Zachary Şells SID 861013217

CS141 ASSIGNMENT 2

due Thursday, October 17

Individual assignment: Problems 1 and 2.
Group assignment: Problems 1,2 and 3.

Problem 1: Each row of the table below contains parameters of the RSA crypto-system: p, q, n, e, and d, with some of them missing. The next-to-last column contains a message x and the last column the corresponding ciphertext y. If the present parameters are correct, fill in the remaining entries in the row. Otherwise, explain why the parameters are not correct. Show your work.

p	q	n	e	d	x	y
5	13	65	7	7	2	63
Error 1	11	121	*Error 1*	5	*Error 1*	2
11	13	143	17	113	24	7
7	11	77	5	*Error 2*	3	*Error 2*
19	7	133	5	65	2	32

Solution 1:

Row 1 -
To find q, we need to compute $q = n/p$ which gives us 13

To find d, we need to compute $d = e^{-1} mod (p-1)(q-1)$

$$d = 7^{-1} mod 48$$

By listing the multiples of 7 and of 48 we can see which multiple of 7 is one less than a multiple of 48
7 multiples - $7, 14, 21, 28, 35, 42, 49, ...$
48 multiples - $48, 96, ...$
We can see that $7 * 7 = (49 * 1) + 1$ which gives us our answer, $d = 7$
To encrypt x, we need to use the formula, $y = x^d mod n$, or $y = 2^7 mod(65)$
Simplifying, we get $128 mod(65)$.
$128/65 = 1$ with remainder 63. So $128 mod(65) = 63 = y$

Error 1 - This configuration of RSA will not work because p and q cannot be equal
$p = n/q = 121/11 = 11 = q$

Error 2 - This configuration of RSA will not work because e must be relatively prime both (q-1) and (p-1)
$e = 5$ and $(q-1) = 10$. 5 and 10 share a common factor of 2. Therefor e is not relatively prime to (q-1)

Problem 2: For an n that is a power of 2, the $n \times n$ Weirdo matrix W_n is defined as follows. For $n = 1$, $W_1 = [1]$. For $n > 1$, W_n is defined inductively by

$$W_n = \begin{bmatrix} W_{n/2} & -W_{n/2} \\ I_{n/2} & W_{n/2} \end{bmatrix},$$

where I_k denotes the $k \times k$ identity matrix (whose diagonal entries are 1 and all other entries 0). For example,

$$W_2 = \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \quad W_4 = \begin{bmatrix} 1 & -1 & -1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & 0 & 1 & -1 \\ 0 & 1 & 1 & 1 \end{bmatrix} \quad W_8 = \begin{bmatrix} 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & 0 & 1 & -1 & -1 & 0 & -1 & 1 \\ 0 & 1 & 1 & 1 & 0 & -1 & -1 & -1 \\ 1 & 0 & 0 & 0 & 1 & -1 & -1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & -1 & -1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$

Give an efficient algorithm that for a vector $\bar{x}$ of length n (where n is a power of 2) computes the product $W_n \cdot \bar{x}$. Your algorithm must run in time $O(n \log n)$. (Hint: use divide-and-conquer, taking advantage of the recursive definition of W_n.)

Solution 2:

If we write out explicitly, the solutions for $W_2X_2, W_4X_4, and\ W_8, X_8$ it becomes apparent that each quadrant of the matrix can be expressed by the previous term in the series of matrices.
Let's label our matrix as follows:

$$W_N = \begin{bmatrix} A & B \\ C & D \end{bmatrix}$$

Now we can construct the following algorithm to compute $W_n * X_n$

def mult(X_n, W_n):
$if(n = 1)$ return X_n
$quadrant_A = mult(X_{n/2}, W_{n/2}))$
$quadrant_B = -1 * mult(X_{n/2-n}, W_{n/2})$
$quadrant_C = [X_n]$
$quadrant_D = quadrant_A$

$$return \begin{bmatrix} A + B \\ C + D \end{bmatrix}$$

From this, we can get the reccurance equation, which is:

$T(n) = 2T(n/2) + O(n)$

Using master's theorem, with condition, $k = 0$, we can obtain the asymptotic value of this recurrance function:

$\theta(nlog(n))$

Problem 3: Prof. Goofy has been trying to speed-up the divide-and-conquer integer multiplication algorithm as follows: Given two numbers x, y with n bits each (you can assume that n is a power of 4), he wants to:

(i) divide each into four equal-length pieces (instead of two pieces as before), and

(ii) express the product $x \cdot y$ using some number p of multiplications of these $n/4$-bit pieces, and some additions, subtractions or shifts.

How small does p need to be for Prof. Goofy's idea to give a faster algorithm than the $O(n^{\log_2 3})$-time algorithm covered in class? Justify your answer.