Fast Multiplication

(Due date: Sunday May 1th)

1. Let R be a commutative ring, with $1 \in R$, and $f, g \in R[x, y]$. Assume that f and g have degrees bounded by m in y and n in x. Let $h = f \cdot g$.

   (a) Using classical univariate polynomial multiplication, and viewing $R[x, y]$ as $R[y][x]$, bound the number of arithmetic operations in R to compute h.

   (b) Using Karatsuba's algorithm bound the number of operations in R to compute h.

   (c) Generalize parts (a) and (b) to polynomials in an arbitrary number of variables.

2. Karatsuba's method for polynomial multiplication can be generalized as follows. Let F be a field, $m, n \in \mathbb{Z}_{>0}$, and $f = \sum_{i=0}^{n} f_i x^i$, $g = \sum_{i=0}^{n} g_i x^i$ in $F[x]$. To multiply f and g, we divide each of them into $m \geq 2$ blocks of size $k = \lceil (n+1)/m \rceil$:
$$f = \sum_{i=0}^{m} F_i x^{ki}, \; g = \sum_{i=0}^{m} G_i x^{ki},$$
with all F_i, $G_i \in F[x]$ of degree less than k. Then $fg = \sum_{i=0}^{2m-1} H_i x^{ki}$, where $H_i = \sum_{j=0}^{i} F_j G_{i-j}$ for $0 \leq i < 2m-1$, and we assume that F_j, $G_j = 0$ if $j \geq m$.

   (a) Find a way to compute H_0, H_1, H_2, H_3 and H_0 when $m = 3$ using at most 6 multiplications of polynomials of degree less than k. Use this method to construct a recursive algorithm à la Karatsuba and analyze its cost when n is a power of 3 (count only the number of multiplications of polynomials which you have to perform with this algorithm).

   (b) Suppose that you have found a scheme to compute $H_0, \ldots, H_{2m-2}$ using d multiplications of polynomials of degree less than k, and made this scheme into a recursive algorithm as in (a). How large may d be at most such that your algorithm is assymptotically faster than Karatsuba's? Compare with your result from (a).

3. Let $F = \mathbb{F}_{17}$ and $f = 5x^3 + 3x^2 - 4x + 3$, $g = 2x^3 - 5x^2 + 7x - 2$ in $F[x]$.

   (a) Show that $\omega = 2$ is a primitive 8th root of unity in F, and compute the inverse 2^{-1} modulo 17 of ω in $\mathbb{F}$.

   (b) Compute $h = f \cdot g \in F[x]$.

   (c) For $0 \leq j < 8$, compute $\alpha_j = f(\omega^j)$, $\beta_j = g(\omega^j)$, and $\gamma_j = \alpha_j \cdot \beta_j$. Compare γ_j with $h(\omega^j)$.

(d) Show the two matrices $V_1 = V_\omega$ and $V_2 = 8^{-1}V_{\omega^{-1}}$, and compute their product. Compute the matrix-vector products $V_2\alpha$, $V_2\beta$, and $V_2\gamma$, with $\alpha = (\alpha_0, \ldots, \alpha_7)$, $\beta = (\beta_0, \ldots, \beta_7)$, $\gamma = (\gamma_0, \ldots, \gamma_7)$.

(e) Trace the FFT multiplication algorithm to multiply f and g, with ω as above.

4. Let $F = \mathbb{F}_{41}$.

(a) Prove that $\omega = 14 \in F$ is a primitive 8th root of unity. Compute all powers of ω, and mark the ones that are primitive 8th roots of unity.

(b) Let $\eta = \omega^2$, and $f = x^7 + 2x^6 + 3x^4 + 2x + 6 \in F[x]$. Give an explicit calculation of $\alpha = DFT_\omega(f)$, using the FFT. You only have to do one recursive step, and then can use direct evaluation at powers of η.

(c) Let $g = x^7 + 12x^5 + 35x^3 + 1 \in F[x]$. Compute $\beta = DFT_\omega(g)$, $\gamma = \alpha \cdot \beta$ with coordinate-wise product, and $h = DFT_{\omega^{-1}}(\gamma)$.

(d) Compute $f \cdot g$ in $F[x]$, and $f *_8 g$. Compare with your result from (c).

5. Let R be a ring, $n \in \mathbb{Z}_{>0}$ and $\omega \in R$ be a primitive nth root of unity.

(a) Show that ω^{-1} is also a primitive nth root of unity.

(b) If n is even, then show that ω^2 is a primitive $n/2$th root of unity. If n is odd, then show that ω^2 is also a primitive nth root of unity.

