Assignment 1

1/ and 2/ - Due November 4th (extended), 2015

Your implementations may be in the form of a maple worksheet or of maple code in plain text files (in a single tarball).

1/ A matrix Toeplitz is a matrix in which each diagonal is constant, for instance:

$$T = \begin{bmatrix} f_3 & f_2 & f_1 & f_0 \\ f_4 & f_3 & f_2 & f_1 \\ f_5 & f_4 & f_3 & f_2 \\ f_6 & f_5 & f_4 & f_3 \end{bmatrix}.$$

a/ Let R be a ring supporting the FFT. If T is square $n \times n$ with entries in R, and g is a column vector in R^n, then show that the matrix times vector product $T \cdot u$ can be computed using one univariate polynomial multiplication. What is the corresponding cost (with an explicit constant in front of the dominating term) using Karatsuba's algorithm? Using the FFT?

b/ We consider the following algorithm[1]:

Algorithm 1. "Middle Product"
Input: $f = f_0 + f_1 x + f_2 x^2$, and $g = g_0 + g_1 x$
1. $m_1 := (f_0 + f_1) g_1$
2. $m_2 := (f_1 + f_2) g_0$
3. $m_3 := f_1 (g_1 - g_0)$
1. $h_1 := m_1 - m_3$
5. $h_2 := m_2 + m_3$
Ouput: h_1, and h_2.

Given f of degree less than $2n-1$ and g of degree less than n in R^n, Show that Algorithm 1 can be used recursively for computing the n coefficients of $x^{n-1}, x^n, \ldots, x^{2n-2}$ in the polynomial $h = fg$. What is the corresponding cost (assuming that n is a power of 2)? Conclude that you can compute the product $T \cdot u$ faster than in question a/ using Karatsuba's algorithm.

c/ Improve the cost given in a/ using the FFT.

Implementation. Given f and g as in b/, implement the recursive algorithm for computing the coefficients of $x^{n-1}, x^n, \ldots, x^{2n-2}$ in the product fg.

Note. Another version of the middle product algorithm is given in the "transposition principle" worksheet.

---

[1] G. Hanrot , M. Quercia, and P. Zimmermann, *The Middle Product Algorithm I*, Applicable Algebra in Engineering, Communication and Computing, 14, 6, 415–438, 2004.

2/ Let R be a commutative ring such that $n!$ is invertible in R. We study Brent & Kung's algorithm[2] for the composition modulo powers of x. We consider two polynomials (truncated power series) f and g in $\mathsf{R}[x]$ of degree less than n, with $g'(0)$ invertible.

a/ For f and g given of degree less than n in $\mathsf{R}[x]$, with $g'(0)$ invertible in R, and knowing $f(g) \mod x^n$, show that $f'(g) \mod x^n$ can be computed in $O(\mathrm{M}(n))$ operations.
Hint: use the chain rule $(f \circ g)' = (f' \circ g)g'$.

We write $g = g_0 + g_1 x^m$ with g_0 of degree less than m, and let $k = \lceil n/m \rceil$, and consider the Taylor expansion:

$$f(g_0 + g_1 x^m) \equiv f(g_0) + f'(g_0)x^m g_1 + \frac{f''(g_0)}{2!}x^{2m}g_1^2 + \dots \mod x^n. \tag{1}$$

b/ Use a/ to prove that $f(g) \mod x^n$ can be computed at the cost of computing $f(g_0) \mod x^n$ plus $O(k\mathrm{M}(n))$ operations.

c/ With f of degree less than d, such that d is a power of two, and g_0 of degree less than m, devise a divide-and-conquer algorithm for computing $f(g) \mod x^n$, with cost $O(\mathrm{M}(n)\log n)$ if $dm \leq n$, and $O((dm/n)\mathrm{M}(n)\log n)$ in general.
Hint: divide f into two blocks of size $d/2$.

d/ Prove that $f(g) \mod x^n$ can be computed in $O((m\log n + k)\mathrm{M}(n))$ operations in R. Which choice of m minimizes the bound?

Implementation. Given f, g, give a recursive procedure for c/; a procedure for the whole composition (do not rewrite polynomial multiplication and power series inversion).

[2] R.P. Brent, and H.T. Kung. Fast algorithms for manipulating formal power series. Journal of the ACM, 25, 4, 581–595, 1978. See also Exercise 12.4 in "Modern Computer Algebra".