Fundamental algorithms

(Due date: Sunday March 5th)

1. For a positive integer r, we consider the variable-length radix r representation $(a_0, \ldots, a_{l-1})$ of a positive integer a, with $a = \sum_{i=0}^{l-1} a_i r^i$, $a_i \in \{0, \ldots, r-1\}$, and $a_{l-1} \neq 0$. Prove that its length l is $\lfloor \log_r a \rfloor + 1$.

2. Give an algorithm for multiplying a multiprecision integer b by a single precision integer a, making use of the single precision multiply instruction. Show that your algorithm uses $\lambda(b)$ single precision multiplications, and the same number of single precision additions.

3. Prove that $\max\{\lambda(a), \lambda(b)\} \leq \lambda(a+b) \leq \max\{\lambda(a), \lambda(b)\} + 1$, and $\lambda(a) + \lambda(b) - 1 \leq \lambda(ab) \leq \lambda(a) + \lambda(b)$ for all $a, b \in \mathbb{Z}_{>0}$.

4. Show that one can perform the classical Euclidean algorithm to compute the gcd of two integers a and b such that $\lambda(a), \lambda(b) \leq \lambda$ with complexity $\mathcal{O}(\lambda^3)$.

5. Let $n \in \mathbb{N}$ and consider $\mathbb{Z}/n\mathbb{Z}$, the ring of congruences modulo n. Show that the versions "modulo n" of the classic algorithms for addition and multiplication have complexity $\mathcal{O}(\log_2 n)$ and $\mathcal{O}(\log_2^2 n)$ respectively.

6. Consider the following algorithm given in pseudocode notation:

   Input: The coefficients of $a = \sum_{i=0}^{n} a_i x^i$ and $b = \sum_{i=0}^{m} b_i x^i$ in $R[x]$, where R is a commutative ring.

   Output: The coefficients of $c = a \cdot b \in R[x]$.

   1. for $i = 0, \ldots, n$ **do** $d_i \leftarrow a_i x^i \cdot b$

   2. return $c = \sum_{i=0}^{n} d_i$

   (a) Show that this algorithm computes correctly the product $c = a \cdot b$.

   (b) Show that the number of multiplications and additions of elements of R performed by this algorithm is $(n+1)(m+1)$ and nm, respectively. Conclude that the total number of arithmetic operations of this algorithm is bounded by $2(n+1)(m+1)$.

7. Consider the following algorithm given in pseudocode notation:

   Input: $a = \sum_{i=0}^{n} a_i x^i$, $b = \sum_{i=0}^{m} b_i x^i$, with all $a_i, b_i \in R$, where R is a commutative ring, b_m is a unit of R, and $n \geq m \geq 0$.

   Output: $q, r \in R[x]$ with $a = qb + r$ and $\deg r < m$.

   1. $r \leftarrow a$**,** $u \leftarrow b_m^{-1}$

   2. for $i = n-m, n-m-1, \ldots, 0$ **do**

3. **if** $\deg r = m + i$ **then** $q_i \leftarrow lc(r)u$, $r \leftarrow r - q_i x^i b$
else $q_i \leftarrow 0$

4. return $q = \sum_{i=0}^{n-m} q_i x^i$

(a) Show that this algorithm computes correctly the division with remainder $a = qb + r$ and $\deg r < m$.

(b) Show that the total number of arithmetic operations of this algorithm is bounded by $(2m + 1)(n - m + 1) = (2 \deg b + 1)(\deg q + 1)$.

8. Let R be a ring (commutative, with 1) and $a = \sum_{0 \le i \le n} a_i x^i \in R[x]$ of degree n, with all $a_i \in R$. The *weight* $w(a)$ of a is the number of nonzero coefficients of a besides its leading coefficient:

$$w(a) = \#\{0 \le i < n : a_i \ne 0\}.$$

Thus, $w(a) \le \deg a$, with equality if and only if all coefficients of a are nonzero. The *sparse* representation of a, which is particularly useful if a has small weight, is a list of pairs $(i, a_i)_{i \in I}$, with each $a_i \in R$ and $f = \sum_{i \in I} a_i x^i$. Then, we can choose $\#I = w(a) + 1$.

(a) Show that two polynomials a, $b \in R[x]$ of weight $n = w(a)$ and $m = w(b)$ can be multiplied in the sparse representation using at most $2nm + n + m + 1$ arithmetic operations in R.

(b) Draw an arithmetic circuit for division of a polynomial $a \in R[x]$ of degree less than 9 by $b = x^6 - 3x^4 + 2$ with remainder. Try to get its size as small as possible.

(c) Let $n \ge m$. Show that quotient and remainder on division of a polynomial $a \in R[x]$ of degree less than n by $b \in R[x]$ of degree m, with $\mathrm{lc}(b)$ being a unit, can be computed using $n - m$ divisions in R, and $w(b)(n - m)$ multiplications and subtractions in R each.

9. Let R be a ring and k, m, $n \in \mathbb{N}$. Show that the "classical" multiplication of two matrices $A \in R^{k \times m}$ and $B \in R^{m \times n}$ takes $(2m-1)kn$ arithmetic operations in R.

10. Given the real and imaginary parts a_0, a_1, b_0, $b_1 \in \mathbb{R}$ of two nonzero complex numbers $z_1 = a_0 + a_1 i$ and $z_2 = b_0 + b_1 i$, with $i^2 = -1$, show how to compute the real and imaginary parts of the quotient $\frac{z_1}{z_2}$ using at most 7 multiplications and divisions in $\mathbb{R}$. Draw an arithmetic circuit illustrating your algorithm. Can you achieve at most 6 real multiplications and divisions?

11. For $n \in \mathbb{N}$, let $\ell(n)$ denote the minimum number of multiplications sufficient to compute X^n starting from X.

(a) Design an algorithm that computes $\ell(n)$, for all $n \in \mathbb{N}$.

(b) Show that $\ell(mn) \leq \ell(m) + \ell(n)$ for all $m, n \in \mathbb{N}$.

(c) Show that $\ell(2n) \leq \ell(n) + 1$, and $\ell(2n+1) \leq \ell(2n) + 1$ for all $n \in \mathbb{N}$.

12. Let $\ell(n)$ as before, and denote with $w_2(n)$ the "Hamming weight" (i.e., the number of nonzero coefficients) of the binary expansion of n.

    (a) Show that
    $$\ell(n) \leq \lfloor \log_2 n \rfloor + w_2(n) - 1 \leq 2 \log_2 n.$$

    (b) Show that $\ell(n) \geq \lceil \log_2 n \rceil$.

13. The algorithm to compute X^n by using nested squaring according to the Hamming weight of n is called "the binary method". Show that the binary method is actually an application of Horner's rule.

14. Use the binary method to design an algorithm for multiplication of integers involving only the simple operations of doubling, halving, and adding.

15. Let $d(n)$ be the minimal number of operations necessary to compute X^n starting from X allowing multiplications AND divisions. Clearly we have $d(n) \leq \ell(n)$.

    (a) Show that $d(31) = 6 < 7 = \ell(31)$.

    (b) Show that 31 is the smallest n such that $d(n) < \ell(n)$.