Cryptography and Network Security

Prof. D. Mukhopadhyay

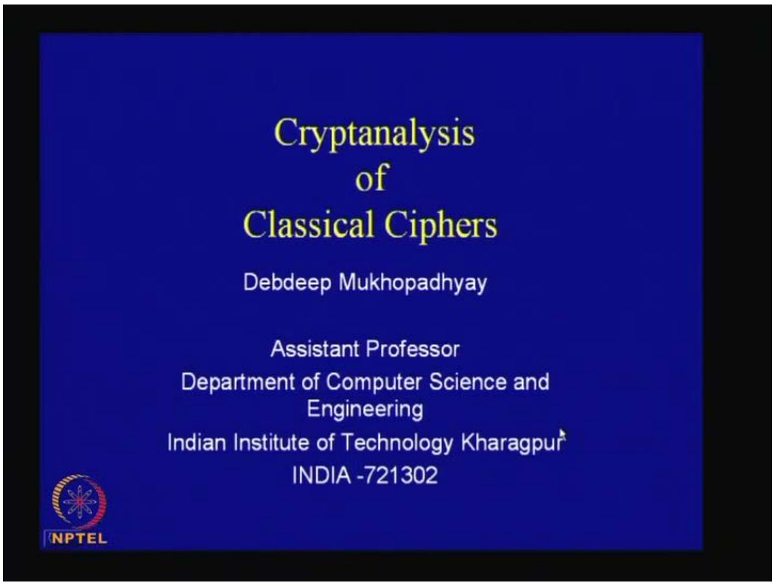
Department of Computer Science and Engineering

Indian Institute of Technology, Karagpur

Lecture No. #06

Cryptanalysis of Classical Ciphers

(Refer Slide Time: 00:24)



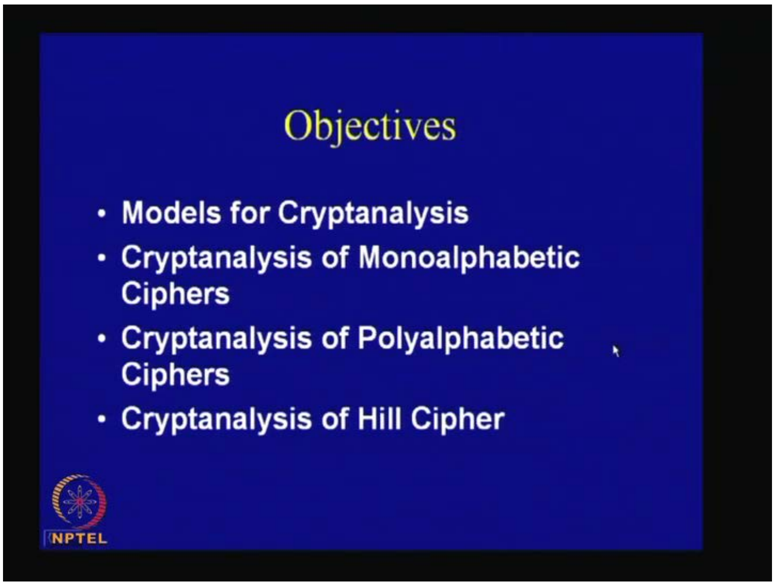
Cryptanalysis
of
Classical Ciphers

Debdeep Mukhopadhyay

Assistant Professor
Department of Computer Science and
Engineering
Indian Institute of Technology Kharagpur
INDIA -721302

Welcome to this class on cryptanalysis of classical ciphers. So, we will essentially be continuing with what we were discussing about classical ciphers, and rather discuss about some cryptanalytic techniques or attacking methods or methods to find out the keys in context to classical ciphers.

(Refer Slide Time: 00:37)

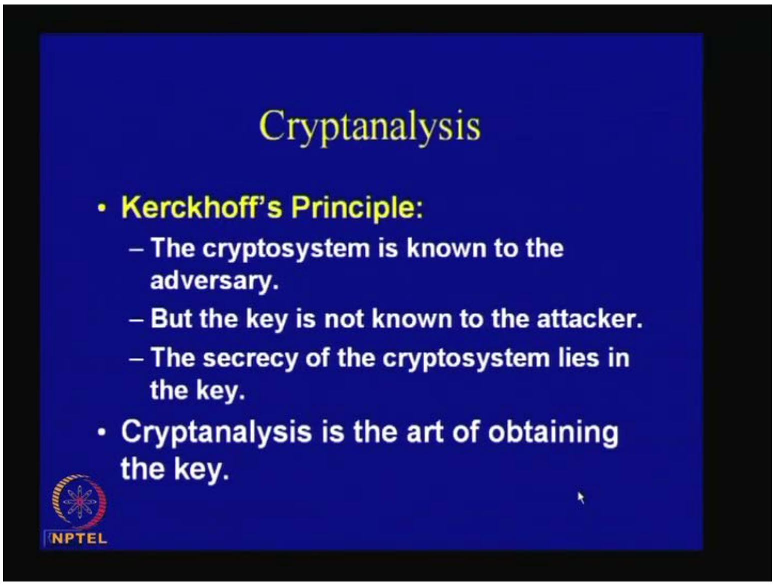


Objectives

- Models for Cryptanalysis
- Cryptanalysis of Monoalphabetic Ciphers
- Cryptanalysis of Polyalphabetic Ciphers
- Cryptanalysis of Hill Cipher

So, in today's class, our objectives are as follows: we will discuss about some models of for which are used for cryptanalysis; we exists for cryptanalysis, and then, essentially discuss about the cryptanalysis of monoalphabetic and polyalphabetic ciphers and conclude with cryptanalysis of hill ciphers.

(Refer Slide Time: 00:55)



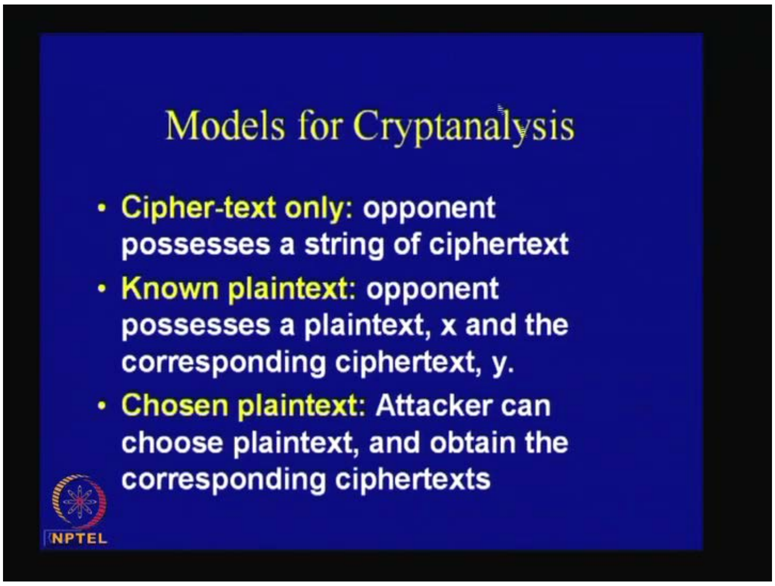
Cryptanalysis

- Kerckhoff's Principle:
  - The cryptosystem is known to the adversary.
  - But the key is not known to the attacker.
  - The secrecy of the cryptosystem lies in the key.
- Cryptanalysis is the art of obtaining the key.

So, to start with we were discussing that one of the fundamental principles in cryptography is or cryptanalysis is Kerckhoff's principle, which says that the cryptosystem is always available in the public domain is known to the adversary, but what is not known is the value of the key.

So, the entire secrecy of the cryptosystem lies in the key, and cryptanalysis is that field of cryptology is tries to obtain the value of the key, and it tries to find out the value of the key better than, typically better than a brute force search that, rather than, rather than doing an exhaustive search, it tries to find out or develop method and methods to find out or rather to obtain the key better than a brute force search of the for the key.

(Refer Slide Time: 01:39)



Models for Cryptanalysis

- Cipher-text only: opponent possesses a string of ciphertext
- Known plaintext: opponent possesses a plaintext, x and the corresponding ciphertext, y.
- Chosen plaintext: Attacker can choose plaintext, and obtain the corresponding ciphertexts

So, there are certain models which are been laid down for cryptanalysis and these models are often useful for the study of ciphers - the first and the most obvious in cipher text only attack, where the opponent posses a string of cipher text and that means that the opponent the attacker has access to only the ciphertext or the crypto text from there it tries to obtain the key.

Now, this is the kind of attack which is squarely would expects in a real life scenario, and therefore, it is kind of the sort of the strongest or the hardest tasks from the point of view of attacker, which has only the information of the cipher text, but in real life or rather in in when we do our studies, we actually sometimes allow the attacker or the cryptanalyser or the or the cryptanalyst some extra information apart from the cipher text.

What we do is there be sometimes, for example, we give an information of the plaintext and try to say that you have the cipher text and you also have the corresponding plaintext, and from, there you try to to deduce the value of the key. Now, this may not be, I mean, I mean intuitively very very practical, but it is often very relevant and there can be lot of examples where these kind of attacks which are known as the known plaintext attacks or chosen plaintext attacks can be also practical. So, in typically in a known plaintext attack, the opponent possesses a plaintext x and also the corresponding cipher text y.

Now, in order to give a practical scenario, imagine that we that is, for example, as a case where we write an email and we respond to the email by generally [press/pressing] pressing the reply button, so, what, what may happen is, for examples, the, the text that I have written remains in your reply message. So, this is a typical example where I, for example, I know what is the content of the plaintext, and if I observed that kind of way to out, then also have an access to a corresponding cipher texts.

So, this is an example of know plaintext attacks, where I have an access to a plaintext and also I obtain the corresponding cipher text. Now also this becomes kind of more relevant in context to asymmetric ciphers where anybody can entry it, because, because as we as we discussing that in a asymmetric cipher, there are two keys - right one of them is the public key and other one is the private key.

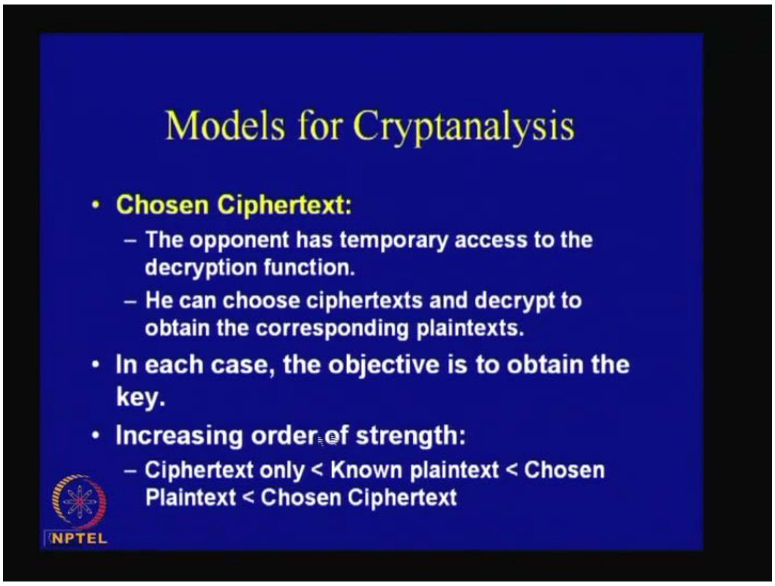
So, so, in this case, everybody who are, I mean knows the [pub/public] public key, right? So, the public key is known to everybody; so, which means that [every/everybody] everybody can or anybody can encrypt a plaintext message, and therefore, obtain I mean to obtain, the, the cipher text for a corresponding plaintext is not a difficult job. So, therefore, you can have ample amount of tupples for plaintext and ciphertext.

So therefore, in such kind of conditions, known plaintext attack is kind of not very kind and not very non intuitive or not very impractical, and therefore, it is also used an important model for sharing. So, they, there are some other model, so, cryptanalysis also like a chosen plaintext attack, where attacker can choose plaintext and obtain the corresponding ciphertext. So, in this case, the attacker can does, not, not only know the plaintext but can also choose the plaintext. So, this is an extra power extra capability which is there in the adversary.

As you can understand all these are actually different levels of adversary. So, which one is the kind of strongest attackers? The chosen plaintext attacker is, stronger, stronger than the known plaintext attacker, which is stronger than the cipher text only attacker, right?

So, in terms of information, the cipher text only attacker as by the least information compare to compare to known plaintext attacker and, which, which has lesser information compare to chosen plaintext attacker.

(Refer Slide Time: 05:24)



Models for Cryptanalysis

- Chosen Ciphertext:
  - The opponent has temporary access to the decryption function.
  - He can choose ciphertexts and decrypt to obtain the corresponding plaintexts.
- In each case, the objective is to obtain the key.
- Increasing order of strength:
  - Ciphertext only < Known plaintext < Chosen Plaintext < Chosen Ciphertext

So, there, there is another important model which is also used, which is known as the chosen ciphertext. So, in this case, the opponent has got temporary access to also the decryption function. What he can do is that he can choose ciphertext and decrypt to obtain the corresponding plaintexts, but what is to be kept in mind is that in this case of chosen ciphertext, the, the idea is like this, that is, an attacker is given a large number of ciphertext and also using its decryption function, it obtains its corresponding plaintext.

But in this decryption function, the key is kind of embedded. Therefore, when we are doing the decryption function, then the attacker does not have the knowledge of the key, right? but the at end of these kind of operation, which we call as oracle queries; what is done is that the attacker is given a challenge ciphertext and is asked to find out the corresponding key. So, this comes at end of all the previous exchanges which has taken place.

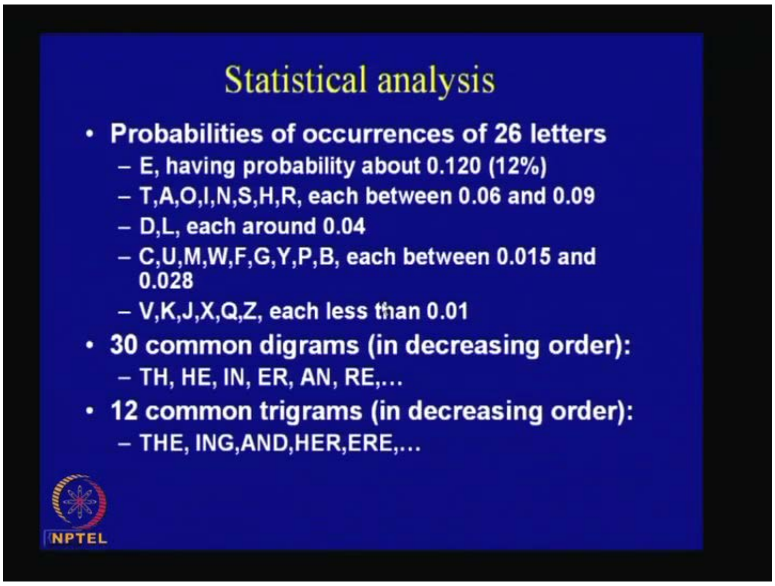
So, in this case, the attacker is kind of the strongest, right? This version is kind of the strongest notion, which has got access to not only the encryption function but also can do decryption functions of significant number of corresponding ciphertext, right?

So, therefore, in these case, in each case, objective is to obtain the key, and we can say that in increasing order of strength, the ciphertext only attacker is the least stronger attacker, strong, strong attacker; then comes known plaintext attack; then comes chosen plaintext attack, and finally, the chosen ciphertext attack which is the strongest form of the attacker, ok?

So, when you are designing a cipher, then ideally you would like to kind of look into the chosen ciphertext attack, and kind of say that my attack, my crypto system is prevented or protected against as a strong form of the attacker such as that of the chosen ciphertext attack, right?

So, therefore, ideally I would like to kind of counter even the chosen ciphertext attack, and from the point of view, when you are kind of attacking system, when you are doing research in crpytanalysis, then you would be more happy if you able to find out a ciphertext only attack, and you kind of less happy as you go down the series, right? So because the attacks becomes less and less stronger or more weak, ok.

(Refer Slide Time: 07:38)



Statistical analysis

- Probabilities of occurrences of 26 letters
  - E, having probability about 0.120 (12%)
  - T,A,O,I,N,S,H,R, each between 0.06 and 0.09
  - D,L, each around 0.04
  - C,U,M,W,F,G,Y,P,B, each between 0.015 and 0.028
  - V,K,J,X,Q,Z, each less than 0.01
- 30 common digrams (in decreasing order):
  - TH, HE, IN, ER, AN, RE,…
- 12 common trigrams (in decreasing order):
  - THE, ING,AND,HER,ERE,…

So, now, we will kind of discuss some tools which are often used for crpytanalysis, but it may be kept in mind these techniques are mainly applicable to cipher which is old that is, the classical ciphers, the machine ciphers or the modern ciphers are much more robust or much more strong. So, do start with, let us make some observations, like we see that English language, for example, has got certain probabilities of occurrences.

So, therefore, typically we have got twenty six letters and there are certain statistics which you can easily observe if you kind of do as do do a analysis of a large number of texts.

For example, e has got the highest probability and occurs with the probability of around twelve percent. The next in order comes, these, these letters, these alphabets like t a o i n s h and r, and d and l comes around 0.04, and then some other letters like as mention they are like c u m w f g y p and b, which comes between 0.015 and 0.028, and then, we have got v k j x q and z which are less than zero point zero one.

So, if, so these are the single operators or letters. If you kind of absorbs diagrams or double occurrences of letters, then t h is supposedly the most commonly offering diagram. Then you have got h e, then you have got i n and you have got these diagrams in decreasing order. Similarly, common tri trigrams would be like t h e t h e is the highest occurring trigram and then you have got these particular series. So, you can actually form statistics and we call this statistics as appriority statistics before we start the cryptanalysis, ok?

