Algorithmique et structures de données

CM 7 – Arbres généraux et forêts

Plan du CM 7

Plan du CM 7

Codage d'une forêt

Algorithmes sur les forêts avec ce codage

Parcours des nœuds sur une forêt

Arbre général

Schéma d'induction

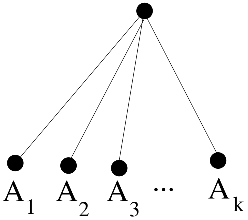
Construction inductive des arbres généraux

(i) l'arbre vide $\emptyset$ est un arbre général.

(ii) L'arbre constitué d'une racine $\cdot$ et d'une suite finie de sous-arbres généraux non vides $A_1, \ldots, A_k$ est un arbre général.

La suite finie peut être vide, on obtient ainsi l'arbre racine $\bullet$ (arbre réduit à une racine).

Schéma général



Arbre général

Schéma d'induction

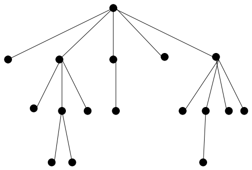
Construction inductive des arbres généraux

(i) l'arbre vide $\emptyset$ est un arbre général.

(ii) L'arbre constitué d'une racine $\cdot$ et d'une suite finie de sous-arbres généraux non vides $A_1, \ldots, A_k$ est un arbre général.

La suite finie peut être vide, on obtient ainsi l'arbre racine $\bullet$ (arbre réduit à une racine).

Exemple d'arbre général

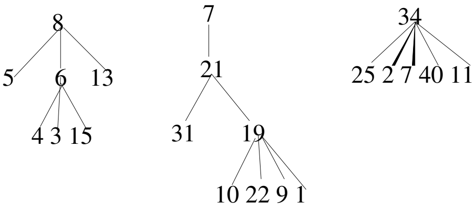


Forêt

Définition

- une forêt est une suite finie d'arbres généraux
- on tient compte de l'ordre entre les arbres généraux (ce n'est pas un ensemble)

Exemple de forêt



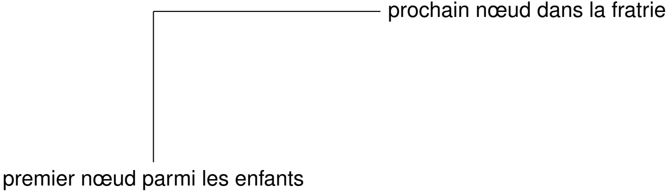
Codage d'une forêt premier enfant–prochain de la fratrie

Structure

```
structure noeudF{
   premierEnfant : pointeur sur noeudF
   valeur : entier
   prochainFratrie : pointeur sur noeudF

}
```

Schéma général



Codage d'une forêt premier enfant–prochain de la fratrie

Structure

```
structure noeudF{
   premierEnfant : pointeur sur noeudF
   valeur : entier
   prochainFratrie : pointeur sur noeudF
}
```

Au niveau de la racine

- la racine de la forêt est la racine du premier arbre général
- le prochain nœud de la fratrie est la racine du prochain arbre général

racine du prochain arbre général

premier nœud parmi les enfants

Codage d'une forêt premier enfant–prochain de la fratrie

Structure

```
structure noeudF{
   premierEnfant : pointeur sur noeudF
   valeur : entier
   prochainFratrie : pointeur sur noeudF

}
```

Type forêt et arbreGeneral

```
type foret = pointeur sur noeudF
type arbreGeneral = pointeur sur noeudF
```

Différence entre arbre général et forêt

- un arbre général A vérifie $A \rightarrow prochainFratrie = None$.

Codage d'une forêt premier enfant–prochain de la fratrie

Différents types de pointeurs

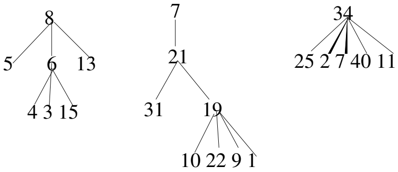
- le pointeur premierEnfant pointe sur le premier enfant du nœud.
  Il vaut donc None si le nœud n'a pas d'enfant.

- le pointeur prochainFratrie pointe sur le prochain de la fratrie du nœud.
  Il vaut donc None si c'est le dernier de la fratrie.

