MATH 587 Project: The Enigma Machine

Nathan Bickel William Edwards Evie Ellis Jackson Ginn
Gavin Hewitt

December 11, 2024

1 Introduction

Cryptography is the process of preventing unauthorized access to information being transmitted over insecure channels via ciphers. Ciphers are algorithms used to encrypt and decrypt plaintext into unreadable forms known as cipher text. Decrypting cipher text without the knowledge of a secret key is, ideally, quite difficult, making different kinds of ciphers very popular tools for communicating sensitive wartime information between various parties.

The Enigma Machine is a cipher device created and used in the early-to-mid twentieth century. It was most notably used to transmit military communications between all branches of the German military in World War II. The machine scrambles the alphabet via en electromechanical rotor that illuminates a light representing a letter. When entering plaintext, the illuminated letter represents cipher text, and the opposite occurs when entering cipher text. A new light is illuminated with each keystroke, making it even more difficult to crack given that a plaintext character is almost never encrypted to the same cipher text character each time it is written. The main security of the system, however, relies on a predetermined list of machine settings altered daily and on additional settings that were changed for each message. When receiving a message, the station had to be aware of the exact settings needed to accurately decrypt a message. The complexity of the machine allowed for it to be used to safely encrypt many highly top-secret messages prior to its cracking in December of 1932.[10]

2 History

The creation of the Enigma Machine relies on the invention of the rotor machine around 1915. The history of the first rotor machine is somewhat complicated; it was originally thought that the first ever rotor machines were developed simultaneously by four different people around the world: Edward Hebern in the U.S., Arvid Damm in Sweden, Hugo Koch in the Netherlands, and Arthur Scherbius in Germany. However, it was discovered nearly a century later in 2003 that the original rotor machine was actually invented by two Dutch naval officers, Theo van Hendel and Rudolf Spengler, in 1915. The two produced the first rotor-based cipher machines for the Dutch War Ministry to use.[8]

The most notable rotor-based cipher machine, the Enigma Machine, was not invented until closer to the end of World War I by Arthur Scherbius in Germany. It was originally

called the Probemaschine, which directly translates to prototype machine, and was the beginning in a long line of iterations of glow lamp cipher machines that led to the design most commonly known today. In 1923, Scherbius invented the Printing Enigma, where the name Enigma was first used. It functioned like a normal typewriter by printing its encrypted output directly onto a sheet of paper. A year later, a second printing Enigma machine was developed, however it wasn't very popular given a long list of mechanical problems. The reliability issues that coincided with both printing Enigmas made them unsuitable for the German military. To mitigate this, Scherbius returned to his original glow-lamp-based prototype and developed the Enigma A. The Enigma A was housed in a wooden transit case and resembles the later Enigma models, except for the fact that the keys were arranged in alphabetical order, and not in the typical QWERTZ German typewriter order. The Enigma A was later succeeded by both the Enigma B and C, with the latter utilizing three rotors to scramble text.[8]

The glow lamp machines all utilized a reflector that made the machine reciprocal. The reflector allowed for the settings for both encryption and decryption to be identical, drastically reducing complexity of the machine. In 1926, the Enigma was drastically improved with a new chassis and the use of the standard German keyboard layout. It also allowed for the reflector to be set to 26 different positions, making encrypted messages more difficult to crack given that there were more settings to configure. It became the basis for the later known Enigma designs, including a long line of more commercialized designs. This machine was known as the Enigma D to some, but the Enigma A26 as model numbers were introduced. The lifespan of the Enigma D was short-lived as it was later replaced in 1927 by the Enigma K, which had the longest life-span of all machines as it was produced till 1944. It's assumed that the 'K' stands for Kommerziell, or commercial in German, being that it was the main commercial machine. It was sold to a variety of national and foreign customers, including the Swiss Army and Foreign Ministry. Additionally, several wartime variants were derived from this model, including the Japanese Enigma T.[9]

