon
– Si la liste est vide elle reçoit l'adresse du nouveau maillon
– Sinon cherche le dernier maillon et on insère l'adresse du nouveau maillon dans le pointeur suivant de celui-ci.

7.7 Les Arbres

7.1 Définition

Un arbre une structure de données représentant une collection de données du même types, organisées hiérarchiquement. Un arbre est donc constitué de nœuds qui peuvent avoir comme enfants d'autres nœuds. Dans cette structure hiérarchisée le sommet de l'arbre est appelé la racine, il n'a pas de prédécesseur. Un nœud qui ne possède pas d'enfant est appelé une feuille. Une branche est une suite consécutive de nœuds partant de la racine à une feuille (figure 5).

7.7 Les Arbres

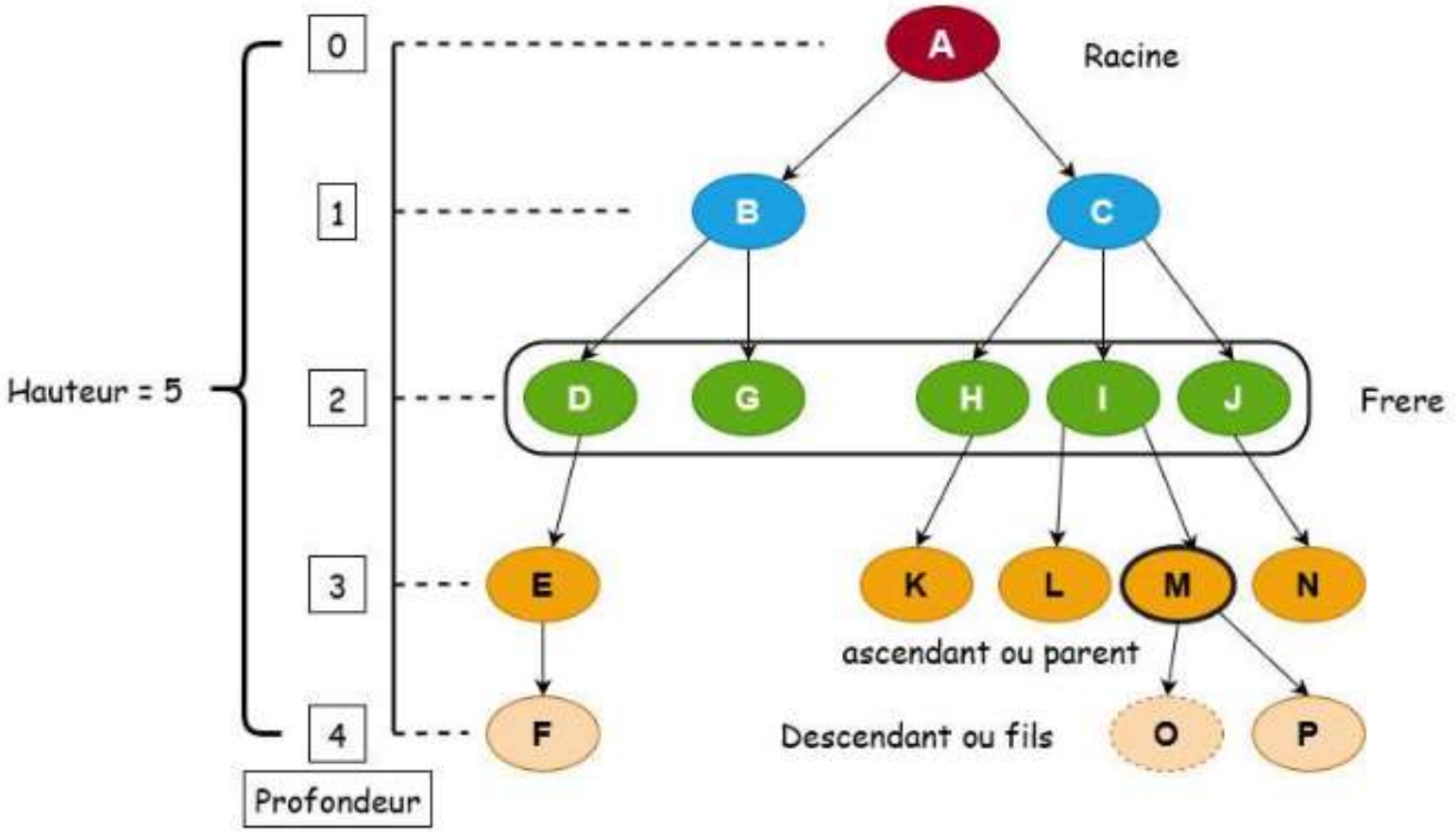


Figure 5 : Représentation d'un arbre

7.7 Les Arbres

Un arbre est une structure de données arborescente récursive. C’est-à-dire qu'un arbre est par essence composé de sous arbre. Dans l'exemple de la figure 5, l'arbre de racine A est composé du sous arbre de racine B et celui de racine C eux même composés de sous arbres et ainsi de suite.

7.7 Les Arbres

7.1.1 Terminologie

- Racine, c'est le nœud qui est au sommet de l'arbre, il n'a pas de prédécesseur (A).
- Arêtes désignent les liens entre les nœuds représentant une relation parent/fils.
- Feuille est un nœud qui n'a pas de fils ou de descendant (F, G, K, L, O, P et N).
- Nœud interne est un nœud en dehors de la racine qui a au moins 1 nœud descendant.
- Chemin (C) d'un nœud x est la suite de nœud par lesquels il faut passer pour aller de la racine au nœud x, C(K) = (A, C, H, K)
