CS 70, Summer 2014 — Homework 3

Harsimran (Sammy) Sidhu, SID 23796591

July 14, 2014

Collaboraters: Chonyi Lama, Jenny Pushkarskaya
Sources:

Problem 1

(a) 4^{273} mod 11 using repeated squaring
$4 \times 4^{272} \equiv 4 \times 16^{136} \equiv 4 \times 5^{136} \equiv 4 \times 25^{68} \equiv 4 \times 3^{68} \equiv 4 \times 9^{34} \equiv 4 \times 81^{17} \equiv 4 \times 4^{17} \pmod{11}$
$4 \times 4^{17} \equiv 4^{18} \equiv 16^{9} \equiv 5^{9} \equiv 5 \times 5^{8} \equiv 5 \times 25^{4} \equiv 5 \times 3^{4} \equiv 5 \times 9^{2} \equiv 5 \times 81 \equiv 5 \times 4 \equiv 20 \pmod{11}$
9 mod 11

(b) 4^{273} mod 11, using Fermat's Little Theorem.
$4^{270+3} \equiv 4^{3} \times 4^{(10\times 27)} \equiv 4^{3} \times (4^{10})^{27} \pmod{11}$
$4^{3} \times (1)^{27} \equiv 4^{3} \equiv 64 \equiv 9 \pmod{11}$
9 mod 11

(c) $4^{5^{273}}$ mod 11
since $4^{5^{273}}$ is in mod 11, that would mean that 5^{273} is in mod 10 due to the fact that 5^{273} will only have 10 congruence classes.

$5^{273} \equiv 5^{270+3} \equiv 5^{9\times 30} \times 5^{3} \equiv (5^{9})^{30} \times 5^{3} \equiv (1)^{30} \times 5^{3} \pmod{10}$
$5^{3} \equiv 125 \equiv 5 \pmod{10}$

So $4^{5^{273}} \equiv 4^{5} \equiv 4 \times 4^{4} \equiv 4 \times 16^{2} \equiv 4 \times 5^{2} \equiv 4 \times 25 \equiv 4 \times 3 \equiv 1 \pmod{11}$
1 (mod 11)

(d) 5^{82} mod 21
Since 21 is not a prime we have to use the size of the set of coprime integers between 1 and 20 to 21 which in this case is 12. So $5^{12} \equiv 1$ mod 21
$5^{(6\times 12+10)} \equiv (5^{12})^{6} \times 5^{10} \equiv (1)^{6} \times 25^{5} \equiv 4^{5} \equiv 4 \times 4^{4} \equiv 4 \times 256 \pmod{21}$
$4 \times 256 \equiv 4 \times 4 \pmod{21}$
16 (mod 21)

Problem 2

$p = 11,\ q = 23,\ e = 7,\ N = 253,\ (p-1)(q-1) = (11-1)(23-1) = 220$

(a) $1 \equiv gcd((p-1)(q-1), e) \equiv a \times (p-1)(q-1) + d \times e \qquad \text{mod } (p-1)(q-1)$
$1 \equiv gcd(220, 7) \equiv a \times 220 + d \times 7 \qquad \text{mod } 220$

$gcd(220, 7) \rightarrow 220 = 31 \times 7 + 3$
$gcd(7, 3) \rightarrow 7 = 2 \times 3 + 1$
$gcd(2, 1) \rightarrow 2 = 2 \times 1 + 0$
$gcd(1, 0) \rightarrow 1 = 0 \times 0 + 1$
$gcd(220, 7) = 1$

EGCD
$1 = 1 \times (1) + 0 \times (0)$
$1 = 0 \times (2) + 1 \times (1)$
$1 = 1 \times (7) - 2 \times (3)$
$1 = -2 \times (220) + 63 \times (7)$

$d = 63$

(b) $y = x^e$ mod N
$y = 44^7$ mod 253
$y = 44 \times 44^6 \equiv 44 \times 44^6 \pmod{253}$
$44 \times 1936^3 \pmod{253}$
$44 \times 165^3 \pmod{253}$
$165 \times 44 \times 165^2 \pmod{253}$
$7260 \times 27225 \pmod{253}$
$176 \times 154 \pmod{253}$
$y = 33 \pmod{253}$

