Solutions to Homework 2

Debasish Das
EECS Department, Northwestern University
ddas@northwestern.edu

1 Problem 1.14

Using the results from 0.4, Fibonacci numbers in terms of matrix can be represented as follows

$$\begin{pmatrix} F_n \\ F_{n+1} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}^n \cdot \begin{pmatrix} F_0 \\ F_1 \end{pmatrix} \tag{1}$$

Since F_n(mod p) can be obtained by taking the first term of the matrix $\begin{pmatrix} F_n \\ F_{n+1} \end{pmatrix}$ (mod p). As the matrix $\begin{pmatrix} F_0 \\ F_1 \end{pmatrix}$ is a constant matrix, computing $\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}^n$ (mod p) is sufficient to compute F_n(mod p). We call the matrix $\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$ as A. An extension of modular exponentiation algorithm can be employed to solve this problem once we prove the following theorem.

Theorem 1 *Given any general matrix A, (A mod p)×(A mod p) = A^2(mod p)*

Proof: Assume A as any matrix $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$. (A mod p) can be written as $\begin{pmatrix} a + k^1 p & b + k^2 p \\ c + k^3 p & d + k^4 p \end{pmatrix}$ where $k^1, .., k^4$ are arbitrary integers. After multiplying (A mod p) with (A mod p), the first term in the final matrix can be written as $b^2 + ac + p(C)$ where C is any constant which is equivalent to the first term of the matrix A^2(mod p). Similarly it holds for other terms as well. $\square$

Using the above theorem, we can establish in general that (A^n (mod p) $\times$ (A mod p) = A^{n+1} (mod p). The algorithm is given as

```
function Modified-modexp(A, p, n)
Input:  2x2 Array A where each element is of n-bits,
       p (n bit) and integer exponent n
Output:  A^n mod p
if y = 0:  return I_2
z = Modified-modexp(A,p,⌊n/2⌋)
temp = (z mod p)×(z mod p)
if y is even:
```

```
  return temp
else:
  return (A mod p)×temp
```

Complexity Analysis : Number of recursive calls are analogous to Modular Exponentiation presented at page 19. Matrix multiplication is O(n^2) (Chapter 3 presents a better bound). Hence the complexity of the algorithm is O(n^3).

2 Problem 1.15

Statement : For any a, b, if ax ≡ bx mod c, then a ≡ b mod c.
Necessary and Sufficient Condition Derivation

$$ax \equiv bx(modc) \Rightarrow c|(a-b)x$$
$$a \equiv b(modc) \Rightarrow c|(a-b)$$

Now since c must divide (a-b)x and c must divide (a-b), we should choose x such that GCD(c,x) = 1 which will ensure that if (a-b)x is divisible by c, then (a-b) must be divisible by c as GCD(c,x) is 1.

3 Problem 1.17

x and y are each n-bit long. We are performing complexity analysis of 2 algorithms for x^y computation.

```
function Iterative-Exponentiation(x,y)
Input:  x and y each n-bit long
Output:  x^y
prod = x
for i = 1 to y-1
  prod = x * prod
return prod
```

Complexity Analysis : After each multiplication, the size of the product becomes i.n where i is the current iteration. Total time is given by $\sum_{i=0}^{y-1} O(in \cdot n)$ = $\frac{(y-1)y}{2}$O(n^2). The complexity is O(y^2n^2) where y is O(2^n).

```
function Recursive-Exponentiation(x,y)
Input:  x and y each n-bit long
Output:  x^y
if y = 0:  return 1
z = Recursive-Exponentiation(x,⌊y/2⌋)
if y is even:
  return z*z
else:
  return x*z*z
```

Complexity Analysis : Since y is n-bit long, the number of iterations is bounded by O(n). Size of z on each return from recursive call increases by a factor of 4. Total running time is given by
$O(n^2) + O(4n^2) + ... + O((2^{n-2}n)^2) = O(n^2 2^{2n})$

4 Problem 1.24

From the set given 0,1,2,...,p^n-1 we have to exclude all numbers which are multiple of p since gcd(kp,p^n) where $0 \leq$ k $< p^{n-1}$ is surely not equal to 1 as p is a common divisor for both the numbers. Now consider numbers of the form kp+i where 0<i<p-1. k and i are integers. Now gcd(kp+i,p^n) = 1 as p is surely not a divisor of kp+i and p is the only prime divisor of p^n. Therefore total numbers in the set which are multiple of p are p^{n-1}. Numbers which have an inverse are $p^n - p^{n-1}$.