- Profondeur (P) d'un nœud est le nombre de nœud constituant son chemin, P(K) = 4
- Hauteur d'un arbre désigne la plus grande profondeur de ses nœuds. Ainsi, elle est égale à la profondeur du nœud le plus profond, H = max(P).
- H(Ø) = 0 ; Arbre vide et H(x) = H(Asc(x)) + 1

7.7 Les Arbres

- Niveau (N) d'un nœud est la distance qui le sépare de la racine, c'est-à-dire, le nombre d'arête qu'il y a entre la racine et lui. Autrement, le niveau d'un nœud est son chemin sans la racine. N(x) = C(x) – 1. Tous les fils d'un nœud ont par conséquent le même niveau, on dit qu'ils sont frère. Tous les fils de parent d'un même niveau sont aussi du même niveau.
- La taille (T) d'un arbre est le nombre de nœud de celui-ci. Nous avons également T(Ø) = 0 ; arbre vide et T(x) = 1+ somme(fils(x))
- Degré (D) ou arité d'un nœud est le nombre de fils qu'il possède. L'arité d'un arbre est le maximum des arités de ses nœuds.
- Etiquette d'un nœud est la valeur (information) contenue dans celui-ci. Cette valeur peut être un réel, entier, enregistrement, tableau… Chaque nœud d'un arbre possède une seule étiquette.

Nous observons généralement 3 types d'arbres, les arbres quelconques, les arbres dégénérés et les arbres binaires.

7.7 Les Arbres

7.2 Arbre binaire

Un arbre binaire est un arbre de degré 2, c'est-à-dire, que tous les nœuds ont au plus 2 fils appelés fils gauche et fils droit. Le fils gauche et ses descendants constituent un sous arbre gauche (SAG) et le fils droit et ses descendants constituent un sous arbre droit (SAD).

