N. Creignou, F. Olive, E. Perdereau et K. Perrot

TP n° 1

Sous-séquences maximales

(À présenter en séance de TP le vendredi 9 octobre.)

Étant donné un tableau d'entiers relatifs $T[1,\ldots,n]$, il s'agit de déterminer une sous-séquence (éléments consécutifs) dont la somme est maximale. Ainsi, notant $S(k,l) = \sum_{j=k}^{l} T[j]$ pour $1 \leq k \leq l \leq n$, il s'agit de déterminer k_{max} et l_{max} tels que $S(k_{max}, l_{max}) = \max_{k \leq l} S(k,l)$.

On propose quatre stratégies algorithmiques pour résoudre ce problème :

i) Premier algorithme : examiner toutes les sous-séquences possibles.

ii) Optimiser l'algorithme précédent en observant que $S(k,l) = S(k,l-1) + T[l]$.

iii) Diviser pour régner : diviser la séquence en deux. Calculer une sous-séquence de somme maximale de chaque moitié. Calculer une sous-séquence de somme maximale qui contient l'élément du milieu. Finalement prendre le maximum des trois.

iv) Supposer le problème résolu pour $T[1,\ldots,i]$. Observer que la solution pour $T[1,\ldots,i+1]$ est soit la solution précédente soit la sous-séquence de somme maximale qui se termine par $T[i+1]$.

Travail demandé

(a) Écrire chacun des algorithmes en langage pseudo-algorithmique.

(b) En faire une analyse de la complexité dans le pire des cas.

(c) Implémenter ces algorithmes.

(d) Comparer leurs performances sur des instances aléatoires.
(Indication : pour mesurer le temps d'exécution d'un programme, vous pouvez utiliser la commande `time` *en Bash.* `time ./max1 1 -2 3 -4`*)*
(Indication : pour générer des instances aléatoires, vous pouvez utiliser la variable `RANDOM` *en Bash.* `echo $(( $RANDOM % 100 ))`*)*

(e) Discuter la pertinence des résultats expérimentaux par rapport à l'analyse faite en (b).