What is Computer Science?

An Information Security Perspective

Daniel Page ⟨dan@phoo.org⟩ and Nigel P. Smart ⟨csnps@bristol.ac.uk⟩

git # 148def3 @ 2018-07-11

THE ENIGMA OF GROUP THEORY

With the advent of the 1920s people saw the need for a mechanical encryption device. Taking a substitution cipher and then rotating it became seen as the ideal solution. This idea had actually been used previously in a number of manual ciphers, but using machines it was seen how this could be done more efficiently. The rotors could be implemented using wires and then encryption could be done mechanically using an electrical circuit. By rotating the rotor we obtain a new substitution cipher.

As an example, suppose the rotor used to produce the substitutions is given by

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
TMKGOYDSIPELUAVCRJWXZNHBQF
```

By this we mean that the plaintext letter A will encrypt to the ciphertext letter T, and so on. In the past a substitution cipher was only used on its own, thus the word EGG would encrypt to the word ODD. This turns out not to be a good idea because the plaintext letter, e.g. G, always encrypts to the same ciphertext letter, e.g. D. This fact allows a cipher based solely on a single substitution, a so-called *substitution cipher*, to be easily broken [9].

A simple way of increasing the security of a substitution cipher is to rotate the ciphertext alphabet by one letter every time we encrypt. So to encrypt the first letter we use the substitutions given by

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
TMKGOYDSIPELUAVCRJWXZNHBQF
```

However, to encrypt the second letter we rotate the rotor by one position and use the substitutions

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
MKGOYDSIPELUAVCRJWXZNHBQFT
```

To encrypt the third letter we use the substitutions

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
KGOYDSIPELUAVCRJWXZNHBQFTM
```

and so on. This the word EGG will become encrypted to ODS. This gives us a polyalphabetic substitution cipher with 26 alphabets.

The most famous of the rotor machines developed in the first half of the twentieth century was the Enigma machine used by the Germans in World War II [2]. We shall describe the most simple version of Enigma which only used three such rotors, chosen from the following set of five.

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
EKMFLGDQVZNTOWYHXUSPAIBRCJ
AJDKSIRUXBLHWTMCQGZNPYFVOE
BDFHJLCPRTXVZNYEIWGAKMUSQO
ESOVPZJAYQUIRHXLNFTGKDCMWB
VZBRGITYUPSDNHLXAWMJQOFECK
```

Machines in use towards the end of the war had a larger number of rotors, chosen from a larger set. Note, the order of the rotors in the machine is important, so the number of ways of choosing the rotors is

$$5 \cdot 4 \cdot 3 = 60.$$

Each rotor had an initial starting position, and since there are 26 possible starting positions for each rotor, the total number of possible starting positions is $26^3 = 17\,576$.

The first rotor would step on the second rotor on each full iteration under the control of a ring hitting a notch, likewise the stepping of the third rotor was controlled by the second rotor. Both the rings were movable and their positions again formed part of the key, although only the notch and ring positions for the first two rotors were important. Hence, the number of ring positions was $26^2 = 676$. The second rotor also had a kick associated to it making the cycle length of the three rotors equal to

$$26 \cdot 25 \cdot 26 = 16\,900.$$

The effect of the moving rotors was that a given plaintext letter would encrypt to a different ciphertext letter on each press of the keyboard. Finally, a plug board was used to swap letters twice in each encryption and decryption operation. This increased the complexity and gave another possible 10^{14} keys.

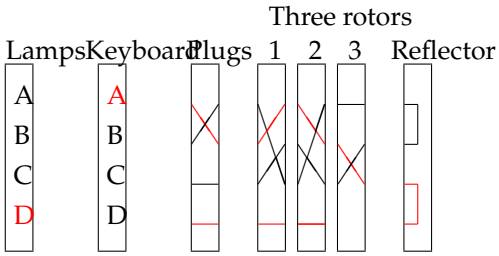
The rotors used, their order, their starting positions, the ring positions and the plug board settings all made up the secret key. Hence, the total number of keys was then around 2^{75}.

To make sure encryption and decryption were the same operation the message letter was passed through the plug board, then through the three rotors, and then through another fixed substitution called the reflector. After passing through the reflector the message letter was sent back through the three rotors and the plugboard again. The reflector was fixed in the machine, and was given by

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
YRUHQSLDPXNGOKMIEBFZCWVJAT
```

The operation of a simplified Enigma machine is described in Fig. 1. By tracing the red lines one can see how the plaintext character A encrypts to the ciphertext character D. Notice that encryption and decryption can be performed by the machine being in the same positions. Now assume that rotor one moves on one step, so A now maps to D under rotor one, B to A, C to C and D to B. You should work out what happens with the example when we encrypt A again.

Figure 1: *Simplified Enigma machine*



The purpose of this note is to present a mathematical model of the Enigma machine and show how some basic facts about permutations allowed the Polish cryptographers to break the cipher.

1 Permutations

We want to find a mathematical description of the Enigma machine. To do this we will first need to introduce the concept of a permutation [7].

We let A be a finite set of size n, we might as well assume that the set is given by $A = \{1, 2, \ldots, n\}$. A function from A to A is said to be one-to-one if every element in A maps to exactly one element in A; for example if $A = \{1, 2, 3\}$, then we have the one-to-one function $f(1) = 2$, $f(2) = 3$ and $f(3) = 1$. Another name for such one-to-one functions from A to A is a permutation, since such a function permutes the elements in the set around.

This is a very cumbersome way to write a permutation. Mathematicians (being lazy people) have invented the following notation, the function f above is written as

$$\begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix}.$$

What should be noted about this notation (which applies for arbitrary n) is that all the numbers between 1 and n occur exactly once on each row. The first row is always given as the numbers 1 to n in increasing order. Any such matrix with these properties represents a permutation, and all permutations can be represented by such a matrix.

