UNE ANALYSE EN MOYENNE D'ALGORITHME
ÉTUDE D'UN TRI : QUICKSORT.

1. Avertissement :

Ce cours est sensé donner, à un niveau fin de DEUG deuxième année, une introduction à quelques techniques d'analyse en moyenne d'algorithmes.

De telles techniques seront ici illustrées par une analyse de l'une des plus célèbres méthodes de tri : Quicksort.

Les prérequis sont quelques notions d'algorithmiques (par exemple programmer en Pascal, légère compréhension de la récursivité), ainsi que la manipulation de séries vue en cours de mathématiques.

2. Pourquoi trier ?

Devant les flux d'informations auxquelles on est confronté en informatique, il est primordial d'y mettre un peu d'ordre. Travail que l'humain délègue désormais à l'ordinateur (notez l'étymologie du mot ordinateur : ordonner<ordre).

Avec quelle efficacité ? Il faut pour cela étudier la "complexité" de la méthode employée.

Il y a de nombreuses méthodes (algorithmes) de tri ; voici un tableau comparatif de leur efficacité :

en moyenne dans le pire des cas tri par insertion (insertion sort) $n^2/2$ comparaisons tri par sélection (selection sort) tri par bulle (bubble sort) tri par (shellsort) tri rapide (quicksort) tri par fusion (merge sort)

Donnons ici l'algorithme "quicksort" dû a Hoare en 1962

```
Procedure Quicksort(l,r:integer);
var v,t,i,j:integer;
Begin
if r>=1 then
 begin
  v:=a[r]; i:=l-1; j:=r;
    repeat
     repeat I:=i+1 until a[i]>=v;
     repeat j:=j-1 until a[j]<=v;
     t:=a[i];a[i]:=a[j];a[j]:=t;
    until j<=i;
  a[j]:=a[i];a[i]:a[r];a[r]:=t;
  quicksort(l,i-1);
  quicksort(i+1,r);
 end
end;
```

Le chapitre suivant va montrer comment on peut affirmer que "Quicksort effectue en moyenne n partitions $2(n+1)(H_{n+1}-1)$ comparaisons $(n+1)(H_{n+1}-5/2)/3+1/2$ échanges".

Cet algorithme est typiquement une illustration de la méthode "diviser pour régner" (divide and conquer).

Date: Mars 1999.

3. Un outil puissant : les séries génératrices

> "Concepts of analysis of algorithms are not really esoteric or difficult, but they are relatively new."

Tel est ce qu'affirme D.E. Knuth, dans la préface du livre de Ph. Flajolet et R. Sedgewick, ouvrage qui est chaleureusement recommandé à quiconque veut approfondir ce qui esquissé ci.

La morale de tout ce chapitre, c'est de relier le discret au continu, c'est là toute la richesse des séries génératrices.

4. Exemple avec les arbres binaires

On a b_n arbres binaires avec n nœuds internes.

$B = FouB - N - B$

$B(z) = \sum_{n\geq 0} b_n z^n$ verifie donc $B(z) = 1 + zB^2(z)$ Ainsi, $B(z) = 1 - \sqrt{1-4z}2z$

$$\rho = \frac{1}{4} b_n = 4^n$$

en fait $b_n = \frac{\binom{2n}{n}}{n+1}$, les nombres de Catalan (1,1,2,5,14...).

