TD Info n°5

PCSI 2 Lycée Pasteur

4 octobre 2007

Le but de ce TD est de manipuler des tableaux en Maple, même si la plupart des exercices proposés ici vont en fait uniquement recoder des fonctions Maple déjà existantes.

1 Quelques éléments sur la complexité des algorithmes

La complexité d'un algorithme correspond grossièrement au temps qu'il va mettre à s'exécuter (complexité temporelle) ou à la place qu'il va prendre en mémoire (complexité spatiale). C'est surtout la première qui va nous intéresser pour l'instant. On peut la mesurer en comptant le nombre d'opérations qui seront effectuées lors de l'exécution de l'algorithme. La complexité est habituellement exprimée en fonction de la taille des données qui seront traitées par l'algorithme. Comme dans ce TD nous manipulerons des données qui seront des tableaux, on notera n le nombre d'éléments du tableau. Quelques définitions pour fixer les idées :

Définition 1. *Un algorithme est dit linéaire si son temps d'exécution est proportionnel à la taille n des données. Il est dit polynomial si son temps d'exécution est proportionnel à une puissance de n (on parle notamment d'algorithme quadratique quand le temps d'exécution est proportionnel à n^2). Il est enfin dit exponentiel si son temps d'exécution est proportionnel à a^n pour un certain réel a. On utilise souvent la notation O pour signifier « proportionnel à ». Ainsi, on parlera d'algorithme en $O(n)$ pour désigner un algorithme linéaire.*

Exemple : Un algorithme linéaire qui met une seconde à traiter un tableau à 100 éléments mettra 100 secondes à traiter un tableau à 10 000 éléments. Un algorithme quadratique qui met également une seconde à traiter le cas d'un tableau à 100 éléments mettra 10 000 secondes (soit près de trois heures) à traiter le cas du tableau à 10 000 éléments. Un algorithme en $O(2^n)$ qui mettrait aussi une seconde avec 100 éléments mettrait 2^{100} secondes à traiter ne serait-ce que 200 éléments, soit environ 4.10^{22} années (4000 milliards de milliards d'années).

2 Exemples avec des polynomes

On considère dans cette section un tableau à $n+1$ éléments supposé contenir les coefficients d'un polynome de P degré n.

1. Écrire un algorithme permettant de calculer le polynome dérivé de P (c'est-à-dire de créer un tableau stockant les coefficients de P'). Quel est le nombre d'opérations nécessaires ?
2. Écrire un premier algorithme permettant d'évaluer le polynome en un réel x donné, c'est-à-dire de calculer $P(x)$. Quel est le nombre d'opérations utilisées ?
3. Écrire un nouvel algorithme utilisant le procédé de Hörner décrit ci-dessous. Cette fois-ci, combien d'opérations sont nécessaires ?

Le procédé de Hörner permet d'évaluer les polynomes de façon légèrement plus rapide que la procédure « normale ». La méthode naturelle est de calculer $P(x) = p_n x^n + \cdots + a_1 x + a_0$ en calculant les puissances successives de x puis en les multipliant par les coefficients p_n, et enfin de faire la somme. Le procédé de Hörner consiste à faire l'opération suivante : $p_0 + x(p_1 + x(p_2 + x(\dots(p_n + x))))$. Vérifier que la valeur calculée est bien égale à $P(x)$.

3 Coefficients binomiaux

Exercice : Ecrire un programme calculant les coefficients binomiaux (tous les $\binom{n}{k}$ pour une valeur de n demandée à l'utilisateur) à partir de leur définition, en commençant par créer un tableau permettant de stocker les valeurs des factorielles.

Exercice : Ecrire un deuxième programme qui utilise la relation de Pascal.

Comparer les deux programmes. À votre avis, lequel est le plus efficace ?