Les Arbres

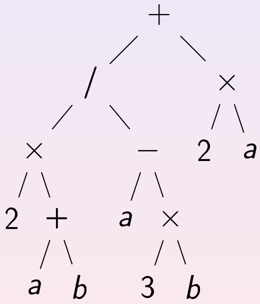
Sandrine Vial
sandrine.vial@prism.uvsq.fr

Janvier 2012

Les arbres

Une des structures les plus importantes et les plus utilisées en informatique

- Arbres généalogiques
- Arbres de classification
- Arbres d'expression



Représentation de l'expression

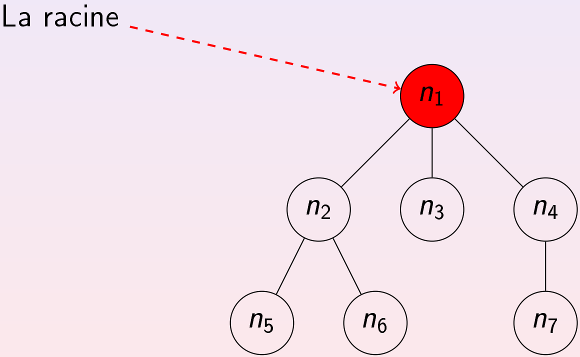
$$(2 \times (a + b))/(x - 3 \times b) + 2 \times x$$

Terminologie

- Un **arbre** : un ensemble de nœuds reliés entre eux par des arêtes.
- Trois propriétés pour les arbres **enracinés** :
  1. Il existe un nœud particulier nommé **racine**. Tout nœud c autre que la racine est relié par une arête à un nœud p appelé **père** de c.
  2. Un arbre est **connexe**.
  3. Un arbre est **sans cycle.**

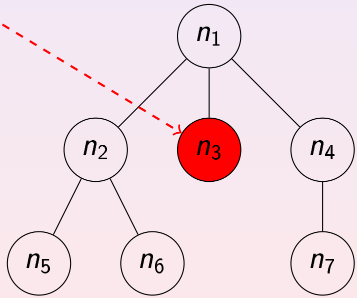
Terminologie

- Un nœud peut avoir 0 ou plusieurs fils.
- Un nœud a exactement un père.



- Un nœud peut avoir 0 ou plusieurs fils.
- Un nœud a exactement un père.

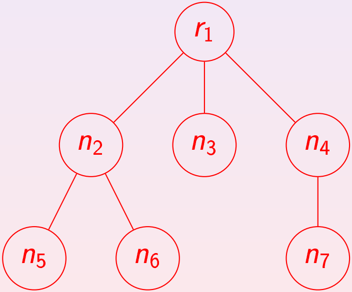
La racine



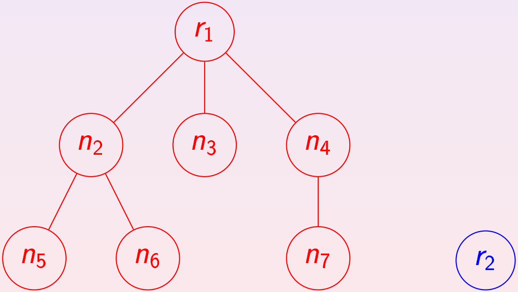
Définition récursive

- **Base :**
  - Un nœud unique n est un arbre
  - n est la racine de cet arbre.
- **Récurrence :**
  - Soit r un nouveau nœud
  - $T_1, T_2, \ldots, t_k$ sont des arbres ayant pour racine $r_1, r_2, \ldots, r_k$.
  - Création d'un nouvel arbre ayant pour racine r et on ajoute une arête entre r et r_1 r et r_2, ..., r et r_k.

