TD 5 - Multiplications rapide

Retrouvez tous les énoncés et les corrections des TP sur ma page personnelle :

`http://perso.ens-lyon.fr/hadrien.croubois/`

Les polynômes

Dans cette première partie nous nous intéresserons à implémenter efficacement les opérations sur les polynômes. Les polynômes serons décrits par le type :

```
type poly == int list;;
```

Question 1 : Comment représenter les polynômes 0, X^2-1, $3X^7-8X^5+64X^4$?

Question 2 : Écrire une fonction `degre` qui renvoi le degré du polynôme

```
val degre :  poly → int
```

Question 3 : Ecrire une fonction d'évaluation de polynôme

```
val eval :  poly → int → int
```

Multiplication naïve

On s'intéresse ici à la multiplication selon la formule théorique :

$$P.Q = \sum_{n\geq 0}\sum_{j+k=n} p_j.q_k X^n = \sum_{j\geq 0} p_j X^j.Q$$

Question 4 : Écrire une fonction `monomial_mult p a n` réalise la multiplication du polynôme p par le monôme $a.X^n$

```
val monomial_mult :  poly → int → int → poly
```

Question 5 : Ecrire une fonction `poly_mult` qui réalise la multiplication de polynôme selon la méthode naïve.

```
val poly_mult :  poly → poly → poly
```

Quel est sa complexité ?

Ainsi multipliais Karatsuba

Pour diminuer la complexité de la multiplication de polynômes, on va utiliser le paradigme "diviser pour régner". Pour cela, on va décomposer les polynômes à multiplier P et Q de degrés strictement inférieurs à $2n$ en

$$P = P_1 + P_2.X^n \quad \text{et} \quad Q = Q_1 + Q_2.X^n$$

avec les degrés de P_1, P_2, Q_1 et Q_2 strictement inférieurs à n.

Question 6 : Ecrire une formule qui réduit la multiplication des polynômes P et Q de degrés strictement inférieurs à $2n$ en multiplications de polynômes de degrés strictement inférieurs à n.

Question 7 : Programmer un algorithme récursif de multiplication qui utilise la formule précédente.

```
val DaC_mult :  poly → poly → poly
```

Quelle est sa
complexité ?

On peut raffiner cette méthode avec la remarque suivante de Karatsuba : le terme intermédiaire de $P.Q$ s'écrit

$$P1.Q2 + P2.Q_1 = (P_1+P_2).(Q_1+Q_2) - P_1.Q_1 - P_2.Q_2$$

On échange donc deux multiplications et une addition contre une multiplication et quatre additions.

Question 8 : Écrire une fonction qui réalise la multiplication de polynômes à la Karatsuba.

Question 9 : Trouver la formule de récurrence qui définit la complexité de la multiplication de Karatsuba. Quelle est sa solution ?

Encore mieux ...

Question 10 : Pouvez vous faire mieux? (objectif $\Theta(n \log n)$)

Les matrices

Question 11 : Comparez les resultats du code suivant :

```
let m = make_vect 10 (make_vect 10 0) in
  m.(0).(0) <- 1;
  m;;
```

Pourriez vous expliquer ce résultat ? Quel solution pour y remédier ?

Question 12 : En vous inspirant de la méthode de multiplication rapide développé pour les vecteurs, concevoir un algorithme récursif de multiplication rapide de matrice. Vous justifierai vos choix.

Quel est la complexité de votre algorithme ?