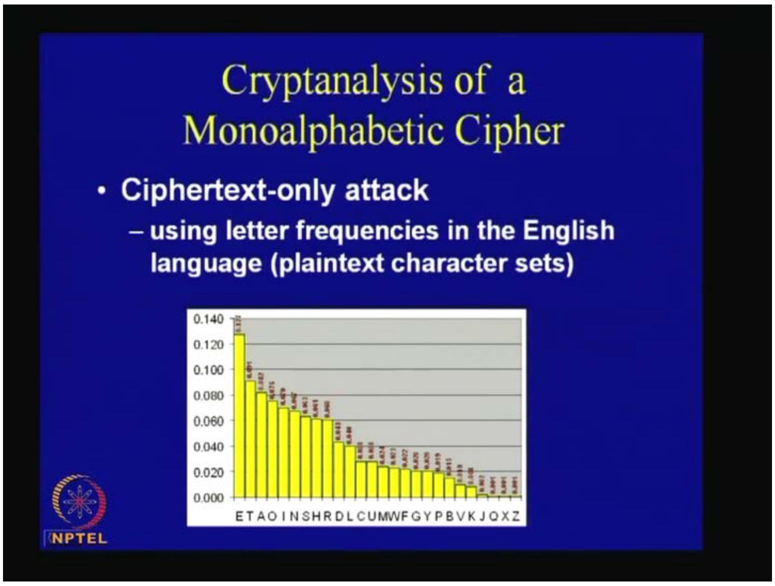
So, therefore, this is the knowledge of the plaintext text that you have. So, you know that a person who is using a cipher, is trying to, is trying to do an encryption over a meaningful message, right? So, this this meaningful message is in our case assume to be, formed, formed of English language, English language characters, it is an is belongs to the English language grammar, and therefore, what we have done is, for example, these statistics, are, are of some English language texts that have been preprocessed, and from there, people have, kind out, found out frequency distribution of single occurrences, of double occurrences, try triple occurrences and so on, and this is actually a very useful information when you are doing cryptanalysis.

So, let us to start with, let us consider a shift cipher. If you remember any, shift, shift cipher which we studied in context to monoalphabetic ciphers, then every letter is kind of given a shift, right?

So, therefore, for example, e which is the most commonly occurring letter is also shifted, right? So, in case of ceaser cipher, you know it has been shifted by say three steps, so, e becomes f g and h, right? Therefore, if e was the most commonly occurring letter in normal plaintext, then in the cipher text which uses the ceaser cipher for example, then h would be the most commonly occurring letter, right?

So, therefore, if you observe, if you take a kind of piece of cryptogram on the ciphertext and you start find now, finding out [thef/the] the later which occurs most frequently, and suppose that, [lat/letter] that letter is h, right? So, in that case, you can kind of conclude that h has got corresponding with e in the normal letter, right? So, that is the way how you can actually obtain the shift which exits incase of shift ciphers.

(Refer Slide Time: 11:03)



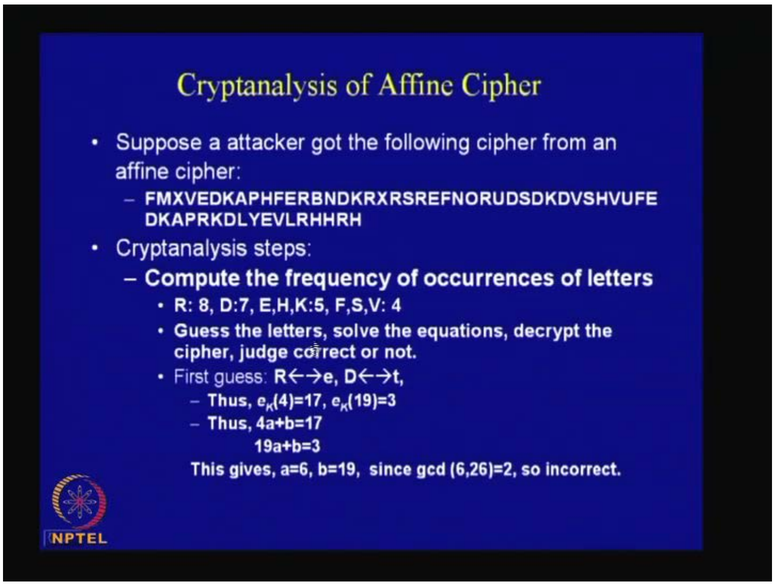
Cryptanalysis of a Monoalphabetic Cipher

- Ciphertext-only attack
  - using letter frequencies in the English language (plaintext character sets)

So, that is what we are said here. So, that incase of, so, this is actually an example of a cipher-text-only attack because here using only a letter frequencies, because you are having an access only to the cipher text, ok?

The other information which you have got is the letter frequencies in the English language or that is the, plaintext, plaintext character sets. So, these are the kind of some frequency distribution. You can see that e is the most commonly according letter. Similarly, here we have got t a o i n and so on. So, this is the gradual frequency bar to show the frequency distribution.

(Refer Slide Time: 11:37)



Cryptanalysis of Affine Cipher

- Suppose a attacker got the following cipher from an affine cipher:
  - FMXVEDKAPHFERBNDKRXRSREFNORUDSDKDVSHVUFE DKAPRKDLYEVLRHHRH
- Cryptanalysis steps:
  - **Compute the frequency of occurrences of letters**
    - **R: 8, D:7, E,H,K:5, F,S,V: 4**
    - **Guess the letters, solve the equations, decrypt the cipher, judge correct or not.**
    - First guess: **R←→e, D←→t,**
      - **Thus, $e_K(4)=17$, $e_K(19)=3$**
      - **Thus, $4a+b=17$**
        $19a+b=3$

      This gives, a=6, b=19, since gcd (6,26)=2, so incorrect.

So, in context of affine cipher, you see the suppose an attacker slow as little consider little bit of, let us consider the affine cipher, you remember the affine cipher, right? So, suppose an attacker has got the following cipher from an affine cipher. So, this is the kind of cipher text which has been kind of the retrieved, by, by an attacker and he knows that the cipher which is been correspondingly used is the affine cipher.

So, let us try to do some cryptanalysis. So, what, first of all we try to find out the frequency of occurrences of the letters. So, we find out for example, that r has got an occurrence of eight; d has the occurrence of seven; e and h and k has an occurrence of five; f s and v has an occurrence of four. So, first of all we would try to guess the letters and solve the equations, and then, decrypt the cipher and judge whether it makes a meaningful sentence or not.

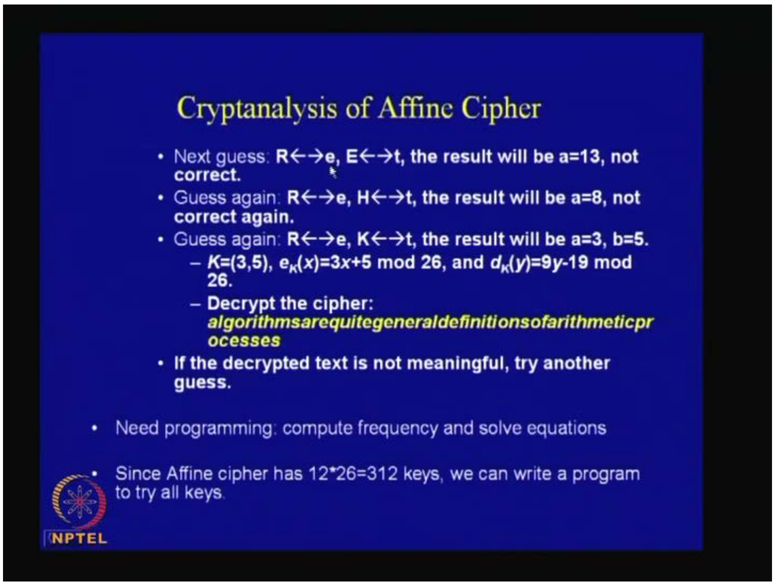
So, the first case would be like I told you that r is the highest according letter, and therefore, it should corresponds to e, because as we are studying in this frequency diagram, that e was an is generally the most frequently available letter, is most frequently occurring letter. So, in the cipher text, r is the most commonly according letter, then we known that the same this is the affine cipher then r should have corresponded to e.

So, therefore, we make an mapping like r is e, and similarly, and the next according letter is in this case d. So, the next occurring letter in this case being d, we say that d must correspond to the next occurring letter which is t. So, we say that d must have been mapped from t.

So, therefore, we can actually write this equations we know that e k. So, if i encode e as four and r i[s]- as seventeen, so, all the alphabets have been encoded by numbers from zero to twenty five. So, similarly, we have got the other equation as e k on nineteen and we know that that is equal to three; so, d is three and nineteen is denotes d.

So then, we have got these equations. So, if I, if I remember the affine ciphers, we had two keys. The key was kind of a tuple a, b, and therefore, you can write equations like 4 a plus b is equal to 17 and 19 a plus b is equal to 3. Therefore, you will solve this, you will find that a is equal to 6 and b is equal to 19. So, immediately you can say that this is a wrong guess, why? Because if you remember in a affine cipher, it is the requirement because of the invert ability of the affine cipher that a has to be co-prime to twenty six. So, if we take the greatest common divisor of a and 26, it should get one, but since this number is 6, and you know that if you take the g c d with 26, you actually get two, and since this is not equivalent to one, so, this is an incorrect decipherment.

(Refer Slide Time: 14:21)



Cryptanalysis of Affine Cipher

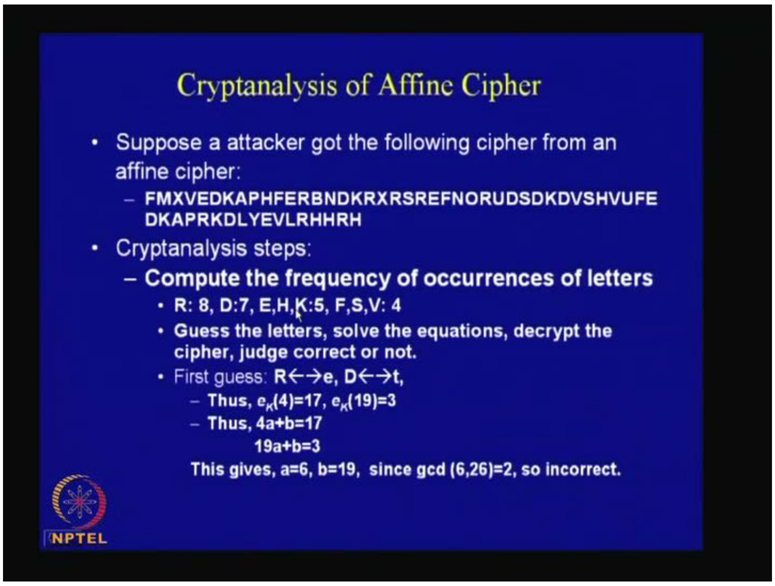
- Next guess: **R←→e, E←→t, the result will be a=13, not correct.**
- Guess again: **R←→e, H←→t, the result will be a=8, not correct again.**
- Guess again: **R←→e, K←→t, the result will be a=3, b=5.**
  - **K=(3,5), $e_K(x)$=3x+5 mod 26, and $d_K(y)$=9y-19 mod 26.**
  - **Decrypt the cipher:** ***algorithmsarequitegeneraldefinitionsofarithmeticpr ocesses***
- **If the decrypted text is not meaningful, try another guess.**

- Need programming: compute frequency and solve equations
- Since Affine cipher has 12*26=312 keys, we can write a program to try all keys.

So, therefore, we go for the next guess. So, keeping r as e, what we do is that we take the next according letter, which is in this case e, and we say that let e mapped to t. So, in this case you get a is equal to 13, and 13 is again not correct because 13 and 26 the g c d will be 13 which is again not equal to one.

So then, we go for the next occurrence. Therefore, we again keep r as e and the next occurring letter is h, which is map say t is map to h, and therefore, you solve for, again you will get a is equal to 18, which is again not correct because of the same reason that the g c d of a and 26 is again not equal to one.

(Refer Slide Time: 15:09)



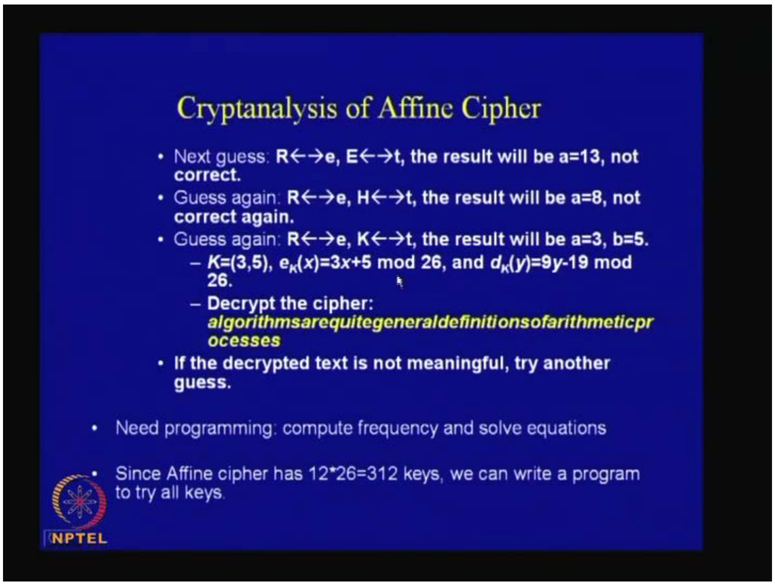
Cryptanalysis of Affine Cipher

- Suppose a attacker got the following cipher from an affine cipher:
  - FMXVEDKAPHFERBNDKRXRSREFNORUDSDKDVSHVUFE DKAPRKDLYEVLRHHRH
- Cryptanalysis steps:
  - **Compute the frequency of occurrences of letters**
    - **R: 8, D:7, E,H,K:5, F,S,V: 4**
    - **Guess the letters, solve the equations, decrypt the cipher, judge correct or not.**
    - First guess: **R←→e, D←→t,**
      - **Thus, $e_K(4)=17$, $e_K(19)=3$**
      - **Thus, 4a+b=17**
        19a+b=3

      This gives, a=6, b=19, since gcd (6,26)=2, so incorrect.

NPTEL

(Refer Slide Time: 15:14)



Cryptanalysis of Affine Cipher

- Next guess: **R←→e, E←→t, the result will be a=13, not correct.**
- Guess again: **R←→e, H←→t, the result will be a=8, not correct again.**
- Guess again: **R←→e, K←→t, the result will be a=3, b=5.**
  - **K=(3,5), $e_K(x)$=3x+5 mod 26, and $d_K(y)$=9y-19 mod 26.**
  - **Decrypt the cipher:** ***algorithmsarequitegeneraldefinitionsofarithmeticprocesses***
- **If the decrypted text is not meaningful, try another guess.**

- Need programming: compute frequency and solve equations
- Since Affine cipher has 12*26=312 keys, we can write a program to try all keys.

NPTEL

So, we can continue like this; so, luckily for us the next guess is correct, and we say that let t get mapped to k. Therefore, you see that the next occurrence letters, so, we take has t has got mapped into the next according letter that is k in this case, and therefore, we see that t has got mapped into k, and therefore, the corresponding equation is in this case, we solve this and get a is equal to three and b is equal to five.

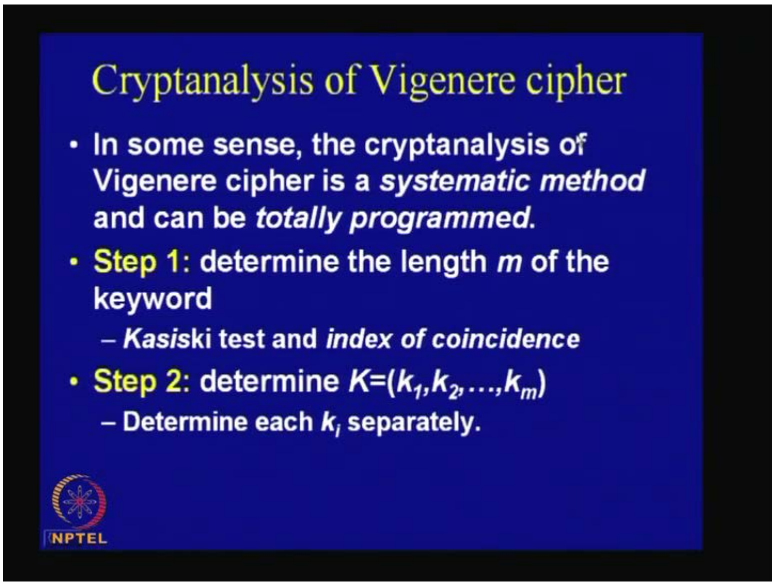
So, we now think that this is correct because if i take a g c d of a and 26 that I indeed get one, and therefore, I say let the formula can be this that is 3 x plus 5 mod 26, which is the encryption of x when k is used as the key. So, corresponding decryption function is actually it could obtain as 9 y minus 19 mod 26 and this decryption function exists, because g c d of a and 26 was equal to one. So, this we are discussed in the last class.

So, using this decryption function, we can actually decrypt the entire ciphertext and we get that algorithm are quite general definition so far and you get kind of a meaningful text, and therefore, if the decrypt and if this [decryption\decrypt] decrypted test, I mean text wouldn’t have been meaningful, then we would have tried another guess. So, therefore, you see that this is a kind of technique, and therefore, it can be programmed where you can actually compute the frequency and solve the equations and check whether the g c d of a and 26 was equal to one.

So, in this case, in case of affine ciphers, you know that the total number of possible keys keys is 12 into 26 which is equal to 312 keys which is quite small, and therefore, you can indeed write a program to try all the keys, but you know that, therefore, this idea about the the idea behind affine cipher is that you can actually use the frequency analysis technique quite deficiently, that is, you can actually form an apriority kind of frequency distribution of the English language text. From the English language text, you can form an apriority frequency distribution of the characters of the alphabet of the [dry/diagrams] diagrams and the trigrams and you can actually use this information to obtain the corresponding the cipherment of an, of the output of an affine cipher, ok?

