TD Algorithmique

July 26, 2024

1 Un premier calcul de a^n

```
Fonction puissance(a, n):
    c ← n
    p ← 1
    Tant que c>0:
        p ← p × a
        c ← c - 1
```

La suite d'entiers c définie par :

$$c_0 = n$$

$$c_{i+1} = c_{i-1}$$

est strictement décroissante, donc au bout d'un nombre fini d'itérations on a $c = 0$ et l'algorithme termine.

On choisit l'invariant $p_i = a^{n-c_i}$. Pour $i = 0$, on a bien $p_0 = 1 = a^{n-c_0} = a^{n-n} = 1$. Si on suppose $p_i = a^{n-c_i}$, on a (cf la boucle) :

$$p_{i+1} = p_i \times a = a^{n-c_i} \times a = a^{n-c_{i+1}} = a^{n-(c_{i-1}} = a^{n-c_{i+1}}$$

Donc la propriété est vraie pour tout i. A la dernière itération, $c_i = 1$ puisque $c_{i+1} = 0$ ce qui fait sortir de la boucle, donc $p_i = a^{n-1}$ et comme $p_{i+1} = a \times p_i = a^n$ l'algorithme est correct.

La complexité est $O(n)$ car la boucle fait n itérations (on fait n multiplications pour calculer la puissance).

2 Exponentiation rapide

i	0	1	2	3
r_i	1	1	a^2	a^6
k_i	a	a^2	a^4	a^8
c_i	6	3	1	0

i	0	1	2	3
r_i	1	a	a^3	a^7
k_i	a	a^2	a^4	a^8
c_i	7	3	1	0

On constate que la variable r contient le résultat à renvoyer, il faut donc à priori compléter la fonction par la ligne :

`retourner r`

- Terminaison :

  A chaque itération, $c_{i+1} = c_i//2$, donc la suite d'entiers positifs c_i est strictement décroissante, en conséquence au bout d'un nombre fini d'itérations on aura $c = 0$.

- Correction :

  Pour $i = 0$, on a $k_0^{c_0} \times r_0 = 1 \times a^n = a^n$.

  Supposons que la propriété soit vraie pour le rang i : $k_i^{c_i} \times r_i = a^n$:

  Il faut distinguer deux cas :

  - si c_i est pair, alors $c_i = 2p$ donc $c_{i+1} = p$, $k_{i+1} = k_i^2$ et $r_{i+1} = r_i$. On en déduit : $k_{i+1}^{c_{i+1}} \times r_{i+1} = (k_i^2)^p \times r_i = k_i^{c_i} \times r_i = a^n$
  - si c_i est impair, alors $c_i = 2p + 1$ donc $c_{i+1} = p$, $k_{i+1} = k_i^2$ et $r_{i+1} = r_i \times k_i$. On en déduit : $k_{i+1}^{c_{i+1}} \times r_{i+1} = (k_i^2)^p \times r_i k_i = k_i^{2p+1} \times r_i = k_i^{c_i} \times r_i = a^n$ Donc la propriété est vraie pour tout i. Comme l'algorithme termine pour $c_i = 0$, on aura donc à la sortie de la boucle principale $k_i^0 r_i = r_i = a^n$, l'algorithme est donc correct.

L'algorithme d'exponentiation rapide effectue beaucoup moins de multiplications que l'algorithme précédent, qui nécessitait n multiplications. En effet l'algorithme termine lorsque $c = n$ a été divisé par 2 jusqu'à ce qu'on obtienne 0. A une unité près, le nombre d'itérations est donc i tel que $n = 2^i$ autrement dit $i = log_2(n)$. On a au plus 2 multiplications par itération, donc la complexité de l'algorithme est $O(log(n))$. Si n est grand, le gain est donc très important.

[ ]: