UNIVERSITÉ DE STRASBOURG

Département d'Informatique
LICENCE 2ème année
Mardi 22 octobre 2013

CC de Structures de Données et Algorithmes 2

Durée : 1h30. Aucun document autorisé.

I Quelques rappels sur les arbres binaires et les arbres parfaits

- arbre binaire (sorte `arbrebin`) :
  - constructeurs :
    - `Vide()`,
    - `Enrac(g,r,d)`.
  - autres opérations de base :
    - `racine(a)` (sélecteur),
    - `arbreg(a)` (sélecteur),
    - `arbred(a)` (sélecteur),
    - `vide(a)` (observateur).
- arbre parfait (sorte `arbrepa`) (arbre dont les étiquettes sont ajoutés en suivant un parcours en largeur) :
  - constructeurs :
    - `Vide_pa()`,
    - `Ajout(a,x)`.
  - autres opérations de base (où k est un entier entre 0 et $n-1$ qui désigne une adresse dans l'arbre et n est le nombre de nœuds) :
    - `enleve(a)` : enlève le noeud d'addresse $n-1$ (transformateur),
    - `remplace(a,x,k)` (transformateur),
    - `nb_noeuds(a)` (observateur),
    - `etiquette(a,k)` (sélecteur),
    - `echange(a,k1,k2)` (transformateur).

II Spécification des arbres *partiellement ordonnés*

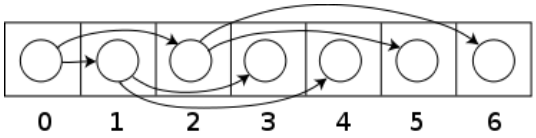
Un arbre est *partiellement ordonné* s'il est parfait et tel que chacun de ses nœuds internes possède une étiquette inférieure ou égale à celles de ses fils. On considère ici des arbres partiellement ordonnés dont les étiquettes sont des entiers.

1. Dessiner un arbre partiellement ordonné avec 10 nœuds étiquetés par les entiers de 1 à 10.

2. Spécifier algébriquement les opérations suivantes en donnant pour chacune d'elles son profil, les préconditions et les axiomes :
   (a) `minpo` : étiquette la plus petite d'un arbre partiellement ordonné.
   (b) `arbrebin_of_arbrepa` : conversion d'un arbre parfait en un arbre binaire en créant un arbre binaire qui a la même structure et les mêmes étiquettes que l'arbre parfait (indication : se servir d'une fonction auxiliaire qui prend une adresse k en argument).
   (c) `estpo` : test si un arbre parfait est partiellement ordonné (se servir de la question précédente).
   (d) `insere` : insertion d'un nouveau nœud étiqueté dans un arbre partiellement ordonné de telle manière que l'arbre reste partiellement ordonné (indication : se servir d'une fonction auxiliare `monte(a,k)` qui fait remonter dans l'arbre une étiquette trop petite en l'échangeant avec celle du noeud père).

III Implémentation des arbres parfaits en utilisant un tableau (tas)

On choisit de représenter les arbres parfaits de taille bornée par N par un tableau où les étiquettes de l'arbre sont contiguës en mémoire (cf. figure ci-dessous).



Une telle représentation s'appelle un tas. Le type des arbres parfaits représentés de cette façon est défini comme suit en C :

```
#define N 100
typedef int S;
struct tas {
  int nb_noeuds;
  S *tab;
};
typedef struct tas *arbrepa;
```

Ecrire en C les définitions des fonctions correspondantes à chacun des opérations sur les arbres parfaits : `Vide_pa()`, `Ajout(a,x)`, `enleve(a)`, `remplace(a,x,k)`, `nb_noeuds(a)`, `etiquette(a,k)` et `echange(a,k1,k2)`.