Colossus and the Breaking of the Lorenz Cipher

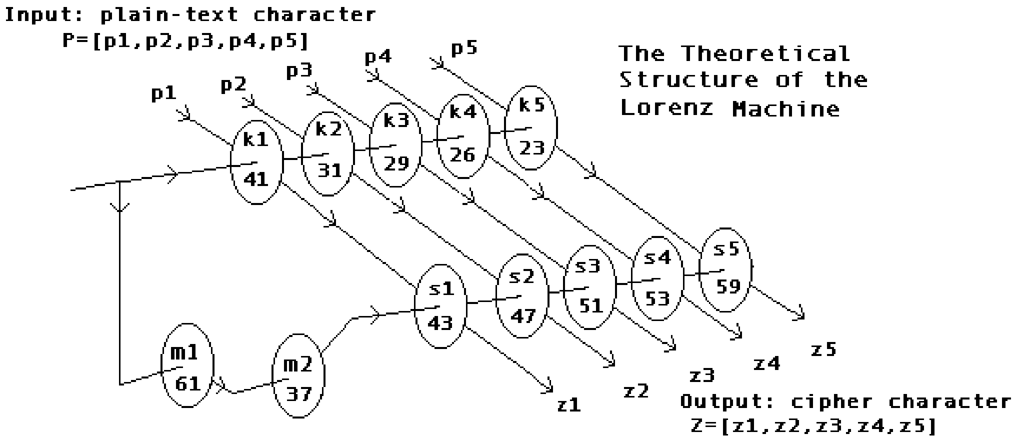
Introduction:

The breaking of the Lorenz cipher at Bletchley Park was a remarkable synergy of mathematics and engineering that had a significant effect on the course of WWII. Several descriptive accounts of this great achievement have appeared in various publications, on TV and also more recently on the Internet. The aim of this account is to focus on some of the mathematics that formed the basis for this success.

The Lorenz cipher machine was intended solely for the use of the German High Command for military communications at the highest level. The system was based on the use of the standard international tele-printer code, in which each plain-text character was converted into a group of five electrical impulses made up of marks (x) and spaces (•), that to-day would be represented by five digit binary numbers. For example the letter 'S' represented by:- x • x • •, becomes 1 0 1 0 0 in modern notation.

The Logical structure of the Lorenz Machine

The function of the Lorenz machine was to transform each character of plaintext into a cipher character, in a way that made it appear that they had no discernable relationship with each other. The machine processed each of the five component 'impulses' of the plain-text characters by means of two sets of five cipher wheels. These wheels were coupled together by a complex set of gears, and each had a number of adjustable tabs on its circumference, with some of them being be set to their 'active' state. When in this state a tab had the effect of changing the incoming impulse (i.e. from 'x' to '•', and vice versa). At the positions on the circumferences where the tabs were in their 'non-active' state, the incoming impulses were not changed. As the wheels moved each tab in turn interacted with the corresponding incoming impulse.
The diagram gives the basic layout of the machine, and shows the number of adjustable tabs on each wheel.



The first set of five wheels was known as the 'K-wheels' and the second set as the 'S-wheels', (originally the Greek letters χ and ψ were used).
The K-wheels stepped on by one position after the input of each plain-text

character, but the motion of the S-wheels was more complex, and sometimes this set did not step on, being under the control of two additional so called ‘motor’ wheels (at a later stage the Germans introduced some additional complications to the control of the movement of the S-wheels that were referred to as ‘*limitations*’). The joint effect of the two sets of wheels was to generate for each character of plain-text, the five binary bits of a pseudo random ‘key’ character which was then combined, with the plain-text character by a process of ‘addition’, to create the corresponding cipher character. The action of the machine can be represented by simple algebra:-
Let a plain-text character be represented by P, the pseudo random key character by C, and the cipher character by Z, then:- Z = P + C.

The process of “addition” was carried out on the five pairs of binary bits using the following rules:- • + • = •, • + x = x, x + • = x, and x + x = • , These correspond to the rules of addition in modulo 2 arithmetic:-
0 + 0 = 0, 0 + 1 = 1, 1 + 0 = 1, and 1 + 1 = 0

Two examples:- Plain-text ‘H’ = • • x • x   Cipher ‘I ’ = • x x • •
key = ‘L’ = • x • • x   key ‘L’= • x • • x
Cipher = • x x • • = ‘I ’   • • x • x = ‘H’

These examples illustrate the property of the addition process that was fundamental to the function of the Lorenz machine, that can be expressed in the general form:- If Z = P + C then Z + C = P.
This implies that C + C = ‘null’ (the additive identity element), which is the character ‘/ ’ (• • • • •). The reader may care to verify that the addition of any character to itself always results in the character ‘/ ’.