(c) $x = y^d$ mod N
$x = 103^{63}$ mod 253
$103 \times 103^{62} \equiv 103 \times 10609^{31} \equiv 103 \times 236^{31} \pmod{253}$
$24308 \times 236^{30} \equiv 20 \times 236^{30} \equiv 20 \times 55696^{15} \equiv 20 \times 36^{15} (\text{mod } 253)$
$36 \times 20 \times 36^{14} \equiv 720 \times 1296^7 \equiv 214 \times 31^7 \equiv 6634 \times 31^6 (\text{mod } 253)$
$56 \times 31^6 \equiv 56 \times 961^3 \equiv 56 \times 202^3 \equiv 11312 \times 202^2 \equiv 180 \times 40804 \equiv 180 \times 71 \pmod{253}$
$12780 \equiv 130 \pmod{253}$
$x = 130$ mod 253

Problem 3

$N = p$, and e is coprime to $p - 1$

(a) If e is coprime to $p - 1$ that would mean that we could find the decryption key, d just knowing the public key, p and running the extended gcd algorithm as $gcd(p-1, e) = a \times (p-1) + d \times e$

Once d is found, we can now compute $(x^e \text{ mod p })^d$ mod p. This can be computed by performing repeated squaring. This will result in x mod p.

(b) In normal RSA encryption we use $N = pq$ where p, q are two primes. So the amount of congruence classes is $(p-1)(q-1)$, and to find d in RSA, we find the inverse of e in mod $(p-1)(q-1)$. However in this example we are using $N = p$ which has $(p - 1)$ congruence classes. So to find d in this example we just find the inverse of e in mod $(p - 1)$. This is dangerous due to the fact that N aka p is public key so anyone can solve for d.

Armed with this knowledge we can find (x mod p).
Using the extended gcd algorithm we can find the inverse of e. The function would be $d = egcd((p - 1), e)$. Once d is found we can use the intercepted message
which is (x^e mod p). Now we just raise the intercepted message to the d power mod p.
So we have $(x^e)^d \equiv (x^{ed}) = (x)$ mod p. We solved for d as the inverse of e in mod $p - 1$ so we know that the exponent is just 1. Therefore the final result of the intercepted message is
(x) mod p.

Lemma for proof above!
d is the inverse of e in mod $(p - 1)$. so $ed = 1 + k(p - 1)$
$(x^{ed}) = x^{1+k(p-1)} = x \times x^{k(p-1)} \pmod{p}$
$x \times (1)^k \equiv x \bmod p$ Using Fermat's Little Theorem

(c) It was proven in lecture and Note 5 that the running time for the repeated squaring algorithm is $O(n)$ operations, where n is the number of bits that p is made of since it is the largest number of p, e. This translates to $O(\log p)$ arithmetic operations. The extended gcd algorithm also is in $O(n)$ arithmetic operations, where n is the amount of bits in x, y according to Note 5. This translates roughly to $O(\log p)$ as well. Together we have $O(\log p + \log p) = O(\log p)$ arithmetic operations. If we assume each arithmetic operation is $O((\log N)^2)$ as James did in lecture then the total runtime is to be $O((\log p)^3)$.

Problem 4

(a) The maximum degree of the product fg would be the sum of the leading degrees of two polynominals. Since they are at most d, the sum would be $d+d$ or $2d$.

(b) If we have two polynominals f, g, we can find the product by multiplying each term of f with every term of g and then summing the similar terms.
$f(x) = a_d x^d + ... + a_1 x^1 + a_0$
$g(x) = b_d x^d + ... + b_1 x^1 + b_0$
$fg(x) = (a_d x^d + ... + a_1 x^1 + a_0) \times (b_d x^d + ... + b_1 x^1 + b_0)$
$fg(x) = (a_d x^d \times (b_d x^d + ... + b_1 x^1 + b_0) + ... + a_1 x^1 \times (b_d x^d + ... + b_1 x^1 + b_0) + a_0 \times (b_d x^d + ... + b_1 x^1 + b_0))$
For each term we have to do $(d+1)$ multiplications which we then do on $d+1$ terms. So we have have a total of $(d+1)^2$ multiplications.
After multiplying, we can analyze the degrees of each set of terms.
The first set of will be $(2d, 2d-1, 2d-2, \ ... \ , d)$, The second would be $(2d-1, 2d-2, 2d-3, ... \ , d-1)$ and so on and so forth until $(d, d-1, d-2, \ ... \ , 0)$. If we look carefully we can see that d appears d times and $d-1$ appears $d-1$ times. This pattern goes both forwards and backwards on d until $2d$ or 0 is hit respectively. so the amount of additions we have to perform is the sum of 1 to d and then back to 1.

