

Option Informatique

Exercice supplémentaire : Représentations contigües d'un arbre

Sujet

On considère un arbre binaire au sens large T dont les sommets sont des entiers. On se propose d'étudier plusieurs implémentations d'un tel arbre et des opérations élémentaires sur les arbres. Dans la suite, on considère l'arbre T de la figure 1.

1 Utilisation d'un tableau

Chaque sommet de l'arbre est représenté par un enregistrement contenant trois champs : un champ *gauche* indiquant l'indice du fils gauche dans un tableau, un champ *droit* indiquant l'indice du fils droit et un champ *info* indiquant la valeur associée au sommet.

La valeur *nil* est associée à -1 . On définit ainsi le type :

```
type element = { droit : int ; info : int ; gauche : int };;
```

L'arbre T est représenté à l'aide d'un tableau et d'une variable *racine* indiquant l'indice de la racine dans le tableau. On définit le type :

```
type arbre = { racine : int ; sommet : element vect };;
```

Ainsi l'arbre précédent peut être représenté de la façon suivante :

```
let T = {racine = 0 ; sommet=[|{droit = 3 ; info = 12 ; gauche = 2} ; ...  
; {droit = -1 ; info = 13 ; gauche = -1}|]}
```

où `T.sommet` est le tableau représenté en figure 2.

Écrire les opérations élémentaires `est_terminal`, `racine`, `fils_gauche`, `fils_droit`,
`cons : arbre → int → arbre → arbre` avec cette implémentation d'un arbre.

2 Utilisation d'une liste

Une deuxième représentation consiste à ranger dans une liste les sommets de l'arbre suivant un parcours préfixé. Un sommet de l'arbre est représenté par un enregistrement contenant trois champs : un champ *gauche* indiquant la présence ou non d'un fils gauche, un champ *droit* indiquant la présence ou non d'un fils droit et un champ *info* indiquant la valeur associée au sommet.

```
type element = {droit : bool ; info : int ; gauche : bool};;
```

```
type arbre == element list;;
```

Ainsi, l'arbre T peut être représenté de la façon suivante :

```
let T = [{droit = true ; info = 12 ; gauche = true}; ...
```

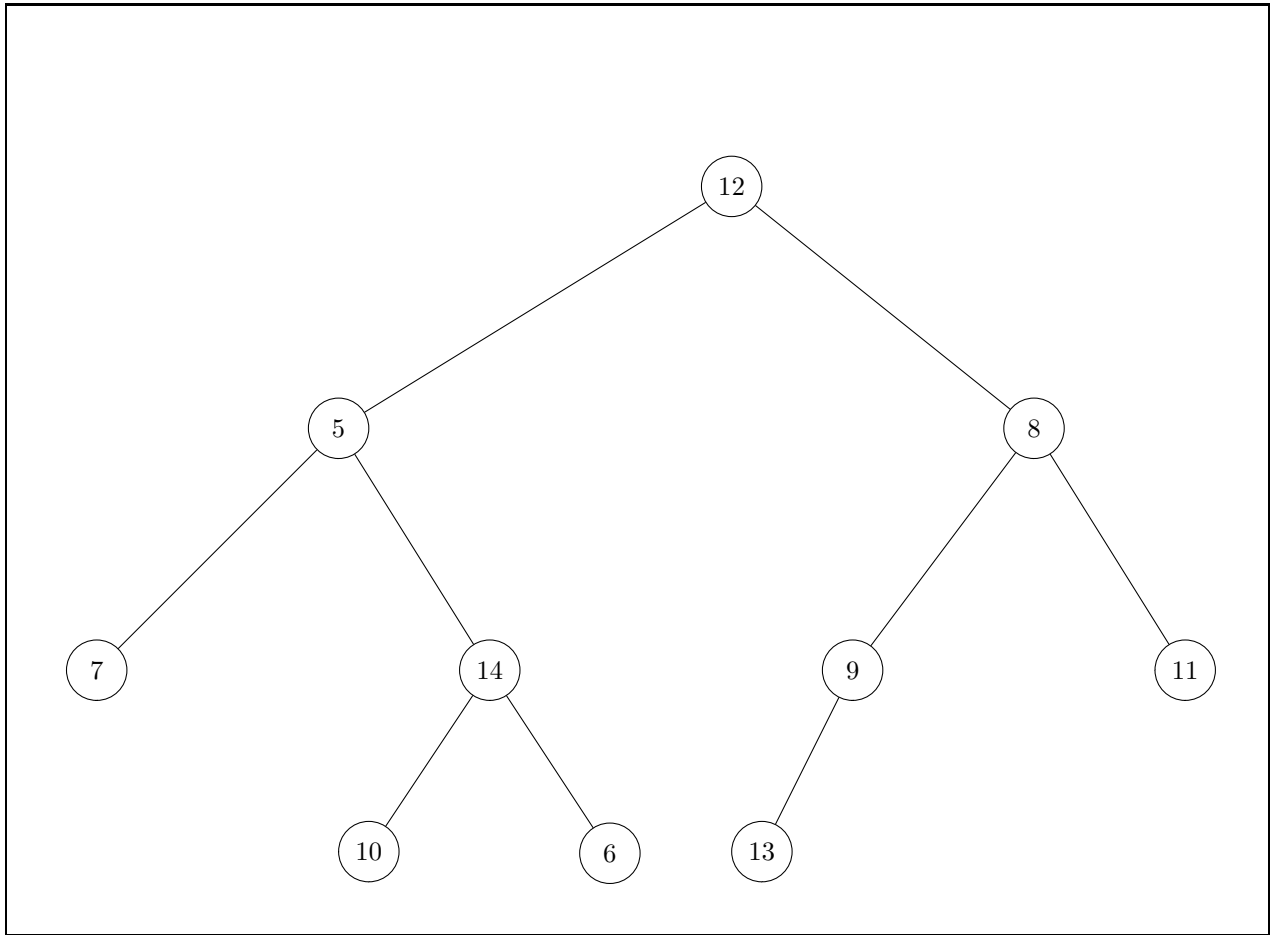


FIG. 1 – L'arbre T .

```
; {droit = false; info = 11; gauche = false}];;
```

où la liste est représentée en figure 3.

Écrire les opérations élémentaires `est_terminal`, `racine`, `fils_gauche`, `fils_droit`,
`cons` : `arbre` → `int` → `arbre` → `arbre` avec cette implémentation d'un arbre.

| | | | | | | | | | | |
|--------|----|---|---|----|----|----|----|----|----|----|
| gauche | 2 | 4 | 6 | -1 | 8 | 10 | -1 | -1 | -1 | -1 |
| info | 12 | 5 | 8 | 7 | 14 | 9 | 11 | 10 | 6 | 13 |
| droite | 3 | 5 | 7 | -1 | 9 | -1 | -1 | -1 | -1 | -1 |

FIG. 2 – Une représentation de T.sommet

| | | | | | | | | | | |
|--------|----|---|---|----|----|---|---|---|----|----|
| gauche | v | v | f | v | f | f | v | v | f | f |
| info | 12 | 5 | 7 | 14 | 10 | 6 | 8 | 9 | 13 | 11 |
| droite | v | v | f | v | f | f | v | f | f | f |

FIG. 3 – Une représentation de T à l'aide d'une liste.