Algorithmique IV (UE-41) - TD 4.

TD 4. Square & Multiply et factorisation de Hörner [1]

EXERCICE 1. Écrivez l'algorithme d'exponentiation naïf sur la machine RAM, en supposant qu'un registre peut stocker des valeurs arbitrairement grandes. Calculez le nombre $C(n)$ d'instructions décodées par cet algorithme en fonction de n. On suppose que la première valeur sur la bande d'entrée est le nombre x à exponentier et le second l'exposant n. L'hypothèse est-elle réaliste sur une machine physique et peut-on considérer que la fonction I est la fonction de complexité de l'algorithme ?

EXERCICE 2. On appelle *chaîne d'additions* de longueur ℓ toute séquence $(a_k)_{k\in[\![0,\ell]\!]}$ strictement croissante de $\ell+1$ entiers naturels telle que $a_0 = 1$ et telle que chaque entier de la séquence est la somme de deux valeurs précédentes quelconques de cette séquence.

(1) Définissez une chaîne d'additions en logique des prédicats.

(2) Écrivez l'algorithme `DoubleAdd` qui est l'adaptation additive de l'algorithme `SquareMultiply` pour un monoïde $(X, +)$ où l'on a simplement remplacé la loi de composition multiplicative par une loi addive.

(3) Soit $(X, \times)$ un monoïde. Montrez que le nombre minimum de multiplications requises pour calculer x^n est la longueur minimale d'une chaîne d'additions $(a_i)_{i\in[\![0,\ell]\!]}$ telle que $a_\ell = n$.

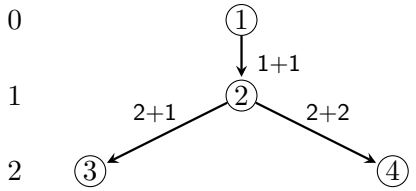

FIGURE 1. Arbre des chaînes d'additions de longueur $k = 2$.

1. Version du 13 janvier 2025, 11 : 13

(4) On construit inductivement l'arbre des chaînes d'additions de la manière suivante : la racine contient la valeur $a_0 = 1$ et on crée en chaque nœud de l'arbre autant de fils que de nouvelles valeurs obtenues en additionnant deux valeurs quelconques sur le chemin qui le relie à la racine. Construisez l'arbre partiel de profondeur $k := 4$ (La figure 1 montre cet arbre partiel pour la profondeur $k = 2$).

(5) Vérifiez que le nombre de fils d'un nœud de profondeur $k \geqslant 1$ est minoré par $k+1$ et majoré par $(k+1)(k-2)/2$. Est-il envisageable de chercher une chaîne d'addition minimale pour atteindre une valeur n à l'aide de l'arbre des chaînes d'additions ?

(6) On se donne un élément x d'un monoïde $(X, \times)$. Quel est le nombre minimal de multiplications requises pour calculer x^{15} ?

(7) Comparez le au nombre de multiplications effectuées par l'algorithme *square & multiply*, le calcul d'un carré étant comptabilisé comme une multiplication.

EXERCICE 3. Écrivez une version de l'algorithme *square & multiply* dans laquelle les bits de l'exposant sont utilisés dans l'ordre inverse, autrement dit du bits de poids faible vers le bit de poids fort.

EXERCICE 4. La suite de Fibonacci est une suite récurrente d'entiers naturels de premiers termes $F_0 := 0$ et $F_1 := 1$ définie par la relation

$$\forall n \in \mathbb{N} \quad F_{n+2} := F_{n+1} + F_n. \tag{1}$$

(1) Écrivez un algorithme sur la machine RAM qui pour l'entrée n, calcule le terme F_n de la suite de Fibonacci. On suppose que les registres peuvent coder des entiers de taille arbitrairement grande. On supposera que $n \geqslant 2$. Calculez le nombre $C(n)$ d'instructions décodées par la machine RAM en fonction de n.

On définit la matrice $\Phi := \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$.

(2) Démontrez que

$$\forall n \in \mathbb{N} \quad \Phi^{n+1} = \begin{pmatrix} F_{n+2} & F_{n+1} \\ F_{n+1} & F_n \end{pmatrix}. \tag{2}$$

(3) Écrivez un algorithme `ProdMat(A,B)` qui renvoie le produit de deux matrices carrés avec $A[i][j]$ désignant le terme à la ligne i et la colonne j.

(4) Écrivez un algorithme avec une complexité $\Theta(\log(n))$ pour calculer F_n.

EXERCICE 5. Soit $N \geqslant 1$ et $b \geqslant 2$ deux entiers naturels. Démontrez que le nombre de chiffres de l'écriture de N en base b est égal à $\lfloor \log_b N \rfloor + 1$. Indication : on admet que pour tout réel $x \geqslant 0$, il existe un unique entier N tel que $N \leqslant x < N + 1$, appelé *partie entière* de x que l'on note $\lfloor x \rfloor$.

EXERCICE 6. On rappelle l'algorithme de Hörner :

```
    ALGORITHME Horner(P,x):réel
    DONNEES
     · P[0,n]: liste de n+1 réels
     · x: réel
    VARIABLES
     · R: réel
     · i: entier
    DEBUT
01>  · R ← P[n]
02>  · i ← 0
03>  · TQ (i < n) FAIRE
04>  ·  · i ← i + 1
05>  ·  · R ← R * x + P[n - i]
06>  ·  FTQ
07>  · RENVOYER R
    FIN
```

Algo. 1. Évaluation après factorisation de Hörner.

Démontrez que l'algorithme s'arrête et démontrez qu'il est correct. Considérez le prédicat $P(i)$ suivant :

$$\text{Après la } i\text{-ème itération } \quad R = \sum_{k=0}^{i} a_{n-k} x^{i-k}. \tag{3}$$