But this may not be so obvious when you are have actually having a polyalphabetic cipher, because in a polyalphabetic cipher, one particular alphabet, if you remember, can get mapped into various alphabets, right? And therefore, the frequency distribution is not exactly maintained in this fashion but, you can actually discuss and we can a we will be discussing and see that actually you can use these idea, but you have to use it in a little bit more cleaver way, ok?

(Refer Slide Time: 17:37)

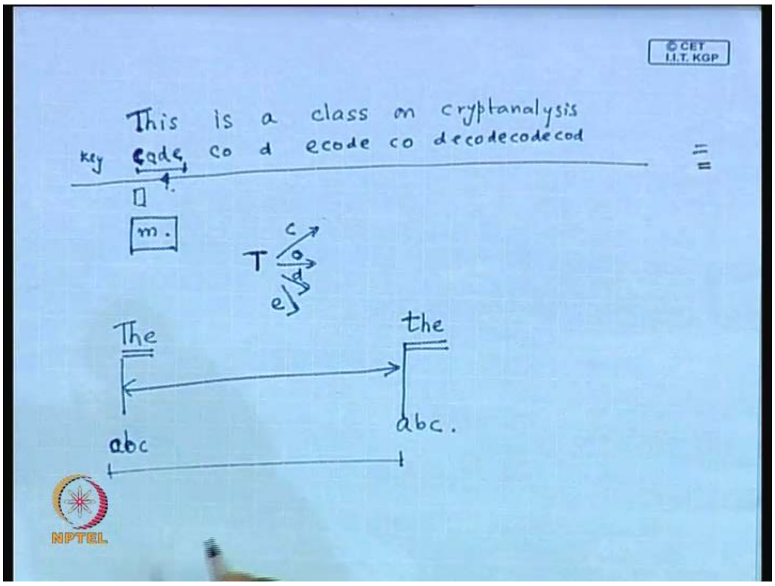


Cryptanalysis of Vigenere cipher

- In some sense, the cryptanalysis of Vigenere cipher is a *systematic method* and can be *totally programmed.*
- Step 1: determine the length *m* of the keyword
  - *Kasiski* test and *index of coincidence*
- Step 2: determine $K=(k_1,k_2,\ldots,k_m)$
  - Determine each k_i separately.

So, let us study that next. So, therefore, we discussed the cryptanalysis of the polyalphabetic cipher. Example of that, we discussed was the vigenere cipher. So, in some sense, the cryptanalysis of vigenere cipher is also a systematic method and can be totally programmed, but first of all let us try to understand that, that in a, in a case of a polyalphabetic cipher, if you remember that, you have to actually obtain the enti[re]- you have to obtain the keywords, right?

(Refer Slide Time: 18:11)



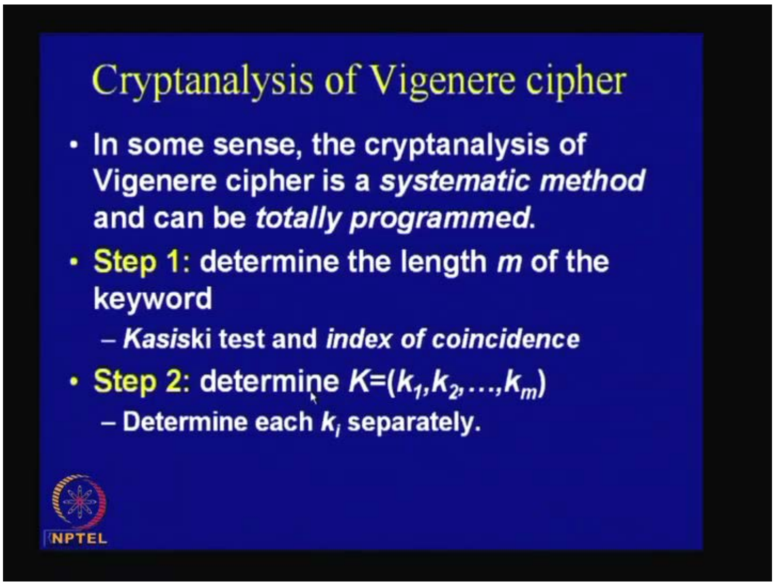
This is a class on cryptanalysis
key code co d ecode co decodecodecod
m.
T c o d e
The
the
abc
abc.

Therefore, the first step is determine the length of the key. So, what is done in a polyalphabetic cipher, if you to just to recap, you remember that you have got this plain text message. So, let us just consider a plaintext message the, for example, this is a class on cryptanalysis and the key could be, for example, code. So, in case of a polyalphabetic cipher, what we did was we took this code, this key was code and we kind of repeated them, right?

So therefore, so we kept like this, and, and we performed an addition operation. So, let as it was normal shift cipher. So, we took t, we added with the corresponding number with c and took a modular and obtained this.

So, the point which we noted here that is the this t because of its different occurrences could actually transform by either c or o or with d or with e. So, therefore, every letter like if I take t for example, can be shifted by c; can be shifted by o; can be shifted by d; can be shifted by e. So, if you are got keyword like this which has got instead of four length has got a length of m, then every letter has if there is a possibility can be encoded in a possibly by m possible transformations, right?

(Refer Slide Time: 19:51)



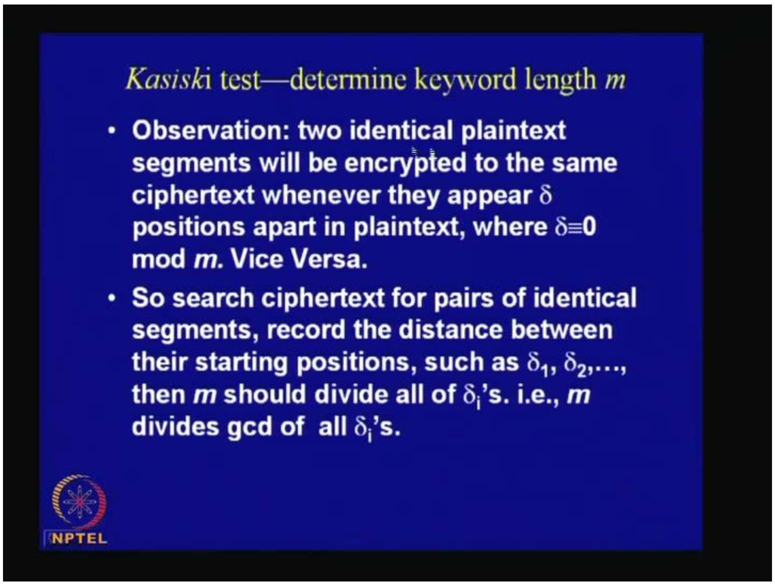
Cryptanalysis of Vigenere cipher

- In some sense, the cryptanalysis of Vigenere cipher is a *systematic method* and can be *totally programmed.*
- Step 1: determine the length *m* of the keyword
  - *Kasiski* test and *index of coincidence*
- Step 2: determine $K=(k_1,k_2,\ldots,k_m)$
  - Determine each k_i separately.

So therefore, the first interesting of the first important step is to determine the length of this key, that is, to find out the how many ways can letter get mapped. So, for this we do a study, and the first step is known as the kasiski step. So, therefore, the first step is to determine the length m of the word; it is called kasiski test, and then, we will confirm them by a term which is call as a index of coincidence.

So, we will first of all try to determine these key, that is, k is equal to k 1 k 2, and therefore, first of all we find out the length of key, and then, in second step, we will determine the key and we will determine each of these key quite separately, separately or is independently of each other each other.

(Refer Slide Time: 20:21)



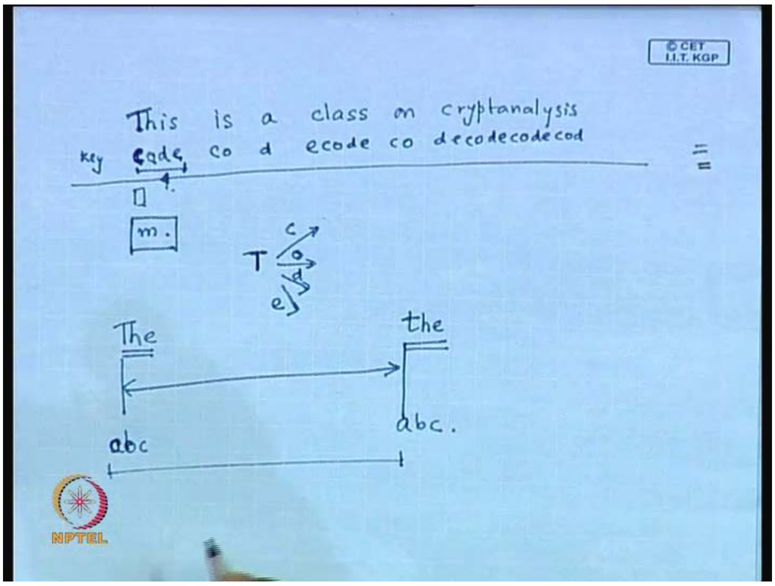
Kasiski test—determine keyword length *m*

- Observation: two identical plaintext segments will be encrypted to the same ciphertext whenever they appear δ positions apart in plaintext, where $\delta \equiv 0$ mod *m*. Vice Versa.
- So search ciphertext for pairs of identical segments, record the distance between their starting positions, such as $\delta_1, \delta_2, \ldots$, then *m* should divide all of δ_i's. i.e., *m* divides gcd of all δ_i's.

So, the first observation is that if there are two identical plaintext segments, then they will be encrypted to the same ciphertext when they appear delta positions apart in the plaintext - where delta is actually congruent to zero module p, module m. So, this is actually holds vice versa.

So, therefore, the idea is that if there are two identical plaintext segments, say for example, there are two identical plaintext segments and they will be encrypted to the same ciphertext whenever they appear a multiple of n number of times apart in the plaintext, ok?

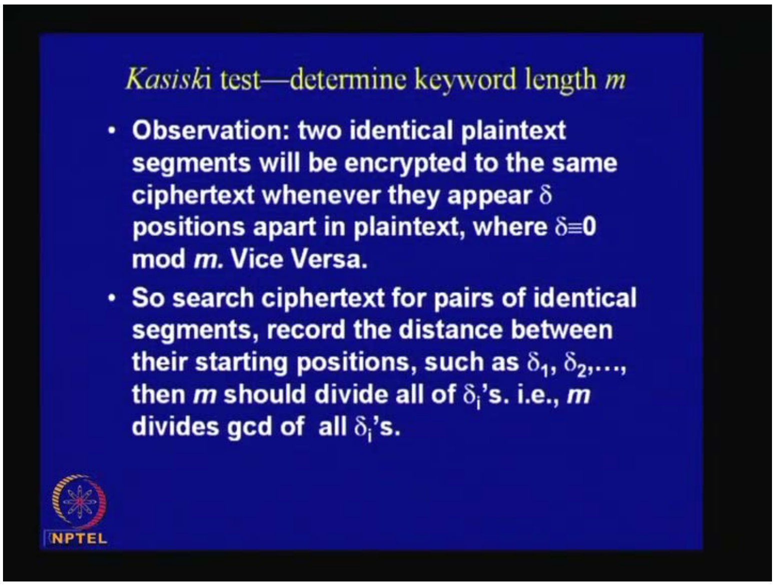
(Refer Slide Time: 20:55)


© CET I.I.T. KGP
This is a class on cryptanalysis
key code co d ecode co decodecodecod
m.
T
c
o
d
e
The
the
abc
abc.
NPTEL

See for example, if there is the particular letter like t h e, so, this is that an example of a trigram, and this is been therefore, if there is a repetition of this plaintext somewhere like if the t h e occurs again somewhere, and in the ciphertext also we have got a kind of suppose, we just take any [let/letter] any corresponding ciphertext like suppose this is map by a b c, then you can actually say that the separation of this and this, that is, this separation is actually a multiple of the size of the key, because the key has to exactly kind of divide, this the size of the key has to exactly divide the separation.

So therefore, this occurs here like this there is an identical kind of plaintext segment and it also encrypts to the same ciphertext. Then we can actually say that there separation, so, if you measure the separation of these two text, then this distance is actually a multiple of the size of the key.

(Refer Slide Time: 22:02)



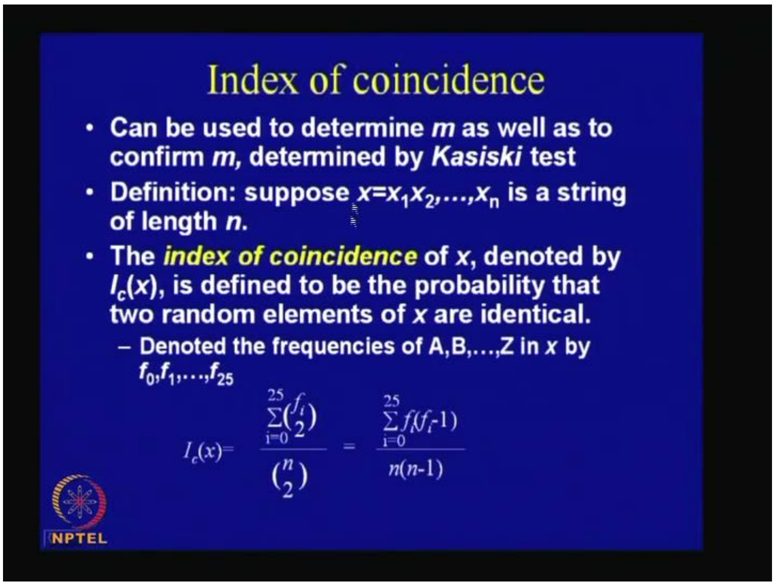
Kasiski test—determine keyword length m

- Observation: two identical plaintext segments will be encrypted to the same ciphertext whenever they appear δ positions apart in plaintext, where $\delta \equiv 0 \bmod m$. Vice Versa.
- So search ciphertext for pairs of identical segments, record the distance between their starting positions, such as δ_1, δ_2,…, then m should divide all of δ_i's. i.e., m divides gcd of all δ_i's.

So, you can actually to take the ciphertext, and from there, you can find out similar such occurrences and you can actually start observing the distances, and because the size of the key, actually divides all these distances. So then, what you can do is the, you can actually find out greatest common divisor of the distances and that can actually serve as a key size; so, that can serve as your key size.

So, what you can do is that you can very well said, that the key, that the key size is actually a, so, therefore, it actually divides all these distances, and therefore, it must divide the greatest common divisor of these distances, and then, in a later test which is known as the index of, coincidence, coincidence test, we kind of confirm this test, I mean we kind of confirm this key size.

(Refer Slide Time: 23:12)



Index of coincidence

- Can be used to determine *m* as well as to confirm *m,* determined by *Kasiski* test
- Definition: suppose $x=x_1x_2,\ldots,x_n$ is a string of length *n.*
- The ***index of coincidence*** of *x*, denoted by $I_c(x)$, is defined to be the probability that two random elements of *x* are identical.
  - Denoted the frequencies of A,B,…,Z in *x* by $f_0,f_1,\ldots,f_{25}$

$$I_c(x)=\frac{\sum_{i=0}^{25}\binom{f_i}{2}}{\binom{n}{2}}=\frac{\sum_{i=0}^{25}f_i(f_i-1)}{n(n-1)}$$

So therefore, so we what we do is that we take the ciphertext and we search them for pairs of identical segments and then we record the distances between them, between the starting positions. So, suppose they are delta 1 delta 2 and so on, then m should divide all of these delta i's, and therefore, m should divide the g c d of all these delta i’s. So, this can be use. So, in the next text, what we do is that we actually use the index of coincidence to determine n as well as to rather we use it to confirm m which is determine by the kasiski test.

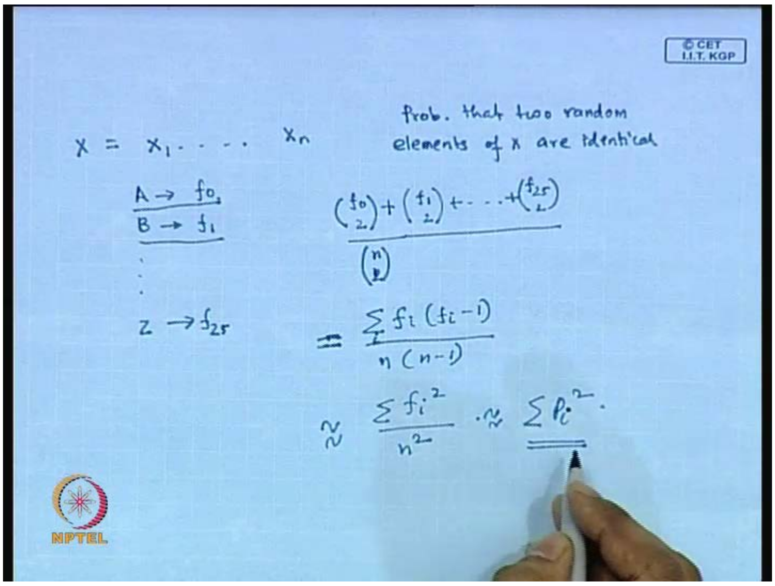
So, what is the definition of the index of coincidence? So, let as see the definition. So, suppose, here x is equal to x 1 to x x 1 x 2 and so on till x n, this is a string of length n, right? Now, the index of coincidence of x is denoted, so, it is often denoted by i c x and it is defined to be the probability that two random elements of x are identical. So, you take an x which is the string and you find out the index of coincidence and it is defined to be the probability that if you take two random elements of x, then they are identical.

So, so, let us try to compute this index of coincidence and let as assume that the frequencies of a b and so on till z in x is actually obtain to be f 0 f 1 and so on till f 25. So, now, we will find out that what is the probability that two random elements of x are identical? So, how many ways can you actually choose randomly to two elements from this string of length n? You can actually choose them by m c to s.

And since you are you want to find out the probably the two random elements of x are actually identical; so, which means that if the first letter which you have chosen is a, then the second letter is also a; if the second letter, if the first letter is b, in a second letter is also b.

