Université de Bordeaux — **Algèbre et calcul formel – Agrégation**
Mathématiques — 2014–2015

FEUILLE D'EXERCICES n° 4

Multiplication de polynômes : Karatsuba, FFT

On propose ici d'écrire des procédures de multiplication de polynômes. Pour cela, on définira les polynômes comme des listes et on utilisera les transformations liste-polynôme et polynôme-liste.

Dans Sage, si `P` est un polynôme (donc défini dans un anneau de polynômes), on obtient son i-ème coefficient en tapant `P[i]`. C'est donc la même syntaxe que pour les listes. On pourrait donc écrire ces procédures en définissant les polynômes directement dans un anneau de polynômes (sans utiliser les opérations de sage sur les polynômes pour que l'exercice garde de l'intérêt).

Une fois cette feuille terminée, on pourra recommencer en prenant ce parti. Les modifications à apporter seront bien sûr mineures, voire inexistantes suivant les commandes utilisées. Certaines commandes fonctionnent différemment pour les listes et les polynômes (essayer par exemple `P[0:3]`).

Exercice 1 – [Karatsuba]
On désire calculer le produit de deux polynômes $P, Q \in R[X]$ de degrés $< n$, où R est un anneau commutatif. L'approche naïve a une complexité algébrique de $O(n^2)$ opérations dans R. Une façon d'améliorer ce résultat est la suivante. Considérons nos polynômes comme des polynômes de degré $< 2^s$ où s est le plus petit entier tel que $n \leqslant 2^s$, i.e. $s = \lceil \log_2 n \rceil$. Supposons $s > 0$ et écrivons

$$P = X^{2^{s-1}} P_1 + P_2 \quad \text{et} \quad Q = X^{2^{s-1}} Q_1 + Q_2,$$

où P_1, P_2, Q_1 et Q_2 sont des polynômes de degré $< 2^{s-1}$. On a alors

$$\begin{aligned} PQ &= X^{2^s} P_1 Q_1 + X^{2^{s-1}} (P_1 Q_2 + P_2 Q_1) + P_2 Q_2 \\ &= X^{2^s} P_1 Q_1 + X^{2^{s-1}} \left((P_1 + P_2)(Q_1 + Q_2) - P_1 Q_1 - P_2 Q_2 \right) + P_2 Q_2, \end{aligned}$$

de telle sorte que nous avons juste à calculer trois produits

$$A = P_1 Q_1, \quad B = P_2 Q_2 \quad \text{et} \quad C = (P_1 + P_2)(Q_1 + Q_2)$$

de polynômes de degré(s) $< 2^{s-1}$. On utilise cette idée de façon récursive, ce qui conduit à un algorithme dont la complexité algébrique est en $O(n^{\log_2 3})$.

1) Soit donc n une puissance de 2. Écrire une procédure récursive `Karatsuba(`P, Q, n`)` utilisant le principe rappelé ci-dessus et renvoyant le polynôme PQ.

Dans cette procédure, P et Q seront rentrés comme des listes, et le polynôme PQ obtenu sera également une liste.

2) Écrire ensuite une procédure `MulK(A,P,Q)` qui, étant donnés deux polynômes P et Q définis dans l'anneau de polynômes `A` utilise `Karatsuba` pour donner le produit PQ, lui aussi défini dans `A`.

3) Tester cette procédure sur des polynômes symboliques de degré 3.

4) La tester numériquement avec de gros polynômes pris au hasard. N.B. si `A` est un objet défini comme un ensemble éventuellement muni d'une structure, on peut obtenir un élément au hasard dans A en utilisant `A.random_element`).

Exercice 2 – [FFT]
Soit $n = 2^k$ une puissance de 2, avec $k > 0$. Soit ω une racine primitive n-ième de l'unité, par exemple $\omega = e^{2i\pi/n}$. On définit un isomorphisme de $\mathbb{C}$-algèbres $\mathcal{F}_\omega : \mathbb{C}[X]/(X^n - 1) \to \mathbb{C}^n$ par

$$\mathcal{F}_\omega(R) = (R(1), R(\omega), \ldots, R(\omega^{n-1})).$$

On identifiera une classe $\bar{R} \in \mathbb{C}[X]/(X^n - 1)$ avec son représentant $R \in \mathbb{C}[X]$ de degré $< n$, ainsi qu'avec le n-uplet $(R_0, \ldots, R_{n-1})$ de ses coefficients, dans $\mathbb{C}^n$.

On évalue $\mathcal{F}_\omega(R)$ en calculant récursivement deux $\mathcal{F}$ de degrés $< m = n/2$ par le biais des formules

$$\begin{aligned} R(\omega^p) &= \sum_{j=0}^{m-1} R_{2j}\alpha^{jp} + \omega^p \sum_{j=0}^{m-1} R_{2j+1}\alpha^{jp} \\ R(\omega^{p+m}) &= \sum_{j=0}^{m-1} R_{2j}\alpha^{jp} - \omega^p \sum_{j=0}^{m-1} R_{2j+1}\alpha^{jp}, \end{aligned}$$

où $0 \leqslant p < m$ et où $\alpha = \omega^2$.

Implanter l'algorithme récursif s'appuyant sur cette remarque. La procédure `FFT` recevra en entrées R, ω et n, et retournera $\mathcal{F}_\omega(R)$. Là encore, le polynôme R sera défini par une liste. On prendra garde à ne pas calculer ω^p à chaque étape de la boucle sur p. Pour cela, on pourra par exemple les stocker en amont.

Exercice 3 – [Produit rapide de polynômes par FFT]
Ici encore $n = 2^k$ avec $k > 0$. Soient P et Q deux polynômes de $\mathbb{C}[X]$ vérifiant $\deg(PQ) < n$. On identifiera encore P et Q aux n-uplets $(P_0, \ldots, P_{n-1})$ et $(Q_0, \ldots, Q_{n-1})$ formés de leurs coefficients. On rappelle que l'on a alors

$$\mathcal{F}_\omega(PQ) = \mathcal{F}_\omega(P) \cdot \mathcal{F}_\omega(Q)^1,$$

et que pour tout polynôme $R \in \mathbb{C}[X]$ de degré $< n$ on a

$$\mathcal{F}_{\omega^{-1}}(\mathcal{F}_\omega(R)) = \mathcal{F}_\omega(\mathcal{F}_{\omega^{-1}}(R)) = nR.$$

Écrire une procédure prenant en arguments P, Q et n et retournant PQ, procédure qui prendra bien sûr appui sur la procédure `FFT` de l'exercice précédent.

Remarque. Pour s'assurer que $\deg(PQ) < n$ on pourra imposer à P et Q d'être tous deux de degrés $< m = n/2$.

[1] ici $(u_i)_{0\leqslant i<n} \cdot (v_i)_{0\leqslant i<n} = (u_i v_i)_{0\leqslant i<n}$.