While the commercial Enigma remained in production and use, a series of more complex machines were developed. The commercial machines had a simpler rotor turnover mechanism, the rightmost rotor makes a single step with each key press and once a full revolution has been complete, the middle rotor begins and so on. The more advanced range of machines was known as the Zahlwerk Enigma or counter Enigma, like because it has a counter showing the length of a message. The advanced turnover mechanism sported by this variation is driven by cogwheels that allow the mechanism to be wound back, which is useful for correcting mistakes. The three roots also have seventeen, fifteen, and eleven turnover notches respectively, each being relative primes of 26, which increased the number of steps before a sequence is repeated.[8]

Since the creation of the first Enigma Machine, the German Army had been looking for a machine that could be used in the field and was secure enough to keep messages secret for longer periods of time. In 1926, the development of a machine for the German Army, known as the Enigma I, began. It housed a unique feature, a switch board that significantly increased the strength of the cipher. The first version was ready in 1927 using a single-ended switch board, but after a few iterations, they settled in 1928 for a simplified double-ended plugboard. The plugboard allowed for users to manually pair up letters, meaning that if you plugged 'A' to 'Z', pressing 'A' would always result in a 'Z' and vice versa. The final version

with the double-ended plugboard was put into service in June of 1930 and was exclusively used on military machines. A few other variants of the Enigma I were created for the German Navy, known as the Enigmas M1 through M4.[8]

Despite the innovative design of the Enigma machine, intelligence efforts from multiple countries eventually succeeded in breaking it by exploiting its inherent weaknesses. These efforts began in Poland, which had regained independence after World War I. The nation's expertise in cryptography, developed during and after the war, provided an early advantage in codebreaking at the onset of World War II. Polish cryptographers successfully decrypted German ciphers until the adoption of the Enigma machine, which introduced mechanical encryption far more advanced than manual methods.[1]

Recognizing the challenge posed by this new device, mathematicians from the University of Poznań—Marian Rejewski, Jerzy Różycki, and Henryk Zygalski—leveraged mathematical techniques to crack the Enigma cipher as early as 1932. Poland's breakthroughs were made possible by these mathematicians' ingenuity and their innovative methods, including Różycki's clock method and Zygalski's perforated sheets.[2]

The Różycki clock helped determine, though inconsistently, which of the machine's three rotors was in the rightmost position, the rotor that stepped forward with every keypress. Identifying this rotor reduced the number of possible configurations, significantly narrowing the search space for deciphering messages. The Zygalski sheets, a set of 26 perforated cardboard sheets, were another crucial tool. They represented various rotor scrambling options, enabling cryptographers to deduce Enigma settings more efficiently.[1]

Marian Rejewski's mathematical prowess, particularly his application of group theory, combined with Różycki and Zygalski's contributions and his fluency in the German language, were pivotal to Poland's early success in breaking Enigma. By Christmas 1932, Polish cryptographers were intercepting and decrypting German military communications. This success continued until 1938 when Germany strengthened the Enigma machine, increasing its complexity by introducing additional rotors and procedures.[3]

After the Germans enhanced Enigma's security in 1938, British mathematician Alan Turing became central to the next phase of the effort. Turing joined Bletchley Park, a top-secret British codebreaking center, where he designed the Bombe, an electromechanical device that expedited the decryption of Enigma-encrypted messages. The Bombe worked by comparing sections of ciphertext and plaintext to identify the daily key settings used by German operators. First deployed in 1940, the Bombe allowed Bletchley Park cryptographers to decrypt Enigma messages consistently.[3] A total of 211 Bombe machines were eventually built, and their work provided vital intelligence, giving Allied forces a distinct new advantage. Additionally, many of those at Bletchley Park were able to exploit common occurrences in German messages to speed up this process. Cryptographers used phrases like 'Heil Hitler' and predictable weather reports to narrow down possibilities in tandem with the Enigma Machine.[4] Overall, the eventual cracking of the Enigma Machine was an incredible triumph that spread across countries and disciplines and shaped the course of World War II.[5]