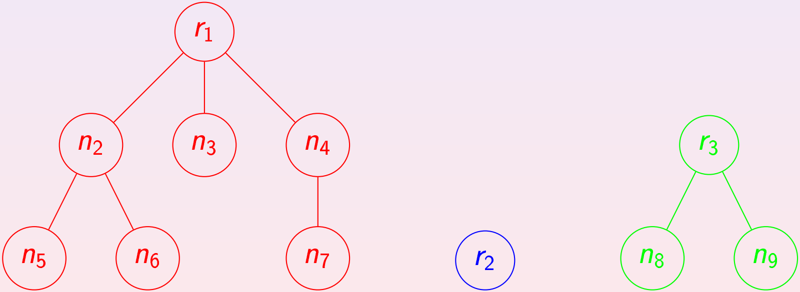
Définition récursive : un exemple



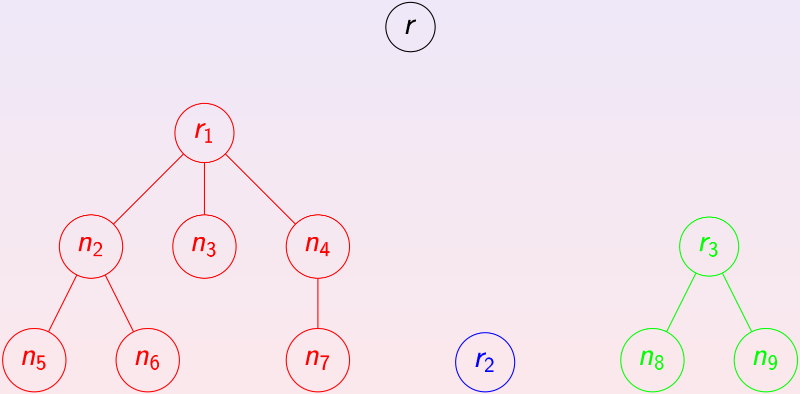
Définition récursive : un exemple



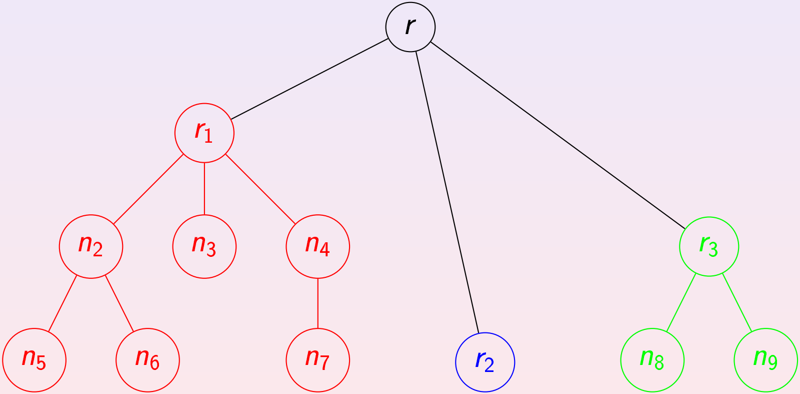
Définition récursive : un exemple



Définition récursive : un exemple



Définition récursive : un exemple



Chemins, ancêtres, descendants, . . .

- Les **ancêtres** d un nœud : *Nœuds trouvés sur le* ***chemin unique*** *entre ce nœud et la racine.*
- Le nœud d est un **descendant** de a si et seulement si a est un ancêtre de d.
- **Longueur** d'un chemin = nombre d'arêtes parcourues.

Généalogie

- La racine est un ancêtre de tous les nœuds.
- Chaque nœud est un descendant de la racine.
- Les nœuds ayant le même père = **frères**.
- Un nœud n et tous ses descendants = **sous-arbre**

Feuilles et nœuds intérieurs

- Une `feuille` est un nœud qui n'a pas de fils
- Un nœud intérieur est un nœud qui a au moins 1 fils.
- Tout nœud de l'arbre est :
  - Soit une feuille
  - Soit un nœud intérieur

Mesures sur les arbres

- **Taille** de l'arbre T, notée *taille(T)* = nombre de nœuds.
- **Nombre de feuilles** noté *nf(T)*.
- **Longueur de cheminement** de l'arbre T, notée *LC(T)* = somme des longueurs de tous les chemins issus de la racine.

$$LC(T) = \sum_{x \text{ nœud de } T} h(x).$$

- **Longueur de cheminement externe** de l'arbre T, notée *LCE(T)* = somme des longueurs de tous les chemins aboutissant à une feuille issus de la racine.

$$LCE(T) = \sum_{x \text{ feuille de } T} h(x).$$

Hauteur

- La **hauteur d'un nœud** n, notée $h(n)$, est la longueur du chemin depuis la racine jusqu'à n.
- La **hauteur de l'arbre** T, notée $h(T)$:

$$h(T) = \max_{x \text{ nœud de l'arbre}} h(x)$$

Mesures

- **Hauteur moyenne** de l'arbre T, notée $HM(T)$= moyenne des hauteurs de tous les nœuds.

$$HM(T) = \frac{LC(T)}{taille(T)}$$

- **Hauteur moyenne externe** de l'arbre T, notée $HME(T)$ = moyenne des longueurs de tous les chemins issus de la racine et se terminant par une feuille.

