La programmation récursive

Une fonction f est récursive lorsqu'il existe une méthode permettant de calculer $f(a)$ à partir de valeurs $f(b)$ avec $b \prec a$.

Fonction		Formule		Formule de récursivité
$f(n) = n!$	$\longrightarrow$	$n! = n \times (n-1)!$	$\longrightarrow$	$f(n) = n.f(n-1)$
$f(a,n) = a^n$	$\longrightarrow$	$a^n = a.a^{n-1}$	$\longrightarrow$	$f(a,n) = a * f(a,n-1)$
$f(l) =$ longueur de la liste l	$\longrightarrow$	$l = x :: q$	$\longrightarrow$	$f(l) = 1 + f(q)$
$f(n,p) = \binom{n}{p}$	$\longrightarrow$	$\binom{n}{p} = \frac{n}{p}\binom{n-1}{p-1}$	$\longrightarrow$	$f(n,p) = \frac{n}{p} f(n-1,p-1)$

La difficulté consiste à déterminer une telle formule !

Exemple 1

Exponentiation rapide : Pour calculer $f(a,n) = a^n$.

On remarque que : $\begin{cases} a^{2p} = [a^p]^2 \\ a^{2p+1} = a * [a^p]^2 \end{cases}$.

Ainsi : $$f(a,n) = \begin{cases} \left(f(a, \frac{n}{2})\right)^2 & \text{lorsque } n \text{ est pair} \\ a * \left(f(a, \frac{n-1}{2})\right)^2 & \text{lorsque } n \text{ est impair} \end{cases}$$

Exemple 2 : Résolution de $g(x) = 0$ par dichotomie :

On suppose ici la fonction g continue, strictement croissante sur $[a,\ b]$ avec $\begin{cases} f(a) < 0 \\ f(b) > 0 \end{cases}$.

On cherche la solution α de $g(x) = 0$ sur $[a,\ b]$ à $\varepsilon > 0$ près.

On note $f(g,a,b,\varepsilon)$ cette solution.

On remarque que a et b sont des approximations de α à ε près lorsque $b - a \leq \varepsilon$.

Ainsi, tant que $b - a \geq \varepsilon$, on aura :

- $f(g,a,b,\varepsilon) = f(g,a,\frac{a+b}{2},\varepsilon)$ lorsque $g(\frac{a+b}{2}) > 0$
- $f(g,a,b,\varepsilon) = f(g,\frac{a+b}{2},b,\varepsilon)$ lorsque $g(\frac{a+b}{2}) < 0$

Nous avons bien la stricte décroissance des arguments. (au sens de la réduction de l'intervalle $[a,\ b]$)

Exemple 3 : Recherche dichotomique dans un tableau trié

On veut savoir si un élément e se trouve dans un tableau trié t.

On note $f(e,t,i,j)$ la fonction qui renvoie $\begin{cases} \text{true lorsque } e \in [t.(i);\ \ldots;\ t.(j)] \\ \text{false lorsque } e \notin [t.(i);\ \ldots;\ t.(j)] \end{cases}$.

On voudra alors calculer $f(0, n-1)$ où n est la longueur de t.

Formule de récursivité : $f(e,t,i,j) = \begin{cases} \text{true} & \text{si} \quad e = t.(\frac{i+j}{2}) \\ f(e,t,i,\frac{i+j}{2}) & \text{si} \quad e < t.(\frac{i+j}{2}) \\ f(e,t,\frac{i+j}{2},j) & \text{si} \quad e > t.(\frac{i+j}{2}) \end{cases}$.

Cas de base : Lorsque $j - i \leq 1 \longrightarrow f(e,t,i,j) = (e = t.(i))\ ||\ (e = t.(j))$

Comment ou quand s'arrêter ? Les cas de base

Fonction		Formule de récursivité		Cas de base		Valeur associée
$f(n) = n!$	$\longrightarrow$	$f(n) = n.f(n-1)$	$\longrightarrow$	$n = 0$	$\longrightarrow$	$f(0) = 1$
$f(a,n) = a^n$	$\longrightarrow$	$f(a,n) = a * f(a,n-1)$	$\longrightarrow$	$n = 0$	$\longrightarrow$	$f(a,0) = 1$
$f(l) =$ longueur de la liste l	$\longrightarrow$	$f(x :: q) = 1 + f(q)$	$\longrightarrow$	$l = [\,]$	$\longrightarrow$	$f([\,] = 0$
$f(n,p) = \binom{n}{p}$	$\longrightarrow$	$f(n,p) = \frac{n}{p} f(n-1,p-1)$	$\longrightarrow$	$p = 0$	$\longrightarrow$	$f(n,0) = 1$
u_n tel que $\begin{cases} u_0 = 0 \\ u_1 = 1 \\ u_{n+2} = u_{n+1} + u_n \end{cases}$	$\longrightarrow$	$f(n) = f(n-1) + f(n-2)$	$\longrightarrow$	$\begin{cases} n = 0 \\ n = 1 \end{cases}$	$\longrightarrow$	$\begin{cases} f(0) = 0 \\ f(1) = 1 \end{cases}$

Il faut indiquer à la fonction :

- La valeur de l'argument pour laquelle il n'est plus utile d'appliquer la formule de récursivité
- Ce qu'elle doit renvoyer pour cette valeur

Résolution de $g(x) = 0$ par dichotomie :