So, if you assume that of the frequency of occurrence of a is denoted by f zero, then how many ways can we choose, can both the [choose/choosing] choosing slide, both the, when you are choosing, both, both times, then both times you are choosing a.

(Refer Slide Time: 25:35)



©CET I.I.T. KGP

Prob. that two random elements of x are identical

$X = x_1 \ldots \ldots x_n$

$A \rightarrow f_0,$

$B \rightarrow f_1$

$\vdots$

$Z \rightarrow f_{25}$

$$\frac{\binom{f_0}{2} + \binom{f_1}{2} + \ldots + \binom{f_{25}}{2}}{\binom{n}{2}}$$

$$= \frac{\sum_i f_i(f_i - 1)}{n(n-1)}$$

$$\approx \frac{\sum f_i^2}{n^2} \approx \underline{\underline{\sum p_i^2}}.$$

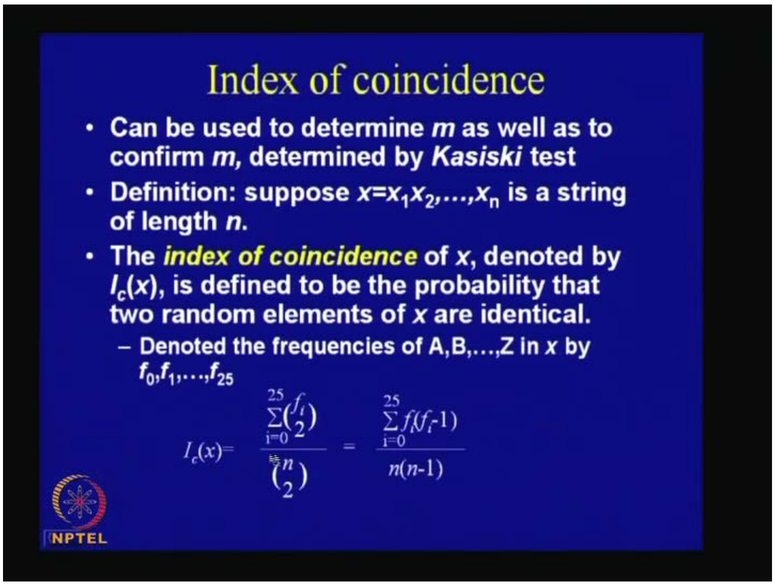
So, how many ways can you do that? It is actually if the first time, if you are chosen this f zero times, right? In the second case, it is the, it can be like the remaining f 0 minus 1 cases because of first cases we have already chosen this; so, that means that if you kind of make a sigma over this, then that means that, so if, if you have got a occurrences, that is, if you have got what I am trying to say is this, that is, there are n letters, so, you have got x 1 to x n and that forms your string x, and you know that the occurrence of a here is denoted by f 0; you are occurrence of d is denoted by f 1; similarly, the occurrence of z is denoted by f 25.

So, how many ways, can, can you choose to two elements in this n choose two, and how many ways can the both the letters be a? So, that is the question, right? You have to determine the probability; you have to find out the probability that two random elements of x are identical. This is what do you have to find now. So, you see that, the, the number of cases where we have got a is actually denoted by f 0.

So therefore, if we have got to choose two letters from here, both of them are a, it is actually f 0 choose 2, because of by choose this two letters from this f 2 is, right? The second thing you can actually choose them by f 1 choose 2, because that gives the number of ways in which you can choose b. So, you can continue like this and the final thing will be f 25 choose 2.

So, you can actually approximate this and this will be actually equal to sigma this actually equal to sigma f i into f i minus 1 divided by n into n minus 1 for all possible i’s. So, this you can actually approximate and make it equal to sigma f i square by n squared. You can actually bring this n square below and you can actually neglect this minus 1. So, this actually will work out to be equal to sigma p i square; so, that is the square of the probability.

(Refer Slide Time: 27:54)



Index of coincidence

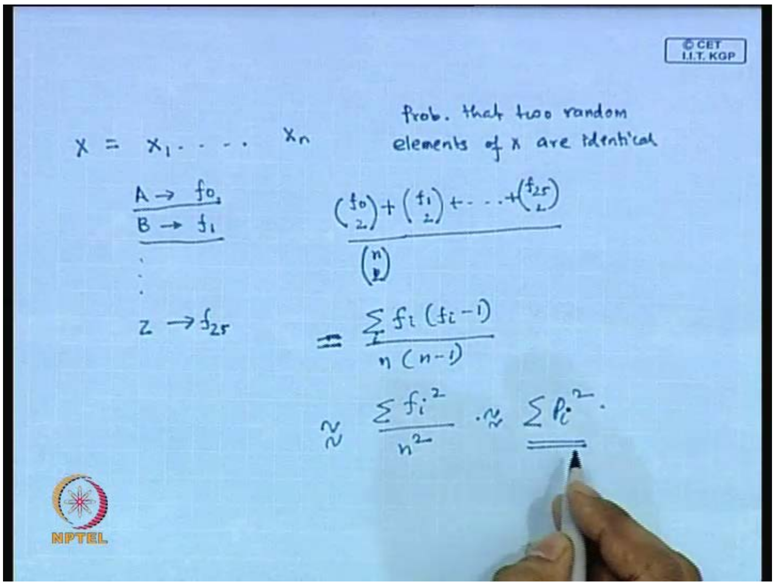
- Can be used to determine *m* as well as to confirm *m,* determined by *Kasiski* test
- Definition: suppose $x=x_1x_2,\ldots,x_n$ is a string of length *n.*
- The *index of coincidence* of *x*, denoted by $I_c(x)$, is defined to be the probability that two random elements of *x* are identical.
  - Denoted the frequencies of A,B,…,Z in *x* by $f_0,f_1,\ldots,f_{25}$

$$I_c(x)=\frac{\sum_{i=0}^{25}\binom{f_i}{2}}{\binom{n}{2}}=\frac{\sum_{i=0}^{25}f_i(f_i-1)}{n(n-1)}$$

So, you can actually deduce this in this passion, and therefore, you obtain this results, that is, the index of coincidence is nothing but sigma f i into f i minus 1 divided by n squared.

So, you can actually neglect this minus 1 for both sides, and then, you can actually divide f i and make it f i by n and that is squared. So, therefore, f i by n is nothing but a probability that direct letter occurs.

(Refer Slide Time: 28:16)



Prob. that two random elements of x are identical

$X = x_1 \cdot \cdot \cdot \cdot \; x_n$

$A \rightarrow f_0,$

$B \rightarrow f_1$

$\vdots$

$Z \rightarrow f_{25}$

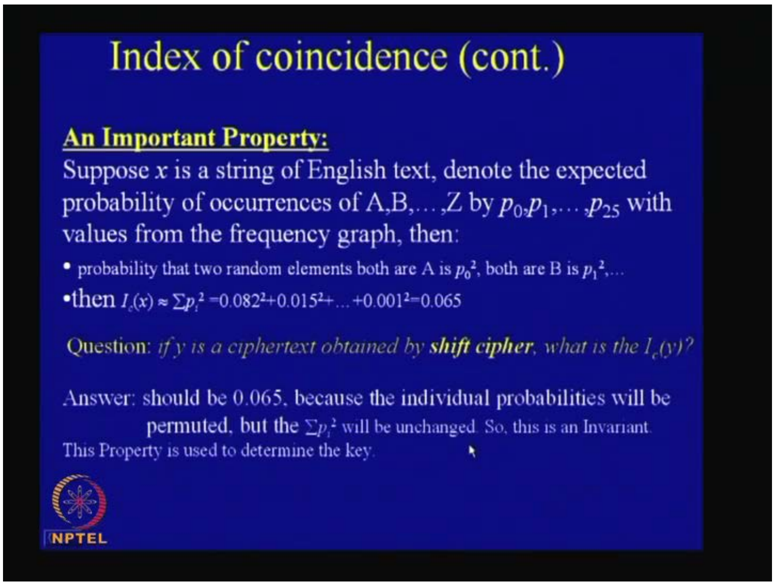
$$\frac{\binom{f_0}{2}+\binom{f_1}{2}+\cdots+\binom{f_{25}}{2}}{\binom{n}{2}}$$

$$= \frac{\sum_i f_i(f_i-1)}{n(n-1)}$$

$$\approx \frac{\sum f_i^2}{n^2} \approx \underline{\underline{\sum p_i^2}}.$$

So therefore, you get sigma p i square, which is denoted by here as sigma p i square; so, that means that you obtain the probability of any alphabet and then find out the square and then you take a sigma of all the possible i values, ok?

(Refer Slide Time: 28:27)



Index of coincidence (cont.)

An Important Property:

Suppose x is a string of English text, denote the expected probability of occurrences of A,B,…,Z by $p_0,p_1,\ldots,p_{25}$ with values from the frequency graph, then:

- probability that two random elements both are A is p_0^2, both are B is p_1^2,…
- then $I_c(x) \approx \sum p_i^2 = 0.082^2+0.015^2+\ldots+0.001^2=0.065$

Question: *if y is a ciphertext obtained by* ***shift cipher****, what is the $I_c(y)$?*

Answer: should be 0.065, because the individual probabilities will be permuted, but the $\sum p_i^2$ will be unchanged. So, this is an Invariant.

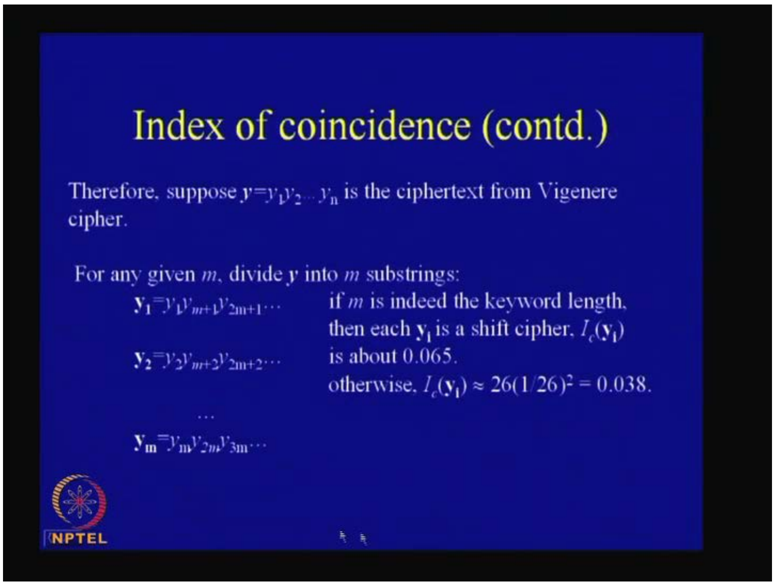
This Property is used to determine the key.

So, this you can actually do here, therefore, suppose x is a English text denoted by the expected probability of occurrences of a, b and z and so on till, a b and so on till z, and then, the frequency are denoted by p 0 p 1 and so on till p 25 with values from the frequency graph. As we have seen before that the probability of two random elements both of them being a is p 2 square; both of them being b is p 1 square, and therefore, you can say that the i c x as we have seen before is sigma p i square, and therefore, if you take the squares like 0.082 square plus 0.015 square plus so on, you get a value of 0.065. Now, this values is very important because of a reason that if, I tell you exactly why it is important, but please remember this numbers; so, it is 0.065, ok?

So now, if y is a ciphertext which is obtained by a shift cipher, then what is, i c, i c y? So, that is the question. So, if i take x and if i kind of transform this x by a by a shift cipher, then what will be the [i/i c y] i c y? So, you note one thing that if you would take a shift cipher, then every alphabet said is kind of just permuted the frequency distribution as such do not get change.

So therefore, the i c y x actually do not get change and it should remain 0.065 because the individual probability is will be just permuted, but the sigma p i square will not change; that will remain invariant. So, this property is actually used or exploited to determine the key value.

(Refer Slide Time: 29:55)



Index of coincidence (contd.)

Therefore, suppose $y=y_1y_2 \dots y_n$ is the ciphertext from Vigenere cipher.

For any given m, divide y into m substrings:

$\mathbf{y}_1=y_1y_{m+1}y_{2m+1}\cdots$

$\mathbf{y}_2=y_2y_{m+2}y_{2m+2}\cdots$

$\dots$

$\mathbf{y}_m=y_my_{2m}y_{3m}\cdots$

if m is indeed the keyword length, then each $\mathbf{y}_i$ is a shift cipher, $I_c(\mathbf{y}_i)$ is about 0.065.

otherwise, $I_c(\mathbf{y}_i) \approx 26(1/26)^2 = 0.038$.

So, we continue with our index of coincidence, and then, what we do is that, so, if you remember that in our kasiski test, we have got the suggestion of m. So, what we do is that starting from one to m, we actually start arranging these entire letters like this.

So, you have got like y 1 y 2 and so on till y n which is the ciphertext, which is obtained from the polyalphabetic vigenere cipher. Then for any given m, if we are to kind of confirm this value of m, what we do is that we start dividing y into m substrings like this. So, what we do is that we make [subst/substrings] m substrings like y 1 y 2 and so on till y m like this.

So, we say got y 1 first then y 2 and so on till y m; then y m plus 1 y m plus 2 and so on till y 2 m; then we have got start from y 2 m plus 1 y 2 m plus 2 and so on till y 3 m and we continue like this, ok?

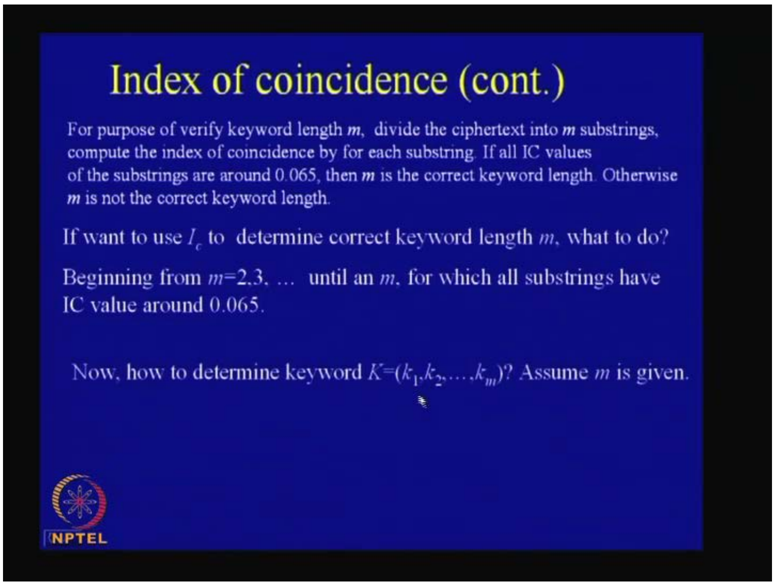
So, if aim is indeed the keyword length, then you see that each of these rows each of these rows will essentially be a shift cipher, because if you remember how the polyalphabetic cipher works, so, if your keyword is actually indeed I mean, indeed, indeed a string of length m, then each of this letters are actually being obtained from the plaintext by shifting name by, suppose, this is the, this is by the first alphabetic in the key; this is shifted by the second alphabetic in the key and so on; this is shifted by the mth alphabet in the key, but what about this? This is again shifted by the first alphabet in the k[ey]- in the key, right?

So therefore, all these elements which are there in the rows or all these, cipher, cipher alphabets are actually shifted by the same letter. So, this is also shifted by the same letter; these also shifted by letters and these are also sifted by the same letters; so, which means that each of these row are nothing but shift ciphers, and therefore, they are individual i c i’s should also be equal to, I mean should also be equal to 0.065, but if it is not so, then that is if m is not a actual keyword, then all of them will be a kind of random string, because this could be shifted by something; this will be shifted by something; this will be, the next one will be shifted by something, right?

So therefore, they will serve a as a kind of random string, and therefore, for the random string, actually you will find that if each of this alphabet occur with the probably, each of this alphabets character, I mean English language alphabets accords with the probability of 1 by 26, then the i c of that corresponding text will be equal to 26 into 1 by 26 square, because that is the sigma p i square in that case and that works out to 0.038.

And now, you see that the 0.038 value and the 0.065 are actually quite distinguishable, right? So therefore, you see that this is the property which exists in these character set, which is actually quite distinct from that of a random string, and therefore, these as I told you at the very beginning of our [disca/discuss] of our classes, that if you of actually observed the property which makes, I mean makes which kind of distinguishes that given cipher from a random distribution, then you can actually exploit that for developing an attack, which is the precisely what is done here also, ok?

(Refer Slide Time: 33:28)



Index of coincidence (cont.)

For purpose of verify keyword length ***m***, divide the ciphertext into ***m*** substrings, compute the index of coincidence by for each substring. If all IC values of the substrings are around 0.065, then ***m*** is the correct keyword length. Otherwise ***m*** is not the correct keyword length.

If want to use I_c to determine correct keyword length m, what to do?

Beginning from m=2,3, … until an m, for which all substrings have IC value around 0.065.

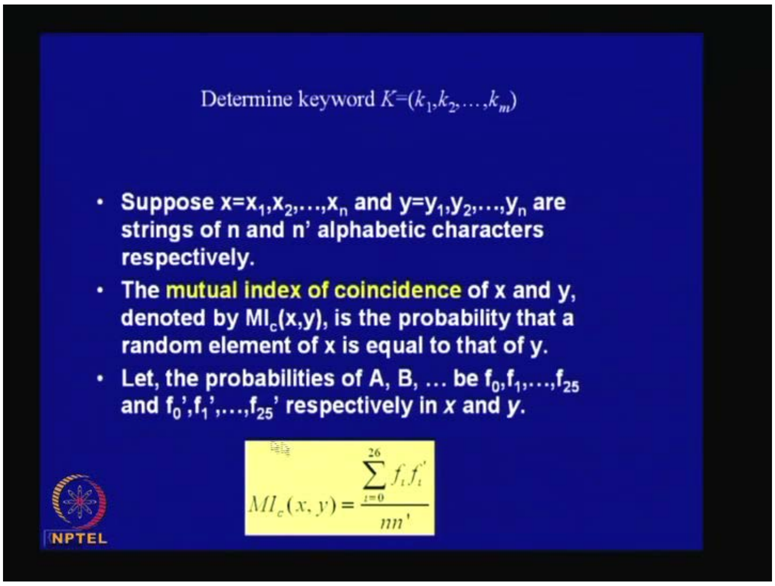
Now, how to determine keyword $K=(k_1,k_2,\ldots,k_m)$? Assume m is given.