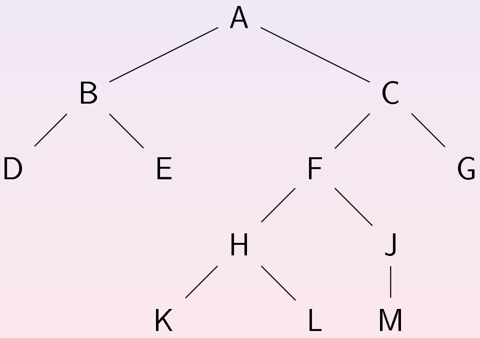
$$HME(T) = \frac{LCE(T)}{nf(T)}$$

Les arbres binaires

1. Etude d'une classe particulière d'arbres
2. Propriétés
3. Algorithmes

Les arbres binaires

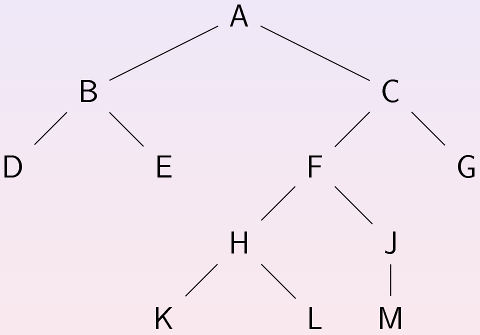
Tous les nœuds d'un arbre binaire ont 0, 1 ou 2 fils.



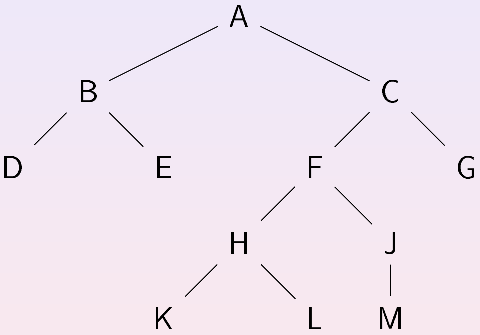
Les arbres binaires : Définitions

- **Fils gauche** de n = racine du sous-arbre gauche de n.
- **Fils droit** de n = racine du sous-arbre droit de n.
- **Bord gauche** de l'arbre = le chemin depuis la racine en ne suivant que des fils gauche.
- **Bord droit** de l'arbre = le chemin depuis la racine en ne suivant que des fils droits.

Exemple : arbre binaire



Exemple : arbre binaire



Taille de l'arbre :

Longueur de cheminement externe :

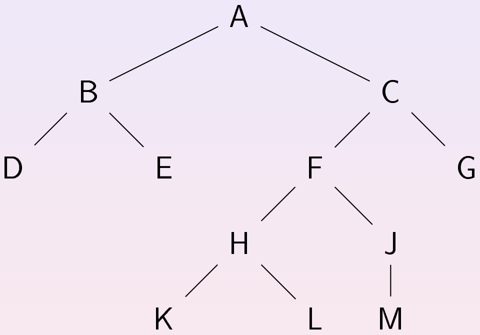
Hauteur moyenne :

Nombre de feuilles :

Longueur de cheminement :

Hauteur moyenne externe :

Exemple : arbre binaire



Taille de l'arbre : 12
Longueur de cheminement externe : 18
Hauteur moyenne : 2.33

Nombre de feuilles :6
Longueur de cheminement : 28
Hauteur moyenne externe : 3

Quelques arbres binaires particuliers

1. Arbre binaire *filiforme*
2. Arbre binaire *complet*
    - 1 nœud à la hauteur 0
    - 2 nœuds à la hauteur 1
    - 4 nœuds à la hauteur 2
    - ...
    - 2^h nœuds à la hauteur h.
    - Nombre total de nœuds d'un arbre de hauteur h :

$$2^0 + 2^1 + 2^2 + \cdots + 2^h = 2^{h+1} - 1$$

Quelques arbres binaires particuliers

1. Arbre binaire *parfait* :
   - Tous les niveaux sont remplis sauf le dernier.
   - Les feuilles sont le plus à gauche possible.
2. Arbre binaire *localement complet* : chaque nœud a 0 ou 2 fils.

Propriétés sur les arbres (1)

Lemme

$$h(T) \leq taille(T) - 1$$

Idée de Preuve

Egalité obtenue pour un arbre filiforme.

Propriétés sur les arbres (2)

Lemme

Pour tout arbre binaire T de taille n et de hauteur h on a :

$$\lfloor \log_2 n \rfloor \leq h \leq n - 1$$

Idée de Preuve

- Arbre filiforme : arbre de hauteur h ayant le plus petit nombre de nœuds : $n = h + 1$ (seconde inégalité).
- Arbre complet : arbre de hauteur h ayant le plus grand nombre de nœuds : $n = 2^{h+1} - 1$ (première inégalité).

