TUTORIAL 3

1 Recursive division

Let a and b be two polynomials in $K[x]$ such that $\deg a = 4n$ and $\deg b = 2n$ and take n to be a power of 2. We decompose a and b such that $a(x) = a_h(x)x^{2n} + a_l(x)$ and $b(x) = b_h(x)x^n + b_l(x)$, where $\deg a_h, \deg a_l \leqslant 2n$ and $\deg b_h, \deg b_l \leqslant n$.

Consider $D(n)$ as the complexity, in number of arithmetic operations over K, required to perform the euclidean division of a degree $2n$ polynomial by a degree n polynomial. Similarly, we denote by $M(n)$ the complexity of multiplying two degree n polynomials over R.

We perform the euclidean division of a_h by b_h (i.e. $a_h = b_h q_h + r_h, \deg r_h < \deg b_h$).

1. Show that $\deg(a - bq_h x^n) < 3n$ and that $a - bq_h x^n$ is computable using $D(n) + M(n) + O(n)$ operations.
2. Show that we can finish dividing a by b using another $D(n) + M(n) + O(n)$ operations.
3. What is the value of $D(n)$ if $M(n) = n^\alpha, \alpha > 1$?
4. Same question as before, for $M(n) = n(\log n)^\alpha, \alpha > 1$.

2 Composition of polynomials

1. What is the cost of computing the coefficients of the composition $f \circ g$ of polynomials f, g of degrees d_1, d_2? (Assume that ring operations have unit cost.) Use that $f(x) = \sum_{i=0}^{n} a_i x^i = a_0 + x(a_1 + x(a_2 + \cdots + x(a_{n-1} + a_n x) \ldots))$.

   Let $N > 0$ be a power of 2 and let A and B be two polynomials over $\mathbb{K}$ with $B(0) = 0$ and $B'(0) \neq 0$. We will study a fast algorithm for computing the composition $A(B) \mod X^N$ which is due to Brent and Kung (1978).

   Let $m > 0$ be a parameter which we will tune later. The algorithm is based on the following Taylor's expansion.

2. Writing $B = B_1 + X^m B_2$ where B_1 is a polynomial of degree $< m$ in $\mathbb{K}[X]$, show that

$$A(B) = A(B_1) + A'(B_1)X^m B_2 + A''(B_1)\frac{X^{2m}B_2^2}{2!} + A^{(3)}(B_1)\frac{X^{3m}B_2^3}{3!} + \cdots$$

   Having this decomposition, let us now observe that once we have computed the composition $A(B_1)$ we can compute the other terms efficiently.

3. Let F and G be two polynomials with $G'(0) \neq 0$, and assume that we have computed $F(G) \mod X^N$. Show how to compute $F'(G) \mod X^N$ using $\mathcal{O}(\mathrm{M}(N))$ operations in $\mathbb{K}$, where $\mathrm{M}(n)$ stands for the complexity of multiplying two polynomials of degree n over $\mathbb{K}$.

4. Denoting by $\mathcal{C}(m, N)$ the number of operations used for computing $A(B_1) \mod X^N$, deduce from the previous question a cost bound for computing $A(B) \mod X^N$.

   To obtain a fast algorithm, it remains to give an efficient method to compute $A(B_1) \mod X^N$. To this end, we will study the following more general situation.

5. Let F and G be polynomials over $\mathbb{K}$ of degrees k and m respectively, with $G(0) = 0$. Give a divide-and-conquer algorithm which computes $F(G) \mod X^N$ using $\mathcal{O}(\frac{km}{N}\mathrm{M}(N)\log(N)\log(k))$ operations in $\mathbb{K}$.

6. Deduce an upper bound for $\mathcal{C}(m, N)$, and a cost bound for computing $A(B) \mod X^N$. Conclude by giving the whole algorithm, including a good choice of m and the corresponding cost bound.

3 Logarithm and exponential

For polynomials $S, T \in \mathbb{K}[X]$ such that $S(0) = 0$ and $T(0) = 0$ we define

$$\exp_n(S(X)) = \sum_{k=0}^{n-1} \frac{S(X)^k}{k!} \bmod X^n$$

$$\log_n(1 + T(X)) = \sum_{k=1}^{n-1} (-1)^{k+1} \frac{T(X)^k}{k} \bmod X^n$$

1. Assume $A(0) = 0$, prove that

$$(A(X) + 1)^{-1} = \sum_{k=0}^{m-1} (-1)^k A(X)^k \bmod X^m$$

2. Recall that $S(0) = 0$. Let $U_n(X) = S'(X)/(S(X) + 1) = \sum_{k=0}^{n-2} u_k X^k \bmod X^{n-1}$ (remark that $S(X) + 1$ is invertible modulo X^n because $S(0) + 1 \neq 0$). Prove that

$$\log_n(1 + S(X)) = \sum_{k=1}^{n-1} u_{k-1} \frac{X^k}{k} \bmod X^n$$

   (Hint: use question 1).

3. Deduce a quasi-linear time algorithm to compute $\log_n(S(X) + 1)$.

4. Prove that if $T(0) = 0$, then $\log_n(\exp_n(T(X)) = T(X)$ (remark that this is well defined because $\exp_n(T(0)) = 1$). (Hint: derive the two terms of the expression above).

5. Let $Y = \exp_N(T(X)) - 1 \mod X^N$. Using question above, we have that

$$f(Y) = \log_N(1 + Y) - T(X) = 0 \mod X^N.$$

   Using Hensel lifting, deduce an algorithm computing $Y = \exp_N(T(X)) - 1 \mod X^N$ in time $O(M(N))$. (Hint: remember that as M is super-linear we have that $M(N) + M(N/2) + \cdots + M(N/2^k) + \cdots \leq 2M(N)$).