So therefore, you first of all you in order to confirm the value of the keys of the key size, then what you do is that you start arranging them like this and then you start finding out the i c's of the corresponding rows. If your i c's work out to be 0.065 for all these rows, then the value of m is confirmed; otherwise, if the value of i c's works out to around 0.038, then the value of m is wrong value. So, you can start with the next value until and unless m.

So for the purpose of in order to verifying the keyword length m, what you do is that divide the ciphertext into m substrings and then you compute the index of coincidence for each substring. If all the i c values of the substrings that around 0.065, then m is the correct keyword length; otherwise, m is not the correct keyword length, ok?

So, if you want to use i c to determine the correct keyword length m, then what we will do? You start from beginning from m equal to 2 3 and so on until a m for which all substrings have i c values of around 0.065. Now, the question is how to determine the actual key? Or the now you have got, you have rather you have confirmed the size of the key. Now, your objective is to find out each of the keys.

(Refer Slide Time: 34:16)



Determine keyword $K=(k_1,k_2,\ldots,k_m)$

- Suppose $x=x_1,x_2,\ldots,x_n$ and $y=y_1,y_2,\ldots,y_n$ are strings of n and n' alphabetic characters respectively.
- The mutual index of coincidence of x and y, denoted by $MI_c(x,y)$, is the probability that a random element of x is equal to that of y.
- Let, the probabilities of A, B, … be $f_0,f_1,\ldots,f_{25}$ and $f_0',f_1',\ldots,f_{25}'$ respectively in *x* and *y*.

$$MI_c(x,y)=\frac{\sum_{i=0}^{26} f_i f_i'}{nn'}$$

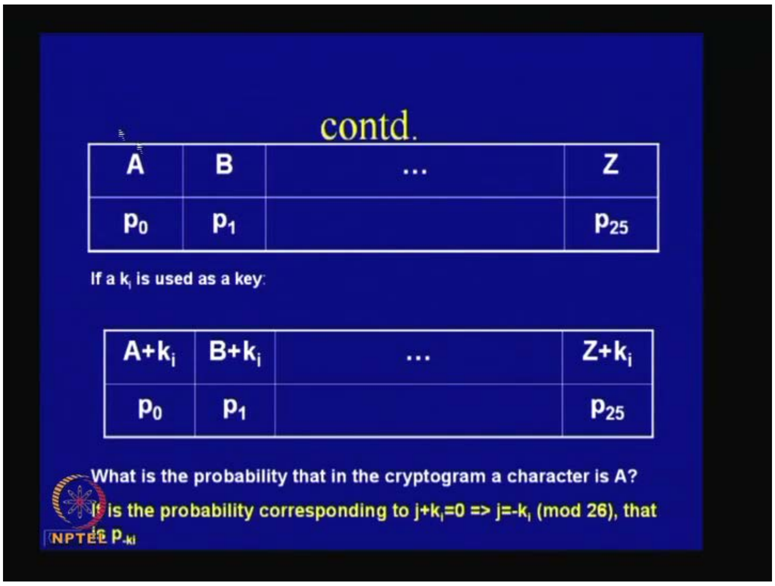
So, for doing that or rather to in order to determine the keyword, we use another concept which is known as the mutual index of coincidence. So, mutual index of coincidence works as follows: so, consider a two strings like x and y which have formed of x 1 x 2 and so on till x n, and y is formed of y 1 y 2 and so on till y n. These are strings of n and n dash alphabetic characters respectively. Then the mutual index of coincidence of x and y is denoted by this m i c x, y. Here it is the probability that a random element of x is actually equal to the random element of y.

So therefore, using a i mean the calculation is exactly the same as that of index of coincidence. If your probabilities of occurrence in case of a b and so on, so, it is actually not the probabilities but the frequencies. The frequencies are like f 0 f 1 and till so on till f 25, and for the next 1 is f 0 dash f 1 dash and so on till f 25 dash, then the mutual index of coincidence is obtain approximately as sigma f i and it obtained as sigma f i f i dash divided by n into n dash.

So, this n into n dash is nothing but the tutorial number of a ‘s in which can choose to alphabets, and this is the total number of a’s in which you can actually choose so that the both that, so both the thing which you choose are the same, that is, both of them are either a or both of them are b and both of them or c and so on.

So, number of a’s in which you can choose the ith from x from the first string is a so that both of them are the same letters is f i and the number of a’s in which you can choose the ith element. In the second street is a f i dash because there of that is the frequency of occurrences, and therefore, the probability of [cho/choose] I mean of the probability that random element of x is equal to the run an element of y is given by this ratio.

(Refer Slide Time: 36:05)



contd.

A	B	…	Z
p_0	p_1		p_{25}

If a k_i is used as a key:

$A+k_i$	$B+k_i$	…	$Z+k_i$
p_0	p_1		p_{25}

What is the probability that in the cryptogram a character is A?

It is the probability corresponding to $j+k_i=0 \Rightarrow j=-k_i \pmod{26}$, that is p_{-ki}

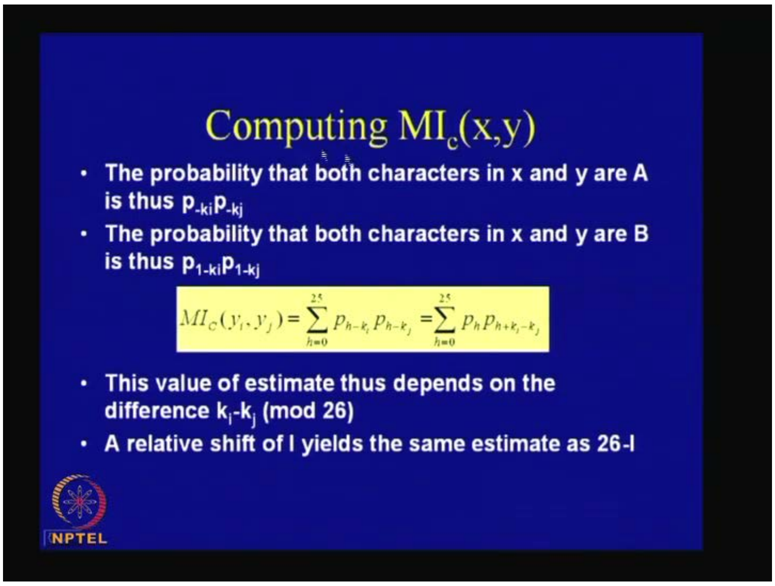
So, if you see that if you’ve got a b and so on till z and these are the corresponding probabilities like p 0 p 1 and till so on till p 25, and if a i is used as a key, then each of these letters get transformed as a plus k I; b get transformed to b plus k i and so on till z gets transformed by z plus k i.

So now, if i ask you like what is the probability that in the cryptogram a character is a. So, a is denoted by the letter 0, by the number 0. So, therefore, it is the probability corresponding to j plus k i which is equal to 0, and therefore, j will be equal to minus k i mod 26, that, is this probability will be equal to p minus k j, right?

So therefore, that is equal to p of minus k i mean p of minus k i. So, that is p j which is p of minus k I; so, that is the corresponding probability.

So what i'm saying is basically that now if i tell you that what is the probability that in the cryptogram a character is a, then this is not a; it is actually a plus k i. So, therefore, among all these strings among all these alphabets, you have to find out which one is corresponds to 0. So, suppose j plus k i corresponds to 0, therefore, the corresponding probability here will be p j and this p j is nothing but p of minus k i.

(Refer Slide Time: 37:21)



Computing $MI_c(x,y)$

- The probability that both characters in x and y are A is thus $p_{-ki}p_{-kj}$
- The probability that both characters in x and y are B is thus $p_{1-ki}p_{1-kj}$

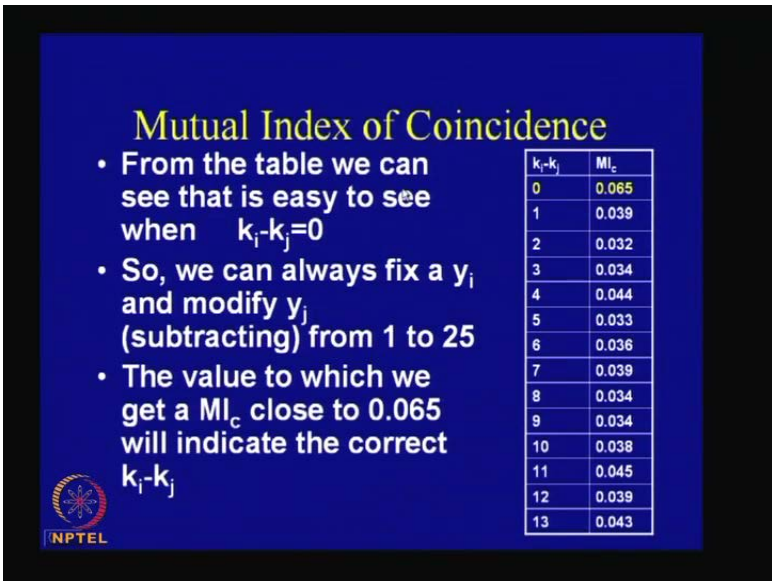
$$MI_C(y_i, y_j) = \sum_{h=0}^{25} p_{h-k_i} p_{h-k_j} = \sum_{h=0}^{25} p_h p_{h+k_i-k_j}$$

- This value of estimate thus depends on the difference k_i-k_j (mod 26)
- A relative shift of l yields the same estimate as 26-l

So similarly, the probability that both character in x and y are a is therefore found out by p of minus k i multiplied by p of minus k j because of they are independent choosing, and similarly, the probability that both characters x and y are b it can obtain by p 1 minus k i multiplied by p 1 minus k j and you can continue in this passion, and therefore, the mutually index of coincidence of these two strings will be equal to sigma p h minus k i multiplied by p h minus k j - where h varies from 0 to 25, and that is equal to sigma; h is equal to 0 to 25 and you can actually make some changes in the in the variables and you will get the p h multiplied by p h plus k i minus k j and then you take a sigma from h is equal to 0 to h equal to 25.

So therefore, you actually obtain a mutual index of coincidence, and this value, therefore this mutual index of coincidence actually realize upon k i minus k j; so, that is the shift, right? Therefore, it is the it depends upon the difference k i minus k j mod 26 and you can actually prove this technical exercise that a relative shift of i yields the same estimate as that of 26 minus I; that is quite trivial from this formula, right?

(Refer Slide Time: 38:44)



Mutual Index of Coincidence

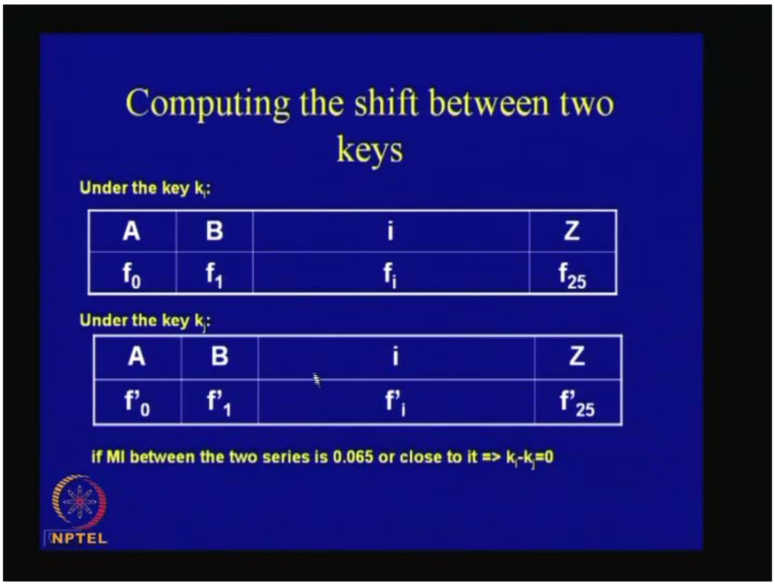
- From the table we can see that is easy to see when k_i-k_j=0
- So, we can always fix a y_i and modify y_j (subtracting) from 1 to 25
- The value to which we get a MI_c close to 0.065 will indicate the correct k_i-k_j

k_i-k_j	MI_c
0	0.065
1	0.039
2	0.032
3	0.034
4	0.044
5	0.033
6	0.036
7	0.039
8	0.034
9	0.034
10	0.038
11	0.045
12	0.039
13	0.043

So now you see that they there some typical values of mutual index of coincidence for different values of k i minus k j. So, you see that if k i minus k j is 0, then it is same as that of the index of coincidence, and therefore, you get the value of 0.065, but for other value of k i minus k j, you actually get a value which actually varies, varies around 0.03, and therefore, it is quite distinct from this fact that k i minus k j is equal to 0.

So, what we can do is that you can always fix a y i and you can modify the corresponding y j by subtracting from 1 to 25 and then the value to which we get a m i c which is close to 0.065 will actually indicate the correct value of k i minus k j.

(Refer Slide Time: 36:25)



Computing the shift between two keys

Under the key k_i:

A	B	i	Z
f_0	f_1	f_i	f_{25}

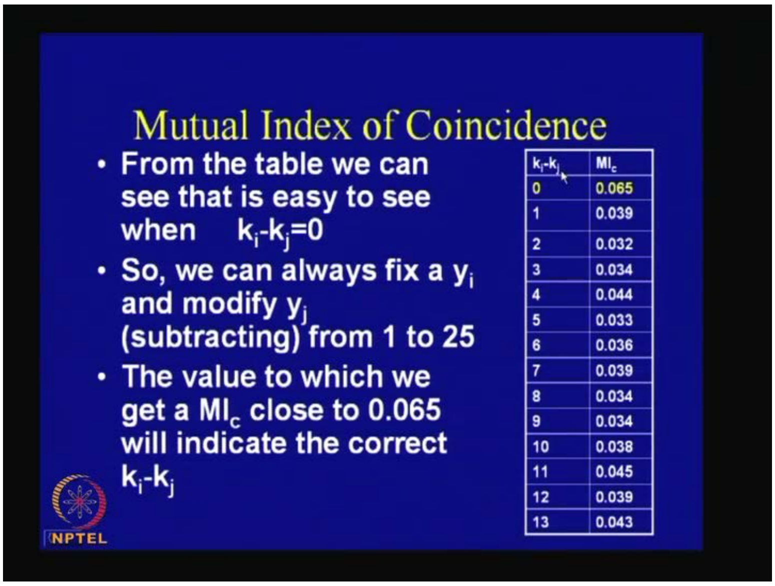
Under the key k_j:

A	B	i	Z
f'_0	f'_1	f'_i	f'_{25}

if MI between the two series is 0.065 or close to it => k_i-k_j=0

So, you can actually I mean try to understand using this, that is, so, if you if you want to compute the shift between two keys, what we do is that under the key k i, you obtain the this is the corresponding frequency of occurrences, and if under the corresponding key k j, this is the frequency of occurrences and you consider the m i between these two series and it works out to 0.065, then you can say that k i and k minus k j is equal to 0, right?

(Refer Slide Time: 39:51)

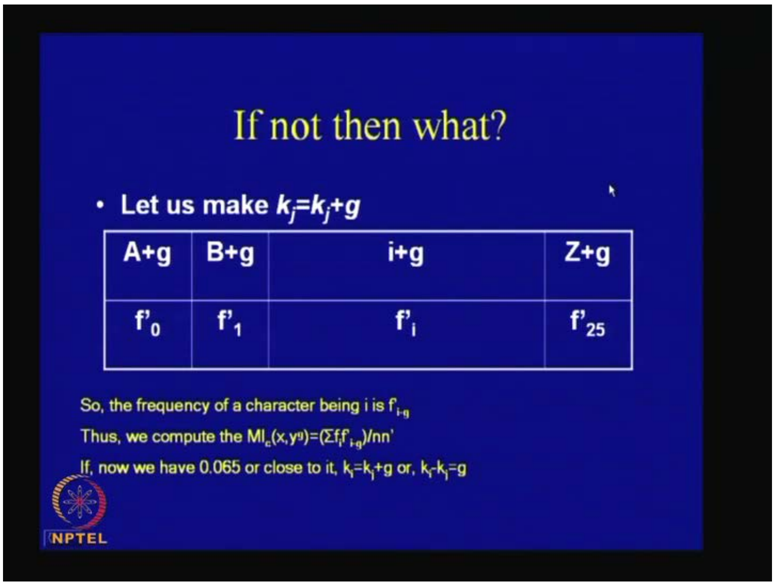


Mutual Index of Coincidence

- From the table we can see that is easy to see when k_i-k_j=0
- So, we can always fix a y_i and modify y_j (subtracting) from 1 to 25
- The value to which we get a MI_c close to 0.065 will indicate the correct k_i-k_j

k_i-k_j	MI_c
0	0.065
1	0.039
2	0.032
3	0.034
4	0.044
5	0.033
6	0.036
7	0.039
8	0.034
9	0.034
10	0.038
11	0.045
12	0.039
13	0.043

(Refer Slide Time: 39:58)



If not then what?