Propriétés sur les arbres

Corollaire

Tout arbre binaire non vide T ayant f feuilles a une hauteur $h(T)$ supérieure ou égale à $\lceil \log_2 f \rceil$.

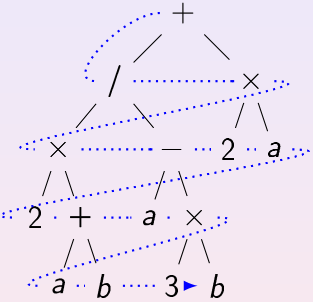
Lemme

Un arbre binaire localement complet ayant n nœuds internes a $(n+1)$ feuilles.

Exploration

- Pas aussi simple que dans le cas des listes.
- Pas d'ordre naturel.
- Deux types de parcours :
  - En largeur d'abord
  - En profondeur d'abord

Parcours en largeur d'abord



Ordre d'évaluation des nœuds :

$$+ / \times \times - 2\ a\ 2 + a \times a\ b\ 3\ b$$

Type de données

Mise en œuvre chaînée

```
Enregistrement Nœud {
    Val      : entier;
    Gauche : ↑ Nœud;
    Droit : ↑ Nœud;
}
```

Parcours en largeur d'abord

Algorithme 1 Parcours en largeur d'un arbre binaire

```
ParcoursEnLargeur(r : Nœud)
▷ Entrée : r (la racine d'un arbre)
▷ Sortie : traitement de tous les nœuds de l'arbre enraciné en r
▷ Variables locales :
    ce_niveau, niveau_inférieur : File ;
    o : Nœud ;
Début
    ce_niveau ← { r } ;
    tant que (ce_niveau est non vide)  faire
        niveau_inférieur = { } ;
        pour chaque nœud o de ce_niveau  faire
            traiter o ;
            niveau_inferieur ← niveau_inférieur ∪ enfants de o.
        fin pour
        ce_niveau ← niveau_inférieur ;
    fin tant que
Fin
```

Parcours en profondeur d'abord

Ordre d'évaluation des nœuds : ça dépend ....

Parcours en profondeur d'abord

- Parcours infixe :

$$2 \times a + b / a - 3 \times b + 2 \times a$$

- Parcours préfixe :

$$+ / \times 2 + a \ b - a \times 3 \ b \times 2 \ a$$

- Parcours postfixe

$$2 \ a \ b + \times a \ 3 \ b \times - / 2 \ a \times +$$

Algorithme 2 Parcours en profondeur d'un arbre binaire

```
ParcoursEnProfondeur(r : Nœud)
▷ Entrée : r (la racine d'un arbre)
▷ Sortie : traitement de tous les nœuds de l'arbre enraciné en r
Début
    si r = ∅
        traitement de l'arbre vide
    sinon
        traitement_prefixe(r) ;
        ParcoursEnProfondeur(r.Gauche) ;
        traitement_infixe(r) ;
        ParcoursEnProfondeur(r.Droit) ;
        traitement_postfixe(r) ;
    fin si
Fin
```

Arbres généraux et forêts

- **Arbre général** : arbre où les nœuds peuvent avoir un nombre quelconque de fils.
- **Forêt** : collection d'arbres en nombre quelconques

Représentation

- Tableau de fils
- Fils ainé et frère droit (bijection avec les arbres bianires).

Arbres généraux : affichage préfixé

Algorithme 3 Parcours prefixe d'un arbre général

```
Prefixe(r : Nœud)
▷ Entrée : r (la racine d'un arbre)
▷ Sortie : traitement de tous les nœuds de l'arbre enraciné en r
▷ Variable locale :
    t : Nœud ;
Début
    traiter le nœud r
    t ← r.FilsAine ;
    tant que t ≠ NIL faire
        Prefixe(t) ;
        t ← t.FrereDroit ;
    Ftque
Fin
```

Arbres généraux : affichage postfixé

Algorithme 4 Parcours postfixe d'un arbre général

```
Postfixe(r : Nœud)
▷ Entrée : r (la racine d'un arbre)
▷ Sortie : traitement de tous les nœuds de l'arbre enraciné en r
▷ Variable locale :
    t : Nœud ;
Début
    t ← r.FilsAine ;
    tant que t ≠ NIL  faire
        Postfixe(t) ;
        t ← t.FrereDroit ;
    Ftque
    traiter le nœud r
Fin
```