Les arbres (I)

Nour-Eddine Oussous, Éric Wegrzynowski

Licence ST-A, USTL - API2

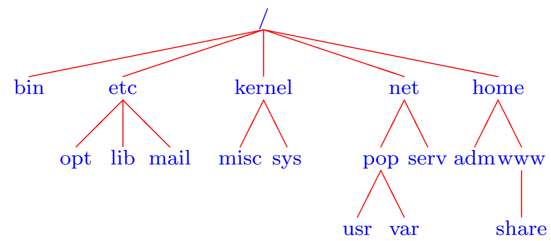
16 novembre 2009

Introduction

- Les arbres offrent une des plus importantes structures de données non linéaires en informatique.
- Ils permettent une organisation naturelle des données, par exemple
  - système de fichiers
  - base de données
  - sites web,. . .
- Dans les arbres, on a une relation hiérarchique entre les objets.
- Ils permettent parfois d'obtenir des algorithmes plus performants que lorsqu'on utilise des structures de données linéaires (listes, tableaux,. . .)

Exemple

L'arborescence des répertoires d'un système d'exploitation imaginaire.



Notion d'arbres

Un arbre est une sructure de données qui peut

- soit être vide
- soit comporter un nombre fini de nœuds tels que
  - à chaque nœud est associée une valeur
  - chaque nœud possède un nombre fini de successeurs
  - un (et un seul) nœud n'est le successeur d'aucun autre, c'est la racine
  - tout autre nœud est le successeur d'un seul nœud, son père.

Branches et profondeur des nœuds

- Un nœud qui ne possède aucun successeur est appelé nœud externe ou feuille.
- Les autres nœuds sont appelés nœuds internes.
- Une branche est une suite finie $x_0, x_1, \ldots, x_p$ de nœuds telle que
  - x_0 est la racine
  - x_p est une feuille
  - $\forall k \in [\![1, p]\!] \;\; \text{pere}(x_k) = x_{k-1}$
- La longueur d'une branche est le nombre de nœuds qui la composent moins 1 : p.

Profondeur d'un nœud

- Un nœud y est un descendant d'un nœud x s'il existe une branche qui va de x à y. Autrement dit, $\exists x_0, \ldots, x_p$ tels que
  - $x_0 = x$
  - $x_p = y$
  - $\forall\, k \in [\![1, p]\!],\; \text{pere}(x_k) = x_{k-1}$
- Le nombre p est appelé profondeur du nœud y par rapport à x.
- La profondeur d'un nœud dans un arbre est la profondeur de ce nœud par rapport à la racine.

Hauteur d'un arbre

- La hauteur d'un arbre non vide A, notée $h(A)$, est la longueur maximale d'une branche de cet arbre.
- La hauteur d'un arbre est aussi la profondeur maximale d'un nœud.
- La hauteur n'est pas définie pour un arbre vide. lorsque nous programmerons la hauteur, nous définirons par convention la hauteur de l'arbre vide.

Exemple

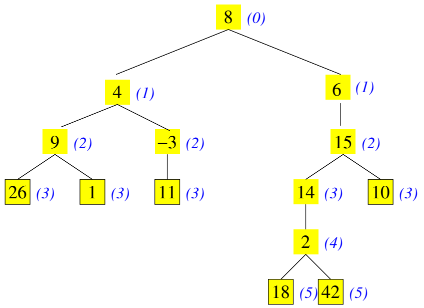


Fig.: Un arbre de hauteur 5 et de taille 14

nœuds internes : feuilles : *(profondeurs)*

Arité et taille d'un arbre

- Un arbre est dit d'arité a si chacun de ses nœuds possède au maximum a successeurs. L'arbre vide est de toute arité.
- La taille d'un arbre A est son nombre de nœuds.

Relation entre hauteur, arité et taille

Théorème

Soit A un arbre d'arité a, de taille n et de hauteur h.

- Le nombre n_p de nœuds de A à profondeur $0 \leq p \leq h$ vérifie
$$1 \leq n_p \leq a^p$$
- La taille n vérifie l'encadrement
$$h + 1 \leq n \leq \frac{a^{h+1} - 1}{a - 1}$$
- La hauteur h vérifie l'encadrement
$$\log_a (n(a-1) + 1) - 1 \leq h \leq n - 1$$

Définition récursive des arbres binaires

Définition

Un arbre binaire composés d'éléments d'un ensemble E est

- soit l'arbre vide Δ ;
- soit un triplet $< e; g; d >$ composé d'un élément $e \in E$, et de deux arbres binaires g et d