- Let us make $k_j = k_j + g$

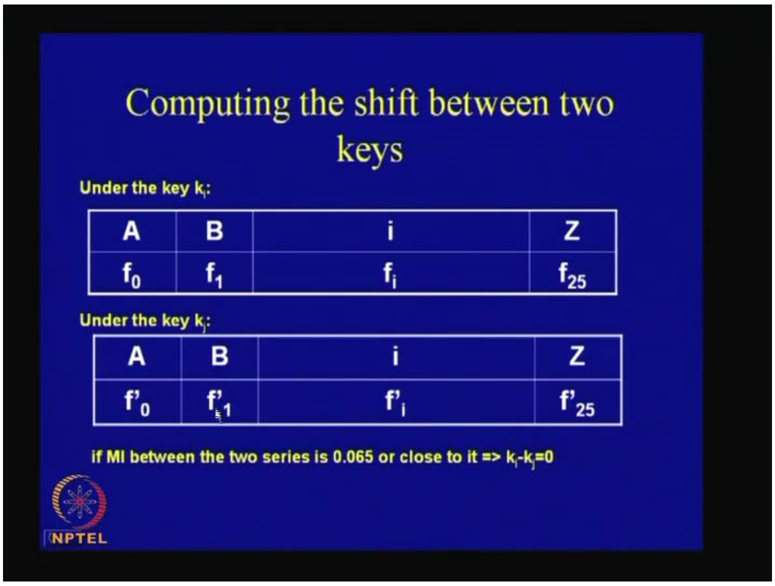
A+g	B+g	i+g	Z+g
f'_0	f'_1	f'_i	f'_{25}

So, the frequency of a character being i is f'_{i-g}

Thus, we compute the $MI_c(x, y^g) = (\Sigma f_i f'_{i-g})/nn'$

If, now we have 0.065 or close to it, $k_i = k_j + g$ or, $k_i - k_j = g$

(Refer Slide Time: 40:06)



Computing the shift between two keys

Under the key k_i:

A	B	i	Z
f_0	f_1	f_i	f_{25}

Under the key k_j:

A	B	i	Z
f'_0	f'_1	f'_i	f'_{25}

if MI between the two series is 0.065 or close to it => $k_i - k_j = 0$

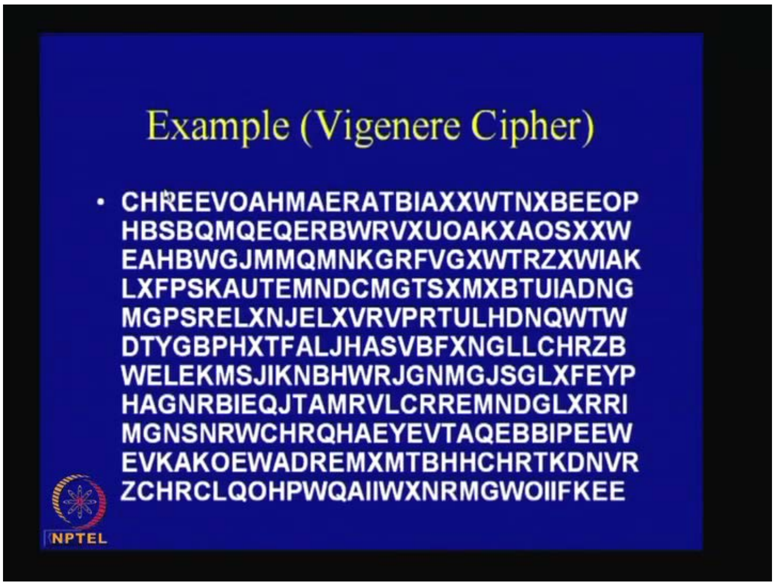
That is absorbed from this table that is if k i minus k j is 0, then this mutual of index of index of coincidence is 0.065. For what if not? If, if it is not equal to so, then what we do is that you shift this keep the first one same, that is, keep the one of the frequency same but you start shifting the next one.

So, you just start shifting them then we say that g like this, and therefore, all of these frequencies, therefore, the frequency of a character being i is now f dash i minus g. So, this because of the exactly of the same thing which I told you in context to the, in, I told you previously.

So therefore, the corresponding frequency is f dash i minus g and thus we compute the mutual index of coincidence of x and y g as sigma f i multiplied by with f i minus g dash divided by n n dash, and now, if we are got a value of 0.065 or close to it, then you can say that k i is equal to k j plus g. So, therefore, k i becomes equal to k j plus g, and from there you can actually compute the difference of k i minus k j being equal to g.

So therefore, you keep one of them constant and then keep on shifting the other values and you have also start computing the mutual index of coincidence, in this, in this fashion, and then, if this mutual index of coincidence matches 0.065, then you can say that this particular shift is actually the correct shift, and k i and k j are actually having a difference of this shift which is equal to g in this case.

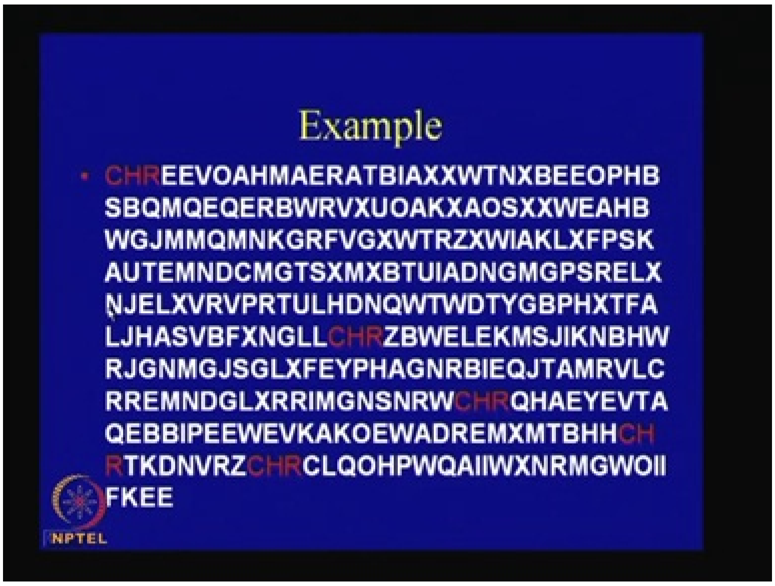
(Refer Slide Time: 41:23)



Example (Vigenere Cipher)

- CHREEVOAHMAERATBIAXXWTNXBEEOP
HBSBQMQEQERBWRVXUOAKXAOSXXW
EAHBWGJMMQMNKGRFVGXWTRZXWIAK
LXFPSKAUTEMNDCMGTSXMXBTUIADNG
MGPSRELXNJELXVRVPRTULHDNQWTW
DTYGBPHXTFALJHASVBFXNGLLCHRZB
WELEKMSJIKNBHWRJGNMGJSGLXFEYP
HAGNRBIEQJTAMRVLCRREMNDGLXRRI
MGNSNRWCHRQHAEYEVTAQEBBIPEEW
EVKAKOEWADREMXMTBHHCHRTKDNVR
ZCHRCLQOHPWQAIIWXNRMGWOIIFKEE

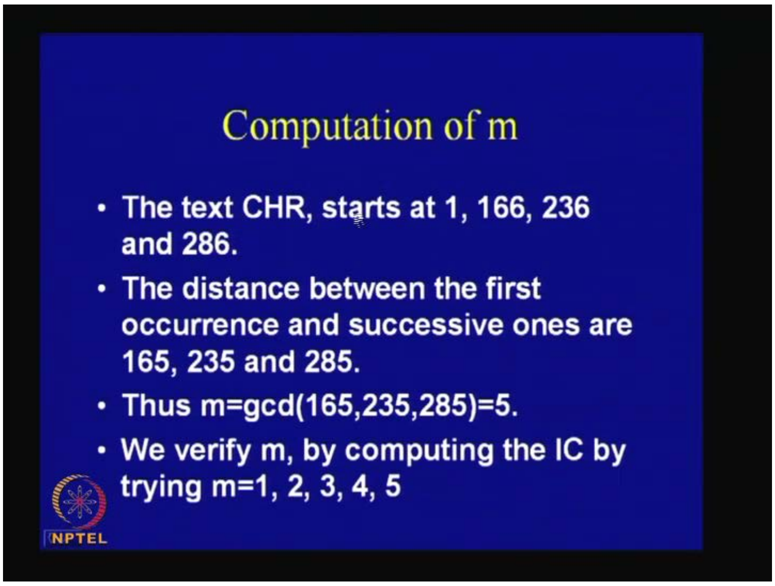
(Refer Slide Time: 41:29)



Example

- CHREEVOAHMAERATBIAXXWTNXBEEOPHB
SBQMQEQERBWRVXUOAKXAOSXXWEAHB
WGJMMQMNKGRFVGXWTRZXWIAKLXFPSK
AUTEMNDCMGTSXMXBTUIADNGMGPSRELX
NJELXVRVPRTULHDNQWTWDTYGBPHXTFA
LJHASVBFXNGLLCHRZBWELEKMSJIKNBHW
RJGNMGJSGLXFEYPHAGNRBIEQJTAMRVLC
RREMNDGLXRRIMGNSNRWCHRQHAEYEVTA
QEBBIPEEWEVKAKOEWADREMXMTBHHCH
RTKDNVRZCHRCLQOHPWQAIIWXNRMGWOII
FKEE

(Refer Slide Time: 41:38)



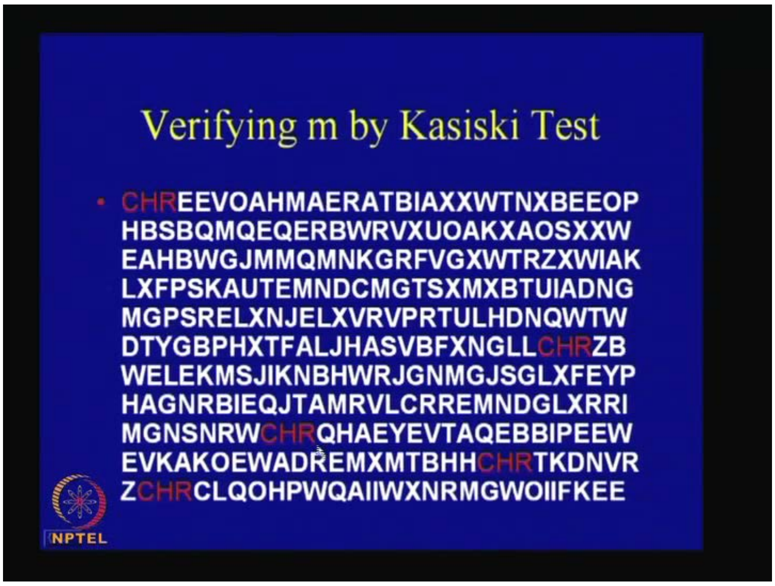
Computation of m

- The text CHR, starts at 1, 166, 236 and 286.
- The distance between the first occurrence and successive ones are 165, 235 and 285.
- Thus m=gcd(165,235,285)=5.
- We verify m, by computing the IC by trying m=1, 2, 3, 4, 5

So, I give you some examples to show how it works. So, this is an example to show of, this is an example of a ciphertext. The first important thing is to obtain the kind of common occurrences using that this c h r is a trigram, which occurs at certain distances; occurs kind of repeats, right? And we, we observed to the text c h r starts at 1, 166, 236 and 286 positions, and therefore, the distance that between the first occurrence and the successive ones are 165, 235 and 285.

So, if you know that if I take a g c d of these [dis/distances] distances, then it works out to five, and therefore, we verify m by computing the i c by trying m is equal to 1, 2, 3, 4 and 5, ok?

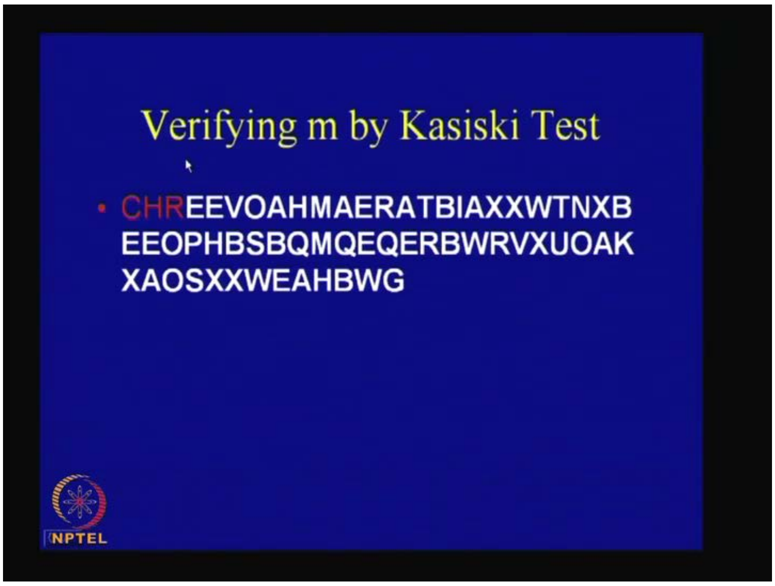
(Refer Slide Time: 42:03)



Verifying m by Kasiski Test

- CHREEVOAHMAERATBIAXXWTNXBEEOP
  HBSBQMQEQERBWRVXUOAKXAOSXXW
  EAHBWGJMMQMNKGRFVGXWTRZXWIAK
  LXFPSKAUTEMNDCMGTSXMXBTUIADNG
  MGPSRELXNJELXVRVPRTULHDNQWTW
  DTYGBPHXTFALJHASVBFXNGLLCHRZB
  WELEKMSJIKNBHWRJGNMGJSGLXFEYP
  HAGNRBIEQJTAMRVLCRREMNDGLXRRI
  MGNSNRWCHRQHAEYEVTAQEBBIPEEW
  EVKAKOEWADREMXMTBHHCHRTKDNVR
  ZCHRCLQOHPWQAIIWXNRMGWOIIFKEE

(Refer Slide Time: 42:09)

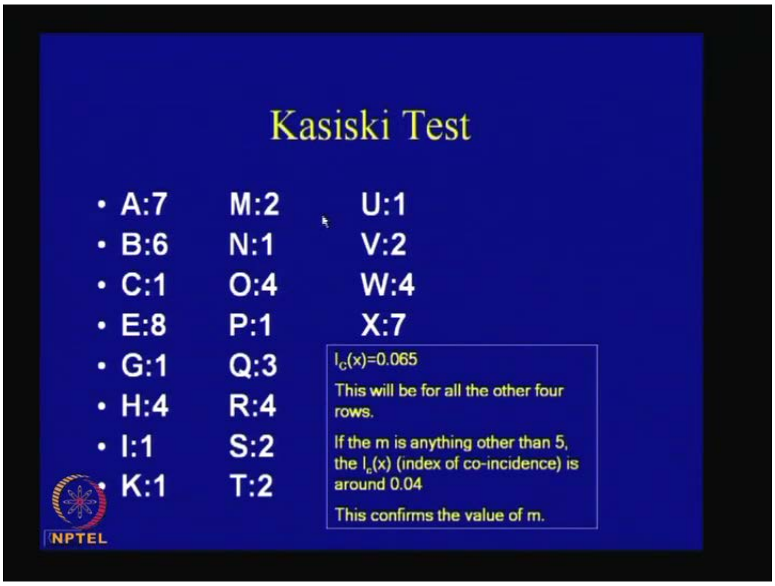


Verifying m by Kasiski Test

- CHREEVOAHMAERATBIAXXWTNXB
  EEOPHBSBQMQEQERBWRVXUOAK
  XAOSXXWEAHBWG

So, so, we will like to verify this m by, the, the index of coincidence test. So, what we do first of all is that we kind of take one of these rows, therefore, this, this is the one of the rows, we have actually divided them and we have formed. So, what we do is that we start forming five rows and the first row is given as this; this is a first row.

(Refer Slide Time: 42:27)



Kasiski Test

- A:7
- B:6
- C:1
- E:8
- G:1
- H:4
- I:1
- K:1

M:2
N:1
O:4
P:1
Q:3
R:4
S:2
T:2

U:1
V:2
W:4
X:7

$I_c(x)=0.065$

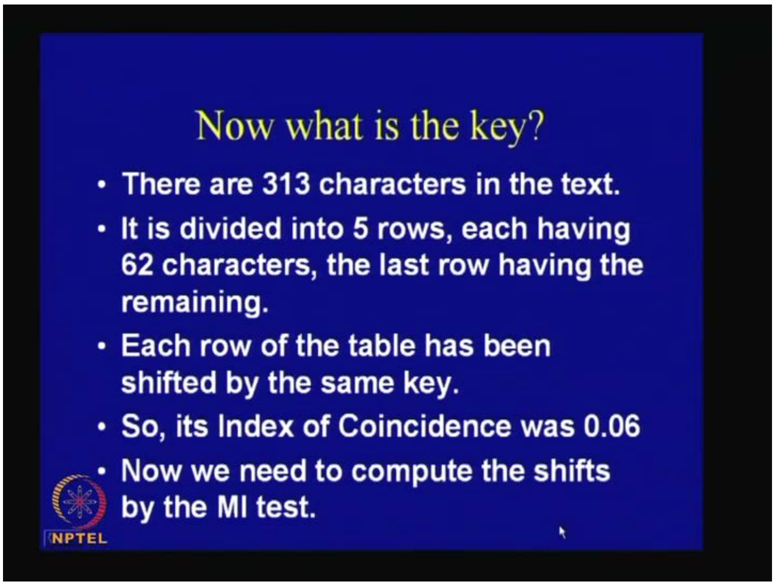
This will be for all the other four rows.

If the m is anything other than 5, the $I_c(x)$ (index of co-incidence) is around 0.04

This confirms the value of m.

So, what we do is that we actually from, I mean a index of coincidence so we, [ob/obtain] we again obtain the corresponding frequencies, and after obtaining the frequencies, we obtain the i c value. So, if you obtain this i c value, this value comes on 0.065, which can and this actually is holds for the other four rows also. So, therefore, if the m is anything other than five, then as we have discuss the i c x would have been around 0.04, but since you are getting a 0.065 value, then the value of m equal to five is actually conformed by this text.