The set of all permutations on a set of size n is denoted by S_n. The above matrix notation allows us to see very easily that the size of the set S_n is $n!$. To see this we notice that there are n choices for the first element in the second row of the above matrix. Then there are $n-1$ choices for the second element in the second row and so on.

Suppose we define the permutations

$$g = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix} \text{ and } f = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{pmatrix}$$

As permutations are nothing but functions we can compose them. Remembering that $g \circ f$ means apply the function f and then apply the function g we see that

$$\begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix} \circ \begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{pmatrix}$$

means $1 \to 3 \to 1$, $2 \to 2 \to 3$ and $3 \to 1 \to 2$. Hence, the result of composing the above two permutations is

$$\begin{pmatrix} 1 & 2 & 3 \\ 1 & 3 & 2 \end{pmatrix}.$$

If σ is a permutation on a set A then we usually think of σ acting on the set. So if $a \in A$ then we write

$$a^\sigma$$

for the action of σ on the element a, i.e. this is another way of writing $\sigma(a)$. However, this can cause confusion when using the standard notation for function composition above. For example

$$1^{g \circ f} = g(f(1)) = 3$$

so we are unable to read the permutation from left to right. However, if we use another notation say $\cdot$ to mean

$$f \cdot g = g \circ f$$

then we are able to read the expression from left to right, i.e.

$$1^{f \cdot g} = g(f(1)).$$

We shall call this operation multiplying permutations.

Mathematicians are, as we said, by nature lazy people and this notation we have introduced is still a little too much. For instance we always write down the numbers $1, \ldots, n$ in the top row of each matrix to represent a permutation. Also some columns are redundant, for instance the first column of the permutation

$$\begin{pmatrix} 1 & 2 & 3 \\ 1 & 3 & 2 \end{pmatrix}.$$

We now introduce another notation for permutations which is concise and clear, which uses the concept of a *cycle*. By a cycle or n-cycle we mean the object $(x_1, \ldots, x_n)$ with distinct $x_i \in \mathbb{N} \setminus \{0\}$. This represents the permutation $f(x_1) = x_2$, $f(x_2) = x_3$, $\ldots$, $f(x_{n-1}) = x_n$, $f(x_n) = x_1$ and for $x \notin \{x_1, \ldots, x_n\}$ we have $f(x) = x$. For instance we have

$$\begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix} = (1,2,3) = (2,3,1) = (3,1,2).$$

Notice that a cycle is not a unique way of representing a permutation. As another example we have

$$\begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{pmatrix} = (1,3)(2) = (3,1)(2).$$

The identity permutation is represented by $()$. Again, as mathematicians are lazy we always write $(1,3)(2) = (1,3)$. This can still lead to ambiguities as $(1,2)$ could represent a function

$$\{1,2\} \to \{1,2\} \text{ or } \{1,2,\ldots,n\} \to \{1,2,\ldots,n\}.$$

Which function it represents is usually however clear from the context.

Two cycles $(x_1, \ldots, x_n)$ and $(y_1, \ldots, y_n)$ are called disjoint if $\{x_1, \ldots, x_n\} \cap \{y_1, \ldots, y_n\} = \emptyset$. It is easy to show that if σ and τ are two disjoint cycles then

$$\sigma \cdot \tau = \tau \cdot \sigma.$$

Note this is not true for cycles which are not disjoint, e.g.

$$(1,2,3,4) \cdot (3,5) = (1,2,5,3,4) \neq (1,2,3,5,4) = (3,5) \cdot (1,2,3,4).$$

This means that multiplying permutations does not follow the usual "commutative rule" of standard multiplication.

Our action of permutations on the underlying set can now be read easily from left to right,

$$2^{(1,2,3,4)\cdot(3,5)} = 3^{(3,5)} = 5 = 2^{(1,2,5,3,4)},$$

as the permutation $(1,2,3,4)$ maps 2 to 3 and the permutation $(3,5)$ maps 3 to 5. With this notation it is easily seen that computing the value of permutations becomes very simple, which is why the notation is used quite a lot.

What really makes disjoint cycles interesting is that every permutation can be written as a product of disjoint cycles. To see this let σ be a permutation on $\{1, \ldots, n\}$ and let σ_1 denote the cycle

$$(1, \sigma(1), \sigma(\sigma(1)), \ldots, \sigma(\ldots\sigma(1)\ldots)),$$

where we keep applying σ until we get back to 1. We then take an element x of $\{1, \ldots, n\}$ such that $\sigma_1(x) = x$ and consider the cycle σ_2 given by

$$(x, \sigma(x), \sigma(\sigma(x)), \ldots, \sigma(\ldots\sigma(x)\ldots)).$$

We then take an element of $\{1, \ldots, n\}$ which is fixed by σ_1 and σ_2 to create a cycle σ_3. We continue this way until we have used all elements of $\{1, \ldots, n\}$. The resulting cycles $\sigma_1, \ldots, \sigma_t$ are obviously disjoint and their product is equal to the cycle σ.