Table of a part of the tele-printer code

A	xx•••	B	x••xx	C	•xxx•	D	x••x•	E	x••••	F	x•xx•
G	•x•xx	H	••x•x	I	•xx••	J	xx•x•	K	xxxx•	L	•x••x
M	••xxx	N	••xx•	O	•••xx	P	•xx•x	Q	xxx•x	R	•x•x•
S	x•x••	T	••••x	U	xxx••	V	•xxxx	W	xx••x	X	x•xxx
Y	x•x•x	Z	x•••x	3	•••x•	4	•x•••	8	xxxxx	+	xx•xx
9	••x••	/	•••••								

(3 = ‘carriage return’, 4 = ‘line feed’, 8 = ‘letter shift’, + = ‘figure shift’, 9 = ‘space’, / = ‘null’.)

Note: At BP all the tele-printer control characters were suppressed and replaced by the symbols shown above, and then printed as normal characters. This was done to avoid the difficulties that would otherwise have arisen as a consequence of the control characters appearing at random positions in the intercepted cipher messages.

In reality both the K and S sets of wheels generated their own five bit pseudo random character, and so, with an extension of the notation, the key character C = K + S’, so that the basic cipher equation for the machine can be expressed in the form:- Z = P + K + S’ (The symbol S’ represents the combined effect of the S-wheels and their stepping motion. The distinction between S and S’ will be made clearer later).
The operating principle of the Lorenz system can now be expressed in

another way using the equation $Z = P + K + S'$. After adding the composite key character $K + S'$ to both sides (modulo 2), the equation is transformed to the alternative form:- $Z + K + S' = P$. At the transmitting station the sequence of plain-text characters were converted into cipher characters by the machine, according to the first equation given above, and at the receiving station each cipher character was converted back into plain-text by adding to it the same composite key character that had been used during the process of encipherment as shown by the alternative form of the equation. In order to bring this about, it was essential that all twelve wheels in both of the Lorenz machines involved had been adjusted to the same set of starting positions.

At BP the secrets of the design of the Lorenz machine were deduced in a most remarkable way by a combination of skilful cryptology and some brilliant mathematics. Subsequently a simulator of the Lorenz machine (known as ‘Tunny’) was constructed. This had the same logical properties as the German machine, although its physical form was completely different. However before Tunny could be used to decipher an intercepted Lorenz message, the correct wheel starting positions (settings) used for the message had to be determined.

Finding the wheel settings:

A major difficulty in finding the correct wheel settings can be illustrated by considering how the task might to be carried out by a process of ‘trial and error’. From the diagram of the machine it can be seen that the number of possible wheel settings is:-

$$41 \times 31 \times 29 \times 26 \times 23 \times 43 \times 47 \times 51 \times 53 \times 59 \times 61 \times 37 \quad (= 1.6 \times 10^{19})$$

If it were possible to test all of them at a rate say of 1000 per second, then the total time taken would be about 500 million years!

A more realistic approach to the problem was to break the task down into a number of less demanding procedures by reducing the number of wheel settings that had to be considered at the same time.

A significant discovery:

From an examination of the few deciphered messages that had been obtained over a period of time by hand methods, it was discovered that the plain-text often contained more pairs of repeated characters than would be expected to occur by chance. One reason for this was that the German operators often repeated the control characters, to make sure that they were not lost during transmission, as such a loss would cause a sequence of errors in the following part of the message. For example it is quite likely that the following (dummy) message:- LUFTWAFFE FLTGR (ROEM XVI) would have been transmitted as:- 88LUFTWAFFE9FLTGR9++K88ROEM9XVI++L88

The likely presence of pairs of repeated characters in the messages was used as a basis for finding some of the wheel settings, but first it was necessary to devise a simple procedure that made it possible to detect these repeats automatically by means of a machine.

