Advanced Data Structures and Algorithms
Exercise Sheet 6

Stuart Golodetz

February 26, 2006

1. (a) We know that iff a and b are relatively prime then $\exists x, y \cdot ax + by = 1$. Using this fact, our proof proceeds as follows:

   If:

   If $gcd(n_1n_2, n_3n_4) = 1$ then $\exists x, y \cdot n_1n_2x + n_3n_4y = 1$. In particular, this means that $n_1(n_2x) + n_3(n_4y) = 1$, i.e. $gcd(n_1, n_3) = 1$ and similarly that $n_1(n_2x) = n_4(n_3y)$, i.e. $gcd(n_1, n_4) = 1$, and that $gcd(n_2, n_3) = gcd(n_2, n_4) = 1$. The missing ones follow from the other equation: if $gcd(n_1n_3, n_2n_4) = 1$, then $\exists x, y \cdot n_1n_3x + n_2n_4y = 1$. In particular, this means that $n_1(n_3x) = n_2(n_4y)$, i.e. $gcd(n_1, n_2) = 1$, etc. We conclude that all the n_is are pairwise relatively prime.

   Only If:

   Recalling Theorem 33.6 in CLRS, which says 'For any integers a, b and p, if $gcd(a, p) = 1$ and $gcd(b, p) = 1$ then $gcd(ab, p) = 1$,' we proceed as follows:

   Note that if $n_1, ..., n_4$ are pairwise relatively prime then $gcd(n_i, n_j) = 1$ for all $i \neq j$, $i, j \in \{1, ..., 4\}$. So in particular, $gcd(n_1, n_3) = gcd(n_2, n_3) = 1$, which by Theorem 33.6 implies that $gcd(n_1n_2, n_3) = 1$. For entirely analogous reasons, $gcd(n_1n_2, n_4) = 1$ as well. But since gcd is symmetric this means that $gcd(n_3, n_1n_2) = gcd(n_4, n_1n_2) = 1$, which implies that $gcd(n_1n_2, n_3n_4) = 1$ by Theorem 33.6.

   An entirely analogous argument shows that $gcd(n_1n_3, n_2n_4) = 1$ as well. □

   (b) TODO: I'm not really sure how to extend it to the general case.

2. We calculate as follows:

   $a_1 = 1, a_2 = 2, a_3 = 3$

   $n_1 = 9, n_2 = 8, n_3 = 7$

   $n = 9 \times 8 \times 7 = 504$

   $m_1 = 8 \times 7 = 56, m_2 = 9 \times 7 = 63, m_3 = 9 \times 8 = 72$

   $p_1 = m_1^{-1} \bmod n_1 = 56^{-1} \bmod 9$
   i.e. $56 \times p_1 \bmod 9 = 1 \leftarrow p_1 = 5$ works

   $p_2 = m_2^{-1} \bmod n_2 = 63^{-1} \bmod 8$
   i.e. $63 \times p_2 \bmod 8 = 1 \leftarrow p_2 = 7$ works

   $p_3 = m_3^{-1} \bmod n_3 = 72^{-1} \bmod 7$
   i.e. $72 \times p_3 \bmod 7 = 1 \leftarrow p_3 = 4$ works

   Now:

   $c_1 = m_1p_1 = 56 \times 5 = 280$
   $c_2 = m_2p_2 = 63 \times 7 = 441$
   $c_3 = m_3p_3 = 72 \times 4 = 288$

   So:

   $a \equiv (a_1c_1 + a_2c_2 + a_3c_3) \bmod n = (280 + 882 + 864) \bmod 504 = 10 \pmod{504}$

In other words:

$a = 10 + 504k$ (for any integer k)

So for instance:

10/9 has remainder 1
10/8 has remainder 2
10/7 has remainder 3

514/9 has remainder 1
514/8 has remainder 2
514/7 has remainder 3

etc.

3. (a) We want:

$$d = e^{-1} \bmod (p-1)(q-1) = 3^{-1} \bmod 280$$

To find this, we use Euclid's Extended Algorithm:

$$gcd(280, 3) = gcd(3, 1) = gcd(1, 0) = 1$$

$$1 = 1 - 0 = 1 - (3 - 3 \cdot 1) = 4 \cdot 1 - 3 = 4 \cdot (280 - 93 \cdot 3) - 3 = -373 \cdot 3 + 4 \cdot 280$$

The value we're after here is -373. If we get this into the right range (i.e. between 0 and 280) by adding multiples of 280 to it (or otherwise), we get $d = 187$ $(= -373 + 2 \cdot 280)$. Sure enough, $187 \cdot 3 \bmod 280 = 1$, as required.