What is nice about the above argument is that it is constructive. Given a permutation we can follow the procedure in the argument above to obtain the permutation as a product of disjoint cycles. To see this consider the permutation

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 2 & 3 & 7 & 6 & 8 & 4 & 1 & 5 & 9 \end{pmatrix}.$$

We have $\sigma(1) = 2,\ \sigma(2) = 3,\ \sigma(3) = 7$ and $\sigma(7) = 1$ so the first cycle is

$$\sigma_1 = (1,2,3,7).$$

The next element of $\{1, \ldots, 9\}$ which we have not yet considered is 4. We have $\sigma(4) = 6$ and $\sigma(6) = 4$ so $\sigma_2 = (4,6)$. Continuing in this way we find $\sigma_3 = (5,8)$ and $\sigma_4 = (9)$. Hence we have

$$\sigma = (1,2,3,7)(4,6)(5,8)(9) = (1,2,3,7)(4,6)(5,8).$$

In the rest of this note we present more details on the Enigma machine and some of the attacks which can be performed on it. However before presenting the machine itself we need to fix some notation which will be used throughout the note. In particular lower-case letters will denote variables, upper-case letters will denote "letters" (of the plaintext/ciphertext languages) and greek letters will denote permutations in S_{26} which we shall represent as permutations on the upper-case letters. Hence x can equal X and Y, but X can only ever represent X, whereas χ could represent (XY) or (ABC).

Our convention of permutations acting on the right of elements of the underlying set, then in this context is exemplified by

$$A^{(ABCD)(XY)} = B.$$

We now collect some basic facts and theorems about permutations which we will need in the sequel.

Fact 1. *Two permutations σ and τ which are conjugate, i.e. for which $\sigma = \lambda \cdot \tau \cdot \lambda^{-1}$ for some permutation λ, have the same cycle structure.*

We define the support of a permutation to be the set of letters which are not fixed by the permutation. Hence, if σ acts on the set of letters $\mathcal{L}$, then as usual we denote by $\mathcal{L}^\sigma$ the set of fixed points and hence the support is given by

$$\mathcal{L} \setminus \mathcal{L}^\sigma.$$

Fact 2. *If two permutations, with the same support, consist only of disjoint transpositions then their product contains an even number of disjoint cycles of the same lengths.*

Fact 3. *If a permutation with support an even number of symbols has an even number of disjoint cycles of the same lengths, then the permutation can be written as a product of two permutations each of which consists of disjoint transpositions.*

In many places we need an algorithm to solve the following problem: Given $\alpha_i, \beta_i \in S_{26}$, for $i = 1, \ldots, m$ find $\gamma \in S_{26}$ such that

$$\alpha_i = \gamma^{-1} \cdot \beta_i \cdot \gamma \text{ for } i = 1 \ldots, m.$$

Note, there could be many such solutions γ, but in the situations we will apply it we expect there to be only a few. For example suppose we have one such equation with

$$\begin{aligned} \alpha_1 &= (AFCNE)(BWXHUJOG)(DVIQZ)(KLMYTRPS), \\ \beta_1 &= (AEYSXWUJ)(BFZNO)(CDPKQ)(GHIVLMRT) \end{aligned}$$

We need to determine the structure of the permutation γ such that

$$\alpha_1 = \gamma^{-1} \cdot \beta_1 \cdot \gamma.$$

We first look at what should A map to under γ. If $A^\gamma = B$, then from α_1 and β_1 we must have $E^\gamma = W$, which in turn implies $Y^\gamma = X$. Carrying on in this way via a pruned depth first search [1] we can determine a set of possible values for γ. Such an algorithm is relatively simple to write down in a computer programming language, using a recursive procedure call [8]. It however of course been a bit of a pain to do this by hand, as one would need to in the 1930's and 1940's.

2 An equation for the Enigma

To aid our discussion in later sections we now describe the Enigma machine as a permutation equation. We first assume a canonical map between letters and the integers $\{0, 1, \ldots, 25\}$ such that $0 = A$, $1 = B$, etc and we assume a standard three wheel Enigma machine.

The wheel which turns the fastest we shall call rotor one, whilst the one which turns the slowest we shall call rotor three. This means that when looking at a real machine rotor three is the left most rotor and rotor one is the right most rotor. This can cause confusion (especially when reading day/message settings), so please keep this in mind.

The basic permutations which make up the Enigma machine are as follows:

2.1 Choice of rotors

We assume that the three rotors are chosen from the following set of five rotors. We present these rotors in cycle notation, but they are the commonly labelled rotors *I*, *II*, *III*, *IV* and *V* used in the actual Enigma machines, which were given earlier. Each rotor also has a different notch position which controls how the stepping of one rotor drives the stepping of the others.

Rotor	Permutation Representation	Notch Position
I	(*AELTPHQXRU*)(*BKNW*)(*CMOY*)(*DFG*)(*IV*)(*JZ*)	16 = *Q*
II	(*BJ*)(*CDKLHUP*)(*ESZ*)(*FIXVYOMW*)(*GR*)(*NT*)	4 = *E*
III	(*ABDHPEJT*)(*CFLVMZOYQIRWUKXSG*)	21 = *V*
IV	(*AEPLIYWCOXMRFZBSTGJQNH*)(*DV*)(*KU*)	9 = *J*
V	(*AVOLDRWFIUQ*)(*BZKSMNHYC*)(*EGTJPX*)	25 = *Z*

2.2 Reflector

There were a number of reflectors used in actual Enigma machines. In our description we shall use the reflector given earlier, which is often referred to as "Reflector B". This reflector has representation via disjoint cycles as

$$\varrho = (AY)(BR)(CU)(DH)(EQ)(FS)(GL)(IP)(JX)(KN)(MO)(TZ)(VW).$$

2.3 An Enigma key

An Enigma key consists of the following information:

- A choice of rotors ρ_1, ρ_2, ρ_3 from the above choice of five possible rotors. Note, this choice of rotors affects the three notch positions, which we shall denote by n_1, n_2 and n_3. Also, as noted above, the rotor ρ_3 is placed in the left of the actual machine, whilst rotor ρ_1 is placed on the right. Hence, if in a German code book it says use rotors
$$I, II, III,$$
this means in our notation that ρ_1 is selected to be rotor III, that ρ_2 is selected to be rotor II and ρ_3 is selected to be rotor I.
- One must also select the ring positions, which we shall denote by r_1, r_2 and r_3. In the actual machine these are letters, but we shall use our canonical numbering to represent these as integers in $\{0, 1, \ldots, 25\}$.
- The plugboard is simply a product of disjoint transpositions which we shall denote by the permutation τ. In what follows we shall denote a plug linking letter A with letter B by $A \leftrightarrow B$.
- The starting rotor positions we shall denote by p_1, p_2 and p_3. These are the letters which can be seen through the windows on the top of the Enigma machine. Remember our numbering system is that the window on the left corresponds to p_3 and that on the right corresponds to p_1.

2.4 The encryption operation

We let σ denote the shift-up permutation given by

$$\sigma = (ABCDEFGHIJKLMNOPQRSTUVWXYZ).$$

The stepping of the second and third rotor is probably the hardest part to grasp when first looking at an Enigma machine, however this has a relatively simple description when one looks at it in a mathematical manner.

Given the above description of the key we wish to deduce the permutation ϵ_j, which represents the encryption of the jth letter, for $j = 0, 1, 2, \ldots,$.

We first set

$$\begin{aligned} m_1 &= n_1 - p_1 - 1 \pmod{26}, \\ m &= n_2 - p_2 - 1 \pmod{26}, \\ m_2 &= m_1 + 1 + 26m. \end{aligned}$$

The values of m_1 and m_2 control the stepping of the second and the third rotors.

We let $\lfloor x \rfloor$ denote the round towards zero function, i.e. $\lfloor 1.9 \rfloor = 1$ and $\lfloor -1.9 \rfloor = -1$. We now set, for encrypting letter j,

$$\begin{aligned} k_1 &= \lfloor (j - m_1 + 26)/26 \rfloor, \\ k_2 &= \lfloor (j - m_2 + 650)/650 \rfloor, \\ i_1 &= p_1 - r_1 + 1, \\ i_2 &= p_2 - r_2 + k_1 + k_2, \\ i_3 &= p_3 - r_3 + k_2. \end{aligned}$$

Notice, how i_3 is stepped on every $650 = 26 \cdot 25$ iterations whilst i_2 is stepped on every 26 iterations and also stepped on an extra notch every 650 iterations. We can now present ϵ_j as

$$\begin{aligned} \epsilon_j = \; & \tau \cdot (\sigma^{i_1+j} \rho_1 \sigma^{-i_1-j}) \cdot (\sigma^{i_2} \rho_2 \sigma^{-i_2}) \cdot (\sigma^{i_3} \rho_3 \sigma^{-i_3}) \cdot \varrho \cdot \\ & \cdot (\sigma^{i_3} {\rho_3}^{-1} \sigma^{-i_3}) \cdot (\sigma^{i_2} {\rho_2}^{-1} \sigma^{-i_2}) \cdot (\sigma^{i_1+j} {\rho_1}^{-1} \sigma^{-i_1-j}) \cdot \tau. \end{aligned}$$

Note that the same equation/machine is used to encrypt the jth letter as is used to decrypt the jth letter. Hence we have

$${\epsilon_j}^{-1} = \epsilon_j.$$

Also note that Each ϵ_j consists of a product of disjoint transpositions.

Note, we shall always use γ_j to represent the internal rotor part of the Enigma machine, hence

$$\epsilon_j = \tau \cdot \gamma_j \cdot \tau.$$

3 The Polish work on Enigma

The polish mathematicians Jerzy Rozycki [5], Henryk Zygalski [4] and Marian Rejewski [6] were the first to find ways of analysing the Enigma machine. To understand their methods one must first understand how the Germans used the machine. On each day the machine was set up with a key, as above, which was chosen by looking up in a code book. Each subnet would have a different day key.

To encipher a message the sending operator decided on a message key. The message key would be a sequence of three letters, say DHI. The message key needs to be transported to the recipient. Using the day key, the message key would be enciphered twice. The double enciphering is to act as a form of error control. Hence, DHI might be enciphered as $XHJKLM$. Note, that D encrypts to X and then K, this is a property of the Enigma machine.

The receiver would obtain $XHJKLM$ and then decrypt this to obtain DHI. Both operators would then move the wheels around to the positions D, H and I, i.e. they would turn the wheels so that D was in the leftmost window, H in the middle one and I in the rightmost window. Then the actual message would be enciphered.

For this example, in our notation, this would mean that the message key is equal to the day key, except that $p_1 = 8 = I$, $p_2 = 7 = H$ and $p_3 = 3 = D$.

However, life was even more difficult for the Poles as they did not even know the rotor wirings or the reflector values. Hence, they needed to break the machine without even having a description of the actual machine. They did have access to a non-military version of Enigma and deduced the basic structure. In this they were very lucky in that they deduced that the wiring between the plugboard and the right most rotor was in the order of the alphabet. If this were not the case there would have been some hidden permutation which would also have needed to be found.

Luckily the French cryptographer Gustave Bertrand [3] obtained from a German spy, Hans-Thilo Schmidt, two months worth of day keys. Thus, for two months of traffic the Poles had access to the day settings. From this information they needed to deduce the internal wirings of the Enigma machine.

Note, in the pre-war days the Germans only used three wheels out of a choice of three, hence the number of days keys is actually reduced by a factor of ten. This is, however, only a slight simplification (at least with modern technology).

