CMSC 389T MIDTERM SOLUTION

Phong Dinh and William Gasarch

Jan 12, 2017

Problem 1

Plutonians use alphabet with $512 = 2^9$ symbols.

Part a: How many affine ciphers are there?

Part b: Alice and Bob want to use a 2×2 Matrix Cipher. Give an example of a matrix that WORKS.

SOLUTIONTO PROBLEM 1

a) We need to know how many numbers in $\{1, \ldots, 512\}$ are rel prime to 512. Since $512 = 2^9$ all of the odd numbers are rel prime to 512 but none of the even numbers are. Alternatively, from homework 1, we know that $\phi(n)$ is the the number in $\{1, \ldots, n-1\}$ that are relatively prime to n, so

$$\phi(512) = \phi(2^9) = 2^8 = 256$$

(NOTE- I might ask on the final a question where you really NEED to look at $\phi(n)$ to get the right answer.)

Therefore, there are $512 - 256 = 256$ numbers from $\{1, \ldots, 512\}$ that are not relatively prime with 512. Since we are using affine cipher, where x maps to $ax + b \pmod{512}$, then there are 256 possible values for a. In addition, b can be arbitrary, so there are 512 possible values for b. B Therefore, there are

$$256 * 512 = 2^8 \times 2^9 = 2^{17}$$

possible affine ciphers.

b) Any matrix whose determinant is relatively prime with 512 will work. For example,

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

works because determinant of this matrix is 1, which is relatively prime with 512.

Problem 2

Alice and Bob want to use a variant of Vigenere cipher where they code a sequence of 2×2 matrices rather than a sequence of shift cipher. Alice and Bob first try to do this by having the key word be FOUR keywords of the same length(like JUSTIN GRADES ALICES FINALS) and use the first words for the a's (left upper corner of the matrix), second word for the b's (right upper corner of matrix), third word for the c's (left lower corner of matrix), fourth word for the d's (right lower corner of matrix).

Part a: Show that there are four-tuples of words for which this is a bad idea. Give such a pair and say WHY its a bad idea. For this problem a word can be a sequence of numbers between 0 and 25 and need not correspond to real English words.
NOTE that we are using $\pmod{26}$, and A is 0, B is 1, $\ldots$, Z is 25.

Part b: Assume that Alice and Bob DO get this to all work. Give a short description (no pseudocode needed) of how Eve can crack the code.
We can now assume that each matrices for encoding has the determinant that is relatively prime with 26.

SOLUTION TO 2a

In order to this version of Vigenere cipher works, each 2×2 matrices has to have its determinant not relatively prime with 26. However, if you pick four words of the same length in English then its quite likely that one of the resulting matrices will have determinant NOT rel prime to 26. An example of a four-tuple keyword that does not work is:

$$\text{KEY} = \text{DAD HAS FAT CAT}$$

This does not work because for the second matrix, it is

$$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

and its determinant is 0, which is not relatively prime with 26. We conclude that this is a BAD idea.

SOLUTION TO 2b

Lets look carefully and what Alice and Bob did. They have a sequence of matrices $M_1, M_2, \ldots, M_L$.
They code the first *pair* of letters with M_1

the second *pair* of letters with M_2

$\ldots$

the Lth *pair* of letters with M_L

the $(L+1)$th *pair* of letters with M_1

the $(L+2)$th *pair* of letters with M_2.

More generally, for all $a \in N$ and all $i \in \{1, \ldots, L\}$ they code the $(aL+i)$th pair with M_i.

STEP ONE: FIND CANDIDATES FOR THE KEYLENGTH L

This is similar (but not identical) to how you find key lengths for ordinary Vig. Lets say that you find a 4-letter sequence of words 3 times with differences 10, 30, 100. Say its *abcd*.

10	11	13	14	$\cdots$	30	31	32	33	$\cdots$	100	101	102	103
a	b	c	d	$\cdots$	a	b	c	d	$\cdots$	a	b	c	d

So we think the same matrix mapped the (10,11) positions, the (30,31) position, and the (100,101) position. Lets cal this matrix M_i. Note that

M_i codes (10,11)

$M_{i+1 \pmod L}$ codes (12,13)

$M_{i+2 \pmod L}$ codes (14,15)

$M_{i+3 \pmod L}$ codes (16,17)

$\ldots$

$M_{i+j \pmod L}$ codes (2j+10,2j+11)

In particular taking $2j + 10 = 30$, so $j = 10$ we have that

$M_{i+10 \pmod L}$ codes (30,31)

HERE IS THE KEY DIFFERENCE: For Ordinary Vig we took the DIFFERENCE, so we would have taken (30-10)=20 as something of interest to look at. Here we take the difference DIVIDED BY TWO. So we take (30-10)/2 = 10.

We can now present the Key length finder:

1. Find some (say) 4 letter word that occurs (say) 5 times.
2. Look at all the differences DIVIDED BY TWO. There are $\binom{5}{2} = 10$ of them, let them be $r_1, \ldots, r_{10}$.
3. Let $L_1, \ldots, L_s$ be all of the numbers that divide all of the r_i's (s will be very small). These $L_1, \ldots, L_s$ the candidates for Key Length.

USING THE KEYLENGTH

Let the text be $T = t_1t_2t_3 \cdots T_N$.

For each keylength candidate L do the following:

Break the text up into

$T_1 = t_1t_2t_{2L+1}t_{2L+2} \cdots$

$T_2 = t_3t_4t_{2L+3}t_{2L+4} \cdots$

$T_3 = t_5t_6t_{2L+5}t_{2L+6} \cdots$

$\ldots$

$T_L = t_{2L-1}t_{2L}t_{2L+(2L-1)}t_{2L+(2L)} \cdots$.