7.7 Les Arbres

7.2.1 Implémentation des arbres binaires

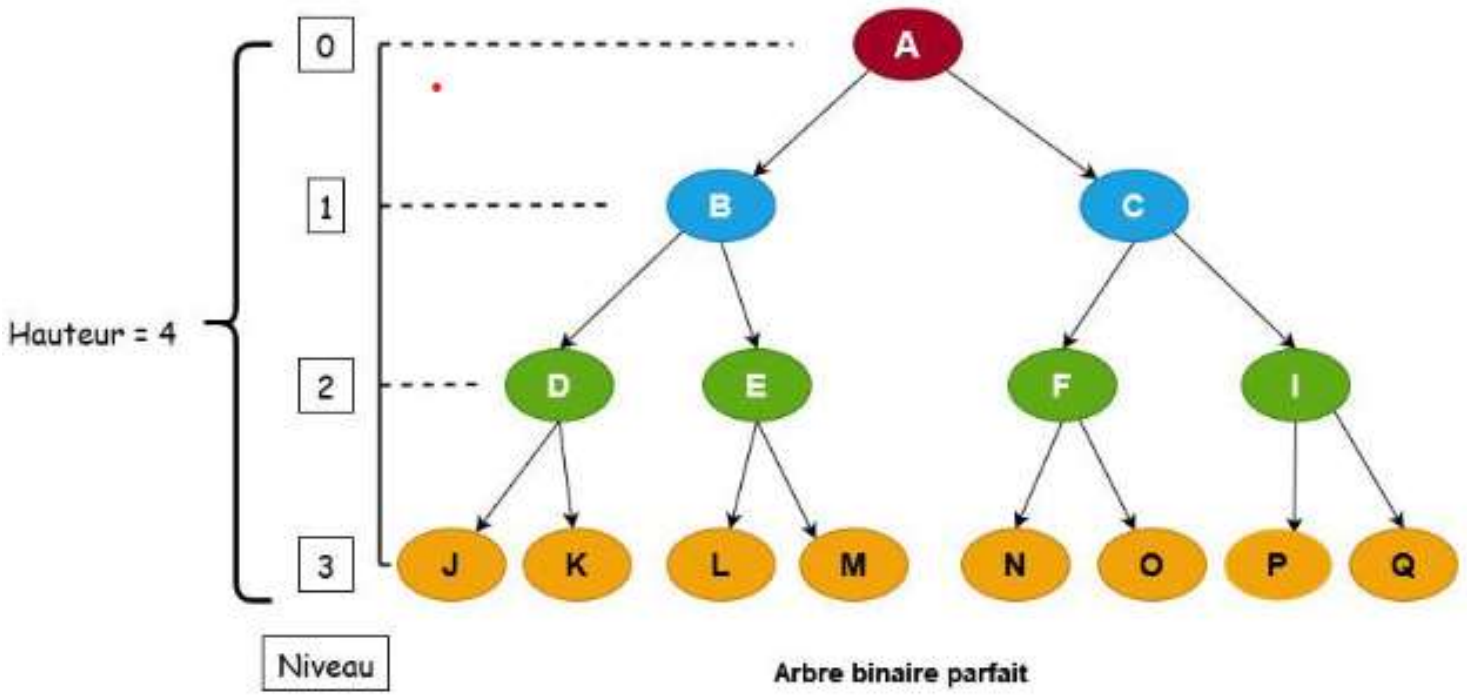


Figure 6 : Arbre binaire parfait

Un arbre binaire dont tous les nœuds internes ont exactement 2 fils sont dit arbres binaires parfaits (figure 6).

Propriétés des arbres binaires parfaits

Toutes les feuilles sont au même niveau. T = 2h – 1 ;
T = taille et h = hauteur

7.7 Les Arbres

7.2.2 Déclaration d'un arbre binaire

Les propriétés d'un arbre sont, le nom, le type de données, la valeur du nœud, le pointeur sur le fils gauche et le pointeur sur le fils droit. On déclare un arbre binaire à l'aide d'une structure de données enregistrement. Un arbre pointe sur la racine.

```
Type
Noeud = enregistrement
    valeur : type ;
    nœud *fg ;
    noeud *fd;
    Fin enregistrement
Nœud *arbre;
```

7.7 Les Arbres

7.2.3 Initialisation des arbres binaires

A l'initialisation, un arbre ne pointe sur rien.
Arbre = Null ;