Suppose we are given that the day setting is

Rotors	Rings	Pos	Plugboard
III, II, I	*TXC*	*EAZ*	(*AMTEBC*)

We do not know what the actual rotors are at present, but we know that the one labelled rotor I will be placed in the rightmost slot (our label one). So we have

$$r_1 = 2, \quad r_2 = 23, \quad r_3 = 19, p_1 = 25, \quad p_2 = 0, \quad p_3 = 4.$$

Suppose we intercept a set of messages which have the following headers, consisting of the encryption of the three letter rotor positions, followed by its encryption again, i.e. the first six letters of each message are equal to

```
UCWBLR  ZSETEY  SLVMQH  SGIMVW  PMRWGV
VNGCTP  OQDPNS  CBRVPV  KSCJEA  GSTGEU
DQLSNL  HXYYHF  GETGSU  EEKLSJ  OSQPEB
WISIIT  TXFEHX  ZAMTAM  VEMCSM  LQPFNI
LOIFMW  JXHUHZ  PYXWFQ  FAYQAF  QJPOUI
EPILWW  DOGSMP  ADSDRT  XLJXQK  BKEAKY
......  ......  ......  ......  ......
DDESRY  QJCOUA  JEZUSN  MUXROQ  SLPMQI
RRONYG  ZMOTGG  XUOXOG  HIUYIE  KCPJLI
DSESEY  OSPPEI  QCPOLI  HUXYOQ  NYIKFW
```

If we take the last one of these and look at it in more detail. We know that there are three underlying secret letters, say l_1, l_2 and l_3. We also know that

$$l_1{}^{\epsilon_0} = N, \quad l_2{}^{\epsilon_1} = Y, \quad l_3{}^{\epsilon_2} = I,$$

and

$$l_1{}^{\epsilon_3} = K, \quad l_2{}^{\epsilon_4} = F, \quad l_3{}^{\epsilon_5} = W.$$

Hence, given that ${\epsilon_j}^{-1} = \epsilon_j$, we have

$$N^{\epsilon_0\epsilon_3} = l_1{}^{\epsilon_0\epsilon_0\epsilon_3} = l_1{}^{\epsilon_3} = K, \quad Y^{\epsilon_1\epsilon_4} = F, \quad I^{\epsilon_2\epsilon_5} = W.$$

Continueing in this way we can compute a permutation representation of the three products as follows:

$$\begin{aligned}\epsilon_0 \cdot \epsilon_3 &= (ADSMRNKJUB)(CV)(ELFQOPWIZT)(HY),\\ \epsilon_1 \cdot \epsilon_4 &= (BPWJUOMGV)(CLQNTDRYF)(ES)(HX),\\ \epsilon_2 \cdot \epsilon_5 &= (AC)(BDSTUEYFXQ)(GPIWRVHZNO)(JK).\end{aligned}$$

From these we wish to deduce the values of ϵ_0, $\epsilon_1, \ldots, \epsilon_5$. We will use the fact that ϵ_j is a product of disjoint transpositions and Facts 2 and 3.

We take the first product and look at it in more detail. We take the sets of two cycles of equal degree and write them above one another, with the bottom one reversed in order, i.e.

$$\begin{array}{cccccccccc} A & D & S & M & R & N & K & J & U & B \\ T & Z & I & W & P & O & Q & F & L & E \end{array} \qquad \begin{array}{cc} C & V \\ Y & H \end{array}$$

We now run through all possible shifts of the bottom rows. Each shift gives us a possible value of ϵ_0 and ϵ_3. The value of ϵ_0 is obtained from reading off the disjoint transpositions from the columns, the value of ϵ_3 is obtained by reading off the transpositions from the "off diagonals". For example with the above orientation we would have

$$\begin{aligned}\epsilon_0 &= (AT)(DZ)(SI)(MW)(RP)(NO)(KQ)(JF)(UL)(BE)(CY)(VH),\\ \epsilon_3 &= (DT)(SZ)(MI)(RW)(NP)(KO)(JQ)(UF)(BL)(AE)(VY)(CH).\end{aligned}$$

But we have $20 = 2 \cdot 10$ such orientations, so there are 20 possible values for ϵ_0 and ϵ_3.

Now, to reduce this number we need to really on stupid operators. Various operators had a tendency to always select the same three letter message key. For example popular choices where QWE (the first letters on the keyboard). One operator used the letters of his girlfriend name, Cillie, hence such "cribs" (or guessed/known plaintexts in todays jargon) became known as "Cillies". Note, for our analysis here we only need one Cillie for the day when we wish to obtain the internal wiring of rotor I.

In our dummy example, suppose we guess (correctly) that the first message key is indeed QWE. This means that $UCWBLR$ is the encryption of QWE twice, this in turn tells us how to align our cycle of length 10 in the first permutation, as under ϵ_0 the letter Q must encrypt to U.

$$\begin{array}{cccccccccc} A & D & S & M & R & N & K & J & U & B \\ L & E & T & Z & I & W & P & O & Q & F \end{array}$$

We can check that this is consistent as we see that Q under ϵ_3 must then encrypt to B. If we guessed one more such Cillies we can reduce the number of possibilities for $\epsilon_1, \ldots, \epsilon_6$. Assuming we carry on in this way we will finally deduce that

$$\begin{aligned}\epsilon_0 &= (AL)(BF)(CH)(DE)(GX)(IR)(JO)(KP)(MZ)(NW)(QU)(ST)(VY),\\ \epsilon_1 &= (AK)(BQ)(CW)(DM)(EH)(FJ)(GT)(IZ)(LP)(NV)(OR)(SX)(UY),\\ \epsilon_2 &= (AJ)(BN)(CK)(DZ)(EW)(FP)(GX)(HS)(IY)(LM)(OQ)(RU)(TV),\\ \epsilon_3 &= (AF)(BQ)(CY)(DL)(ES)(GX)(HV)(IN)(JP)(KW)(MT)(OU)(RZ),\\ \epsilon_4 &= (AK)(BN)(CJ)(DG)(EX)(FU)(HS)(IZ)(LW)(MR)(OY)(PQ)(TV),\\ \epsilon_5 &= (AK)(BO)(CJ)(DN)(ER)(FI)(GQ)(HT)(LM)(PX)(SZ)(UV)(WY).\end{aligned}$$

We now need to use this information to deduce the value of ρ_1, etc. So for the rest of this section we assume we know ϵ_j for $j = 0, \ldots, 5$, and so we mark it in blue.

