

1) Arbres et occurrences ...

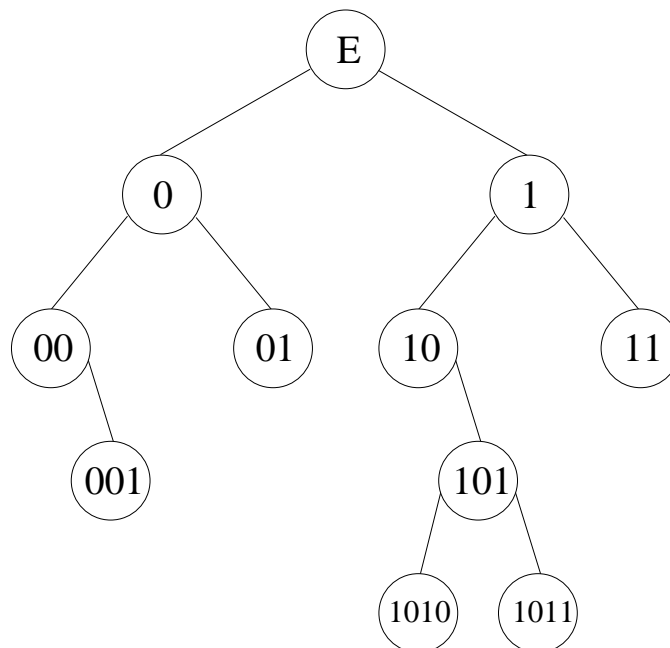
Soit l'arbre B défini, sous forme d'occurrences et représenté à l'aide d'une chaîne de caractères de la manière suivante:

$$B = "E,0,1,00,01,10,11,001,101,1010,1011"$$

1) Sans représenter l'arbre B , pour déterminer :

- $T(B)$, il suffit de compter les virgules présentes dans la chaîne et d'y ajouter 1
- $H(B)$, il suffit de compter le nombre de 0 et de 1 de la dernière occurrence de la chaîne ((1011) dans l'exemple)
- Qu'un nœud n'est pas une feuille, il faut que son occurrence n'en préfixe pas une autre. Il suffit donc de parcourir le reste de la chaîne en vérifiant que l'occurrence de ce nœud ne se trouve pas au début des occurrences suivantes.
- $LC(B)$, il suffit de compter le nombre total de 0 et de 1 se trouvant dans la chaîne.
- $HM(B)$, il suffit simplement d'effectuer l'opération.

2) La représentation de l'arbre est la suivante :



2) Hiérarchie et occurrences ...

Spécifications :

La procédure *occurrence* (*entier no*, *chaîne occ*, *entier hauteur*) donne dans *occ* l'occurrence du nœud de numéro hiérarchique *no*, ainsi que sa hauteur dans l'entier *hauteur*.

On supposera qu'il n'y a pas de problème de limite de taille de la chaîne *occ*.

1) Le principe de la fonction ***occurrence*** est le suivant :

L'entier *no* est divisé "entièrement" par deux jusqu'à ce qu'il atteigne 1. Au départ, la hauteur est nulle et la chaîne *occ* est vide. A chaque division, la hauteur est augmentée de 1 et le reste de la division est, après transformation en chaîne "0" ou "1", concaténé avec la chaîne *occ*, puis l'entier *no* est divisé par deux.

2) Algorithme :

Plutôt que de fournir des solutions en C, Caml ou Pascal, nous avons préféré fournir une solution algorithmique en pseudo-langage facilement implémentable dans un de ces langages.

Cette solution ne tient pas compte de l'aspect débranchant des Retourne et positionne à chaque fois les sinon. C'est bien évidemment optimisable.

Algorithme procedure occurrence

Parametres locaux

entier no

Parametres globaux

chaîne occ

entier hauteur

Variables

chaîne reste

Debut

hauteur ← 0

si no = 1 **alors**

occ ← "ε"

sinon

occ ← ""

tant que no <> 1 **faire**

hauteur ← hauteur + 1

si no mod 2 = 0 **alors** /* calcul du reste sous format chaîne */

reste ← "0"

sinon

reste ← "1"

fin si

occ ← reste + occ /* concaténation en ordre inverse */

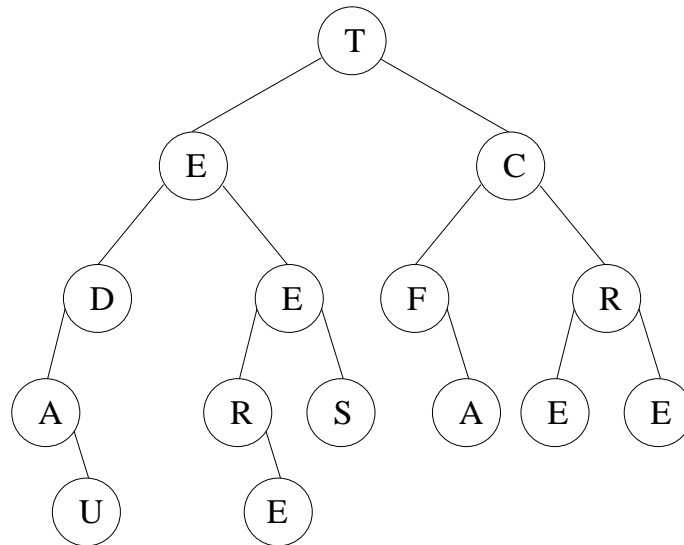
fin tant que

fin si

fin algorithme procedure occurrence

3) Arbres, Hiérarchie et occurrences ...

1) Voici la représentation de l'arbre donné sous sa forme hiérarchique :



2) Le parcours infixe de cet arbre donne :

A U D E R E E S T F A C E R E (Audere est facere, « Oser c'est faire »)
devise du club de foot de Tottenham

3) La représentation sous forme d'occurrences de cet arbre est la suivante :

B = "E,0,1,00,01,10,11,000,010,011,101,110,111,0001,0101"