7.2.4 Primitives des arbres binaires

Test de non vide

Un arbre est vide lorsqu'il n'a aucun nœud.

```
Estvide (arbre : A)
    Si A = Null alors
            Retourner vrai
    Sinon
            Retourner faux
Fin
```

7.7 Les Arbres

Test de feuille

Un nœud est une feuille si la valeur de ses pointeurs fg et fd sont null.

```
Estfeuille (noeud : A)
    Si A.fg = Null et A.fd = Null alors
            Retourner vrai
    Sinon
            Retourner faux
Fin
```

7.7 Les Arbres

Test de nœud interne

Un nœud est interne s'il n'est pas un nœud feuille.

Estinterne(A) = non estfeuille(A)

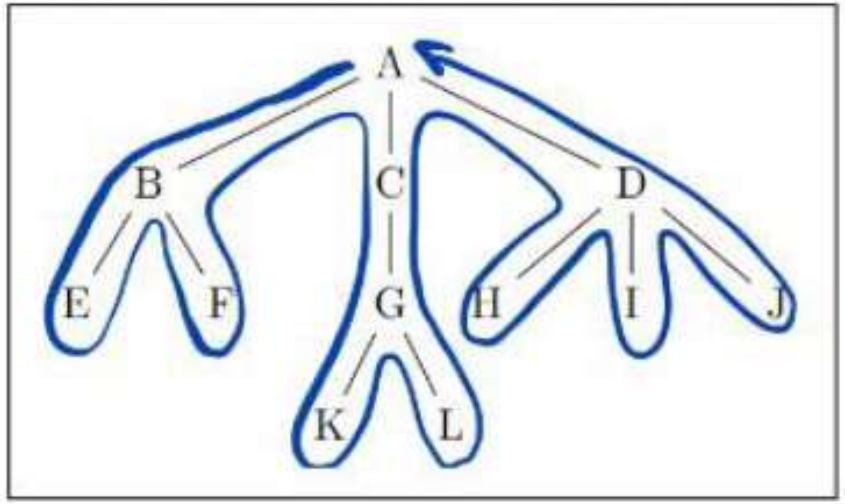
Recupere les fils d'un noeud

```
filsgauche (noeud : A)
    Retourner A.fg
Fin
filsdroit (noeud : A)
    Retourner A.fd
Fin
```

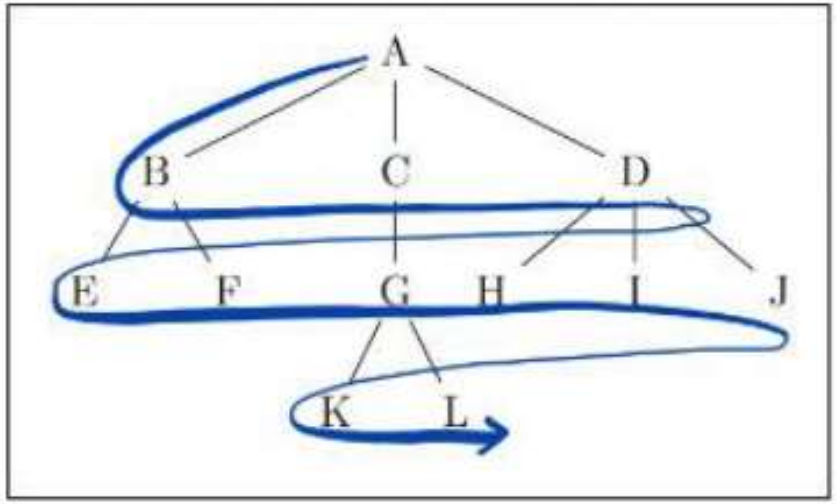
7.7 Les Arbres

7.2.5 Parcours d'un arbre

Parcourir un arbre, c'est visiter la totalité des nœuds de celui-ci, dans un certain ordre. Cette visite se fait de deux sortes, le parcours en profondeur et le parcours en largeur (figure 7).