The “Delta” process:

Consider part of the message given above, together with the same part printed again under it, but with the letters displaced one to the right:-

```
9 F L T G R 9 + + K 8 8 R O E M 9 X V I + + L 8 8
  9 F L T G R 9 + + K 8 8 R O E M  9 XV I + + L 8 8
```

The vertical pairs of characters can be combined or ‘added’ using the rules described earlier, as illustrated by the following examples:-

$$\begin{array}{llcll} F = & x \bullet x\ x \bullet & \quad & + = & x\ x \bullet x\ x \\ 9 = & \underline{\bullet \bullet x \bullet \bullet} & & + = & \underline{x\ x \bullet x\ x} \\ & \underline{\underline{x \bullet \bullet x \bullet}} \text{ (= D)} & & & \underline{\underline{\bullet \bullet \bullet \bullet \bullet}} \text{ (= /)} \end{array}$$

If the original sequence of message characters is represented by the symbol ***P***, then the result of the ‘summation’ formed was called ‘delta ***P’*** and written as Δ***P***.

It follows that if ***P*** = 9 F L T G R 9 + + K 8 8 R O E M 9 X V I + + L 8 8

then Δ***P*** = D 8 4 R T C 8 / H T / Y L B X O B A O X / D F /

Note that for every repeated character in the plain-text sequence ***P***, there is a ‘/’ character in the corresponding sequence Δ***P***. The character ‘/’ is represented in the tele-printer code by ‘• • • • •’, which means that every repeated character in the plain-text sequence ***P*** will lead to a dot in each of the five irregular sequences of dots and crosses formed by the characters in Δ***P***. The discovery of this characteristic of Δ***P*** was of great importance. Subsequent developments were based mainly on the ‘delta’ characters ΔZ, ΔP, etc. and not on the original characters Z, P etc.

Some important equations:

The basic cipher equation is:- Z = P + K + S’. By adding K to both sides (modulo 2) this becomes:- Z + K = P + S’. It will be useful to introduce the additional symbolic term D = Z + Ḳ so that D = P + Ṣ’

The following corresponding ‘delta’ equations are also true:-

i.e.. ΔD = ΔZ + ΔK and ΔD = ΔP + ΔS’

So far all the symbols used (Z, P, D K etc), have represented complete characters, but as the aim was to reduce as far as possible the number of wheels that have to be considered simultaneously, it was necessary to work instead with the component binary bits from which the complete cipher characters were composed (the sequences of these components formed the five ‘bit-streams’).

Bit-stream equations:

Sets of equations identical in form to those for complete characters (with the addition of appropriate suffixes), can be used to describe the sequences of individual binary bits, so that for the 1st bit-stream:-

$Z_1 = P_1 + K_1 + S'_1$ and for the 2nd bit-stream $Z_2 = P_2 + K_2 + S'_2$ and so on for the other three. These lead to the corresponding ‘delta’ relationships:-

$\Delta D_1 = \Delta Z_1 + \Delta K_1$ ----- (i) and $\Delta D_1 = \Delta P_1 + \Delta S'_1$ -----(ii)

$\Delta D_2 = \Delta Z_2 + \Delta K_2$ ------(iii) and $\Delta D_2 = \Delta P_2 + \Delta S'_2$ -----(iv) etc.

These equations ultimately provided methods for finding the settings of all the K-wheels that were based upon an analysis of probabilities. It was considered logical to begin with the K wheels, and if a method involving only one wheel could be found this would have been a great advantage since, for example if the K_1 wheel was selected, then there would have been only 41 different settings to test.

The following is an investigation (using probability theory) on the possibility of finding the setting of the K_1 wheel on its own. Let $\Pr[\Delta P_1 = \bullet\ ] = p$, (from the presence of repeats in the plain-text it was expected that $p > ½$).
Let Pr[the S wheels step on] = a and $\Pr[\Delta S_1 = x\ ] = b$. The conditions required for the event $[\Delta S'_1 = x]$ to occur are that the two events $[\Delta S_1 = x\ ]$ and [the S wheels step on] must both occur, for if the S wheels did <u>not</u> step on then the event $[\Delta S'_1 = x\ ]$ could not happen. Hence $\Pr[\Delta S'_1 = x\ ] = a.b$