(Refer Slide Time: 43:00)

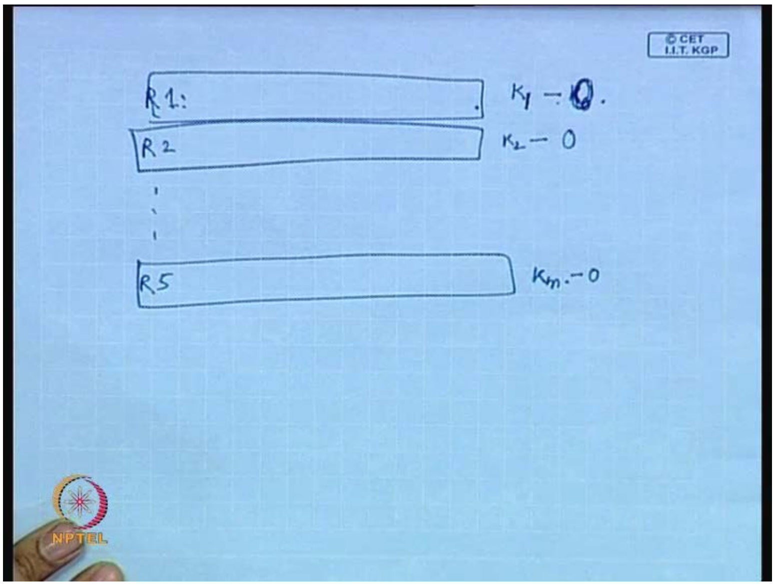


Now what is the key?

- There are 313 characters in the text.
- It is divided into 5 rows, each having 62 characters, the last row having the remaining.
- Each row of the table has been shifted by the same key.
- So, its Index of Coincidence was 0.06
- Now we need to compute the shifts by the MI test.

So, next thing is to obtain the key. So, how do you obtain the key? Now, there are 313 characters in the text, it is divided into five rows because five is the length of the key each having 62characters; the last row having the remaining. Now, each row of the table has been shifted as we have discussed by the same key. So, its index of coincidence was 0.06. So, we are have actually observed that. Now, we need to obtain or rather compute the shifts by the mutual index of coincidence text.

(Refer Slide Time: 43:53)

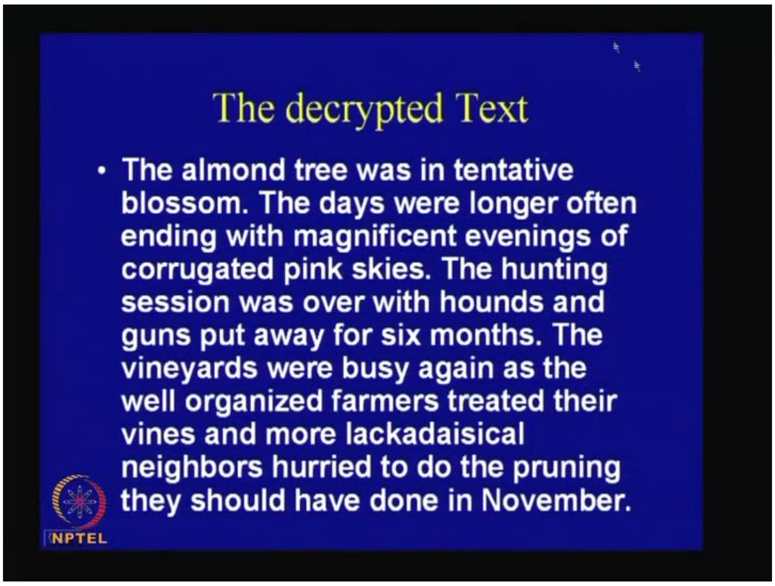


So, what we do is that we are actually form each of these rows, and for each of these rows, we actually assume that an English language text would have been shifted by a zeros; in that case, k i was equal to 0, and we, we try to find out for a every character or other for every row, what is the shift? For every row, what is the shift? So, if you have got five rows, so, if you have got row 1 row 2 and so on till row 5, then you assume that this row 1 is shifted by the first letter in the key which is k 1. The second one is been is been shifted by k 2 and so on this one is been shifted by k m, right?

So, the first objective is to find out what is k 1 minus k 2. So, what is the value of k 1 minus k 0? So, what is k 0 in this case? k 0 is 0 because that corresponds to the normal English language text. So similarly, we find out k 2 minus 0 and so on till k n minus 0 to obtain the corresponding values of the key.

So therefore, we take, we have obtained the frequency distributions here. We know the frequency distribution in context of the normal English language. So, we if kind of compute the mutual index of coincidence between these string and an normal English language string.

(Refer Slide Time: 45:52)

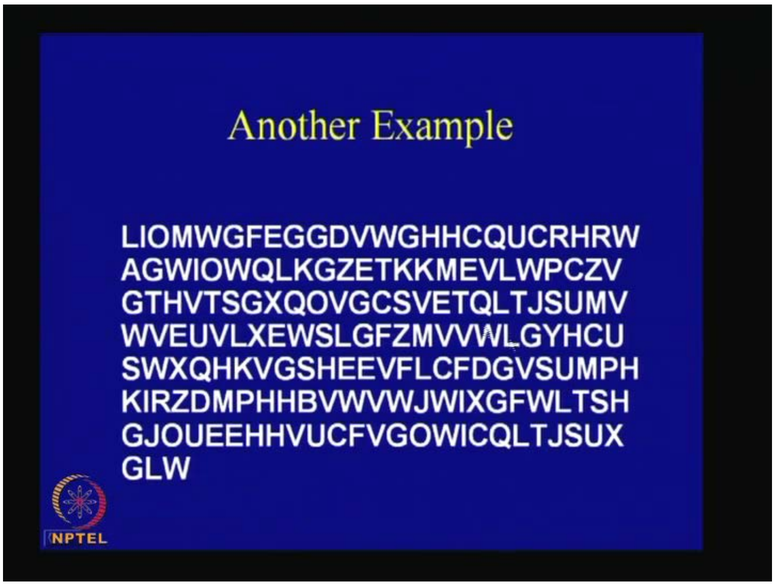


The decrypted Text

- The almond tree was in tentative blossom. The days were longer often ending with magnificent evenings of corrugated pink skies. The hunting session was over with hounds and guns put away for six months. The vineyards were busy again as the well organized farmers treated their vines and more lackadaisical neighbors hurried to do the pruning they should have done in November.

So that kind of gives out the estimate of k i minus 0 k one minus 0. So, we can actually automate these process, and using that automation, we are actually kind of we can actually obtain the key and decrypt it.

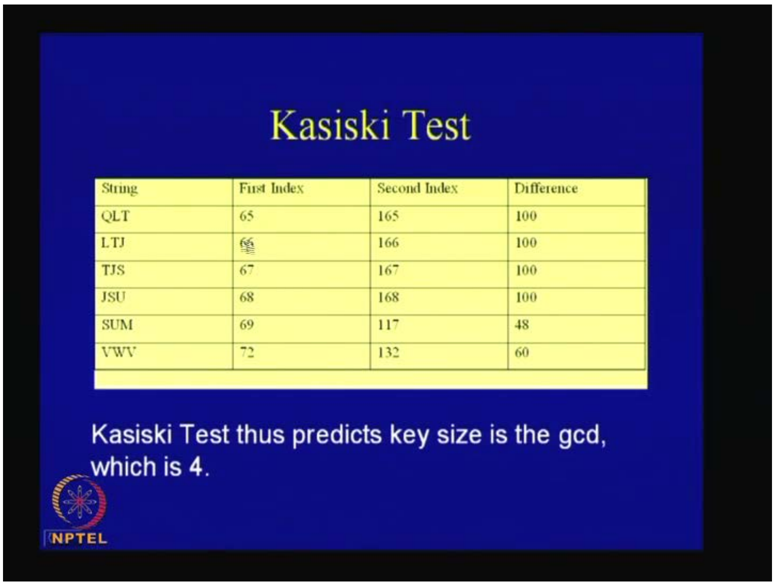
(Refer Slide Time: 44:57)



Another Example

LIOMWGFEGGDVWGHHCQUCRHRW
AGWIOWQLKGZETKKMEVLWPCZV
GTHVTSGXQOVGCSVETQLTJSUMV
WVEUVLXEWSLGFZMVVWLGYHCU
SWXQHKVGSHEEVFLCFDGVSUMPH
KIRZDMPHHBVWVWJWIXGFWLTSH
GJOUEEHHVUCFVGOWICQLTJSUX
GLW

(Refer Slide Time: 45:02)



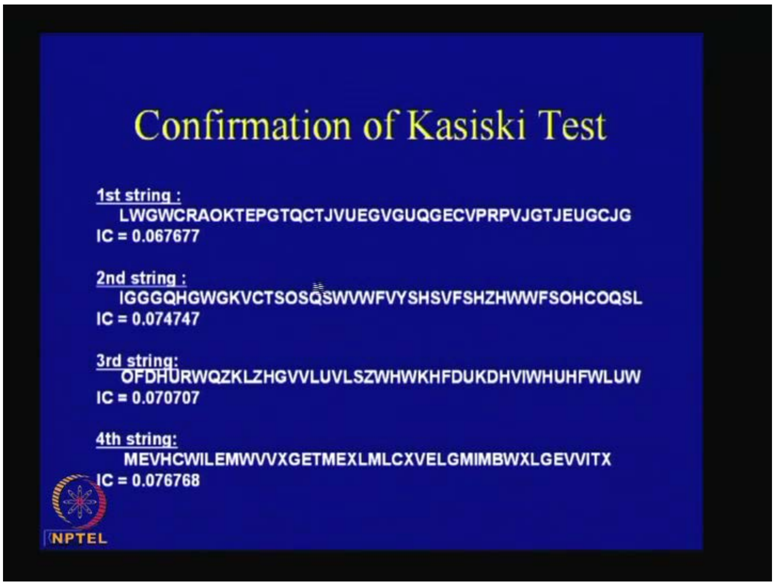
Kasiski Test

String	First Index	Second Index	Difference
QLT	65	165	100
LTJ	66	166	100
TJS	67	167	100
JSU	68	168	100
SUM	69	117	48
VWV	72	132	60

Kasiski Test thus predicts key size is the gcd, which is 4.

So, I will give you another example. So, this is an example of another ciphertext. In this case also you see that, we have, we have obtained the common video according strings and this is the first index and the second index the difference is denoted here, and therefore, the kasiski test thus [predi/predict] predict that the key size is the g c d which is in this case 4; we take the g c d these strings we get 4, ok?

(Refer Slide Time: 45:18)



Confirmation of Kasiski Test

1st string :
LWGWCRAOKTEPGTQCTJVUEGVGUQGECVPRPVJGTJEUGCJG
IC = 0.067677

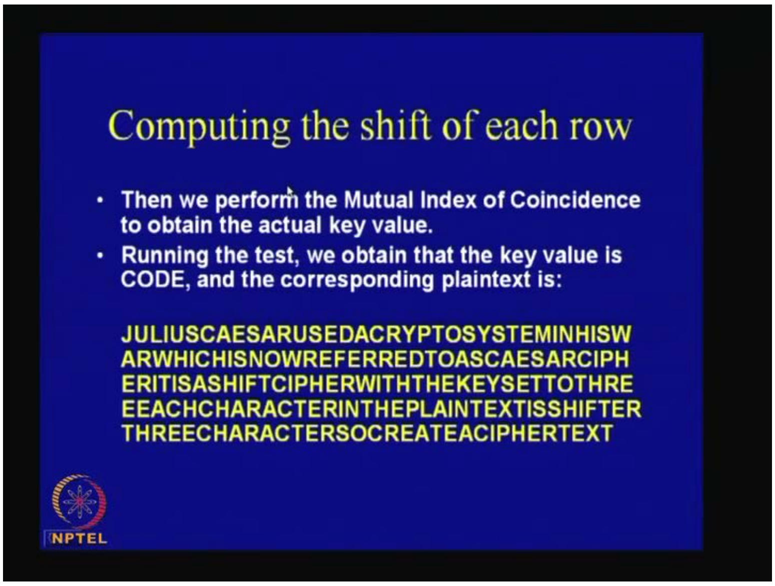
2nd string :
IGGGQHGWGKVCTSOSQSWVWFVYSHSVFSHZHWWFSOHCOQSL
IC = 0.074747

3rd string:
OFDHURWQZKLZHGVVLUVLSZWHWKHFDUKDHVIWHUHFWLUW
IC = 0.070707

4th string:
MEVHCWILEMWVVXGETMEXLMLCXVELGMIMBWXLGEVVITX
IC = 0.076768

So, we will confirm these strings. So therefore, we again kind of break it into four rows in this case. So, the first string is this; second string is this; third string is this and fourth string is this. So, we actually obtain quite high values more than 0.06, and therefore, the size of the key is kind of confirm is confirmed, ok?

(Refer Slide Time: 45:36)

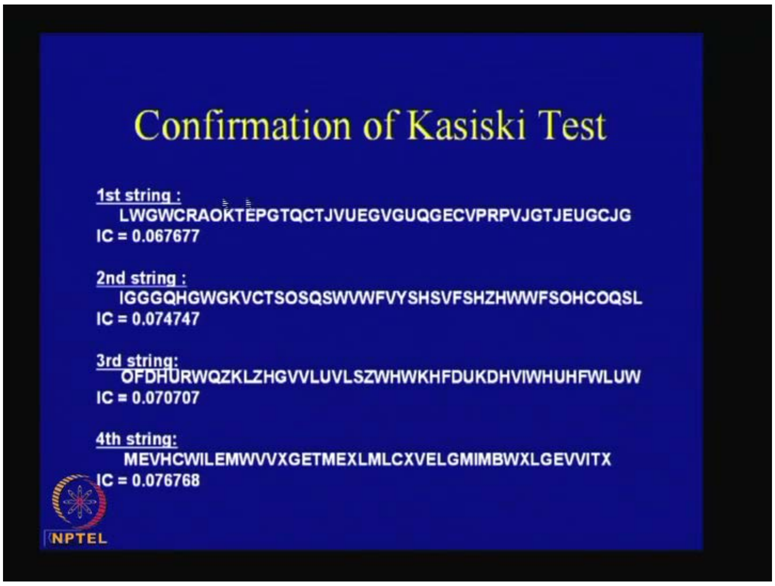


Computing the shift of each row

- Then we perform the Mutual Index of Coincidence to obtain the actual key value.
- Running the test, we obtain that the key value is CODE, and the corresponding plaintext is:

JULIUSCAESARUSEDACRYPTOSYSTEMINHISW
ARWHICHISNOWREFERREDTOASCAESARCIPH
ERITISASHIFTCIPHERWITHTHEKEYSETTOTHRE
EEACHCHARACTERINTHEPLAINTEXTISSHIFTER
THREECHARACTERSOCREATEACIPHERTEXT

(Refer Slide Time: 45:49)



Confirmation of Kasiski Test

1st string :
LWGWCRAOKTEPGTQCTJVUEGVGUQGECVPRPVJGTJEUGCJG
IC = 0.067677

2nd string :
IGGGQHGWGKVCTSOSQSWVWFVYSHSVFSHZHWWFSOHCOQSL
IC = 0.074747

3rd string:
OFDHURWQZKLZHGVVLUVLSZWHWKHFDUKDHVIWHUHFWLUW
IC = 0.070707

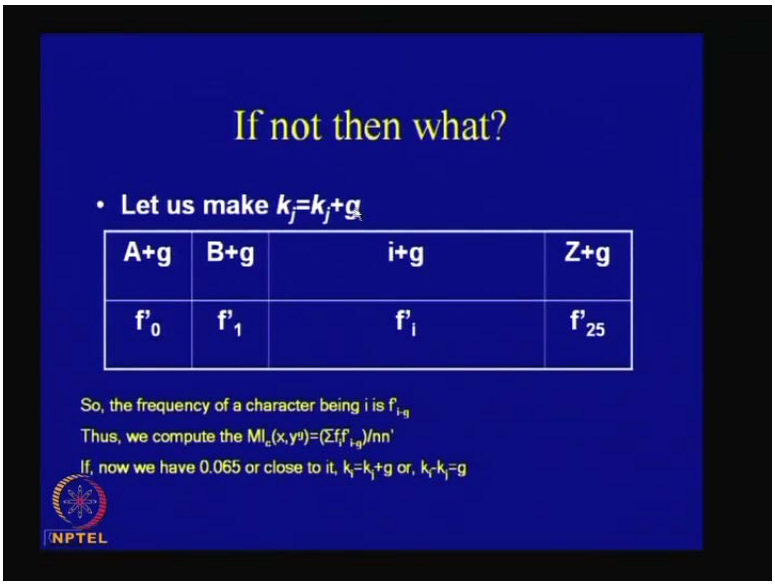
4th string:
MEVHCWILEMWVVXGETMEXLMLCXVELGMIMBWXLGEVVITX
IC = 0.076768

So then, we would need to compute the shift of each row. So, what we do is that we perform the mutual index of coincidence to obtain the actual key value. So, we run this test, that is, we find out we have got the English language. So, we have got the I mean corresponding we obtained the corresponding frequency distribution of these string, and we also know the frequency distribution which is the applied frequency distribution in

English language. We use these two frequencies to compute the mutual index of coincidence of this string, and I am actually and the corresponding value for which actually we get 0.065 is actually the correct string, ok? So therefore, we actually obtain the, is the actual value of the key, that is, we obtain the shift.

So, what we do is that we assumed that the shift in this case is 0; the next we assume that the shift in this case is 1, then and so on till shift is 25. Whichever shift actually gives us the mutual index of value to be 0.065 is the correct shift and that we do exactly as this that is like this.

(Refer Slide Time: 46:40)



If not then what?

