Preuve et Analyse des Algorithmes

9 janvier 2014

Q 1. Soit n un entier positif. Soit $q \neq 1$ un nombre complexe.

1. Que vaut la somme de la série géométrique $\sum_{i=0}^{n-1} q^i$?
2. Montrer que si $M_n = 2^n - 1$ est premier alors n est premier.
3. La réciproque est-elle vraie ?

Q 2. Soit n un entier positif.

1. Que vaut la somme de la série arithmétique $\sum_{k=0}^{n-1} k$?
2. Cette somme apparaît naturellement dans l'analyse du temps de calcul d'un algorithme du cours. Lequel ?

Q 3. On note $I(a, b)$ le nombre d'itérations de l'algorithme d'Euclide pour calculer le PGCD des entiers a et b.

1. Quelle suite numérique est impliquée dans l'analyse de $I(a, b)$?
2. Donner une instance (a, b) pour laquelle $I(a, b) = 11$.

Q 4.

1. Déterminer une solution (entière) de $127u + 107v = 1$.
2. Quel est l'inverse de 107 modulo 127 ?
3. Que vaut 107^{126} modulo 127 ?

Q 5. Une implantation du tri linéaire en langage C

```
typedef unsigned long long ullong;
void trilin( ullong *t , ullong n );
```

sur la machine `obelix` trie une instance de taille 2^{20} en 1 microseconde.

1. Rappeler les préconditions d'utilisation du tri linéaire.
2. Estimer le temps de calcul d'une instance de taille 2^{24} sur la machine `obelix`.
3. A quoi faut-il s'attendre pour la taille 2^{26} ?

Q 6. Soit m un entier positif. On rappelle que le coefficient de Fourier-Hadamard en a d'une table t de taille 2^m est donné par : $\hat{t}(a) = \sum_{0 \leq x < 2^m} t(x)(-1)^{a.x}$.

1. Rappeler l'algorithme récursif vu en cours.
2. Calculer la tranformée de Hadamard-Fourier de la table

$$0, 1, 2, 3, 4, 5, 6, 7$$

Q 7. La figure [**??**] rapporte sur le temps de calcul d'un test de primalité de Fermat appliqué au n-ième nombre de Fermat $F_n = 2^{2^n} + 1$.

```
[drm@obelix] $ lscpu
Architecture: i686    Mode: 32-bit Vitesse proc. en MHz: 3000.000
[drm@obelix] $ free -b
             total       used       free     shared
Mem:    1582931968  837398528  745533440          0
-/+ buffers/cache:  389427200 1193504768
Swap:   3187666944          0 3187666944
[drm@obelix] $ echo 'l(1582931968)/l(2)'  | bc -l
30.559952105962841097 56
```

FIGURE 1 – capacités de la machine `obelix`

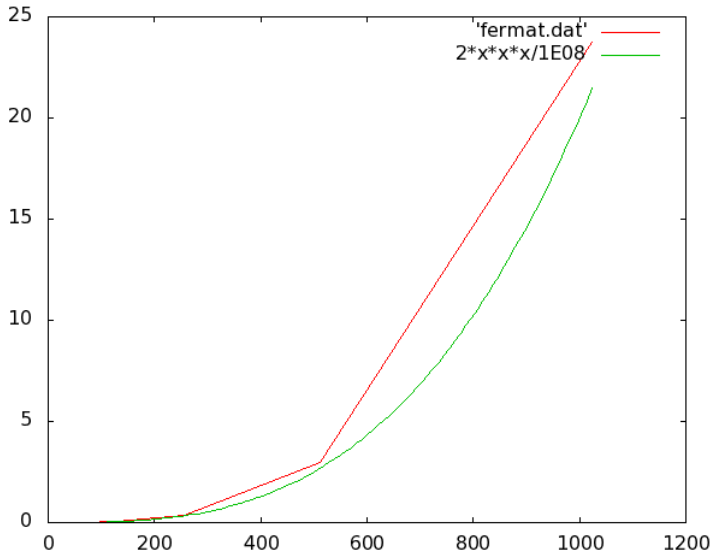

FIGURE 2 – Temps de calcul en secondes du test de primalité de Fermat appliqué à F_n. En abscisse la taille binaire des nombres.

1. Commenter le grahique.
2. Estimer le temps de calcul pour F_{13}.

Q 8. Pour un entier n, on note $\pi(n)$ le nombre de nombres premiers strictement inférieurs à n. L'algorithme du crible d'Eratostène est un procédé efficace pour déterminer tous les nombres premiers inférieurs à un entier donné n. On utilise une table t de n booléens initialisés à 0, sauf $t[0] = 1$ et $t[1] = 1$. L'algorithme procède par élimination en $\sqrt{n}$ étapes. A l'étape i, si $t[i] = 0$ c'est que i est premier, et, on marque "composé" les entiers multiple de i : $t[2i] = 1$, $t[3i] = 1$, ..., $t[ij] = 1$. A la fin de ce processus, les les entiers premiers correspondent aux cellules marquées 0.

On rappelle que la série harmonique : $H_n = \sum_{i=1}^{n} \frac{1}{i}$ vérifie la relation $H_n \sim \ln n$.

1. Utiliser le crible pour déterminé $\pi(100)$.
2. Ecrire une fonction `int pi( int n )` qui retourne $\pi(n)$.
3. Etablir que le temps de calcul est proportionnel à $\sum_{p \leq \sqrt{n}} \frac{n}{p}$, où la somme porte sur des entiers premiers.
4. Montrer que le temps de calcul est majoré par $nH_{\sqrt{n}}$.
5. Montrer que le temps de calcul est $O(n \log n)$.