3 Design and Operation

As discussed above, the Enigma machine is an advanced implementation of the rotor machine utilizing three rotors selected from a set of five rotors and a reflector, with some models including an additional plugboard. To configure the Enigma for encryption and decryption, the user needed to select the three rotors to be used, the order in which they were to be inserted into the machine, the plugboard connections to be made, in some models the rotor configurations (which determined where the turnover mechanism would be), and even the reflector's wiring in some of the final models. These all came from a codebook given to operators, and the configurations changed daily. The core of the Engima is the electrical wiring running through the individual spokes of the rotor. Each rotor acts like a Caesar cipher, where a current running through a wire associated with a letter on the input side produces a current associated with a different letter on the output side, such as A to E. The rotating nature of the rotor complicates the encryption further, similar to a Vigenère cipher. However, rather than a fixed pattern like in the Vigenère cipher, Enigma machines' substitutions depended on all of the configurations along with the position in the message. When all of the rotors were combined along with the varying turnover positions, the encryption becomes even more complex.[6]

When encrypting a plaintext message, the operator first sets all of the Engima's features to the desired configuration. The operator then types the first character on the keyboard which starts an electrical signal. If the character pressed is mapped in the plugboard, that signal is switched accordingly. The signal then turns the first rotor and goes through to the associated output spoke. If the turnover mechanism is activated, the next rotor turns before the signal travels to it, and if that turnover mechanism is activated then the final rotor also turns. The signal travels through the rotors and then reaches the reflector. The reflector acts similarly to a rotor with a wire mapping, but when the signal travels through the wire it then travels back through the rotors in reverse. This does not cause any rotor movement. Once the signal has traveled through all of the rotors a second time, a light turns on that corresponds to the encrypted letter. The wiring of the machine ensures that no letter can map to itself. Decryption works the same way, where the input to the machine is the ciphertext.[7]

4 Mathematical Setting

We first state some preliminary results that allow for abstraction of the Enigma machine.

Definition 4.1. *By an* **alphabet***, we mean any finite non-empty set.*

We call an element in an alphabet Σ a *symbol* of Σ, and a (possibly empty) sequence of symbols a *string over* Σ. We denote the set of all possible strings by Σ^*. In particular, the Enigma machine is concerned with encoding strings over the 26-symbol Latin-script alphabet.

Definition 4.2. *A* **permutation** *of an alphabet* Σ *is a bijection from* Σ *to* Σ*.*

Permutations of the Latin-script alphabet are key to Enigma's encryption using a common operation: the usual function composition. We note that for any alphabet Σ, the set of permutations of Σ is a group under composition, and we call this group the *symmetric group* S_Σ. Much of abstract algebra is concerned with the behavior of symmetric groups: in fact, every group is isomorphic to a subgroup of a symmetric group.

It is convenient to have a compact notation for a permutation $\sigma \in S_\Sigma$. One natural way is to consider some $a \in \Sigma$, and then write down $\sigma(a)$, $\sigma(\sigma(a))$, and so forth until "cycling" back to a.

Definition 4.3. *We say that* $(a_0\ a_1\ a_2\ \cdots\ a_{k-1})$ *is a* **cycle** *of a permutation* $\sigma \in S_\Sigma$ *if the entries are pairwise distinct and, for all* i, $a_i \in \Sigma$ *and* $\sigma(a_i) = a_{(i+1) \bmod k}$.

For example, for $\Sigma := \{a, b, c, d\}$, the permutation σ with

$$\sigma(a) = b,\ \sigma(b) = a,\ \sigma(c) = d,\ \sigma(d) = c$$

contains both the cycles $(a\ b)$ and $(c\ d)$. We note that we could also equivalently write $(a\ b)$ as $(b\ a)$. For this reason, it is conventional to impose a strict total order $<$ on an alphabet called a lexicographical order, and then write cycles such that the first symbol is the minimum in $<$ for all symbols in the cycle.

We say two cycles are *disjoint* if no symbol in Σ is included in both cycles: for example, $(a\ b)$ and $(c\ d)$ are disjoint. We note that there is a natural correspondence between a cycle of σ and the permutation in S_Σ that maps every symbol that is not in the cycle to itself. For example, we can consider $(a\ b)$ to be the permutation

$$a \mapsto b,\ b \mapsto a,\ c \mapsto c,\ d \mapsto d.$$

Then, it is sensible to compose cycles, and write that the above σ is the composition $(a\ b) \circ (c\ d)$, or $(a\ b)(c\ d)$ for short. Since disjoint cycles do not "interact", we can compose them in either order. Thus, it is equally valid to write $\sigma = (c\ d)(a\ b)$.