Recall that we have,

$$\begin{aligned}\epsilon_j &= \tau \cdot (\sigma^{i_1+j}\rho_1\sigma^{-i_1-j}) \cdot (\sigma^{i_2}\rho_2\sigma^{-i_2}) \cdot (\sigma^{i_3}\rho_3\sigma^{-i_3}) \cdot \varrho \cdot\\ &\quad \cdot(\sigma^{i_3}{\rho_3}^{-1}\sigma^{-i_3}) \cdot (\sigma^{i_2}{\rho_2}^{-1}\sigma^{-i_2}) \cdot (\sigma^{i_1+j}{\rho_1}^{-1}\sigma^{-i_1-j}) \cdot \tau\end{aligned}$$

We now assume that no stepping of the second rotor occurs during the first six encryptions under the day setting. This occurs with quite high probability, namely $20/26 \approx 0.77$. If this assumption turns out to be false we will notice this in our later analysis and it will mean we can deduce something about the (unknown to us at this point) position of the notch on the first rotor.

Given that we know the day settings, so that we know τ and the values of i_1, i_2 and i_3 (since we are assuming $k_1 = k_2 = 0$ for $0 \le j \le 5$), we can write the above equation for $0 \le j \le 5$ as

$$\begin{aligned}\lambda_j &= \sigma^{-i_1-j} \cdot \tau \cdot \epsilon_j \cdot \tau \cdot \sigma^{i_1+j}\\ &= \rho_1 \cdot \sigma^{-j} \cdot \gamma \cdot \sigma^{j} \cdot {\rho_1}^{-1}.\end{aligned}$$

Where λ_j is now known and we wish to determine ρ_1 for some fixed but unknown value of γ. The permutation γ is in fact equal to

$$\gamma = (\sigma^{i_2-i_1}\rho_2\sigma^{-i_2}) \cdot (\sigma^{i_3}\rho_3\sigma^{-i_3}) \cdot \varrho \cdot (\sigma^{i_3}{\rho_3}^{-1}\sigma^{-i_3}) \cdot (\sigma^{i_2}{\rho_2}^{-1}\sigma^{i_1-i_2}).$$

In our example we get the following values for λ_j,

$$\begin{aligned}
\lambda_0 &= (AD)(BR)(CQ)(EV)(FZ)(GP)(HM)(IN)(JK)(LU)(OS)(TW)(XY),\\
\lambda_1 &= (AV)(BP)(CZ)(DF)(EI)(GS)(HY)(JL)(KO)(MU)(NQ)(RW)(TX),\\
\lambda_2 &= (AL)(BK)(CN)(DZ)(EV)(FP)(GX)(HS)(IY)(JM)(OQ)(RU)(TW),\\
\lambda_3 &= (AS)(BF)(CZ)(DR)(EM)(GN)(HY)(IW)(JO)(KQ)(LX)(PV)(TU),\\
\lambda_4 &= (AQ)(BK)(CT)(DL)(EP)(FI)(GX)(HW)(JU)(MO)(NY)(RS)(VZ),\\
\lambda_5 &= (AS)(BZ)(CV)(DO)(EM)(FR)(GQ)(HK)(IL)(JT)(NP)(UW)(XY).
\end{aligned}$$

We now form, for $j = 0, \ldots, 4$,

$$\begin{aligned}
\mu_j &= \lambda_j \cdot \lambda_{j+1},\\
&= \rho_1 \cdot \sigma^{-j} \cdot \gamma \cdot \sigma^{-1} \cdot \gamma \cdot \sigma^{j+1} \cdot {\rho_1}^{-1},\\
&= \rho_1 \cdot \sigma^{-j} \cdot \delta \cdot \sigma^{j} \cdot {\rho_1}^{-1},
\end{aligned}$$

where $\delta = \gamma \cdot \sigma^{-1} \cdot \gamma \cdot \sigma$ is unknown. Eliminating δ via $\delta = \sigma^{j-1}{\rho_1}^{-1}\mu_{j-1}\rho_1\sigma^{-j+1}$ we find the following equations for $j = 1, \ldots, 4$,

$$\begin{aligned}
\mu_j &= (\rho_1 \cdot \sigma^{-1} \cdot {\rho_1}^{-1}) \cdot \mu_{j-1} \cdot (\rho_1 \cdot \sigma \cdot {\rho_1}^{-1}),\\
&= \alpha \cdot \mu_{j-1} \cdot \alpha^{-1},
\end{aligned}$$

where $\alpha = \rho_1 \cdot \sigma^{-1} \cdot {\rho_1}^{-1}$. Hence, μ_j and μ_{j-1} are conjugate and so by Fact 1 have the same cycle structure. For our example we have

$$\begin{aligned}
\mu_0 &= (AFCNE)(BWXHUJOG)(DVIQZ)(KLMYTRPS),\\
\mu_1 &= (AEYSXWUJ)(BFZNO)(CDPKQ)(GHIVLMRT),\\
\mu_2 &= (AXNZRTIH)(BQJEP)(CGLSYWUD)(FVMOK),\\
\mu_3 &= (ARLGYWFK)(BIHNXDSQ)(CVEOU)(JMPZT),\\
\mu_4 &= (AGYPMDIR)(BHUTV)(CJWKZ)(ENXQSFLO).
\end{aligned}$$

At this point we can check whether our assumption of no-stepping, i.e. a constant value for the values of i_2 and i_3 is valid. If a step did occur in the second rotor then the above permutations would be unlikely to have the same cycle structure.