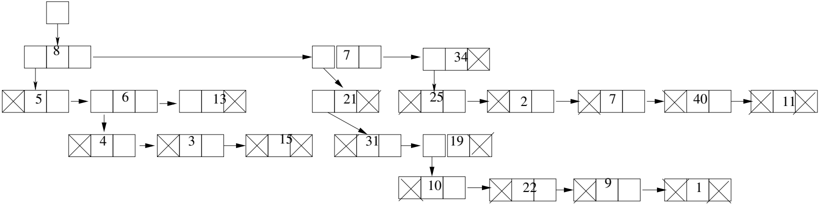
- pour le premier niveau, prochainFratrie pointe sur le prochain arbre.

Codage d'une forêt premier enfant–prochain de la fratrie

Exemple de forêt



Codage correspondant



Plan du CM 7

Codage d'une forêt

Algorithmes sur les forêts avec ce codage

Parcours des nœuds sur une forêt

Nombre de nœuds

Le calcul du nombre de nœuds requiert le même algorithme que pour un arbre binaire.

Sur un arbre binaire

```
nombreNoeuds(A : arbre) : entier
   si A = None alors
      retourner 0
   retourner 1 + nombreNoeuds(A->gauche)
               + nombreNoeuds(A->droit)
```

Sur une forêt

```
nombreNoeudsF(F : foret) : entier
   si F = None alors
      retourner 0
   retourner 1 + nombreNoeudsF(F->premierEnfant)
               + nombreNoeudsF(F->prochainFratrie)
```

Nombre de nœuds

Réduction du nombre d'appels récursifs

On peut souhaiter réduire le nombre d'appels récursifs
en supposant que F est différent de None.

```
nombreNoeudsF(F : foret) : entier
   si F->premierEnfant) = None et F->prochainFratrie = None alors
       retourner 1
   si  F->premierEnfant) = None alors
       retourner  1 + nombreNoeuds(F->prochainFratrie)
   si F->prochainFratrie) = None alors
       retourner  1 + nombreNoeuds(F->premierEnfant)
   retourner 1 + nombreNoeudsF(F->premierEnfant) +
             nombreNoeudsF(F->prochainFratrie)
```

Exercice

Comparez le nombre d'appels récursifs entre les deux algorithmes.

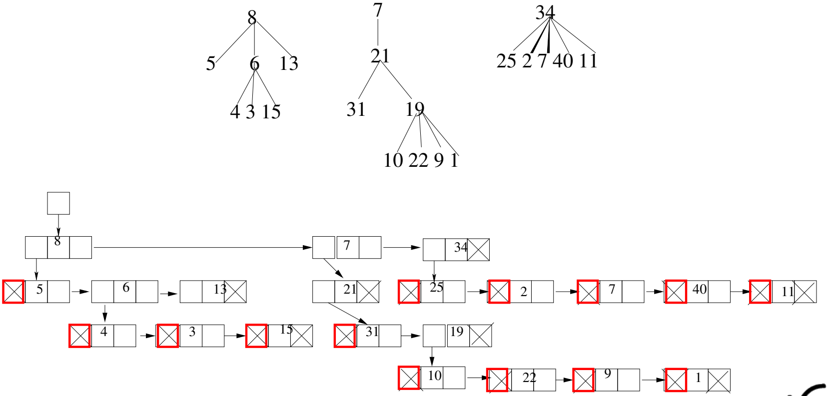
Indication : utilisez le nombre de nœuds internes et le nombre de feuilles.

Feuille d'une forêt ou d'un arbre général

Définition

Une feuille d'un arbre général ou d'une forêt est un nœud qui n'a pas d'enfant.

Exemple précédent



Feuille d'une forêt ou d'un arbre général

Définition

Une feuille d'un arbre général ou d'une forêt est un nœud qui n'a pas d'enfant.