- $f(g,a,b,\varepsilon) = f(g,a,\frac{a+b}{2},\varepsilon)$ lorsque $g(\frac{a+b}{2}) > 0$
- $f(g,a,b,\varepsilon) = f(g,\frac{a+b}{2},b,\varepsilon)$ lorsque $g(\frac{a+b}{2}) < 0$

Cas de base : lorsque $b-a \leq \varepsilon \longrightarrow f(g,a,b,\varepsilon) = \frac{a+b}{2}$

Exemple : Recherche dichotomique dans un tableau :

Formule de récursivité : $f(e,t,i,j) = \begin{cases} \text{true} & \text{si} \quad e = t.(\frac{i+j}{2}) \\ f(e,t,i,\frac{i+j}{2}) & \text{si} \quad e < t.(\frac{i+j}{2}) \\ f(e,t,\frac{i+j}{2},j) & \text{si} \quad e > t.(\frac{i+j}{2}) \end{cases}$.

Cas de base : Lorsque $j-i \leq 1 \longrightarrow f(e,t,i,j) = (e = t.(i)) \;||\; (e = t.(j))$

Le principe : Recherche d'une formule de récursivité

Exemple : $f(n) = \sum_{k=1}^{n} k$

- Formule de récursivité : $f(n) = n + f(n-1)$
- Cas de base : $f(1) = 1$

OCaml
```
let rec f = function
          | 1 -> 1
          | n -> n + (f (n-1));;
```

Exemple : $f(a,n) = a^n$

- Formule de récursivité : $f(a,n) = a * f(a,n-1)$
- Cas de base : $f(a,0) = 1$

OCaml
```
let rec f a = function
          | 0 -> 1
          | n -> a*(f a (n-1));;
```

Programmation : Comment faire ?

Exemple : $g(x) = 0$ par dichotomie

- Formule de récursivité : Tant que $b-a \geq \varepsilon$, on a :
  - $f(a,b,\varepsilon) = f(a,\frac{a+b}{2},\varepsilon)$ lorsque $f(\frac{a+b}{2}) > 0$

- $f(a,b,\varepsilon) = f(\frac{a+b}{2}, b, \varepsilon)$ lorsque $f(\frac{a+b}{2}) < 0$

- Cas de base : Lorsque $b - a \leq \varepsilon \longrightarrow f(a,b,\varepsilon) = a$

OCaml
```
let rec f g a b eps =
    if (b-.a) > eps then let m = (a+.b)/.2. in
                         match (g m) < 0. with
                           | true  -> f g m b eps
                           | false -> f g a m eps
                    else a;;
f (fun x -> x-.1.) 0.9 1.1 0.0001;;
```

ASTUCE : Comment trouver la formule de récursivité ?

On veut calculer $f(a)$...
Pour cela, on suppose être en mesure de calculer $f(b)$ pour un ou plusieurs $b \prec a$.
On se demande alors comment, obtenir $f(a)$ à partir des $f(b)$ déjà connus...

- $f(n) = n!$
- $f(a,n) = a^n$
- $f(l) =$ la longueur de la liste $l = x :: q$
- Le calcul du terme u_n de la suite de Fibonacci
- La recherche d'un élément dans une liste $l = x :: q$

Développés dans les diapo suivantes...

- Décomposition d'un entier sous la forme $n = p.2^q$
- tri selectif

Exemple : Décomposition d'un entier n sous la forme $p.2^q$

Il s'agit en fait de déterminer le couple (p,q) tel que $n = p.2^q$ avec n impair.

- Si n n'est pas divisible par 2 alors on renvoie $(n, 0)$
- Sinon, on a $n = 2.\frac{n}{2}$.

  → On applique donc la fonction à $\dfrac{n}{2}$

  → On obtient un couple (p,q) tel que $\dfrac{n}{2} = p.2^q$ et donc $n = p.2^{q+1}$

  → On renvoie donc le couple $(p, q+1)$

Exemple : Le tri sélectif

Pour un tableau `t = [t.(0);...;t.(n-1)]`.

- On commence par placer le plus grand élément en position t.(n-1)
- On applique la fonction de tri à [t.(0) ;... ;t.(n-2)]
- Cela n'est possible que si on ajoute à la fonction un argument j indiquant la partie `[t.(0);...;t.(j)]` à laquelle est appliqué le tri

OCaml
```
let tri t = let n = Array.length t and
                tri_aux t j = on place la plus grande composante en t.(j) ;
                              on applique tri_aux t (j-1)
                in tri_aux t (n-1);;
```

La Méthode à suivre

1. On commence par rechercher la formule de récursivité
2. On en déduit les cas de base, puis les valeurs que doit renvoyer la fonction pour chacun d'eux
3. On passe à l'implémentation OCaml

A vous de jouer !

Proposer une formule de récursivité et étudier les cas de base pour :

- La recherche d'un élément maximal dans une liste `l = x::q`
- La recherche d'un élément maximal dans un tableau `t` (avec une fonction auxilliaire)
- L'insertion d'un élément à la fin d'une liste `l = x::q`
- Le calcul d'un coefficient binomial avec la formule d'addition
- Le calcul du pgcd de deux entiers avec la formule d'Euclide
- La détermination des parties d'un ensemble représenté sous la forme d'une liste
- Ecriture d'un nombre entier en base 2

Il ne reste plus qu'à proposer une implémentation OCaml