We need to determine the structure of the permutation α, this is done by looking at the four equations simultaneously. We note that since σ and α are conjugates, under ρ_1, we know that α has cycle structure of a single cycle of length 26.

In our example we only find one possible solution for α, namely

$$\alpha = (AGYWUJOQNIRLSXHTMKCEBZVPFD).$$

To solve for ρ_1 we need to find a permutation such that

$$\alpha = \rho_1 \cdot \sigma^{-1} \cdot {\rho_1}^{-1}.$$

We find there are 26 such solutions

(AELTPHQXRU)(BKNW)(CMOY)(DFG)(IV)(JZ)
(AFHRVJ)(BLU)(CNXSTQYDGEMPIW)(KOZ)
(AGFIXTRWDHSUCO)(BMQZLVKPJ)(ENY)
(AHTSVLWEOBNZMRXUDIYFJCPKQ)
(AIZN)(BOCQ)(DJ)(EPLXVMSWFKRYGHU)
(AJEQCRZODKSXWGI)(BPMTUFLYHVN)
(AKTVOER)(BQDLZPNCSYI)(FMUGJ)(HW)
(AL)(BR)(CTWI)(DMVPOFN)(ESZQ)(GKUHXYJ)
(AMWJHYKVQFOGLBS)(CUIDNETXZR)
(ANFPQGMX)(BTYLCVRDOHZS)(EUJI)(KW)
(AOIFQH)(BUKX)(CWLDPREVS)(GN)(MY)(TZ)
(APSDQIGOJKYNHBVT)(CX)(EWMZUL)(FR)
(AQJLFSEXDRGPTBWNIHCYOKZVUM)
(ARHDSFTCZWOLGQK)(BXEYPUNJM)
(ASGRIJNKBYQLHEZXFUOMC)(DT)(PVW)
(ATE)(BZYRJONLIKC)(DUPWQM)(FVXGSH)
(AUQNMEB)(DVYSILJPXHGTFWRK)
(AVZ)(CDWSJQOPYTGURLKE)(FXIM)
(AWTHINOQPZBCEDXJRMGV)(FYUSK)
(AXKGWUTIORNP)(BDYV)(CFZ)(HJSLM)
(AYWVCGXLNQROSMIPBEF)(DZ)(HK)(JT)
(AZEGYXMJUVD)(BF)(CHLOTKIQSNRP)
(BGZFCIRQTLPD)(EHMKJV)(NSOUWX)
(ABHNTMLQUXOVFDCJWYZG)(EISP)
(ACKLRSQVGBITNUY)(EJXPF)(HOWZ)
(ADEKMNVHPGCLSRTOXQW)(BJY)(IUZ)

These are the values of $\rho_1 \cdot \sigma^i$, for $i = 0, \ldots, 25$.

So with one days messages we can determine the value of ρ_1 upto multiplication by a power of σ. The Polish had access to two months such data and so were able to determine similar sets for ρ_2 and ρ_3 (as different rotor orders are used on different days). Note, at this point the Germans did not use a selection of three from five rotors.

If we select three representatives $\hat{\rho_1}$, $\hat{\rho_2}$ and $\hat{\rho_3}$, from the sets of possible rotors, then we have

$$\begin{aligned} \hat{\rho_1} &= \rho_1 \cdot \sigma^{l_1}, \\ \hat{\rho_2} &= \rho_2 \cdot \sigma^{l_2}, \\ \hat{\rho_3} &= \rho_3 \cdot \sigma^{l_3}. \end{aligned}$$

However, we still do not know the value for the reflector ϱ, or the correct values of l_1, l_2 and l_3.

Any one of these versions of $\hat{\rho_i}$ will give a valid Enigma machine (with a different reflector). So it does not matter which choice we make for ρ_i,

4 Determining the day settings

Now having determined the internal wirings, given the set of two months of day settings obtained by Bertrand, the next task is to determine the actual key when the day settings are not available. At this stage we assume the Germans are still using the encrypt the message setting twice routine.

The essential trick here is to notice that if we write the cipher as

$$\epsilon_j = \tau \cdot \gamma_j \cdot \tau,$$

then

$$\epsilon_j \cdot \epsilon_{j+3} = \tau \cdot \gamma_j \cdot \gamma_{j+3} \cdot \tau.$$

So $\epsilon_j \cdot \epsilon_{j+3}$ is conjugate to $\gamma_j \cdot \gamma_{j+3}$ and so by Fact 1 they have the same cycle structure. More importantly the cycle structure does not depend on the plug board τ.

Hence, if we can use the cycle structure to determine the rotor settings then we are only left with determining the plugboard settings. If we can determine the rotor settings then we know the values of γ_j, for $j = 1, \ldots, 6$, from the encrypted message keys we can compute ϵ_j for $j = 1, \ldots, 6$ as in the previous section. Hence,

determining the plugboard settings is then a question of solving one of our conjugacy problems again, for τ. But this is easier than before as we have that τ must be a product of disjoint transpositions.