Procédure

```
estFeuille(F : foret) : booléen
   retourne F->premierEnfant = None
```

Nombre de feuilles

```
nombrefeuilles(F : forêt) : entier
si F = None alors
   retourner 0
si estFeuille(F) alors
   retourner 1 + nombreFeuilles(F->prochainFratrie)
retourner nombrefeuilles(F->premierEnfant) +
          nombreFeuilles(F->prochainFratrie)
```

Bijection entre arbre général et arbre binaire

Racine de l'arbre général

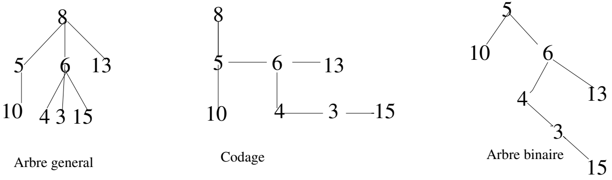
Avec notre codage, la racine d'un arbre général n'a pas de prochain nœud dans la fratrie.

Il suffit de retirer la racine pour obtenir un arbre binaire quelconque

Bijection entre arbres généraux et arbres binaires

Le nombre d'arbres généraux à n nœuds est égal au nombre d'arbres binaires à $n - 1$ nœuds.

Exemple



Arbre general

Codage

Arbre binaire

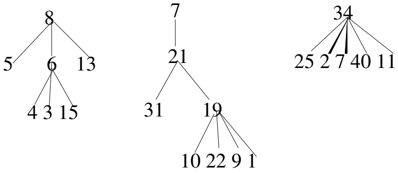
Degré d’un arbre

Degré d’un nœud

Le degré d’un nœud est le nombre de ses enfants.

Exemple de forêt

- 7 est de degré 1
- 6 est de degré 3
- 19 de degré 4



Degré d'un arbre

Degré d'un arbre général

Le degré d'un arbre général est le maximum des degrés de ses nœuds.
Sur l'exemple, le degré est 5, il est obtenu avec le nœud 34.

Calcul du degré d'un nœud

On calcule le degré d'un nœud en parcourant ces enfants.
Si le nœud courant est le dième enfant, nous avons

```
d --- d+1
|
1
```

- on met le degré à 1 pour le premier enfant
- on met le degré à $d + 1$ pour le prochain de la fratrie lorsque le nœud courant est de degré d

Exercice (voir TD)

Donnez une procédure calculant le degré d'un arbre général.
Indication : on utilise le schéma précédent.

Hauteur d'une forêt

Définition

La hauteur d'une forêt est la plus grande hauteur d'un des arbres généraux.
C'est donc la plus grande profondeur d'un nœud des arbres généraux.

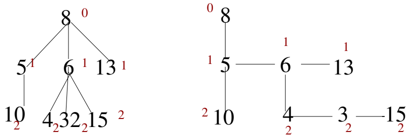
Evolution de la profondeur

Supposons que le nœud courant soit un nœud de profondeur (ou niveau) k.

```
k --- k
|
k+1
```

- le prochain de la fratrie est aussi de profondeur k
- le premier enfant est de profondeur $k+1$

Exemple



Arbre general

Codage

Calcul de la hauteur d'une forêt

Evolution de la profondeur

Supposons que le nœud courant soit un nœud de profondeur (ou niveau) k.

```
k --- k
|
k+1
```

Calcul de la profondeur maximale à partir d'un nœud

On suppose que la forêt F possède au moins un arbre général.

```
profondeurMax(F : foret, k : entier) : entier
   int maxEnfant, maxFratrie
   si F->premierEnfant = None alors
      maxEnfant = k
   sinon maxEnfant = profondeurMax(F->premierEnfant,k+1)
   si F->prochainFratrie = None alors
      maxfratrie = k
   sinon maxFrere = profondeurMax(F->prochainFratrie,k)
   retourner max(maxEnfant,maxFratrie)
```

Calcul de la hauteur d'une forêt

```
hauteur(F : foret) : entier
   retourner profondeurMax(F,0)
```

Plan du CM 7

Parcours en largeur

Parcours en largeur

Les nœuds sont traités par niveau.
Les nœuds de niveau 0 de gauche à droite, puis les nœuds de niveau 1 de gauche à droite...

