Introduction à l'algorithmique: tri et mélanges

Nicolas Gast - nicolas.gast@ens.fr

27 novembre 2006

But du TD

1. Comprendre le sens des mots "`local`" et "`global`"
2. Découvrir des algorithmes de tri
3. Réussir à écrire soi même un algorithme
4. Approfondir vos connaissances sur le calcul de la complexité d'une fonction.

Prérequis

1. Connaitre l'instruction `if`
2. Savoir écrire une procédure
3. Savoir écrire une boucle `for` et `while`
4. Opérations élémentaires sur les listes (nombre d'éléments `nops(l)` et échange de deux éléments).

Exercice 1. `local` et `global`

a. Deviner (sur papier) le résultat des lignes suivantes :

```
x := 1;  y := 1;
glocal := proc()
  local x,z;   global y,t;
  x := 2+x;    y := 2+y;
  z := x;      t := y;
  x,y,z,t;
end;

glocal();
x,y,z,t;
```

b. Taper ces lignes en Maple. Est-ce conforme à vos prévisions ?

Exercice 2. Le Tri bulle

Le *tri à bulle* ou tri par propagation est un algorithme de tri très simple à mettre en œuvre. Il consiste à faire remonter le plus grand élément du tableau (comme une bulle d'air remonte à la surface) en comparant les éléments successifs. C'est-à-dire qu'on va comparer le 1^{er} et le $2^{\text{ième}}$ élément du tableau et les échanger si le premier est plus grand que le deuxième. On recommence cette opération jusqu'à la fin du tableau. Ensuite, il ne reste plus qu'à renouveler l'opération jusqu'à ce que le tableau soit trié.

a. Montrer qu'au bout d'une itération, le plus grand élément du tableau se trouve en dernière position.

b. En déduire qu'au bout de n itérations, le tableau est entièrement trié.

c. Programmer l'algorithme de tri-bulle

d. Combien votre algorithme fait-il de comparaisons ?

e. Si ce n'est pas déjà fait, proposer une amélioration (simple) permettant de passer le nombre de comparaisons à $n(n-1)/2$.

Exercice 3. Dé-trier un tableau

Il peut être parfois utile de mélanger les éléments d'un tableau, ce qui revient à tirer une permutation aléatoire de n éléments (une permutation est une bijection de $[1;n]$ dans $[1;n]$). Il existe beaucoup de mauvaises idées pour faire ça (par exemple, tirer N couples au hasard et les échanger), le but de cet exercice est de programmer un algorithme qui tire une permutation aléatoire de n éléments en temps linéaire :

1. Tirer un nombre au hasard a entre 1 et n et échanger l'éléments n avec l'élément a
2. Tirer un nombre au hasard a entre 1 et $n-1$ et échanger l'éléments $n-1$ avec l'élément a
3. ... Continuer en remplaçant $n-1$ par $n-2$ puis $n-3$, ... jusqu'à 2.

a. Programmer l'algorithme en Maple prenant en entrée un tableau et rendant un nouveau tableau dans lequel les éléments sont les éléments du premier tableau permutés. (note : pour tirer un nombre aléatoire entre `0` et `n-1` inclus, utilisez la commande `rand(n)()`)

b. Quelle est le nombre d'opérations que fait votre algorithme. Peut-on faire mieux ?

c. Pourquoi cet algorithme tire bien chaque permutation de l'ensemble de départ avec la même probabilité.

Exercice 4. Tri fusion

Le tri repose sur le fait que pour fusionner deux listes/tableaux trié(e)s dont la somme des longueurs est n, n-1 comparaisons au maximum sont nécessaires. Pour trier un tableau de taille n, l'algorithme est :

Tri-fusion(l) =

1. Couper **l** en deux sous-tableaux **l1** et **l2** de taille $n/2$
2. Trier **l1** et **l2**
3. Fusionner **l1** et **l2**

a. Écrire une fonction `fusion := proc(l1,l2)` qui prend en entrée deux tableaux (qu'on suppose triés) `l1` et `l2` et qui rend la réunion (triée) des deux tableaux.

b. Écrire la procédure `tri-fusion`

c. Calculer la complexité de votre algorithme.

Exercice 5. Le docteur No a encore frappé

Le cruel Docteur No a capturé 50 mathématiciens pour les soumettre à une épreuve diabolique. Il a créé sur son île une pièce vide à l'exception d'une lampe et d'un interrupteur qui sert à l'allumer ou l'éteindre. Après avoir permis aux mathématiciens de se concerter, il va les soumettre à son épreuve dont il leur communique les termes : il empêchera toute communication entre eux (autre que celle qui va être décrite) et emmènera chaque jour un mathématicien tiré au hasard dans la pièce où se trouve l'interrupteur. La lumière est initialement éteinte. Chaque fois qu'un mathématicien passe dans la pièce, il constate si le précédent avait laissé la lumière allumée ou éteinte (c'est la seule forme de communication possible), et peut lui-même choisir de la laisser dans son état précédent ou de l'allumer ou de l'éteindre. Le but des mathématiciens est de savoir à quel moment tous les mathématiciens sont passés par la pièce : si un mathématicien en acquiert la certitude, il peut l'annoncer au Docteur No. Il est hors de question de se tromper, bien sûr, car cela serait puni de la mort avec des tortures particulièrement raffinées pour tous les mathématiciens ; en revanche, si l'affirmation est juste, tous les mathématiciens sont libérés. le but du problème est de trouver une stratégie telle que les mathématiciens soient (« presque ») certains, un jour, d'être tous libérés et d'être sûr de ne jamais se tromper.

a. On définit $p(n,k)$ la probabilité qu'au bout de n jours, k mathématiciens différents soient passés dans la pièce. Trouver une relation de récurence entre $p(n,k)$, $p(n-1,k)$ et $p(n-1,k-1)$.

b. Écrire une procédure calculant $p(n,k)$.

c. Trouver une stratégie.

d. Évaluer les performance de cette stratégie en maple.