The encryption of the message $M = 100$ is $100^3 \bmod 319 = 254$.

For what it's worth, decrypting this again gives us:

$$\begin{aligned}
& 254^{187} \bmod 319 \\
= \; & (254^7 \bmod 319 \times ((254^{10} \bmod 319)^9 \bmod 319)^2 \bmod 319) \bmod 319 \\
= \; & (144 \times (111^9 \bmod 319)^2 \bmod 319) \bmod 319 \\
= \; & (144 \times 199^2 \bmod 319) \bmod 319 \\
= \; & (144 \times 45) \bmod 319 \\
= \; & 100
\end{aligned}$$

(b) We know two values initially, $x = m^e \bmod n$ and $y = m^f \bmod n$. Given that $gcd(e, f) = 1$, we can use the Extended Euclidean Algorithm to compute a, b s.t. $ae + bf = 1$. Now, we note that:

$$m^{ae+bf} \bmod n = m^1 \bmod n = m \bmod n = m \text{ (since } 0 \leq m < n)$$

We further note that by the usual properties of modular arithmetic:

$$\begin{aligned}
m^{ae+bf} \bmod n &= [(m^e \bmod n)^a \bmod n \times (m^f \bmod n)^b \bmod n] \bmod n \\
&= [x^a \bmod n \times y^b \bmod n] \bmod n
\end{aligned}$$

So we can quite easily compute m using the values we have.

For an example of this, consider the following: use $p = 5$, $q = 7$, $n = 35$, $e = 11$, $f = 13$, $m = 12$, then $m^e \bmod n = 3$ and $m^f \bmod n = 12$. We can decrypt this as follows: use the Extended Euclidean Algorithm (or random guesswork, which is what I used here) to derive that $6 \cdot 11 - 5 \cdot 13 = 1$, whence $a = 6$ and $b = -5$. Then compute:

$$\begin{aligned} & [x^a \bmod n \times y^b \bmod n] \bmod n \\ = \; & [3^6 \bmod 35 \times 12^{-5} \bmod 35] \bmod 35 \\ = \; & [29 \times (12^{-1} \bmod 35)^5 \bmod 35] \bmod 35 \\ = \; & [29 \times 3^5 \bmod 35] \bmod 35 \\ = \; & 12 \end{aligned}$$

Note that we used the multiplicative inverse of 12 modulo 35 in this calculation: in practice this would be computed by using the Extended Euclidean Algorithm, but here it was obvious ($12 \cdot 3 \bmod 35 = 36 \bmod 35 = 1$).

4. (a) No. If we try lots of different values of a and $a^{n-1} = 1 \bmod n$ for all of them, it becomes more likely that n is prime, but it's still not necessarily the case. The point is that this is a probabilistic method of checking whether a number's a prime. We might found a counterexample (thus proving the number's composite), but even if we don't we can't be certain that the number actually is prime.

   (b) Yes, but sometimes it requires a lot of work. If we can find an integer a s.t. $0 < a < n$ and $a^{n-1} \neq 1 \bmod n$ then n is assuredly composite by FLT. The trick is finding such an a without going through all $n-1$ possibilities. Often if a number's composite and we pick a randomly a few times, we'll find one satisfying the above, thus proving that the number's not a prime, but in theory it might not be that easy. In particular, certain numbers called *Carmichael numbers* satisfy $a^{n-1} = 1 \bmod n$ for every a relatively prime to them, despite being composite, meaning that we can only prove they're not prime by running the test with a being one of their factors. In less pathological cases, though, if we run through a reasonable number of different values for a, we will tend to find one which gives us what we want.

   (c) A Monte Carlo algorithm is a probabilistic algorithm which may return the wrong answer but whose error probability is bounded. By contrast, we expect a deterministic algorithm to always return the right answer.

   (d) TODO: I'm having trouble doing this question because the notes aren't up on the web and it's not in the book. Sorry!

   (e) TODO: Similarly.