TD Info n°7

PCSI 2 Lycée Pasteur

25 octobre 2007

Nous reprenons cette semaine notre étude succinte des différents algorithmes de tri. Après avoir étudié la semaine dernière des algorithmes assez peu efficaces, nous nous intéressons cette fois-ci à des algorithmes plus complexes mais plus rapides.

1 Tri fusion

Cet algorithme se base sur le principe, fréquemment utilisé en informatique, « diviser pour régner ». On procède en pratique de la façon suivante : on sépare le tableau en deux sous-tableaux de taille égale, on trie chacun de ces deux tableaux, puis on les fusionne. En quoi consiste cette fusion ? On compare le premier élément de chacun des deux tableaux, on sélectionne le plus petit (qui sera forcément le premier élément du tableau final), on l'enlève de son tableau et on recommence. On obtient ainsi, à partir de deux sous-tableaux triés de taille n chacun, un tableau trié de taille $2n$ en faisant au maximum $2n-1$ comparaisons (on place un élément à chaque comparaison). Poussons le principe un peu plus loin. Pour chacun des deux sous-tableaux, on va les trier en effectuant sur eux un tri fusion, et ainsi de suite récursivement. On découpe en fait nos tableaux jusqu'à obtenir des tableaux de taille 1 (qui se trient tout seuls), puis on fait des fusions successives.

Pour un tableau de taille 2^p (c'est plus simple à calculer), on va ainsi avoir 2^{p-1} fusions de 2 tableaux à 1 élément à la première étape (1 comparaison à chaque fois), puis 2^{p-2} fusions de 2 tableaux à 2 elements (au plus 3 comparaisons à chaque fois), etc, jusqu'à la dernière étape où on fait une seule fusion de 2 tableaux à 2^{p-1} éléments (soit 2^p-1 comparaisons au plus). Le nombre total de comparaisons est donc plus petit que $2^{p-1}+3\times 2^{p-2}+\cdots+2^p-1 \leq p2p$ (chaque terme de la somme est plus petit que 2^p). On a donc moins de $n\log_{10}(n)$ comparaisons. Cet algorithme est beaucoup plus efficace que ceux étudiés la semaine dernière.

2 Tri rapide

Le tri rapide (quicksort en anglais) est réputé être l'algorithme le plus utilisé au monde. Il se base également en partie sur le principe diviser pour régner, mais n'utilise pas de fusion. C'est en fait très simple : on choisit un élément dans le tableau, et on le compare à tous les autres, en formant deux tas : ceux plus petits que lui et ceux plus grands que lui. Il ne reste plus qu'à trier ces deux tas en utilisant la même méthode, l'élément ayant servi de pivot se trouvant au milieu des deux sous-tableaux une fois le tri terminé. Cet algorithme souffre d'un léger défaut : son efficacité dépend fortement du choix du pivot à chaque étape. Si par malheur on tombe sur le plus grand ou le plus petit élément du tableau on fait une étape pour rien. En pratique, on prend usuellement le premier élément du tableau, en supposant que statistiquement ce sera un bon pivot. Si on tombe toujours sur l'élement central pour faire office de pivot, on aura $\log(n)$ étapes à notre algorithme (et même un peu moins, car on enlève le pivot à chaque étape), chaque étape ayant moins de n comparaisons, donc au total une complexité de l'ordre de $n\log n$, comme le tri fusion. En réalité, le tri rapide est légèrement plus efficace en pratique, sauf dans les cas où on tombe souvent sur un mauvais pivot.

3 Tri par casiers

C'est une méthode de tri extrêmement efficace (elle ne nécessite essentiellement aucune comparaison) mais qui ne fonctionne qu'avec des entiers ou assimilés (en tout cas un ensemble fini pas trop gros). Supposons qu'on veuille trier un tableau d'entiers tous compris entre 1 et n, avec n connu à l'avance ou calculé en cherchant le maximum du tableau ($n-1$ comparaisons. On crée un tableau de taille n rempli de 0, et on parcourt ensuite le tableau à trier, en incrémentant pour chaque élément la case du tableau correspondant à la valeur de l'élément. À la fin, il suffit de reconstituer un tableau contenant le bon nombre de fois chaque valeur ayant été incrémentée. On peut en fait utiliser des variantes de cet algorithme dans des cas plus généraux : on veut trier des noms (dans l'ordre alphabétique), on crée un tableau de 26^2 cases correspondant à chaque couple de lettres possibles et on place chaque mot dans la case correspondant à ses deux premières lettres. Il ne reste ensuite plus qu'à trier les éventuels mots commençant par les mêmes lettres, ce qui met assez peu de temps comparé au tri global du tableau.

4 Comparaisons

Si vous êtes courageux, faites à la main les étapes du tri à bulle, du tri fusion et du tri rapide pour trier le tableau suivant : $[4; 13; 11; 6; 9; 1; 7; 2; 10; 16; 14; 3; 8; 5; 15; 12]$, et compter le nombre de comparaisons pour chaque méthode.

Pour le tri d'un tableau de 10000 éléments, on met environ 2 minutes en utilisant un tri à bulles (119.1 secondes avec un tableau aléatoire), 98.1 secondes avec un tri par sélection, 62.5 secondes avec un tri par insertion, 0.109 secondes pour un tri fusion et 0.063 secondes pour un tri rapide.