(c) Let $k \in \mathbb{Z}$ and $d = n/\gcd(n,k)$. Show that ω^k is a primitive dth root of unity.

6. Let $n \in \mathbb{Z}_{>0}$ and R be an integral domain of characteristic coprime with n.

(a) Show that the set R_n of all nth roots of unity is a subgroup of the multiplicative group $R^\times$.

(b) Prove that the following are equivalent for an nth root of unity $\omega \in R$:

i. ω is a primitive nth root of unity,

ii. $\omega^\ell \neq 1$ for $0 < \ell < n$,

iii. $\omega^\ell \neq 1$ for all $\ell | n$, $0 < \ell < n$,

iv. $\omega^{n/p} \neq 1$ for all prime divisors p of n.

We now assume that R contains a primitive nth root of unity ω.

(c) Show that R_n is cyclic and isomorphic to the additive group $\mathbb{Z}_n$ of integers modulo n.

(d) Prove that there are precisely $\varphi(n)$ primitive nth roots of unity, where φ is the Euler's function.

7. Let q be a prime power, $\mathbb{F}_q$ a finite field with q elements, and $n \in \mathbb{N}$ a divisor of $q-1$ with prime factorization $n = p_1^{e_1} \dots p_r^{e_r}$. For $a \in \mathbb{F}_q^\times$, we denote by $ord(a)$ the order of a in the multiplicative group $\mathbb{F}_q^\times$, and want to show that $ord(a) = q-1$ for some $a \in \mathbb{F}_q^\times$. Prove
   (a) $ord(a) = n$ if and only if $a^n = 1$ and $a^{n/p_j} \neq 1$ for $1 \leq j \leq r$.
   (b) $\mathbb{F}_q^\times$ contains an element b_j of order $p_j^{e_j}$, for $1 \leq j \leq r$.
   (c) If $a,\, b \in \mathbb{F}_q^\times$ are elements of coprime order, then $ord(ab) = ord(a) \cdot ord(b)$.
   (d) $\mathbb{F}_q^\times$ contains an element of order n.
   (e) $\mathbb{F}_q^\times$ is cyclic.

8. Prove that for a primer power q and $n \in \mathbb{N}$, a finite field $\mathbb{F}_q$ contains a primitive nth root of unity if and only if n divides $q-1$.

9. Let F be a field supporting the FFT, and $a, b, q, r \in F[x]$ such that $a = qb + r$ and $\deg r < \deg b \leq \deg a < n$ for a power n of 2. We assume that b is coprime to $x^n - 1$. Give an algorithm which on input a, b decides whether $r = 0$, and if so, computes the quotient q using essentially three $n-$point FFTs.

10. A different approach to DFT is to split f into its odd and even parts, that is, to write $f = f_0(x^2) + x f_1(x)^2$, with $f_0,\, f_1 \in R[x]$ of degree less than $n/2$, and then to compute $DFT_{\omega^2}(f_0)$ and $DFT_{\omega^2}(f_1)$ recursively. Work out the details and prove that your algorithm uses $cn \log n$ operations in R for some positive constant $c \in \mathbb{Q}$ when n is a power of 2. Modify, if necessary, your algorithm so that $c = 3/2$.

11. Let R be a ring (commutative, with 1) containing a primitive 3^kth root of unity for any $k \in \mathbb{N}$.
    (a) Designa 3-adic FFT algorithm, taking as input $k \in \mathbb{N}$, a polynomial $f \in R[x]$ of degree less than $n = 3^k$, and a list of powers $1,\, \omega,\, \omega^2,\, \dots,\, \omega^{n-1}$ of a primitive nth root of unity $\omega \in R$, and returning $f(1),\, f(\omega),\, \dots,\, f(\omega^{n-1})$. Prove the correctness of your algorithm.
    (b) Let $T(n)$ denote the cost of your algorithm in operations in R when $n = 3^k$ for some $k \in \mathbb{N}$. Set up a recursion for $T(n)$ and solve it.
    (c) Assuming that R contains primitive nth root of unity for any $n \in \mathbb{N}$, generalize the above to an m-adic FFT algorithm for arbitrary $m \in \mathbb{Z}_{\geq 2}$.

12. Let F be a field containing a primitive 2^kth root of unity for all $k \in \mathbb{N}$. Let $f,\, g \in F[x]$ and $m \in \mathbb{N}$ be a power of 2 such that $m/2 < 2n \leq m$, and set $a = f(x^{m-n} + 1)$ and $b = g$. Show how to obtain the coefficients of $f *_n g$ from those of $a *_m b$.