Theorem 4.4. *Every permutation can be written as the composition (or product) of disjoint cycles. Moreover, this composition is unique up to reordering.*

We do not prove this formally, but intuitively, this is true because every symbol is involved in exactly one disjoint cycle, so we can construct the unique cycle decomposition by writing disjoint cycles until each symbol is represented. We note that we usually omit 1-cycles when we write the cycle decomposition.

Definition 4.5. *A* **transposition** *is a 2-cycle (a cycle with two elements).*

We note that every permutation can be written as the product of transpositions, but this product is not unique and the transpositions will not be disjoint (unless the permutation's unique cycle decomposition has cycles only of length 2). We now give one more result.

Proposition 4.6. *Let* $\sigma \in S_\Sigma$ *be a permutation and* $\tau := (a_1\ a_2\ \cdots\ a_k) \in S_\Sigma$ *be a cycle. Then,*

$$\sigma(a_1\ a_2\ \cdots\ a_k)\sigma^{-1} = (\sigma(a_1)\ \sigma(a_2)\ \cdots \sigma(a_k))\,.$$

We leave the proof as an exercise to the reader. We call the $\sigma\tau\sigma^{-1}$ the *conjugation* of τ by σ. Since every permutation has a unique cycle decomposition, the conjugation of any permutation by σ can be described by the conjugation of each of the permutation's cycles by σ.

We are now ready to define Enigma's behavior more formally.

5 Mathematical Description

An Enigma machine has constant settings, which are baked into the physical machine and are not designed to change, and variable settings, which are intended to be changed regularly.

The constant settings of an Enigma machine are:

- Σ, an alphabet with even cardinality m,
- $<$, a strict total order on Σ,
- $n \in \mathbb{N}$, the number of all available rotors,
- $R_1, R_2, \ldots, R_n \in S_\Sigma$, the available rotors,
- $k \in [n]$[1], the number of rotors that can be inserted into the machine at any one time.

The choice of $<$ uniquely determines a lexicographical order of symbols in Σ. Let $a_0 < a_1 < \cdots < a_{m-1}$ be this ordering, and let $\rho := (a_0\ a_1\ \cdots\ a_{m-1})$.

The variable settings of an Enigma machine are:

- $U \in S_\Sigma$, a permutation composed of $m/2$ disjoint transpositions that encodes the reflector (each transposition represents a pair of symbols to connect),
- $P \in S_\Sigma$, a permutation composed of t disjoint transpositions where $0 \leq t \leq m/2$ that encodes the plugboard (in the same way as the reflector encoding),
- $(i_1, i_2, \ldots, i_k) \in [n]^k$, a tuple of k pairwise distinct entries that encodes the indices of the rotors to be inserted into the machine and in what order to do so (for each $j \in [k]$, rotor R_{i_j} should be inserted in the j-th slot from the right),
- $(s_1, s_2, \ldots, s_k) \in \{0, 1, \ldots, m-1\}^k$, a tuple of k entries that encode the initial rotor settings (discussed below).

We can calculate the number of possible variable settings in terms assuming the constant settings are fixed. We will also assume t is fixed, since in practice it is rarely changed from one setting to another.

Theorem 5.1. *There are*

$$\frac{n! \cdot (m!)^2 \cdot m^k}{(n-k)! \cdot (m-2t)! \cdot (m/2)! \cdot t! \cdot 2^{t+m/2}}$$

possible variable settings for a given Enigma machine.

[1]By $[n]$ we mean $\{1, 2, \ldots, n\}$.

Proof. We compute the number of possibilities for each of the four variable settings and then multiply them:

- To choose U, we can arrange the symbols of Σ in any way and then "pair off" the consecutive symbols to be the transpositions. There are $m!$ ways to choose the arrangements. Each of the $m/2$ transpositions is then counted twice since the symbols can appear in either order in the arrangement, so we divide by $2^{m/2}$. Also, the order we select the transpositions does not matter, so we divide by $(m/2)!$. Thus, there are
$$\frac{m!}{2^{m/2} \cdot (m/2)!}$$
choices for this variable.