- Let us make k_j=k_j+g

A+g	B+g	i+g	Z+g
f'_0	f'_1	f'_i	f'_{25}

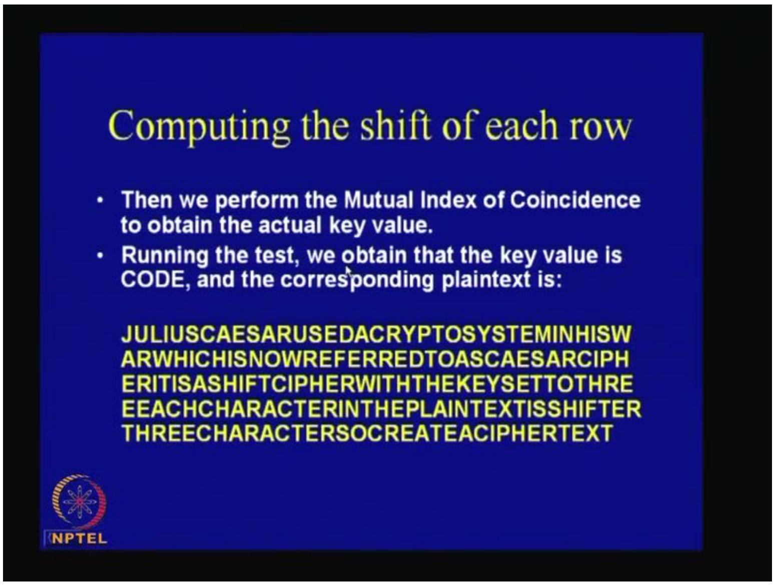
So, the frequency of a character being i is f'_{i-g}

Thus, we compute the $MI_c(x,y^g)=(\Sigma f_i f'_{i-g})/nn'$

If, now we have 0.065 or close to it, k_i=k_j+g or, k_i-k_j=g

So, if, if this not then, we actually start warring this g from 0 25; from 1 to 25 and whichever value actually gives the value of 0.065 is the correct test in it. So, that is the way of obtaining the corresponding key bit or the key value key alphabet, right?

(Refer Slide Time: 47:03)



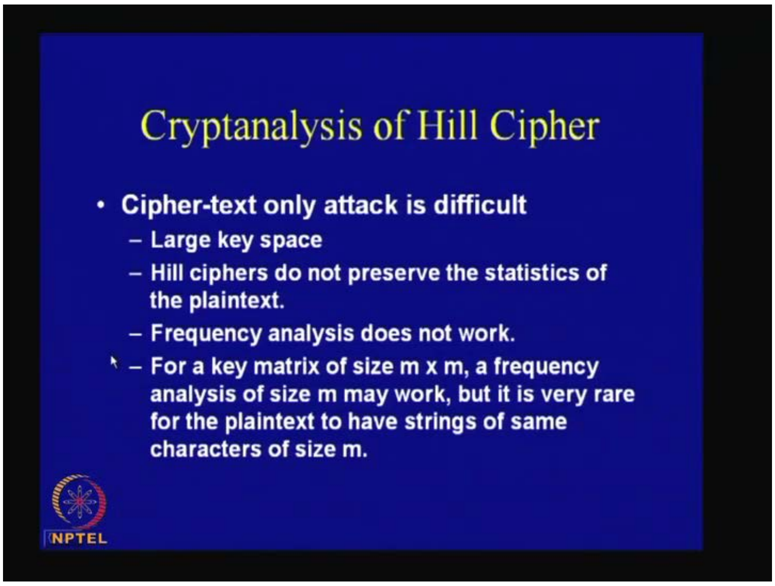
Computing the shift of each row

- Then we perform the Mutual Index of Coincidence to obtain the actual key value.
- Running the test, we obtain that the key value is CODE, and the corresponding plaintext is:

JULIUSCAESARUSEDACRYPTOSYSTEMINHISW
ARWHICHISNOWREFERREDTOASCAESARCIPH
ERITISASHIFTCIPHERWITHTHEKEYSETTOTHRE
EEACHCHARACTERINTHEPLAINTEXTISSHIFTER
THREECHARACTERSOCREATEACIPHERTEXT

So, in this case, we perform this and the corresponding if you run this test, that is actually automated this and we can obtain this the key values in this case code and the corresponding plaintext is this and it makes many meaningful is a meaningful test, and therefore, we kind of conclude that our decryption has been correct.

(Refer Slide Time: 47:18)



Cryptanalysis of Hill Cipher

- Cipher-text only attack is difficult
  - Large key space
  - Hill ciphers do not preserve the statistics of the plaintext.
  - Frequency analysis does not work.
  - For a key matrix of size m x m, a frequency analysis of size m may work, but it is very rare for the plaintext to have strings of same characters of size m.

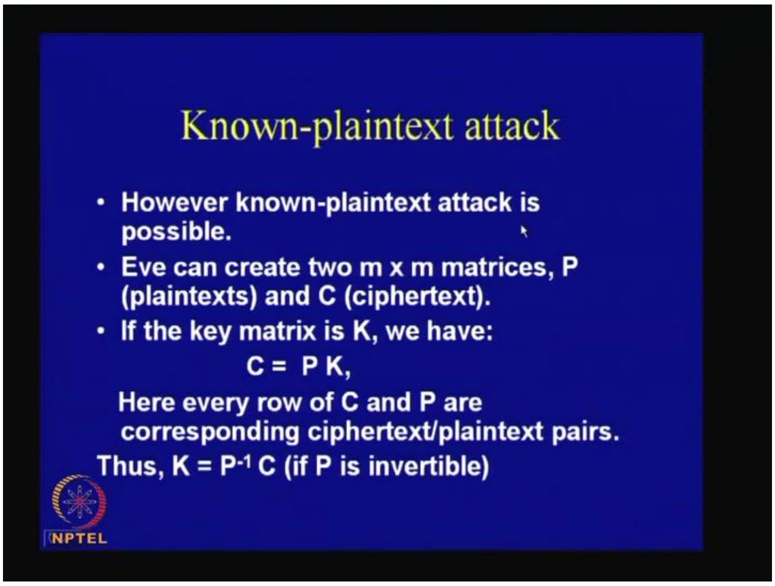
So then, we actually discuss and conclude with the cryptanalysis of hill cipher. So, in this case, the cipher-text only attack is difficult because there is a large key space. The key space is actually for m cross m matrix it would be and it can be as high as 26 to the power of m square but actually it will be not so high, because all the matrixes are not invertible, right? but in a hill cipher, it is important that the matrix also has to be invertible.

So the hill ciphers but there is the point is that where the important point to be stressed, that is, hill ciphers not preserve the statistics of the plaintext, and therefore, frequency analysis does not work. Now for a key matrix of size m cross m a frequency analysis of size m may work but it is very rare for the plaintext to have strings of same character; so, i mean it may work for a size of m but it is very rare for a plaintext to have strings of same characters of size m because m is quite large.

See for example, t h e can occur again frequently see a but the probability of a kind of trigram to repeat is kind of more than a, an, than a kind of a string which has got a ten letters, which has to repeat the probability kind of reduces.

So therefore, if you have you got a letter, I mean if the value of m is suppose something like fifty, then a that the probability in your text is particular string of length 50 will repeat is quite small. Therefore, you will get very small sample to work with and since this is statistical technique this may not work.

(Refer Slide Time: 48:55)



Known-plaintext attack

- However known-plaintext attack is possible.
- Eve can create two m x m matrices, P (plaintexts) and C (ciphertext).
- If the key matrix is K, we have:

  C = P K,

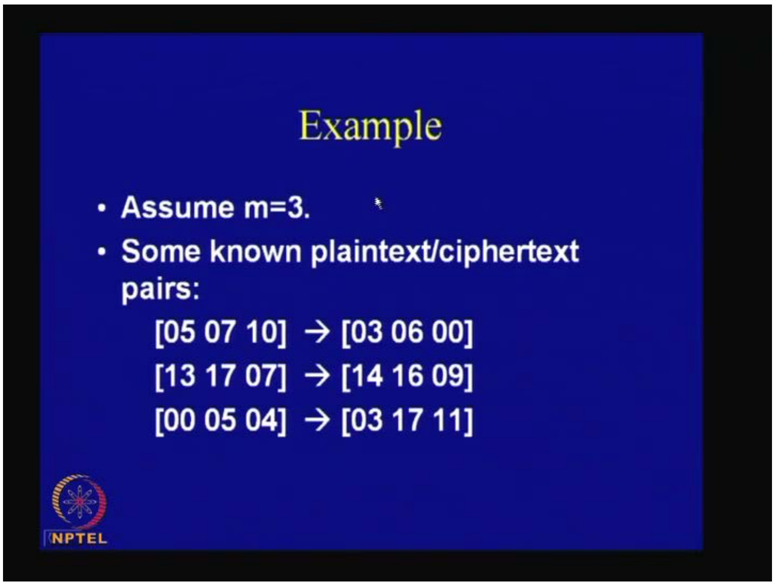
  Here every row of C and P are corresponding ciphertext/plaintext pairs.

Thus, $K = P^{-1} C$ (if P is invertible)

So therefore, I mean a plains and simple analysis a plain and simple cipher-text only attack like what we have seen previously may not work; however, a known plain text attack can easily work like.

So, in this case, you see that e has you can actually create 2 m cross m matrix matrices 1 for the plaintext and 1 for the ciphertext and this, and if the key matrix is k is, is, denoted by k, and then. you can actually represent, it, it by c is equal to p into k, and here, every row of c and p are corresponding ciphertext and plaintext pairs, and therefore, you can actually obtain the k if this plaintext in matrix is invertible as simply as multiplying p inverse with c.

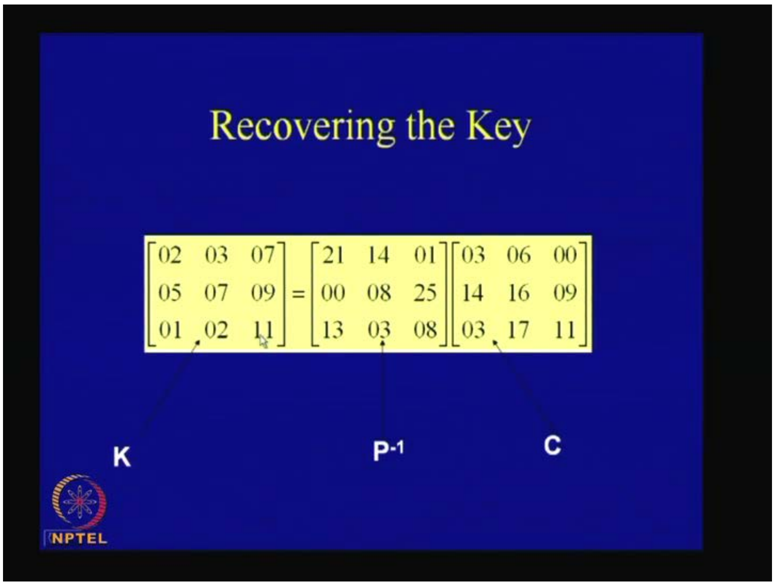
(Refer Slide Time: 49:27)



Example

- Assume m=3.
- Some known plaintext/ciphertext pairs:
  - [05 07 10] → [03 06 00]
  - [13 17 07] → [14 16 09]
  - [00 05 04] → [03 17 11]

(Refer Slide Time: 49:52)

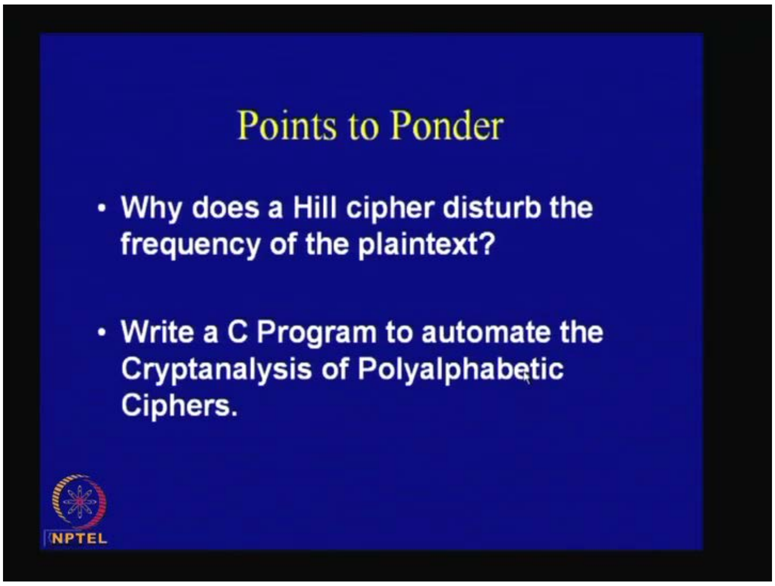


Recovering the Key

$$\underbrace{\begin{bmatrix} 02 & 03 & 07 \\ 05 & 07 & 09 \\ 01 & 02 & 11 \end{bmatrix}}_{K} = \underbrace{\begin{bmatrix} 21 & 14 & 01 \\ 00 & 08 & 25 \\ 13 & 03 & 08 \end{bmatrix}}_{P^{-1}} \underbrace{\begin{bmatrix} 03 & 06 & 00 \\ 14 & 16 & 09 \\ 03 & 17 & 11 \end{bmatrix}}_{C}$$

So here, here we have an example. We says that assume that m is equal three and some known plaintext ciphertext pairs are given here, like suppose 0 5 0 7 1 0 is getting mapped into 0 3 0 6 0 0 1 3 1 7 0 7 is getting mapped to 1 4 1 6 0 9 0 0 0 5, and 0 4 is getting mapped into 0 3 1 7 and 1 1. So, you can actually from two [pl/plus] matrices plaintext and ciphertext matrixes and use them to obtain the inverse of the matrix p and actually in this case p is luckily invertible. If p is not invertible, then you have to actually obtain more plaintext ciphertext spheres, and find out which one is invertible, and from there, you can actually obtain the corresponding key k by multiplying p inverse with c. So, this quite straight forward and can be done and the other. So, essentially this gives us technique of doing a or mounting a known plaintext attack on the finite hill cipher.

(Refer Slide Time: 50:27)



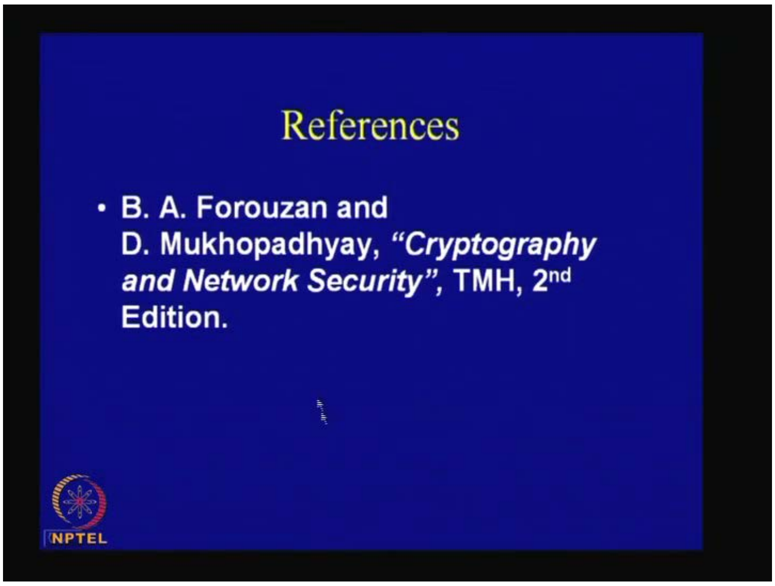
Points to Ponder

- Why does a Hill cipher disturb the frequency of the plaintext?
- Write a C Program to automate the Cryptanalysis of Polyalphabetic Ciphers.

I will give you give a certain points to think on like, why does a hill cipher at all disturb the frequency of the plaintext? So, you can take an hill cipher of size m and you can tried to kind of find out why essentially if the frequency of the plaintext is disturbed. The other important thing which you can tried to do is that we can write a c program to automate the cryptanalysis of polyalphabetic ciphers and you can try to play around with various cipher, like you can take a a normal English text and you can kind of encrypt using vigenere cipher; a polyalphabetic cipher choose some key values and obtain the ciphertext, and then, you give fit or rather feel to your program and see whether you can, you are able to retrieve the key. If you are able to retrieve the size of the key, the actual key, and from there, you are actually able to decipher the ciphertext and, this, the interesting thing of kasiski of this test is that it can be automated.

So therefore, you can actually write a nice program and play around and experiment with them.

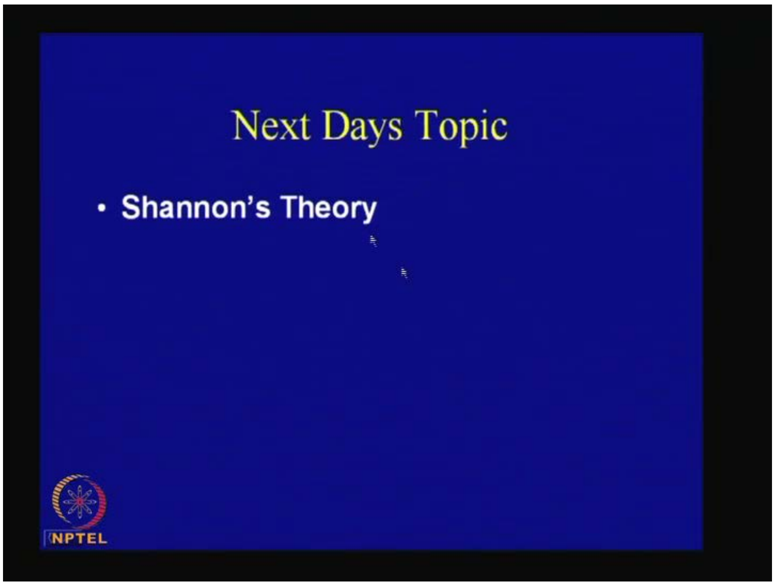
(Refer Slide Time: 51:23)



References

- B. A. Forouzan and D. Mukhopadhyay, *“Cryptography and Network Security”*, TMH, 2nd Edition.

(Refer Slide Time: 51:31)



Next Days Topic

- Shannon’s Theory

So, the references that I have followed is cryptography and network security the second edition of the book and by forouzan and myself, and, and next day, we shall actually [con/continue] continue with shannon’s theory. .