Parcours en profondeur
le parcours se fait branche après branche de gauche à droite



Parcours en largeur
le parcours se fait en visitant tous les frères d'un même niveau avant de passer au niveau suivant de haut en bas

Figure 7 : Les différents types de parcours

7.7 Les Arbres

Parcours en profondeur

Il existe 3 méthodes de parcours en profondeur (figure 8) :

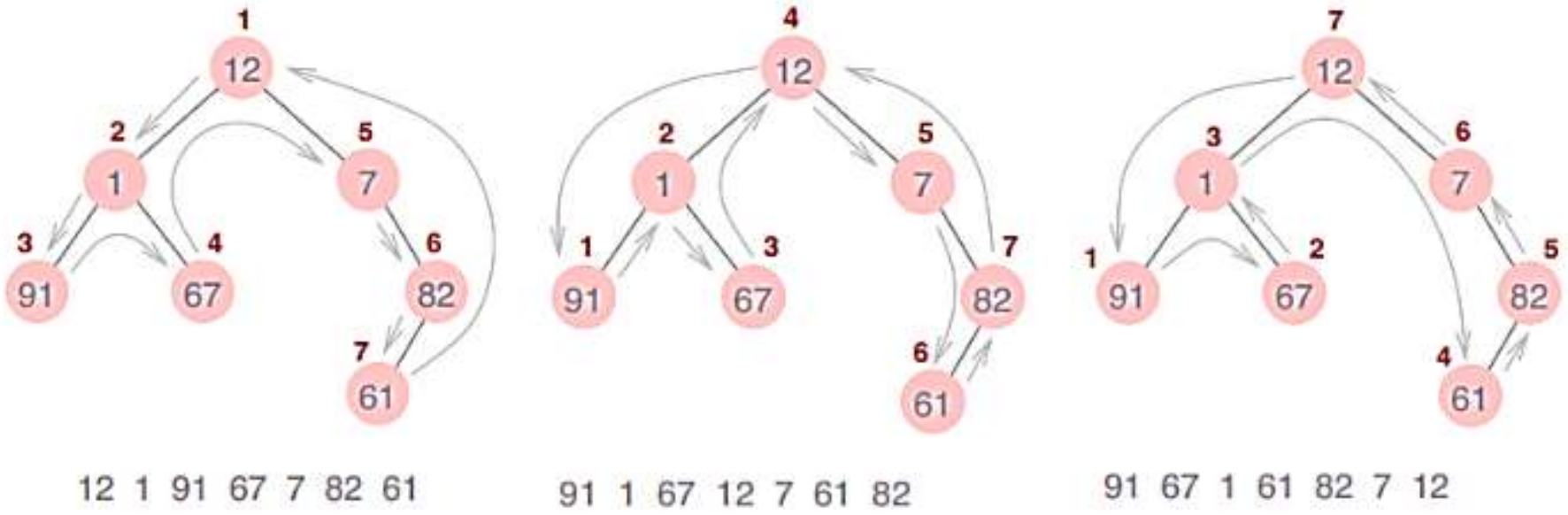


Figure 8 : Parcours Préfixé, Infixé et Postfixé respectivement de gauche à droite

7.7 Les Arbres

– Le parcours préfixé descend sur les fils gauches des sous arbres jusqu'au dernier avant d'attaquer le fils droit en remontant.
– Le parcours infixé se fait de bas en haut et de gauche à droite sous arbre après sous arbre.
– Le parcours postfixé se fait de bas en haut et de gauche à droite niveau après niveau.

Parcours en largeur

Le parcours en largeur se fait en visitant tous les niveaux de la racine au dernier niveau. On commence par la racine puis on descend au niveau inférieur en visitant tous les nœuds de gauche à droite

7.7 Les Arbres

7.2.6 Insertion d'un nœud dans un arbre binaire

On insert le nouveau nœud chez le premier nœud non complet ou la première feuille en partant du haut vers le bas (parcours en largeur).

FIN DU COURS