- To choose P, we can arrange the symbols of Σ in any way, and then choose the first $2t$ symbols to "pair off" to be the transpositions and then map the last $m - 2t$ symbols to themselves. Each transposition is then counted twice since the symbols can appear in either order in the arrangement, so we divide by 2^t. Also, the order we select the transpositions does not matter, so we divide by $t!$. Finally, the order we select the symbols that map to themselves does not matter, so we divide by $(m - 2t)!$. Thus, there are
$$\frac{m!}{2^t \cdot t! \cdot (m - 2t)!}$$
choices for this variable.

- To choose $(i_1, i_2, \ldots, i_k)$, we select k rotors from the n available and preserve order. This is n permute k, so there are $\frac{n!}{(n-k)!}$ choices for this variable.

- To choose $(s_1, s_2, \ldots, s_k)$, we have m choices for each of the k rotors. Thus, there are m^k choices for this variable.

Multiplying these yields the result. □

Enigma machines used by the German military in World War II set Σ as the 26-symbol Latin-script alphabet, $<$ as the typical alphabetical order (which sets $\rho = (a\ b\ c\ \cdots\ y\ z)$), n between 5 and 8 depending on the branch of the military, k as 3, and t (generally) as 10. The available rotors were wired with the creation of the machine. There was then a secret book distributed that had a new U, P, $(i_1, i_2, \ldots, i_k)$ and $(s_1, s_2, \ldots, s_k)$ listed for each day. Using Theorem 5.1 and these standard values with $n = 8$, there were approximately $7.038 \cdot 10^3 3$ variable settings the German military could choose from.

We can now give a formal description of how Enigma encrypts messages. When the settings above are initialized, an Enigma machine with the above parameters will act as the permutation

$$E = P\left[\left(\rho^{s_1} R_{i_1} \rho^{-s_1}\right) \cdots \left(\rho^{s_k} R_{i_k} \rho^{-s_k}\right)\right] U \left[\left(\rho^{s_k} R_{i_k}^{-1} \rho^{-s_k}\right) \cdots \left(\rho^{s_1} R_{i_1}^{-1} \rho^{-s_1}\right)\right] P^{-1} \qquad (\star)$$

on the first symbol it encodes. This product is long, so the reader should pause to convince themselves using the description in Section 3 and Proposition 4.6 that this expression is what we would expect.

Since Enigma is a poly-alphabetic cipher, E permutation changes each time a letter is encrypted. In particular, a non-empty subset of the s_j's change after each encryption. We say the machine in its initial setting is in *state* σ_0, where a state encodes the permutation E depending on the setting of the s_j's. For a state σ_r, we can obtain the state σ_{r+1} that follows by updating the s_j's as follows: set $s_1 := s_1 + 1 \pmod{n}$, and then successively for each $1 < j \leq k$, set

$$s_j := \begin{cases} s_j + 1 \pmod{n} & \text{if } n \mid s_{j-1}, \\ s_j & \text{otherwise.} \end{cases} \qquad (\star\star)$$

Then, for any string $a := a_0 a_1 \cdots a_\ell \in \Sigma^*$, we have

$$E(a) = \sigma_0(a_0)\sigma_1(a_1)\cdots\sigma_\ell(a_\ell),$$

where each σ_r is the permutation described by $(\star)$ with the setting of $(s_1, s_2, \ldots, s_k)$ described by $(\star\star)$.

6 Mathematical Analysis of Decryption

An important aspect of the Enigma Machine was its simple decryption properties: Let σ_0 be an arbitrary initial state of an Enigma Machine and $m = a_0 a_1 \ldots a_n \in \Sigma^*$ be an arbitrary message for encryption. Denote $m' = a'_0\, a'_1 \ \ldots \ a'_n \in \Sigma^*$ as the string resulting from encoding m using a machine beginning in the state σ_0. Then, we claim $m = \sigma_0(a'_0)\, \sigma_1(a'_1) \ \ldots \ \sigma_n(a'_n)$. In other words, by typing in any encrypted message into a machine in the same initial state as the machine originally used for encryption, we will get back our original message.