$(1 + 2 + ... + (d-1) + (d) + (d-1) + (d-2) + ... + 2 + 1)$

$$\text{num of additions} = \sum_{i=1}^{d} i + \sum_{i=1}^{d-1} i = \frac{(d)(d+1)}{2} + \frac{(d-1)(d)}{2} = \frac{(d)(d+1+d-1)}{2} = \frac{(d)(2d)}{2} = d^2$$

The total amount of operations is $(d+1)^2$ multiplications and d^2 additions which is $O(d^2)$

(c) Since both f, g are at most degree d we know the product is at most degree $2d$. That would mean we would need $2d+1$ points or $t \geq 2d+1$ to perform Lagrange interpolation.
By the definition of multiplying a function we know that $fg(x) = f(x) \times g(x)$.
So to compute the point value representation of fg. The points would simply be
$(x_1, \ f(x_1) \times g(x_1)), \ (x_2, \ f(x_2) \times g(x_2)), \ ... \ , (x_t, \ f(x_t) \times g(x_t)) \qquad t \geq 2d+1$

(d) Case 1: $g(x_i) = 0$ for one the points
This would mean that x_i is a root of g. If we went through and performed polynomial division, this root would still exist. Infact there would be at most d roots in the denominator. This implies that there are at most d holes in the function acting as vertical asymptotes. Therefore it is impossible for f/g to have a real value at one of these roots.

Case 2: $g(x_i) \neq 0$ for all the points
In this case we can represent f/g in point-value form. But the condition is that we have $t = d+1$ points where $g(x_i) \neq 0$. It would simply be

$(x_1, \ f(x_1)/g(x_1)), \ (x_2, \ f(x_2)/g(x_2)), \ ... \ , (x_t, \ f(x_t)/g(x_t)) \qquad g(x_i) \neq 0, \ t = d+1$

Problem 5

(a) let d be the degree of some polynomial.
Case 1: $d < p - 1$
Since the degree of the polynomial is less than $p-1$, the equivalent polynomial that is at most degree $p - 1$ is itself.

Case 2: $d \geq p - 1$
In this case where we have a polynomial with $d \geq p - 1$ in GF(p), we can use Fermat's Little Theorem to reduce the degree of the polynomial to at most $p - 1$.
Looking at FLT we see that $x^{p-1} \equiv 1 \pmod{p}$ $\forall x \in \{1, 2,\ ...,\ p-1\}$

But what about the case where $x = 0$? If $x = 0$ then the polynomial terms will also become 0 giving a polynomial of degree 0 which is just a constant and also the y intercept.
Knowing that $(d \geq p - 1)$, we can rewrite d as $(k(p-1) + r)$ this would give

$x^d \equiv x^{k(p-1)+r} \equiv (x^{p-1})^k \times x^r \equiv (1)^k \times x^r \equiv x^r \pmod{p}$
Since r is the remainder of $(d/(p-1))$ it has to be $(0 \leq r < p - 1)$. We have shown that any polynomial with a degree larger than $p - 1$ can we reduced to at most degree $(p - 1)$ which completes the proof.

(b) In GF(p), we have a finite set of congruence classes which classifies inputs as well as outputs. The set for both is the same which is $\{0,\ 1,\ 2,\ ...,\ p-1\}$. We also know that a function must have a single output for every input. This means that any x can only make an appearance once in the point-value representation. So if we have a set that is size p that means there is at most p points. We know that Lagrange interpolation requires $d + 1$ points where d is the degree of the resultant polynomial. From this we can report that the polynomial in GF(p) will be at most degree $(p - 1)$.

