Algorithmique et Programmation

TD n° 8 : Polynômes et FFT

École normale supérieure – Département d'informatique
algoL3@di.ens.fr

2014-2015

Exercice 1 MULTIPLICATION DE POLYNÔMES BIVARIÉS

Donner un algorithme qui multiplie deux polynômes (representés par leur coefficients) $P, Q \in \mathbb{C}[X, Y]$ de degrés inférieurs à n en $O(n^2 \log n)$ opérations élémentaires dans $\mathbb{C}$.

Exercice 2 DIVISION EUCLIDIENNE RAPIDE PAR LA MÉTHODE DE NEWTON

Soit A un anneau commutatif unitaire (en pratique nous considérerons $A = \mathbb{Z}$ ou $A = \mathbb{Z}_N$). Soient $S, T \in A[X]$ avec $\deg(S) = n$, $\deg(T) = m$ et T unitaire.

1. Montrer que l'algorithme classique de division euclienne de S par T a une complexité arithmétique en $O(n^2)$ (*i.e.* le nombre d'opérations nécessaires dans l'anneau A est en $O(n^2)$).
2. Pour $P \in A[X]$ et $k = \deg(P)$, nous notons $\mathrm{Rec}(P(X)) = X^k P(1/X)$ le polynôme réciproque de P. Montrer que
$$\mathrm{Rec}(Q) = \mathrm{Rec}(S)\mathrm{Rec}(T)^{-1} \bmod X^{n-m+1},$$
où Q est le quotient de la division euclidienne de S par T.
3. Soit $F \in A[X]$ avec $F(0) = 1$. Considérons la suite de polynômes $G_i \in A[X]$ définie par $G_0 = 1$ et
$$G_{i+1} = 2G_i - F \cdot G_i^2 \bmod X^{2^{i+1}}$$
pour $i \geq 0$. Montrer que pour tour entier $i \geq 0$, nous avons
$$F \cdot G_i \equiv 1 \bmod X^{2^i}.$$
4. En déduire un algorithme pour calculer Q et le reste R de la division euclidienne de S par T.
5. Montrer que la complexité arithmétique de cet algorithme de division euclidienne appliqué à deux polynômes de degrés $< n$ est en $O(M(n))$ où $M(n)$ est la complexité arithmétique du produit de deux polynômes de degré $< n$ de $A[X]$ (avec $M(n_1 + n_2) \geq M(n_1) + M(n_2)$ pour $n_1, n_2 \in \mathbb{N}$).

Exercice 3 MULTI-ÉVALUATION D'UN POLYNÔME UNIVARIÉ

Soit A un anneau commutatif unitaire (en pratique nous considérerons $A = \mathbb{Z}$ ou $A = \mathbb{Z}_N$). Soient $a_1, \ldots, a_d \in A$. Nous notons $M(d)$ la complexité arithmétique du produit et de la division euclidienne de deux polynômes de degré $< d$ de $A[X]$ (avec $M(d_1 + d_2) \geq M(d_1) + M(d_2)$ pour $d_1, d_2 \in \mathbb{N}$).

1. Supposons que $d = 2^k$ est une puissance de 2 et considérons l'arbre binaire complet T à d feuilles défini par :
   - chacune des d feuilles est associée à un polynôme $X - a_j$ pour $j \in \{1, \ldots, d\}$;
   - pour chaque nœud interne u ayant les fils v et w associés aux polynômes $M_v(X)$ et $M_w(X)$ respectivement, u est associé au polynôme $M_u(X) = M_v(X) \cdot M_w(X)$.

   Donner un algorithme pour construire l'arbre T avec une complexité arithmétique en $O(M(d) \log d)$.
2. Soit $P \in A[X]$ unitaire avec $\deg(P) = d$. Donner un algorithme qui prenant en entrée P, $(a_1, \ldots, a_d)$ et l'arbre T, calcule $P(X) \bmod M_u(X)$ pour tout $u \in T$, avec une complexité arithmétique en $O(M(d) \log d)$.
3. Déduire des questions précédentes un algorithme qui calcule $P(a_1), \ldots, P(a_d)$ pour *tout* $d \in \mathbb{N}$ avec une complexité arithmétique en $O(M(d) \log d)$.

Exercice 4 RECHERCHE DE MOTIFS

Le but de cet exercice est de proposer des algorithmes *numériques* de recherche de motifs (classique ou avec *jokers*). Nous supposerons sans perte de généralité que l'alphabet $\Sigma = \{1, \ldots, \sigma\} \subset \mathbb{N}$ est un ensemble d'entiers consécutifs.

