Introduction to Cryptography

Homework 2

October 16, 2006

1. Assume that m and n are coprime integers; for solving the system of congruences

$$x \equiv a \bmod m,$$
$$x \equiv b \bmod n,$$

compute integers r, s with $mr + ns = 1$, and put $x = ans + bmr$.

(a) Show that this x solves the system.

(b) Estimate the complexity of this algorithm; here you may assume that $0 \le a < m$ and $0 \le b < n$.

We have $ans + bmr \equiv ans \equiv a \bmod m$ since $ns \equiv 1 \bmod m$; similarly we can prove that $ans + bmr \equiv b \bmod n$.

Now put $M = mn$; note that $\log M = \log m + \log n$. Since $|r| < n$ and $|s| < m$, the multiplications mr and ns require $O(\log m \log n)$ steps, which is bounded by $O((\log M)^2$. Multiplying ns by a requires about $\log ns \log a = (\log n + \log s) \log a < (\log n + \log m) \log m < (\log M)^2$ bit operations, hence is bounded by $O((\log M)^2)$. The same is true for finding the product bmr, and for adding the resulting products. Thus the complexity of applying the Chinese Remainder Theorem is $O((\log M)^2)$ with $M = mn$.

2. Let $f, g \in \mathbb{Z}[X]$ be polynomials. What is the complexity for computing $f + g$ and fg?

Let $f(x) = \sum_{i=0}^{d} a_i x^i$ and $g(x) = \sum_{j=0}^{e} b_j x^j$, and put $D = \max\{d, e\}$. Also, let A denote the maximum of the coefficients a_i, b_j. For adding the polynomials we have perform at most $D+1$ additions of numbers $\le A$, so the complexity is $O((D+1)\log A)$. If the degree of the polynomials involved is bounded, D is a constant, and then the complexity of addition is just $O(\log A)$.

For somputing the coefficient c_k of the product $fg = \sum_{k=0}^{d+e} c_k x^k$, we have to compute the products $a_i b_j$; there are $(d+1)(e+1)$ of them. Since each such product occurs in exactly one sum, there are at most $(d+1)(e+1)$

additions to perform. The multiplication require $O((d+1)(e+1)(\log A)^2)$ bit operations, and the additions of the resulting products, which are $\le A^2$, cost at most $O((d+1)(e+1)(\log A^2)) = O(2(d+1)(e+1)\log A)$ bit operations. Thus the overall complexity can be bounded by $O((d+1)(e+1)(\log A)^2)$.

3. Let f and g be polynomials in $(\mathbb{Z}/m\mathbb{Z})[X]$. What is the complexity for computing $f+g$ and fg? Here the coefficients are bounded by m, hence

   we find the complexities $O(D\log m)$ and $O((d+1)(e+1)(\log m)^2)$ for addition and multiplication. Afterwards, we have to reduce the $d+e+1$ coefficients mod m, which costs $O((d+e+1)(\log m)^2$ bit operations since, in $a = bq + r$, we have $a < m^2$ and $b = m$, so $q < m$ as well.

   Adding these numbers gives an upper bound $O((d+1)(e+1)(\log m)^2)$.

4. Prove the following rules for gcd's of natural numbers:

$$\gcd(a,b) = \begin{cases} 2\gcd(\frac{a}{2}, \frac{b}{2}) & \text{if } 2 \mid a, 2 \mid b; \\ \gcd(\frac{a}{2}, b) & \text{if } 2 \mid a, 2 \nmid b; \\ \gcd(\frac{a-b}{2}, b) & \text{if } 2 \nmid ab. \end{cases}$$

   If a and b are even and $d = \gcd(a,b)$, then we have to show that

$$2\gcd(\tfrac{a}{2}, \tfrac{b}{2}) \mid d \quad \text{and} \qquad d \mid 2\gcd(\tfrac{a}{2}, \tfrac{b}{2}) \mid d.$$

   This is easy, and so are the other claims.

5. Show how to compute $\gcd(91, 77)$ using these rules. This algorithm is due to Stein (1961).

$$\begin{aligned} \gcd(91,77) &= \gcd(\frac{91-77}{2}, 77) = \gcd(7,77) \\ &= \gcd(\frac{77-7}{2}, 7) = \gcd(35,7) \\ &= \gcd(\frac{35-7}{2}, 7) = \gcd(14,7) \\ &= \gcd(\frac{14}{2}, 7) = \gcd(7,7) \\ &= 7. \end{aligned}$$