Equation (ii) above states that $\Delta D_1 = \Delta P_1 + \Delta S'_1$, and it follows that
$\Pr[\Delta D_1 = \bullet\ ] = \Pr[\Delta P_1 = \bullet\ ].\Pr[\Delta S'_1 = \bullet\ ] + \Pr[\Delta P_1 = x\ ].\Pr[\Delta S'_1 = x\ ]$
(A consequence of the facts that: $\bullet + \bullet = \bullet$ and also that: $x + x = \bullet$)
This leads to $\Pr[\Delta D_1 = \bullet\ ] = p(1 – a.b)+(1 - p)a.b$ (after the substitutions)
$= p + a.b(1 - 2p)$.
Let this expression = u. If $u > ½$, then this would imply that at the correct setting the event $[\Delta D_1 = \bullet\ ]$ is not random, and that the setting of K_1 might be found by counting the dots in the long sequence of ΔD_1 bits obtained from the whole cipher message. It would be necessary to repeat this counting process for all of the 41 possible starting positions of the K_1 wheel, the correct setting giving a dot count that was greater than any of the others. (The counts made at all the other settings would be associated with the random probability value ½.). A practical dot counting process could be based on equation (i) above, using the known values of $\Delta Z_1 + \Delta K_1$

Unfortunately the Germans had imposed the rule a.b=½ on the wheel patterns they used, and introducing this value, the expression $u = p + a.b(1 -2p) = ½$. Hence the sequence of ΔD binary bits derived from *one* bit-stream was virtually random no matter what the value of ‘p’ happened to be, and so it was found not possible to find the setting for a single wheel on its own.

Another investigation on the feasibility of finding simultaneously the correct settings of the pair of wheels K_1 and K_2, gave a more promising result, although the number of possible pairs of settings for these two wheels which then had to be tested was much greater:- (41 x 31) = 1271.
A probability analysis of this two wheel situation is given below:-
Consider the 1st and 2nd bit-streams:-
let $\Pr[\Delta S_1 = x\ ] = b_1$ and $\Pr[\Delta S_2 = x\ ] = b_2$ .The rule a.b = ½ that the Germans had imposed now worked against them. Since the value of ‘a’ is fixed, it follows that $b_1 = b_2$. Replacing both by the single symbol ‘b’:-

$$\Pr[\Delta S_1 + \Delta S_2 = \bullet\ ] = \Pr[\Delta S_1 = \bullet\ ].\Pr[\Delta S_2 = \bullet\ ] + \Pr[\Delta S_1 = x\ ].\Pr[\Delta S_2 = x\ ] \quad .$$
$$= (1 - b)(1 - b) + b.b = (1 - b)^2 + b^2$$

Next consider the event [$\Delta S'_1 + \Delta S'_2 = \bullet$], for this to happen then either the S wheels do not step on (probability = 1 - a), or they do step and $\Delta S_1 + \Delta S_2 = \bullet$

Hence $\Pr[\Delta S'_1 + \Delta S'_2 = \bullet\ ] = (1 - a) + a\{(1 - b)^2 + b^2\} = 1 - 2ab + 2ab^2$

$= b$ (as $a.b = ½$)

From the known structure of the Lorenz machine and the probable patterns of the tabs set up on the wheels, BP assumed that 0.703 to be a realistic estimate for the value of Pr[the S wheels step on], (i.e. a = 0.70).

Since ab =½ it follows that b = 1/(2a) = 0.71.

So $\Pr[\Delta S'_1 + \Delta S'_2 = \bullet\ ] = 0.71$ (= 0.7 approx.)

Previously messages had indicated that $\Pr[\Delta P_1 + \Delta P_2 = \bullet\ ] = 0.6$ (approx.)

Adding equations (ii) and (iv) gives:- $\Delta D_1 + \Delta D_2 = (\Delta P_1 + \Delta P_2) + (\Delta S'_1 + \Delta S'_2)$
hence $\Pr[\Delta D_1 + \Delta D_2 = \bullet\ ] = \Pr[(\Delta P_1 + \Delta P_2) + (\Delta S'_1 + \Delta S'_2) = \bullet\ ]$.
This expression can be expanded to give:- $\Pr[\Delta D_1 + \Delta D_2 = \bullet\ ] =$
$\Pr[(\Delta P_1 + \Delta P_2) = \bullet\ ].\Pr[(\Delta S'_1 + \Delta S'_2) = \bullet\ ] + \Pr[(\Delta P_1 + \Delta P_2) = x\ ].\Pr[(\Delta S'_1 + \Delta S'_2) = x\ ]$

Substituting the numerical approximations given earlier:-

$\Pr[\Delta D_1 + \Delta D_2 = \bullet\ ] = 0.6 \times 0.7 + (1 - 0.6)(1 - 0.7)$

$= 0.42 + 0.12 = 0.54$

This result showed that the sum of the two delta bit-streams was **not** random.

