info528 : Mathématiques pour l'informatique
TD 1 : diviser pour régner

Pierre Hyvernat
Laboratoire de mathématiques de l'université de Savoie
bâtiment Chablais, bureau 22, poste : 94 22
email : Pierre.Hyvernat@univ-savoie.fr
www : http://www.lama.univ-savoie.fr/~hyvernat/

Exercice 1 : approximations asymptotiques

Question 1. Les affirmations suivantes sont-elles toujours vraies, toujours fausses, parfois vraies ? (Les fonctions sont strictement positives.)
- $f(n) = O\big(f(n)^2\big)$,
- $\min\big(f(n), g(n)\big) = O\big(f(n) + g(n)\big)$,
- $\min\big(f(n), g(n)\big) = \Omega\big(f(n) + g(n)\big)$.

Question 2. Donner une approximation pour $T(n)$ quand
- $T(n) = 2T(n/3) + n\log(n)$,
- $T(n) = T(n/3) + 3^n$,
- $T(n) = 11T(n/3) + n^{\pi/2}$,

Exercice 2 : Complexité.

Question 1. Quelle est la complexité de la recherche dichotomique dans un tableau trié ?

Question 2. Un tableau *unimodal* est un tableau que l'on peut décomposer en
- un segment initial $0, 1, \dots, l$ trié par ordre croissant,
- un segment final $l, l+1, \dots, n$ trié par ordre décroissant.

et qu'il n'y a jamais deux éléments égaux cote à cote.

Par exemple : `[1,5,6,18,21,20,17,4,3,2,1,0]`.

Quelle est la complexité de la recherche du maximum dans un tels tableau ?

Question 3. Quelle est la complexité asymptotique du tri fusion au pire cas / en moyenne ?

Question 4. Le tri rapide (*quicksort*) fonctionne en utilisant le premier élément du tableau comme *pivot* : on s'en sert pour séparer le tableaux en deux parties :
- les élément plus petits que le pivot,
- les éléments plus grands que le pivot.

Chaque partie est ensuite trié récursivement.

Quelle est la complexité du tri rapide au pire cas ?

Pouvez-vous utiliser le théorème du cours (*master theorem*) pour calculer la complexité du tri rapide en moyenne ?

Exercice 3 : Fibonacci

Avec un manque d'originalité flagrant, on définit la suite suivante

$$F_0 = 0 \qquad F_1 = 1 \qquad F_{n+2} = F_{n+1} + F_n \ .$$

Question 1. Écrivez un petit programme récursif naïf pour calculer F_n. Donner une relation de récurrence donnant le nombre d'additions A_n utilisées par cet algorithme pour le calcul de F_n.

Question 2. Calculez les 11 premières valeurs de F_n et de A_n ; conjecturez et prouvez ce qui ressort de cette expérimentation.

Question 3. En sachant que l'on a la formule exacte

$$F_n \quad = \quad \left\lfloor \frac{1}{\sqrt{5}} \left(\frac{1+\sqrt{5}}{2} \right)^n + \frac{1}{2} \right\rfloor$$

que proposez-vous comme méthode de calcul de F_n ?

Question 4. En regardant la question 2, estimez le nombre d'additions que vous avez fait lors du calcul de F_{10}. Déduisez-en un algorithme itératif pour le calcul de F_n et estimez sa complexité (nombre d'additions effectuées).

Question 5. On peut généraliser légèrement et calculer 2 nombres de Fibonacci à la fois avec

$$G_0 = (0,1) \qquad G_{n+1} = (y, x+y) \text{ où } G_n = (x,y)$$

Exprimez la relation entre G_n et G_{n+1} sous forme d'une matrice et déduisez une méthode très efficace pour calculer G_n, et donc F_n.

Exercice 4 : multiplication de Strassen

Question 1. évaluez le nombre de multiplications et d'additions scalaires utilisées pour la multiplication de deux matrices $n \times n$.

Question 2. supposons que n soit une puissance de 2. En utilisant récursivement la multiplication par blocs de taille $n/2$, donnez une relation de récurrence pour calculer le nombre de multiplications scalaires nécessaires à la multiplication des matrices.

On rappelle que la multiplication par blocs correspond à :

$$\left[\begin{array}{c|c} A_{1,1} & A_{1,2} \\ \hline A_{2,1} & A_{2,2} \end{array}\right] \times \left[\begin{array}{c|c} B_{1,1} & B_{1,2} \\ \hline B_{2,1} & B_{2,2} \end{array}\right] \quad = \quad \left[\begin{array}{c|c} C_{1,1} & C_{1,2} \\ \hline C_{2,1} & C_{2,2} \end{array}\right]$$

où les $X_{i,j}$ sont des matrices de taille $n/2 \times n/2$ et $C_{1,1} = A_{1,1} \times B_{1,1} + A_{1,2} \times B_{2,1}$, ...

Question 3. on va maintenant essayer d'améliorer les performances (nombre de multiplications scalaires) : on définit les matrices suivantes :

- $M_1 = (A_{1,1} + A_{2,2}) \times (B_{1,1} + B_{2,2})$
- $M_2 = (A_{2,1} + A_{2,2}) \times B_{1,1}$
- $M_3 = A_{1,1} \times (B_{1,2} - B_{2,2})$
- $M_4 = A_{2,2} \times (B_{2,1} - B_{1,1})$
- $M_5 = (A_{1,1} + A_{1,2}) \times B_{2,2}$
- $M_6 = (A_{2,1} - A_{1,1}) \times (B_{1,1} + B_{1,2})$
- $M_7 = (A_{1,2} - A_{2,2}) \times (B_{2,1} + B_{2,2})$

On pose ensuite

- $C_{1,1} = M_1 + M_4 - M_5 + M_7$
- $C_{1,2} = M_3 + M_5$
- $C_{2,1} = M_2 + M_4$
- $C_{2,2} = M_1 - M_2 + M_3 + M6$

Montrez que la matrice C ainsi obtenue est bien le produit de A et B.

Question 4. donnez une relation de récurrence pour le nombre de multiplications employées par cet algorithme. Résolvez cette récurrence et concluez.

Question 5. discutez les problèmes potentiels liés à cet algorithme.