Algorithmique
Arbres binaires : parcours

Plusieurs manières de parcourir (explorer tous les nœuds) un arbre binaire :

— Le *parcours profondeur* (*depth-first traversal* en anglais), par nature récursif, qui consiste à avancer le plus loin possible.

— Le *parcours largeur* (*breadth-first traversal* en anglais), itératif, qui consiste à passer en revue tous les nœuds de l'arbre niveau par niveau.

1 Le parcours profondeur

Un algorithme récursif

La manière la plus simple d'écrire ce parcours : Il suit la structure récursive de l'arbre :

— L'arbre vide sera un cas d'arrêt.

— Si l'arbre est non vide, les deux sous-arbres sont parcourus récursivement.

Si c'est d'abord le sous-arbre gauche qui est parcouru, on parle de parcours *main gauche* (celui présenté ici). Le parcours *main droite* s'obtient en parcourant d'abord le sous-arbre droit avant le *gauche*.

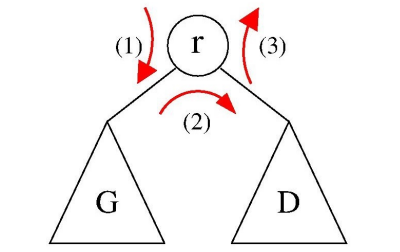
Ordres induits

Lors du parcours en profondeur, chaque nœud est rencontré trois fois. Ce sont les trois *ordres induits* :

(1) avant de descendre sur le sous arbre-gauche : ordre **préfixe**,

(2) en remontant de la gauche, avant de descendre à droite : ordre **infixe** (ou **symétrique**),

(3) en remontant de la droite : ordre **suffixe** (ou **postfixe**).



(1) : préfixe (2) : infixe (3) : suffixe

```
procedure profondeur(arbrebinaire B)
debut
    si B = arbrevide alors
        /* terminaison */
    sinon
        /* traitement préfixe (1) */
        profondeur(g(B))
        /* traitement infixe (2) */
        profondeur(d(B))
        /* traitement suffixe (3) */
    fin si
fin
```

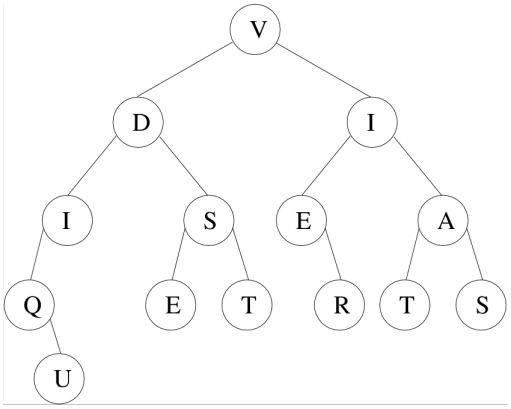
"À la main"

Le parcours en profondeur main gauche consiste à descendre dans l'arbre à gauche le plus loin possible. Lorsqu'on ne peut plus descendre, on remonte d'un niveau : si on vient de la gauche, on descend à droite, sinon on remonte encore... Et on recommence...

Préfixe : V D I Q U S E T I E R A T S

Infixe : Q U I D E S T V E R I T A S

Suffixe : U Q I E T S D R E T S A I V



2 Le parcours largeur

L'algorithme classique utilise une *file* :

- on enfile la racine de l'arbre
- tant que la file n'est pas vide :
  - on récupère et on défile le premier élément de la file
  - on enfile chacun de ses fils (s'ils existent), d'abord le gauche puis le droit.

En général, les nœuds sont traités au moment où ils sortent de la file (on peut dans certains cas effectuer un traitement sur les racines des sous-arbres enfilés, mais c'est plus rare).

```
procedure parcours_largeur(arbrebinaire B)
    variables
        file    f
debut
    si B <> arbrevide alors
        f ← filevide
        f ← enfiler (B,f)
        tant que non estvide(f) faire
            B ← premier(f)
            f ← defiler (f)
             /* traitement contenu(racine(B)) */
            si g(B) <> arbrevide alors
                f ← enfiler(g(B), f)
            fin si
            si d(B) <> arbrevide alors
                f ← enfiler(d(B), f)
            fin si
        fin tant que
    fin si
fin
```

Parcours largeur :
V D I I S E A Q E T R T S U