Equations (i) and (iii) given previously lead to:-

$$\mathbf{\Delta D_1 + \Delta D_2 = (\Delta Z_1 + \Delta K_1) + (\Delta Z_2 + \Delta K_2)}$$

and this provided a practical way for making the ‘dot counts’ on Colossus.

The important result:- $\Pr[\Delta D_1 + \Delta D_2 = \bullet\ ] = 0.54$, shows that a dot count made with the K_1 and K_2 wheels at their correct settings was expected to give a higher score than that obtained for any of the pairs of incorrect starting positions, where the same event would have a numerical probability = 0.5. The difference however is quite small, and to establish a clear distinction between the correct pair of wheel settings and all the remaining 1270 wrong pairs, the dot counts had to be derived from a large number of cipher characters.

A Statistical analysis:

Assume that the number of dots in the counts made for each wrong pair of settings conformed to a Binomial probability distribution (p = ½)
If a cipher message contains N characters, then the expected score when the K_1 & K_2 wheels are at their correct settings = 0.54N. For any other incorrect settings the expected random score = 0.50N, and standard deviation = $\sqrt{N}/2$.
The deviation from the mean = 0.04N, and in order to be significant this should not be less than some chosen multiple of the standard deviation. A value of 4 was adopted at BP for this multiple. Hence $0.04N > 4.\sqrt{N}/2$. After squaring both sides this gives:- $(0.04)^2 N > 4$.

Hence N >2500. This gives the minimum length of code required to give good prospects of discrimination.

The practical procedure:
The dot counts (scores) were carried out on the Colossus computer using the algorithm given earlier:- $(\Delta Z_1 + \Delta K_1) + (\Delta Z_2 + \Delta K_2)$. The machine read optically the required cipher bit-streams from a five hole punched paper tape, and combined them with the wheel patterns of the two K-wheels in the way required by the algorithm, finally printing out the score. This procedure was repeated for every possible pair of wheel settings. The tape was formed into a continuous loop so that after one 'dot count' had been completed for one pair of wheel settings, the count for the next pair could begin. The amount of 'bit processing' required was considerable. For a cipher message of say 3000 characters, the above algorithm would have to be used 41x31x3000 times (= 3,813,000). In order to carry out this huge task in a realistic time, the machine had to be very fast. By reading the cipher characters at the impressive speed of 5000 per second, Colossus would have completed the procedure in about 13 minutes, indicative of a very remarkable engineering achievement.

Finding the settings of the remaining wheels:
A detailed examination of the characteristics of the German plaintext provided the basis for other procedures that could be used to find the settings of all the other Lorenz wheels with Colossus. However it was the normal practice to restrict this work to the five K-wheels. After the settings of the K-wheels had been found by means of the machine, the settings of the S–wheels and the 'motor' wheels were then be recovered by hand methods.

There were good reasons for adopting this practice as the limited number of machines available had other important and more lengthy tasks to perform that could not be done by hand, as in addition to changing the wheel settings for each message the Germans also from time to time also changed the tab settings on the wheels. The considerable task of determining the tab settings on each wheel was known as '*wheel breaking*'. This was a very remarkable procedure that in part resembled an iterative process of the type frequently employed to find the numerical solutions of equations.

In July 1944 on at least one of the most important Lorenz communication links ('Jellyfish') the wheel patterns began to be changed every day, and this presented the three Colossus machines that had by then been installed at BP with a mammoth task (by April 1945 ten of these machines were in operation).

The breaking of the Lorenz cipher was of the greatest significance during the planning of the D-Day operations. (For information about this readers should refer to the 'History' section on this website to see the many references to '*Fish traffic*'.) After the war two of the leading mathematicians who had worked at BP, Prof 'Max' Newman and Dr Alan Turing, became involved with the development of some of the first general-purpose digital computers. This work would at least in part have been motivated by their knowledge of the Colossus machines.

Frank Carter