

Travaux Pratiques

Séance 1 : algorithmes de tri

Dans ce TP, nous allons implanter des algorithmes de tri. Les tableaux contiennent des entiers. Un framework vous est fourni sur moodle.

Exercice 1: tri à bulle et variante du shaker

Le tri à bulles consiste à faire remonter progressivement les plus grands éléments d'un tableau en comparant successivement des paires consécutives d'éléments et en intervertissant le cas échéant (si l'ordre est incorrect). Notez qu'une fois l'élément max « accroché » il est toujours « gagnant » sur les paires consécutives, et remonte donc systématiquement tel une bulle dans le tableau (d'où le nom). Sa variante, le shaker, consiste à faire remonter, resp. descendre en alternance les plus grands, resp. les plus petits, éléments. L'algorithme s'arrête quand plus aucun élément n'a été déplacé.

Quelle est la complexité de ces algorithmes dans le meilleur et le pire cas ?

Exercice 2: tri par fusion

Ce tri consiste à diviser un tableau en deux moitiés de tailles égales. Chacun des deux sous-tableau est trié par appel « récursif ». Après les deux appels récursifs, les deux sous-tableaux (triés) sont fusionnés, c'est à dire recombines, pour obtenir un tableau unique trié.

Quelle est la complexité de la recombinaison ? Celle de toutes les divisions ? Faire un dessin des appels et du nombre d'éléments à recombinaison.

En déduire que ce tri est de complexité $\theta(N \log N)$

Exercice 3: tri rapide

Cette méthode de tri consiste à choisir un élément du tableau appelé pivot et à permuter tous les autres éléments de telle sorte que tous ceux qui sont inférieurs ou égaux au pivot soient à sa gauche et que tous ceux qui sont supérieurs au pivot soient à sa droite. Une fois ce partitionnement réalisé, la méthode est appelée récursivement sur les parties gauches et droites. Aucune recombinaison n'est nécessaire après les appels récursifs, puisque le tableau est nécessairement trié.

Nous avons vu en cours, qu'en moyenne ce tri est de complexité $\theta(N \log N)$.