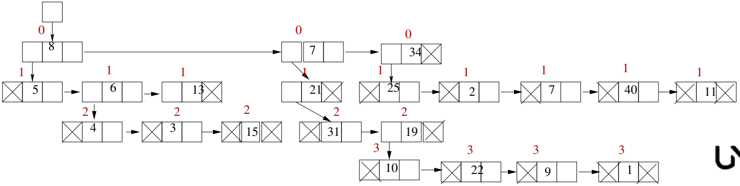
Rappel de l'évolution du niveau à partir d'un nœud

```
k --- k
|
k+1
```

Utilisation d'une file

On utilise une file contenant les nœuds et leur niveau.

Etiquetage du niveau



Parcours en largeur – affichage des valeurs

Méthode

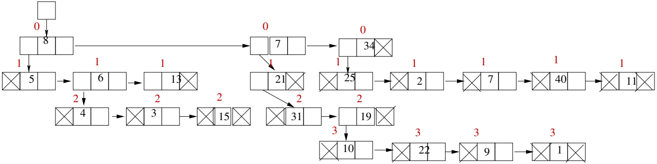
On enfile toujours d'abord la fratrie avant d'enfiler le premier enfant.

Algorithme

```
affichageLargeur(F : foret)
   si F <> None alors
      i : entier ; i = 0
      f : file ; f = initFile() ; f = enfiler(f, (F,i))
      tant que F <> None faire
         f = enfiler(f, (F,i))
         F = F->prochainFratrie
      tant que nonVide(f) faire
          B : foret ; B, i = tete(f)[1] ; f = defiler(f)
          afficher B->val -- niveau i
          si B->premierEnfant <> None faire
               F = B->premierEnfant
               tant que F <> None faire
                  f = enfiler(f, (F,i+1))
                  F = F->prochainFratrie
```

Parcours en largeur – affichage des valeurs

Exemple de forêt



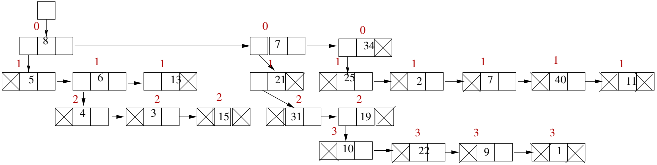
Etat de la file

Exécution

- au début la file est vide

Parcours en largeur – affichage des valeurs

Exemple de forêt



Etat de la file

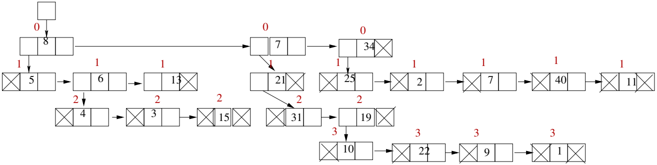
$(8, 0)$	$(7, 0)$	$(34, 0)$

Exécution

- on enfile les racines des trois arbres généraux

Parcours en largeur – affichage des valeurs

Exemple de forêt



Etat de la file

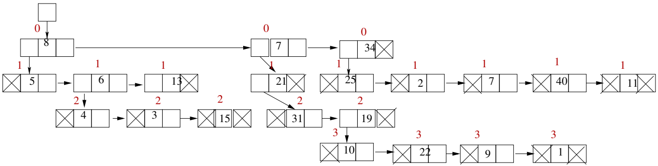
$(7, 0)$	$(34, 0)$	$(5, 1)$	$(6, 1)$	$(13, 1)$

Exécution

- on affiche 8 – niveau 0
- on défile $(8, 0)$
- i prend la valeur 1
- on enfile les trois enfants de 8 avec la valeur 1

Parcours en largeur – affichage des valeurs

Exemple de forêt



Etat de la file

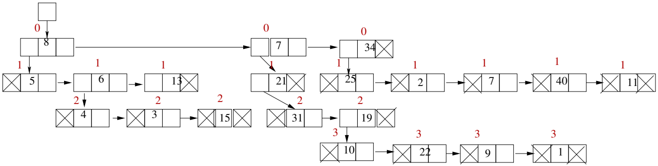
$(34, 0)$	$(5, 1)$	$(6, 1)$	$(13, 1)$	$(21, 1)$

Exécution

- on affiche 7 – niveau 0
- on défile $(7, 0)$
- i prend la valeur 1
- on enfile 21 avec la valeur 1

