Algorithmique 2, TD n° 1

Devan SOHIER

Exercice 1 : opérations arithmétiques *« comme en primaire »*

On dispose des deux variables tableaux de chiffres décimaux et indexés par les chiffres décimaux suivants :

Add	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9	0
2	2	3	4	5	6	7	8	9	0	1
3	3	4	5	6	7	8	9	0	1	2
4	4	5	6	7	8	9	0	1	2	3
5	5	6	7	8	9	0	1	2	3	4
6	6	7	8	9	0	1	2	3	4	5
7	7	8	9	0	1	2	3	4	5	6
8	8	9	0	1	2	3	4	5	6	7
9	9	0	1	2	3	4	5	6	7	8

AddRet	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	1
2	0	0	0	0	0	0	0	0	1	1
3	0	0	0	0	0	0	0	1	1	1
4	0	0	0	0	0	0	1	1	1	1
5	0	0	0	0	0	1	1	1	1	1
6	0	0	0	0	1	1	1	1	1	1
7	0	0	0	1	1	1	1	1	1	1
8	0	0	1	1	1	1	1	1	1	1
9	0	1	1	1	1	1	1	1	1	1

On représente un nombre par un tableau de chiffres décimaux. En utilisant ces tableaux, écrire un algorithme pour l'addition de deux nombres.

Ecrire un algorithme pour la multiplication (vous écrirez également les tableaux auxiliaires nécessaires). Ecrire un algorithme pour la division.

Faire ce même travail en binaire.

Calculer la complexité de ces fonctions.

Concrètement, c'est ainsi que sont réalisées les opérations sur les grands nombres (dépassant la taille implémentée sur la machine) ou sur les polynômes (pour la multiplication et la division).

Exercice 2 : crible d'Eratosthène

On cherche à déterminer tous les nombres premiers entre 1 et n. Pour ce faire, on crée un tableau de n booléens initialement à vrai dans lequel on passe successivement à faux tous les multiples des entiers successifs de 1 à n. Ecrivez cet algorithme, expliquez-le et calculez sa complexité.

Exercice 3 : Algorithme d'Euclide étendu (obtention d'une relation de Bezout)

Algorithm 1 Algorithme d'Euclide

```
Données : a et b deux entiers
r ← a%b
while r ≠ 0 do
    a ← b
    b ← r
    r ← a%b
end while
Ecrire b
```

Le théorème de Bezout énonce que pour tous entiers a et b, il existe des entiers u et v tels que $ua + vb = pgcd(a, b)$. En vous appuyant sur le fait qu'à chaque itération $a = b \times q + r$, adaptez l'algorithme d'Euclide vu en cours pour obtenir ces entiers u et v (importants pour le calcul sur les corps finis que vous verrez si vous faîtes de la cryptographie).

Vous justifierez cet algorithme, en montrant qu'à chaque itération, on a bien $r = a \times u + b \times v$, et que le pgcd de b et r est bien le même que celui de a et b. Vous montrerez que r est décroissant, et que l'algorithme termine donc.

On s'intéresse maintenant à la complexité de cet algorithme. A chaque itération, r diminue d'au moins b. Le pire cas est quand le quotient de a par b est 1. En partant de la dernière itération, et de deux entiers premiers entre eux (si ce n'est pas le cas, les itérations ont lieu à l'identique au facteur près de leur pgcd), montrez que dans le pire des cas, la séquence des valeurs de r est une suite de Fibonacci $F_{n+2} = F_{n+1} + F_n$. Déduisez-en que le pire des cas est $a = F_{n+1}$ et $b = F_n$, auquel cas l'algorithme prend $n + 1$ itérations.

On sait que la suite de Fibonacci $F_n \sim \left(\frac{1+\sqrt{5}}{2}\right)^n$. Déduisez en la complexité de cet algorithme.