5. Application a Quicksort

$C(z) = \sum_{n\geq 0} c_n z^n$ c_n : nombre moyen de comparaisons pour trier n elements.

récurrence

$$c_0 = 0 \quad c_n = n + 1 + \frac{1}{n} \sum_{1\leq j\leq n} c_{j-1} + c_{n-j}$$

nb de comparaisons pour la premiere partition

proba de la partition en j

rec. sur $j-1$

rec. sur $n-j$

On multiplie par n, on somme et on a :

$$\sum_{n\geq 1} nc_n z^n = \sum_{n\geq 1} n(n+1)z^n + 2\sum_{n\geq 1}\sum_{1\leq k\leq n} c_{k-1}z^n$$

$$C'(z) = \frac{2}{(1-z)^3} + 2\frac{2C(z)}{1-z}$$

$$\iff ((1-z^2)^2 C(z))' = (1-z)^2 C'(z) - 2(1-z)C(z) = \frac{2}{1-z}$$

$$C(z) = \frac{2}{(1-z)^2} \ln \frac{1}{1-z}$$

Théorème 1. *Le nombre moyen de comparaisons pour Quicksort est*

$$c_n = [z^n]\frac{2}{(1-z)^2} \ln \frac{1}{1-z} = 2(n+1)(H_{n+1} - 1)$$

La première égalité découle du lemme et la deuxième découle de ce que $C(z) = 2(\frac{1}{1-z} \ln \frac{1}{1-z})' - \frac{2}{(1-z)^2}$ or $\frac{1}{1-z} \ln \frac{1}{1-z} = \sum_{n\geq 1} H_n z^n$ car $\ln \frac{1}{1-z} = \frac{1}{1-z} = \int \sum z^n = \sum \frac{1}{n} z^n$ puis on multiplie par $\frac{1}{1-z}$. □

Remarque : dans le tableau comparatif j'avais du $\ln n$ et ici nous avons H_n mais c'est la meme chose car $H_n = \sum_{k=1}^{n} \frac{1}{k} \sim \int \frac{dx}{x} = \ln n$.

On pourrait de même montrer que la variance est $n^2(7 - \frac{2\pi^2}{3})$.

6. Conclusion

Quicksort mérite bien son nom car la meilleure efficacité que l'on soit en droit d'attendre d'un tri par comparaison est $\lceil \log_2 n! \rceil \sim 1.44 n \ln n$ (aux implémentations près).

On peut se demander si plutôt que de partager la liste à trier en deux, on n'aurait pas pu appliquer d'avantage le principe diviser pour régner en la divisant par exemple en trois. La reponse est oui et une analyse similaire permet de dire que l'algorithme du Quicksort trichotomique se comporte en $12/7 n \ln n$. En revanche, partager la liste en davantage de morceaux n'apporte plus de gain notable.

Après avoir vu et étudié une méthode de tri, il est cohérent d'analyser les méthodes de recherches dans une telle liste désormais triée, les différents algorithmes (qui reposent souvent sur la notion d'arbres) succomberont au même type d'analyse que celle vue plus haut.

Notons pour finir que les techniques esquissées ici sont succeptibles de s'appliquer à toute structure combinatoire et donne naissance à ce que l'on appelle la combinatoire analytique.

book MR83i:68003, AUTHOR = Knuth, Donald E., TITLE = The art of computer programming. Vol. 2, EDITION = Second, NOTE = Seminumerical algorithms, Addison-Wesley Series in Computer Science and Information Processing, PUBLISHER = Addison-Wesley Publishing Co., Reading, Mass., YEAR = 1981, PAGES = xiii+688, ISBN = 0-201-03822-6, @book MR56:4281, AUTHOR = Knuth, Donald E., TITLE = The art of computer programming, NOTE = Volume 3. Sorting and searching, Addison-Wesley Series in Computer Science and Information Processing, PUBLISHER = Addison-Wesley Publishing Co., Reading, Mass.-London-Don Mills, Ont., YEAR = 1973, PAGES = xi+722 pp. (1 foldout),

@book MR86k:68037, AUTHOR = Sedgewick, Robert, TITLE = Algorithms, PUBLISHER = Addison-Wesley Publishing Co., Reading, Mass., YEAR = 1983, PAGES = viii+552, ISBN = 0-201-06672-6,

Références :

Flajolet/Sedgewick Analysis of Algorithms Sedgewick Algorithms in C

(toutes en anglais chez Addison-Wesley, certaines ont ete traduites en francais)