Parcours en largeur – affichage des valeurs

Exemple de forêt



Etat de la file

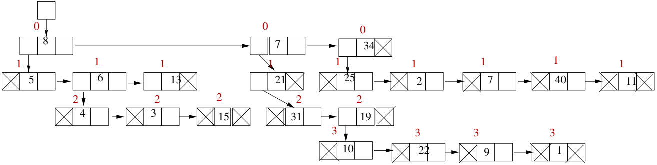
$(5, 1)$	$(6, 1)$	$(13, 1)$	$(21, 1)$	$(25, 1)$	$(2, 1)$	$(7, 1)$	$(40, 1)$	$(11, 1)$

Exécution

- on affiche 34 – niveau 0
- on défile $(34, 0)$
- *i* prend la valeur 1
- on enfile les 5 enfants de 34
- la file contient tous les nœuds de niveau 1

Parcours en largeur – affichage des valeurs

Exemple de forêt



Etat de la file

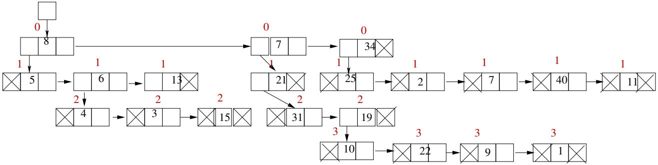
$(6,1)$	$(13,1)$	$(21,1)$	$(25,1)$	$(2,1)$	$(7,1)$	$(40,1)$	$(11,1)$

Exécution

- on affiche 5 – niveau 1
- on défile $(5,1)$
- i prend la valeur 2
- on s'arrête car 5 n'a pas d'enfant

Parcours en largeur – affichage des valeurs

Exemple de forêt



Etat de la file

$(13, 1)$	$(21, 1)$	$(25, 1)$	$(2, 1)$	$(7, 1)$	$(40, 1)$	$(11, 1)$	$(4, 2)$	$(3, 2)$	$(15, 2)$

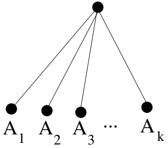
Exécution

- on affiche 6 – niveau 1
- on défile $(6, 1)$
- i prend la valeur 2
- on enfile les trois enfants de 6 avec la valeur 2

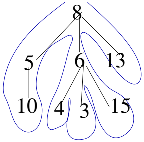
Parcours en profondeur sur un arbre général – ordre préfixe

Définition

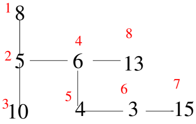
La définition est la même que pour un arbre binaire. On parcourt d'abord les nœuds de A_1, puis de A_2, . . .



Exemple



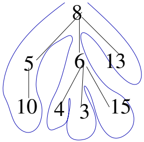
Arbre general



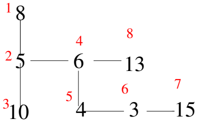
Codage

Parcours en profondeur sur un arbre général

Exemple



Arbre general



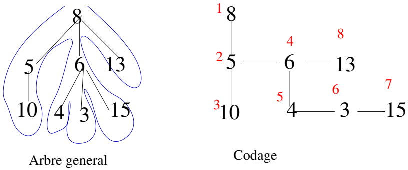
Codage

ordre préfixe et suffixe

- pour l'arbre binaire nous avions trois passages pour chaque nœud. Ici le nombre de passages dépend du degré du nœud.
- nous pouvons définir l'ordre préfixe (premier passage)
- nous pouvons définir l'ordre suffixe (dernier passage)
- pas d'ordre infixe

Parcours en profondeur sur un arbre général – ordre préfixe

Exemple



Algorithme

On retrouve le même algorithme que pour l'affichage en ordre préfixe sur les arbres binaires.

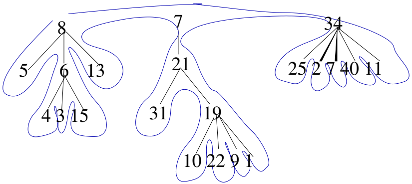
```
affichagePrefixe(A : arbreGeneral)
   si A <> None alors
       afficher A->valeur
       affichagePrefixe(A->premierEnfant)
       affichagePrefixe(A->prochainFratrie)
```

Parcours en profondeur sur une forêt

Définition

Il faut choisir un ordre entre les arbres généraux.

Exemple



Dans ce cas, nous avons le même algorithme que pour un arbre général.