Université Paris 8
Christiane Frougny

Licence d'Informatique — L2
Structures de données

Arbres généraux

L'arbre vide est noté Λ.
Soit A un arbre non vide de racine r ayant k sous-arbres A_1, A_2, ..., A_k.

Parcours préfixe P
$P(\Lambda) = \Lambda$
$P(A) = rP(A_1)P(A_2)\cdots P(A_k)$

Parcours interne I
$I(\Lambda) = \Lambda$
$I(A) = I(A_1)rI(A_2)\cdots I(A_k)$

Parcours suffixe S
$S(\Lambda) = \Lambda$
$S(A) = S(A_1)S(A_2)\cdots S(A_k)r$

Remarque : si l'arbre est réduit à une feuille f, $P(f) = I(f) = S(f) = f$.

Implantation par chaînage FilsGauche - FrereDroit

Un nœud est une cellule contenant un champ FilsGauche, un champ Etiquette et un champ FrereDroit. Un arbre est un pointeur sur une cellule de ce type.

Si n est un nœud, $n \to FilsGauche$ pointe sur le fils aîné de n, $n \to FrereDroit$ pointe sur le frère de n qui suit immédiatement n, et $n \to Etiquette$ contient l'étiquette (ou la valeur) de n.

Algorithme 1 Fonction ParcoursPrefixe(n)

```
si n ≠ NULL alors
  Afficher(n → Etiquette);
  p := n → FilsGauche;
  tant que p ≠ NULL faire
    ParcoursPrefixe(p);
    p := p → FrereDroit
  fin tant que
fin si
```

Appel avec ParcoursPrefixe(A).

Parcours par niveau

On utilise une queue de pointeurs Q, et les fonctions définies dans le cours sur les queues.

Algorithme 2 Fonction ParcoursNiveau(A)

```
si A ≠ NULL alors
  Initialiser(Q);
  AjouterQueue(A, Q)
  tant que non Vide(Q) faire
    p := Tete(Q);
    EnlQueue(Q);
    Afficher(p → Etiquette);
    p := p → FilsGauche;
    tant que p ≠ NULL faire
      AjouterQueue(p, Q);
      p := p → FrereDroit
    fin tant que
  fin tant que
fin si
```