1. (a) En remarquant que le motif $P = (p_1, \ldots, p_m) \in \Sigma^m$ apparaît à la position $i \in \{1, \ldots, n\}$ de $T = (t_1, \ldots, t_n) \in \Sigma^n$ si et seulement si

$$\sum_{j=1}^{m} (p_j - t_{i+j-1})^2 = 0$$

proposer un algorithme de complexité $O(n \log n)$ opérations arithmétiques pour le problème de recherche de motifs (classique).

(b) En découpant le texte T en n/m blocs de longueur $2m$ (se recouvrant partiellement), montrer comment adapter l'algorithme précédent pour avoir une complexité en $O(n \log m)$ opérations arithmétiques.

2. Adapter cet algorithme pour la recherche de motifs lorsque le motif P et le texte T contiennent des caractères spéciaux *jokers* (?), *i.e.* qui détermine l'ensemble

$$\mathcal{M}_{??} = \{i \in \{1, \ldots, n\} | \forall j \in \{1, \ldots, m\}, (p_j = t_{i+j-1}) \vee (p_j = ?) \vee (t_{i+j-1} = ?)\}.$$

3. Nous allons adapter cet algorithme pour qu'il détermine le nombre de lettres communes d'un motif à chaque position d'un texte.

(a) Donner un algorithme en $O(n\sigma \log m)$ opérations arithmétiques qui prenant en entrée P et T retourne un vecteur $H(P, T) = (h_1, \ldots, h_{n-m+1}) \in \mathbb{N}^{n-m+1}$ tel que $h_i = \sum_{j=1}^{m} \mathrm{eq}(p_j, t_{i+j-1})$ où $\mathrm{eq}(a, a) = 1$ et $\mathrm{eq}(a, b) = 0$ si $a \neq b$.

(b) Soit $f \in \mathbb{N}$. Une lettre de Σ est dite *f-fréquente* si elle apparaît au moins f fois dans P. En supposant que le motif P ne contient aucune lettre f-fréquente, donner un algorithme de complexité $O(nf)$ (et donc indépendant de σ) qui prenant en entrée P et T retourne le vecteur $H(P, T)$.

(c) En appliquant ces deux algorithmes en fonction de la fréquence des lettres de P, donner un algorithme qui prenant en entrée un motif $P \in \Sigma^m$ arbitraire et un texte $T \in \Sigma^n$ retourne le vecteur $H(P, T)$ en $O(n\sqrt{m \log m})$ opérations arithmétiques.

Exercice 5 (Bonus - non vu en TD) Évaluation de toutes les dérivées d'un polynôme en un point

À partir de la représentation par coefficients d'un polynôme $A(x)$ de degré au plus $n - 1$, on souhaite déterminer $A^{(t)}(x_0)$ pour $t \in [\![0, n-1]\!]$.

1. Connaissant les coefficients $b_0, \ldots, b_{n-1}$ tels que

$$A(x) = \sum_{j=0}^{n-1} b_j (x - x_0)^j$$

montrer comment calculer $A^{(t)}(x_0)$ pour $t \in [\![0, n-1]\!]$ en $O(n)$ opérations.

2. Expliquant comment on peut trouver $b_0, \ldots, b_{n-1}$ en $O(n \log n)$ opérations connaissant $A(x_0 + \omega_n^k)$ pour $k \in [\![0, n-1]\!]$ (où $\omega_n = e^{2i\pi/n}$).

3. Montrer que

$$A(x_0 + \omega_n^k) = \sum_{r=0}^{n-1} \left(\frac{\omega_n^{kr}}{r!} \sum_{j=0}^{n-1} f(j) g(r - j) \right)$$

où $f(j) = a_j \cdot j!$ et

$$g(\ell) = \begin{cases} x_0^{-\ell} f(-\ell) & \text{si } -(n-1) \leq \ell \leq 0 \\ 0 & \text{si } 1 \leq \ell \leq (n-1) \end{cases}$$

4. Expliquer comment on peut évaluer $A(x_0 + \omega_n^k)$ pour $k \in [\![0, n-1]\!]$ en $O(n \log n)$ opérations. En conclure que toutes les dérivées non triviales de $A(x)$ peuvent être évaluées au point x_0 en $O(n \log n)$ opérations.