En notant $AB(E)$ l'ensemble des arbres binaires composés d'éléments de E, on a donc

$$AB(E) = \{\Delta\} \cup (E \times AB(E) \times AB(E))$$

Représentations abstraites d'un arbre binaire

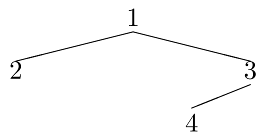


Fig.: Représentation graphique

```
<1; <2; Δ; Δ> ;
    <3; <4; Δ; Δ>; Δ>>
```

Fig.: Représentation triplet

Opérations primitives sur les arbres binaires

1. Constructeur
2. Sélecteurs
3. Prédicat
4. Opérations modificatrices

Création d'un arbre binaire

Spécification

$$\begin{array}{lcccl} \texttt{creerArbre} & : & E \times AB(E) \times AB(E) & \longrightarrow & AB(E) \\ & & e, g, d & \longmapsto & < e; g; d > \end{array}$$

Exemples

$$\begin{array}{rcl} \texttt{creerArbre}(3, \Delta, \Delta) & = & <3; \Delta; \Delta> \\ \texttt{creerArbre}(1, <3; \Delta; \Delta>, \Delta) & = & <1; <3; \Delta; \Delta>; \Delta> \end{array}$$

C'est une opération de construction d'arbres.

Accès à la racine

Spécification

$$\begin{array}{lcccl} \texttt{racine} & : & AB(E) & \longrightarrow & E \\ & & < e; g; d > & \longmapsto & e \end{array}$$

CU : l'arbre passé en paramètre ne peut pas être vide

Exemple

$$\texttt{racine}(<1; <3; \Delta; \Delta>; \Delta>) = 1$$

C'est un sélecteur.

Accès au sous-arbre gauche

Spécification

$$\begin{array}{lcccc} \texttt{gauche} & : & AB(E) & \longrightarrow & AB(E) \\ & & <e;g;d> & \longmapsto & g \end{array}$$

CU : l'arbre passé en paramètre ne peut pas être vide

Exemple

$$\texttt{gauche}(<1;<3;\Delta;\Delta>;\Delta>) = <3;\Delta;\Delta>$$

C'est un sélecteur.

Accès au sous-arbre droit

Spécification

$$\begin{array}{lcccc} \texttt{droit} & : & AB(E) & \longrightarrow & AB(E) \\ & & <e;g;d> & \longmapsto & d \end{array}$$

CU : l'arbre passé en paramètre ne peut pas être vide

Exemple

$$\texttt{droit}(<1;<3;\Delta;\Delta>;\Delta>) = \Delta$$

C'est un sélecteur.

Test de vacuité d'un arbre

Spécification

$$\begin{array}{lcccc} \texttt{estArbreVide} & : & AB(E) & \longrightarrow & Booleen \\ & & a & \longmapsto & \begin{cases} \text{Vrai} & \text{si } a \text{ est vide} \\ \text{Faux} & \text{sinon} \end{cases} \end{array}$$

Exemples

$$\begin{array}{rcl} \texttt{estArbreVide}(\Delta) & = & \text{Vrai} \\ \texttt{estArbreVide}(<1;\Delta;\Delta>) & = & \text{Faux} \end{array}$$

C'est un prédicat.

Changer la valeur de la racine

Spécification

$$\begin{array}{lcccc} \texttt{modifierRacine} & : & AB(E) \times E & \longrightarrow & AB(E) \\ & & <e;g;d>, e' & \longmapsto & <e';g;d> \end{array}$$

CU : l'arbre passé en paramètre ne doit pas être vide

Exemple

$$\texttt{modifierRacine}(<1;<3;\Delta;\Delta>;\Delta>, 4) = <4;<3;\Delta;\Delta>;\Delta>$$

C'est une opération de modification d'arbre.

Changer le sous-arbre gauche

Spécification

$$\texttt{modifierGauche} : \begin{array}{ccc} AB(E) \times AB(E) & \longrightarrow & AB(E) \\ < e; g; d >, g' & \longmapsto & < e; g'; d > \end{array}$$

CU : le premier arbre passé en paramètre ne doit pas être vide

Exemple

$$\texttt{modifierGauche}(< 1; < 3; \Delta; \Delta >; \Delta >, \Delta) = < 1; \Delta; \Delta >$$

C'est une opération de modification d'arbre.

Plan Introduction Définitions et vocabulaire Arbres binaires
Opérations modificatrices

Changer le sous-arbre droit

Spécification

$$\texttt{modifierDroit} : \begin{array}{ccc} AB(E) \times AB(E) & \longrightarrow & AB(E) \\ < e; g; d >, d' & \longmapsto & < e; g; d' > \end{array}$$

CU : le premier arbre passé en paramètre ne doit pas être vide

Exemple

$$\texttt{modifierDroit}(< 1; \Delta; \Delta >, < 2; \Delta; \Delta >) = < 1; \Delta; < 2; \Delta; \Delta >>$$

C'est une opération de modification d'arbre.

Les arbres (I) Licence ST-A, USTL - API2