We have already discussed how to compute $\epsilon_j \cdot \epsilon_{j+3}$ from the encryption of the message keys. Hence, we simply compute these values and compare their cycle structures with those obtained by running through all possible

$$60 \cdot 26^3 \cdot 26^3 = 18,534,946,560$$

choices for the rotors, positions and ring settings. Note, that when this was done by the Poles in the 1930's there was only a choice of the ordering of three rotors. The extra choice of rotors did not come in till a bit later. Hence, the total choice was 10 times less than this figure.

The above simplifies further if we assume that no stepping of the second and third rotor occurs during the calculation of the first six ciphertext characters. Recall this happens around 77 percent of the time. In such a situation the cycle structure depends only on the rotor order and the difference $p_i - r_i$ between the starting rotor position and the ring setting. Hence, we might as well assume that $r_1 = r_2 = r_3 = 0$ when computing all of the cycle structures. So, for 77 percent of the days our search amongst the cycle structures is then only among

$$60 \cdot 26^3 = 1,054,560 \text{ (resp. } 105,456)$$

possible cycle structures. Whilst this may seem a lot to do without a modern computer the Polish cryptographers found various ingenious ways of making this tractable.

After the above procedure we have determined all values of the initial day setting bar p_i and r_i, however we know the differences $p_i - r_i$. We also know for any given message the message key p'_1, p'_2, p'_3. Hence, in breaking the actual message we only require the solution for r_1, r_2, the value for r_3 is irrelevant as the third rotor never moves a fourth rotor. Most German messages started with the same two letter word followed by space (space was encoded by 'X'). Hence, we only need to go through 26^2 different positions to get the correct ring setting. Actually one goes through 26^2 wheel positions with a fixed ring, and use the differences to infer the actual ring settings. Once, r_i is determined from one message the value of p_i can be determined for the day key and then all messages can be trivially broken.

Once the rotor positions had been found for the day, the next task was to compute the positions of the plugs. However, once the rotors are determined the Enigma machine becomes very much like a substitution cipher. It is not quite a substitution cipher, however the same cryptanalytic techniques can be applied. That is one determines the plugboard settings from the viewing the enough ciphertext and from the statistics of the underlying language, which in this case is German.

References

[1] *Wikipedia: Depth-first search*. http://en.wikipedia.org/wiki/Depth-first_search (see p. 7).

[2] *Wikipedia: Enigma machine*. http://en.wikipedia.org/wiki/Enigma_machine (see p. 3).

[3] *Wikipedia: Gustave Bertrand*. http://en.wikipedia.org/wiki/Gustave_Bertrand (see p. 9).

[4] *Wikipedia: Henryk Zygalski*. http://en.wikipedia.org/wiki/Henryk_Zygalski (see p. 9).

[5] *Wikipedia: Jerzy Rozycki*. http://en.wikipedia.org/wiki/Jerzy_R%C3%B3%C5%BCycki (see p. 9).

[6] *Wikipedia: Marian Rejewski*. http://en.wikipedia.org/wiki/Marian_Rejewski (see p. 9).

[7] *Wikipedia: Permutation*. http://en.wikipedia.org/wiki/Permutation (see p. 4).

[8] *Wikipedia: Recursion*. http://en.wikipedia.org/wiki/Recursion (see p. 7).

[9] *Wikipedia: Substitution cipher*. http://en.wikipedia.org/wiki/Substitution_cipher (see p. 3).

I have a question/comment/complaint for you. Any (positive *or* negative) feedback, experience or comment is very welcome; this helps us to improve and extend the material in the most useful way.

However, keep in mind we are far from perfect; mistakes are of course possible, although hopefully rare. Some cases are hard for us to check, and make your feedback even more valuable: for instance

1. minor variation in software versions can produce subtle differences in how some commands and hence examples work, and
2. some examples download and use online resources, but web-sites change over time (or even might differ depending on where you access them from) so might cause the example to fail.

Either way, if you spot a problem then let us know: we will try to explain and/or fix things as fast as we can!

Can I use this material for something ? We are distributing these PDFs under the Creative Commons Attribution-ShareAlike License (CC BY-SA)

http://creativecommons.org/licenses/by-sa/4.0/

This is a fancy way to say you can basically do whatever you want with them (i.e., use, copy and distribute it however you see fit) as long as a) you correctly attribute the original source, and b) you distribute the result under the same license.

Is there a printed version of this material I can buy? Yes: Springer have published selected Chapters in

http://www.springer.com/computer/book/978-3-319-04041-7

while *also* allowing us to keep electronic versions online. Any royalties from this published version are donated to the UK-based Computing At School (CAS) group, whose ongoing work can be followed at

http://www.computingatschool.org.uk/

Why are all your references to Wikipedia? Our goal is to give an easily accessible overview, so it made no sense to reference lots of research papers. There are basically two reasons why: research papers are often written in a way that makes them hard to read (even when their intellectual content is not difficult to understand), and although many research papers are available on the Internet, many are not (or have to be paid for). So although some valid criticisms of Wikipedia exist, for introductory material on Computer Science it certainly represents a good place to start.

I like programming; why do the examples include so little programming? We want to focus on interesting topics rather than the mechanics of programming. So even when we include example programs, the idea is to do so in a way where their meaning is fairly clear. For example it makes more sense to use pseudo-code algorithms or reuse existing software tools than complicate a description of something by including pages and pages of program listings.

But you need to be able to program to do Computer Science, right? Yes! But only in the same way as you need to be able to read and write to study English. Put another way, reading and writing, or grammar and vocabulary, are just tools: they simply allow us to study topics such as English literature. Computer Science is the same. Although it *is* possible to study programming as a topic in itself, we are more interested in what can be achieved *using* programs: we treat programming itself as another tool.