5 Problem 1.26

We have to compute $17^{17^{17}}$ (mod 10) to get the least significant digit.

$$\begin{aligned}
10 &= 2 \cdot 5 \text{ where 2 and 5 are primes} \\
p &= 2 \;\; q = 5 \text{ and } a = 17 \\
a^{(p-1)(q-1)} &= 1(mod10) \Rightarrow 17^4 = 1(mod10) \\
17^{17} = (4^2+1)^{17} &= 4 \cdot C + 1 \text{where C is some constant} \\
17^{17^{17}}(mod10) &= 17^{4 \cdot C}(mod10) \cdot 17(mod10) = 7
\end{aligned}$$

6 Problem 1.44

Alice and her three friends are communicating using RSA cryptosystem. Respective public keys are (N_i,e_i) where i $\in$ 1,2,3. Alice sent the same message MSG to all three of her friends. Since $e_i = 3$, we get following cyphertext M_i

$$M_1 = MSG^3 mod(N_1)$$
$$M_2 = MSG^3 mod(N_2)$$
$$M_3 = MSG^3 mod(N_3)$$

Rearranging terms the above equations can be written as

$$MSG^3 = M_1 mod(N_1)$$
$$MSG^3 = M_2 mod(N_2)$$
$$MSG^3 = M_3 mod(N_3)$$

Someone who intercepts all the 3 encrypted messages M_1, M_2 and M_3 along with the public keys N_1, N_2, N_3 and e can compute MSG^3 using Chinese Remainder Theorem.

Theorem 2 *Suppose m_1, m_2,...,m_s are s integers, no two of which have a common divisor other than 1. Let $M = m_1 m_2 ... m_s$ and suppose a_1,a_2,...,a_s are integers such that gcd(a_i,m_i) = 1 for each i. Then the s congruences*

$$
\begin{aligned}
a_1 x &\equiv b_1 (mod m_1) \\
a_2 x &\equiv b_2 (mod m_2) \\
&..., \\
a_s x &\equiv b_s (mod m_s)
\end{aligned}
$$

have a simultaneous solution that is unique modulo M.

Proof: From each particular congruence we construct one common to the entire set. We choose integers c_1,c_2,...,c_s such that

$$a_i c_i \equiv b_i (mod m_i) \tag{2}$$

Note that one possibility of choose c_i is to take them equal to b_i. Now let $n_i = \frac{M}{m_i}$. No two m_i have a common factor, therefore gcd(n_i,m_i)=1. Therefore an inverse $\hat{n_i}$ exist such that $n_i \hat{n_i} \equiv 1(mod m_i)$. Thus the x_0 defined by

$$x_0 = c_1 n_1 \hat{n_1} + c_2 n_2 \hat{n_2} + ... + c_s n_s \hat{n_s} \tag{3}$$

is a solution to the original system of s congruences. Note that by definition m_i divides each n_j except n_i. Thus

$$
\begin{aligned}
a_i x_0 &= a_i c_1 n_1 \hat{n_1} + a_i c_2 n_2 \hat{n_2} + ... + a_i c_s n_s \hat{n_s} \\
&\equiv a_i c_i n_i \hat{n_i} (mod m_i) \\
&\equiv a_i c_i (mod m_i) \\
&\equiv b_i (mod m_i)
\end{aligned}
$$

Hence x_0 is a solution of each congruence. ∎

Using the theorem getting M is straightforward. For the given problem a_1, a_2 and a_3 are 1 respectively. $c_1 = M_1$, $c_2 = M_2$ and $c_3 = M_3$. $M = N_1 N_2 N_3$. $n_1 = N_2 N_3$, $n_2 = N_1 N_3$ and $n_3 = N_1 N_2$. Compute $\hat{n_1}$ by the equation $n_1 \hat{n_1} = 1(mod N_1)$ using Extended Euclid Algorithm. Similarly compute $\hat{n_2}$ and $\hat{n_3}$. Following that generate x_0 as the solution for the congruences. Now $x_0(mod N_1 N_2 N_3)$ is the required solution. Thus $MSG^3 = x_0$ and $MSG = x_0^{\frac{1}{3}}$

7 Errata for Homework 1

Problem 2 (0.1) (l) $n^{\frac{1}{2}}$ = O($5^{\log n}$)