Expressions postfixées

21 mai 2003

1 Expression postfixée

1.1 définition générales

définition : On définit un alphabet $\mathcal{A}$ comme un ensemble et des mots sur un alphabet comme tout suite finie d'éléments de $\mathcal{A}$. Il est parfois utile de rajouter un mot vide $\emptyset$.
définition : On appelle *longueur d'un mot*, le nombre de lettres qui compose ce mot (la longueur du mot vide étant nulle).
définition : On appelle *concaténation* des mots A et B le mot obtenu en accolant le mot B à la suite du mot A. on le note $A.B$ ou AB. exemple : si $A = unique$ et $B = ment$ on a $AB = uniquement$.
Si on fait le parallèle avec les chaînes de caractères Caml, l'opérateur de concaténation est ^.
on appelera expression algébrique tout mot défini par un alphabet contenant des constantes, des variables, des fonctions et des opérateurs.
exemple :

- ensemble des constantes : $\mathcal{C} = \mathbb{Z}$,
- ensemble des variables : $\mathcal{V} = \{a, b, c, \ldots, z\}$
- ensemble des opérateurs : $\mathcal{O} = \{+, -, \times, /\}$
- ensemble des fonctions : $\mathcal{F} = \{\cos, \sin, \exp, \log\}$
- alphabet : $\mathcal{A} = \mathcal{C} \cup \mathcal{V} \cup \mathcal{O} \cup \mathcal{F}$

ainsi $\sin 2 + \cos \log a$ est une expression algébrique pour cet alphabet.

1.2 expression postfixée

On considère un alphabet $\mathcal{A} = \mathcal{C} \cup \mathcal{V} \cup \mathcal{O} \cup \mathcal{F}$ permettant de définir des expressions algébriques contenant des constantes, des variables, des opérateurs et des fonctions. On définir de manière inductive ce qu'est une expression algébrique postfixée.

- tout mot de longueur 1, a avec $a \in \mathcal{C} \cup \mathcal{V}$ est une expression postfixée
- tout mot de la forme Af où A est une expression postfixée et f une fonction
- tout mot de la forme ABo où A et B sont des expressions postfixées et o est un opérateur.

exemple : $\sin 2 + \cos \log a$ n'est pas une expression postfixée, par contre $3 \sin 5 \times \ln$ est une expression postfixée de longueur 5.

1.3 données du problème

Les expressions postfixées ont un avantage par rapport aux expressions infixes habituellement utilisées, c'est qu'elle ne sont pas ambiguës. Ainsi, si on ne donne pas d'ordre de priorité sur les opérateurs, l'expression préfixée $1+2*3$ peut s'interpréter de deux manières différentes.

- soit $(1+2)*3$ qui donne le résultat 9
- soit $1+(2*3)$ qui donne le résultat 7.

dans le cas des expressions algébriques post fixées, la première s'écrit $12{+}3{*}$ alors que la deuxième s'écrit $123*+$.
Les expressions postfixées sont utilisées sur les calculatrices HP depuis la première calculatrice de poche mise sur le marché vers la fin des années soixante début des années soixante-dix. Le problème est de reconnaître les expressions post fixées parmi toutes les expressions algébriques.
On peut déjà faire un certain nombre de remarques simples :

- toute expression post fixée non vide commence par une constante ou une variable
- toute expression post fixée de longueur supérieure ou égale à 2 se termine par une fonction ou un opérateur

<u>démonstration</u> : Pour la deuxième propriété, il suffit de remarquer que les expressions post fixée de longueur supérieure ou égale à 2 sont nécessairement de la forme Af ou ABo avec A et B expressions post fixées, f fonction et o opérateur ce qui nous donne le résultat.
Pour la première propriété, on peut le faire par récurrence sur la longueur de l'expression post fixée. Soit H_n : toute expression E post-fixée de longueur inférieurs ou égale à n vérifie

- E commence par une constante ou une variable.

Les seules expressions post fixées de longueur 1 sont les mots de la forme a avec $a \in \mathcal{C} \cup \mathcal{V}$ ce qui nous donne H_1. Pour $H_n \Rightarrow H_{n+1}$ avec $n \geqslant 1$, il suffit d'appliquer H_n pour les expressions de longueurs inférieures ou égales à n. Pour les expressions post fixées de longueur $n+1$, elles sont de la forme Af ou ABo avec A et B expressions post fixées, f fonction et o opérateur. L'expression post fixée A étant dans les 2 cas de longueur inférieur ou égale à n, on peut appliquer H_n pour dire que A (et donc l'expression de départ) commence par une constante ou variable, ce qui achève la démonstration.

1.4 Évaluation par une pile

1.5 Fonction de poids

<u>définition</u> : on définit une fonction de poids par

- $p(a) = 1$ si $a \in \mathcal{V} \cup \mathcal{C}$,
- $p(f) = 0$ si $f \in \mathcal{F}$,
- $p(o) = -1$ si $o \in \mathcal{O}$

et pour toute expression $E = a_1a_2\dots a_n$, poids de l'expression est défini comme la somme des poids des «lettres» qui la compose : $p(E) = \sum_{i=1}^{n} p(a_i)$.
<u>théorème</u> : Une expression algébrique est une expression post fixée si et seulement

- Elle est de poids total 1
- Tous ses préfixes non vide ont un poids supérieur ou égal à 1

démonstration :

1.6 implémentation

On peut prendre la convention de travailler avec des listes d'expressions de longueur 1 pour former une expression algébrique de longueur quelconque :

```
type operateur =   add | mult | sous  | div ;;
type fonction = Exp | Log | Cos | Sin  | Carré | Rac_carrée;;

type expression =
  reel of float
| var  of string
| op   of operateur
| fct  of fonction;;
```

La fonction de reconnaissance des expressions post fixées peut se programmer de la façon suivante :

```
let test l =
     (* <<poids>> donne le poids d'un élément simple d'une expression *)
     let poids = function
         reel(x) -> 1              (* Cas des constantes réelles *)
       | var(x)  -> 1              (* Cas des variables *)
       | fct(f)   -> 0             (* Cas des fonctions *)
       | op(o)   -> -1  in         (* Cas des opérateurs *)
     let p = ref(poids (hd l)) and l = ref(tl l) in
          while (!l <> []) & (!p >= 1) do (* Si le poids devient < 1 on sort ou
                                             si on a parcouru toute la liste *)
             p := !p + (poids (hd !l));
             l := tl(!l)
            done;
         if !p = 1 then true   (* On a bien une expression postfixée *)
                   else false  (* On n'a pas une expression postfixée  *);;
```