TP4 – Encore un peu de tableau

Projet de programmation M1

13 Octobre 2015

Exercice 1. [Plus grande somme contiguë] Dans cet exercice, on souhaite implémenter une fonction `int max_sub(int t[], int n)` qui, étant donné un tableau d'entier `t` et sa taille `n`, renvoie la plus grande somme qu'on peut faire en additionnant des éléments consécutifs du tableau. Par exemple pour le tableau `int t[] =` {`-2,10,-7,10,1,-5`}, la plus grande somme d'éléments consécutifs qu'on peut faire est $10 + -7 + 10 + 1 = 14$.

1. Proposez une première implémentation en calculant directement la valeur maximale pour tout i, j, $0 \leq i \leq j < n$, de la somme `t[i]+t[i+1]+...+t[j]`. Combien d'opérations votre algorithme effectue-t-il ?
2. On va trouver une meilleure solution qui fait un nombre linéaire (en la taille du tableau) d'étapes. Pour cela, on va calculer pour tout $i < n$, une valeur m_i qui contient la plus grande somme d'éléments consécutifs se terminant à la case i.
   (a) Que vaut m_0 ? Exprimez ensuite m_{i+1} en fonction de m_i.
   (b) Comment trouver à partir de cela la somme qui nous intéresse ?
   (c) Implémentez cet algorithme, connu sous le nom d'algorithme de Kadane.

Exercice 2. [Polynôme] On se propose ici d'encoder les polynômes (à coefficients entiers) par des tableaux de la manière suivante : un polynôme $P(X) = \sum_{i=0}^{d} a_i X^i$ de degré d est représenté par un tableau `t` de taille au moins $d + 1$ tel que `t[i] =` a_i.

1. Écrivez une fonction `int evalue(int x, int p[], int degre)` qui évalue le polynôme p sur x. Combien d'additions/multiplications votre fonction effectue-t-elle en fonction du degré de p ? En remarquant qu'on peut écrire un polynôme ainsi $((a_d X + a_{d-1})X + \ldots + a_1)X + a_0$ (forme de Horner), proposez une implémentation de `evalue` qui effectue seulement d multiplications et d additions.
2. Écrivez une fonction `derive(int p[],int degre, int retour[])` qui calcule la dérivée du polynôme p et stocke le résultat dans le tableau `retour`.
3. Écrivez une fonction `int somme(int p[], int degrep, int q[], int degreq, int retour[])` qui $p + q$ dans le tableau retour. Quel taille doit avoir ce tableau ? Faites de même pour le produit $p \times q$.

Exercice 3. [Fusion] Dans cet exercice, on va implémenter le tri fusion vu en cours et on adaptera la technique pour calculer le nombre d'inversions du tableau.

1. Supposons qu'on ait deux tableaux t_1 et t_2 triés de taille n_1 et n_2 respectivement. Comment peut-on fusionner les deux tableaux en un tableau t trié en faisant $n_1 + n_2$ opérations ?
2. Implémentez une fonction `fusion(int t[], int i, int j, int k)` qui modifie `t` pour que si les sous-tableaux `t[i..j]` et `t[j+1..k]` sont triés alors à la fin de la fonction, le sous-tableau `t[i..k]` est trié. Utilisez pour cela un tableau annexe de taille $i - k + 1$,
3. Implémentez une fonction `tri_fusion_aux(int t[],int i, int j)` qui trie le tableau `t` entre les indices `i` et `j` récursivement : avec un appel récursif, vous triez la première moitié du tableau puis avec un autre appel récursif, vous triez la deuxième moitié du tableau. Puis vous utilisez votre fonction `fusion` pour trier tout le tableau.
4. En déduire une fonction `tri_fusion` qui trie un tableau de taille n.

Pour le 19 octobre

On appelle une inversion dans un tableau `t`, deux entiers i et j tels que $i < j$ et `t[i]>t[j]`. Supposons que t_1 et t_2 soient deux tableaux triés. Comment modifier votre fonction `fusion` pour qu'elle renvoie le nombre d'inversion de la concaténation de t_1 et t_2 ? En déduire un algorithme pour compter le nombre d'inversion d'un tableau en temps $O(n \log(n))$.