(The T_i are not infinite but I still use $\cdots$.)

KEY GOOD NEWS: If you decode T_i into English what you get will have the same single-letter and double-letter freq as English has.

For each T_i do the following:

There are TWO ways to proceed from here.

USING THE KEYWORD USING FREQ ANALYSE

Run a program to tell how many times each PAIR of letters occurs in the text- though do this the following way which we show by example

If the text is *abcdabdz*

count *ab*, then *cd*, then *ab*, then *dz*.

DO NOT count the *bc* since that was not coded by a matrix.

Once you do this note the first, second, third, and fourth most common pairs. Assume they map to the real first, second, third, and fourth most common pairs (they happen to be th, ne, in, er). Assume the matrix is $M = \begin{bmatrix} w & x \\ y & z \end{bmatrix}$.

You can get out FOUR equations from where the FOUR pairs map. You need FOUR equations since there are FOUR variables. Solve the equations to get w, x, y, z. Once you have the matrix apply it to T_i to get text. AH- but is it English? It might not be since the keylength might be wrong. Run IS-ENGLISH on the decoded Text to see if its English.

USING THE KEYWORD USING BRUTE FORCE

We'll be brief here since we do a similar thing in problem 4.

Look at EVERY matrix $M = \begin{bmatrix} w & x \\ y & z \end{bmatrix}$

For each such matrix decode T_i into what might be English. Test if its English using IS-ENGLISH. If NONE of the matrices give you English then you have the WRONG keylength.

Problem 3

Alice and Bob are going to use the Diffie-Helman key exchange to obtain a shared secret key. They use p and $g = 10$. Fill in the XXX in the following question and explain your answer.

If Alice uses a *and Bob uses* b *and* $(a, b) \in$ *XXX then Eve can find the shared secret key EASILY.*

ANSWER

If either $g^a \pmod{p}$ or $g^b \pmod{p}$ LESS THAN p then Eve can compute a and b easily.

$$XXX = \{(a, b) | 0 \le g^a \le p \text{ OR } g^b \le p\}$$

Problem 4

For this problem the alphabet is the standard 26 letter alphabet. Alice and Bob are going to use a 2×2 matrix cipher. Eve KNOWS they are going to use this. Describe how Eve can crack the code. Your description should be such that someone who has never seen any of the material in this course (except what the 2×2 matrix code is) can use it. You will need to write several programs; however, you can assume you have a program $MATRIXDECODE(T, M)$ which will, in input T a text (probably coded by a matrix code) and M a matrix, outputs what happens if you decode T using M.

If there is a parameter that you need then describe how you would determine that parameter.

You can assume the text is a sequence of numbers that represent letters.

SOLUTION TO PROBLEM 4

There are two solutions. Here is one of them:

Let $\vec{p}$ be the probability vector of English. It is KNOWN that $\vec{p} \cdot \vec{p} \approx 0.68$.

Let $\vec{q}$ be the probability vector if you use 2×2 matrix cipher. We need to know what $\vec{p} \cdot \vec{q}$ is (the approximation value). Let's call this value M. We will write pseudocode to approximate M.

Algorithm 1 Approximate the value of M

```
procedure APPROXIMATE_M
    $T_1, \ldots, T_N \leftarrow$ A large set of English texts
    $M \leftarrow 0.0$
    for $i = 1$ to $N$ do
        for $(a, b, c, d) \in \{0, \ldots, 25\}$ do
            $Q \leftarrow \begin{bmatrix} a & b \\ c & d \end{bmatrix}$
            if $\det(Q)$ is not relatively prime with 26 then
                break
            $T_{i,Q} \leftarrow \vec{p} \cdot$ (prob vector of$MATRIXDECODE(T_i, Q)$
            $M \leftarrow \max(M, T_{i,Q})$
    return $M$
```

Again, we will allow $\vec{p} \cdot \vec{q} \leq M + 0.02$.

We will need a function $COMPUTE\_PROB\_VECTOR(T)$ that can compute the probability vector of a large text T. However, we will OMIT it here since we already provided the pseudocode in our homework 2 solution.

Now we know M, then we are ready to write $IS\_ENGLISH?$ function.

Algorithm 2 Check a text is in English using matrix cipher

```
procedure IS_ENGLISH_MATRIX(T)
    $\vec{p} \leftarrow$ prob vector of English
    $\vec{q} \leftarrow$ COMPUTE_PROB_VECTOR(T)
    $DOT \leftarrow d(\vec{p}, \vec{q})$
    if $DOT \geq 0.66$ then
        THIS IS ENGLISH
        return True
    else if $DOT \leq M + 0.02$ then
        THIS IS NOT ENGLISH
        return False
    else
        THIS IS NOT ENCODED BY AFFINE CIPHER
        return False
```

Then we can start writing the decoding program.

Algorithm 3 Decode matrix cipher

```
procedure DECODE_MATRIX_CIPHER(T)
    $n \leftarrow$ length(T)
    for $(a, b, c, d) \in \{0, \ldots, 25\}$ do
        $Q \leftarrow \begin{bmatrix} a & b \\ c & d \end{bmatrix}$
        if det($Q$) is relatively prime with 26 then
            $T' \leftarrow MATRIXDECODE(T, Q)$
            if IS_ENGLISH_MATRIX($T'$) = true then
                T' is the plaintext
                return T' (and HALT the whole procedure)
```

The other solution is to use freq analysis- similar to the first solution to problem 2.

NOTE: My INTENT was that you find the parameter M and use it. I didn't penalize if you didn't do that. Think about if you really need it or not.