Problem 6

Let's call the secret we want to share S, we want to divide it up so that we need two groups out of 3 in which for each you need 4 out of 7 people in a group. Let's divide the groups into group $A,\ B,\ C$. each group has 7 people who we call $(A1,\ A2,\ ...,\ A7)$, $(B1,\ B2,\ ...,\ B7)$, $(C1,\ C2,\ ...,\ C7)$.
Since we only need 2 groups to unlock the secret we can use a degree 1 polynomial which is a line to share S to all 3 groups.
The function comes out to be $y = mx + S$ for some m.
We now get 3 points from this function. $(a,\ S + am)$, $(b,\ S + bm)$, $(c,\ S + cm)$. $(a \neq b \neq c)$. We now share each x coordinate with each group respectively and make each y a secret to be shared among each 7 per group. Let's call each y coordinate $(S_A,\ S_B,\ S_C)$ respectively. We need 4 people in a group to solve one of these sub-secrets. That would mean we need to use 3 unique polynomials each of degree 3.

$y_A = a_3x^3 + a_2x^2 + a_1x + S_A$ For some $(a_3,\ a_2,\ a_1)$
$y_B = b_3x^3 + b_2x^2 + b_1x + S_B$ For some $(b_3,\ b_2,\ b_1)$
$y_C = c_3x^3 + c_2x^2 + c_1x + S_C$ For some $(c_3,\ c_2,\ c_1)$

Now we give out a unique point to each member of each group using their group function.
If the members wanted to figure out the secret they would have to form 2 groups of 4 and each 4 would interpolate their points to solve for their group function. Once they did that, they would use the y intercept as well as the x that was published to their group to interpolate with another group who also has a point to solve for S.

Problem 7

We have a polynomial that is at most a degree of 2.
We also know that $P(0) = 4,\ P(3) = 1,\ P(4) = 2.$
$(0, 4),\ (3, 1),\ (4, 2)$

$$\Delta_1 = \frac{(x-3)(x-4)}{(0-3)(0-4)} = \frac{(x-3)(x-4)}{12} = 12^{-1}(x-3)(x-4) \pmod 5$$
$$\Delta_2 = \frac{(x-0)(x-4)}{(3-0)(3-4)} = \frac{(x)(x-4)}{-3} = (-3)^{-1}(x)(x-4) \pmod 5$$
$$\Delta_3 = \frac{(x-0)(x-3)}{(4-0)(4-3)} = \frac{(x)(x-3)}{4} = 4^{-1}(x)(x-3) \pmod 5$$

Finding an alternate inverse of 12. If we multiply 12 by 3... $(12 \times 3 = 36) \pmod 5$ we get 1. So 3 is also an inverse of 12 mod 5.

If we multiply -3 by -2... $(-3 \times -2 = 6) \pmod 5$ we get 1. So -2 is also an inverse of -3 mod 5.

If we multiply 4 by 4... $(4 \times 4 = 16) \pmod 5$ we get 1. So 4 is also an inverse of 4 mod 5.

Now we can replace the inverses with the newly calculated ones.

$\Delta_1 = 3(x-3)(x-4) \pmod 5$

$\Delta_2 = -2(x)(x-4) \pmod 5$

$\Delta_3 = 4(x)(x-3) \pmod 5$

$y = y_1\Delta_1 + y_2\Delta_2 + y_2\Delta_2 = 4 \times 3(x-3)(x-4) + 1 \times -2(x)(x-4) + 2 \times 4(x)(x-3) \mod 5$
$y = 12(x-3)(x-4) - 2(x)(x-4) + 8(x)(x-3) \mod 5$
$y = 12(x^2 - 7x + 12) - 2(x^2 - 4x) + 8(x^2 - 3x) \mod 5$
$y = (12x^2 - 84x + 144) + (-2x^2 + 8x) + (8x^2 - 24x) \mod 5$
$y = (18x^2 - 100x + 144) \mod 5$
$y = 3x^2 + 4 \mod 5$
$a_0 = P(0) = 3(0)^2 + 4 = 4 \mod 5$
$a_1 = P(1) = 3(1)^2 + 4 = 7 = 2 \mod 5$
$a_2 = P(2) = 3(2)^2 + 4 = 6 = 1 \mod 5$

$(a_0,\ a_1,\ a_2) = (4,\ 2,\ 1)$