Furthermore, note $a'_0 = \sigma_0(a_0),\ a'_1 = \sigma_1(a_1),\ \ldots,\ a'_n = \sigma_n(a_n)$. Thus, we can rewrite our above claim as $m = a_0 a_1 \ldots a_n = \sigma_0(\sigma_0(a_0))\, \sigma_1(\sigma_1(a_1)) \ \ldots \ \sigma_n(\sigma_n(a_n))$. Since both the states and string are assumed to be arbitrary, our claim is equivalent to the following:

Claim 1. *For an arbitrary Enigma Machine state σ and $a \in \Sigma$, $a = \sigma(\sigma(a))$. By definition, this means $\sigma^{-1} = \sigma$.*

Proof. Let σ be an arbitrary Enigma Machine state. Like in the above section, we assume the Enigma Machine has 3 rotors, a reflector, and a plugboard allowing for 10 pairs of letters to be formed. From above, the permutation representing the state of such a machine is:

$$\sigma = P\left[\left(\rho^{s_1} R_{i_1} \rho^{-s_1}\right) \cdots \left(\rho^{s_k} R_{i_k} \rho^{-s_k}\right)\right] U \left[\left(\rho^{s_k} R_{i_k}^{-1} \rho^{-s_k}\right) \cdots \left(\rho^{s_1} R_{i_1}^{-1} \rho^{-s_1}\right)\right] P^{-1}$$

We want to show $\sigma^{-1} = \sigma$, or in other words, $\sigma \circ \sigma = id_\Sigma$ where id_Σ is the identity permutation. For ease of writing, denote $N_j = (\rho^{s_j} R_{i_j} \rho^{-s_j})$ and $N = N_1 \cdots N_k$. Note then that $N_j^{-1} = (\rho^{s_j} R_{i_j}^{-1} \rho^{-s_j})$ and $N^{-1} = N_k^{-1} \cdots N_1^{-1}$.Then:

$$\begin{aligned} \sigma \circ \sigma &= PNUN^{-1}P^{-1}PNUN^{-1}P^{-1} \\ &= PNUN^{-1}NUN^{-1}P^{-1} \\ &= PNUUN^{-1}P^{-1} \end{aligned} \qquad (1)$$

U is a permutation composed of disjoint transpositions. Because disjoint cycles commute and transpositions are their own inverses, $U \circ U = id_\Sigma$. Thus:

$$\begin{aligned} PNUUN^{-1}P^{-1} &= PNN^{-1}P^{-1} \\ &= PP^{-1} \\ &= id_\Sigma \end{aligned} \tag{2}$$

Therefore, $\sigma \circ \sigma = id_\Sigma$ meaning $\sigma^{-1} = \sigma$. □

This shows that the Enigma Machine's decryption mechanism is well-defined for any state.

7 Background Reading

- Polish contributions to cracking the Enigma[1]
- Jerzy Różycki[2]
- Alan Turing[3]
- Bombe Construction[4]
- Bletchley Park[5]
- Enigma operation [6]
- Enigma design[7]
- Brilliant.org article
- Explanatory paper
- Quora post about the math

8 References

1. The British Library. "Polish Mathematicians and Cracking the Enigma." *European Studies Blog*. January 18, 2018. https://blogs.bl.uk/european/2018/01/polish-mathematicians-and-cracking-the-enigma.html#:~:text=In%201932%20a%20team%20of,code%2C%20in%20only%20ten%20weeks.

2. National Cryptologic Museum Foundation. "1942: Jerzy Różycki, Polish Cipher Bureau Mathematician, Died." *This Day in History Calendar*. January 9, 2023. https://cryptologicfoundation.org/community/bytes/this_day_in_history_calendar.html/event/2023/01/09/1673240400/1942-jerzy-r-ycki-polish-cipher-bureau-mathematician-died-.

3. Imperial War Museum. "How Alan Turing Cracked the Enigma Code." *IWM*. https://www.iwm.org.uk/history/how-alan-turing-cracked-the-enigma-code.

4. The National Museum of Computing. "Bombe." *TNMOC*. https://tnmoc.org/bombe.

5. Bletchley Park. "Our Story." *Bletchley Park*. https://bletchleypark.org.uk/our-story/.

6. Graham Ellsbury. "How the Enigma was Set Up and Operated" *The Enigma Machine* http://www.ellsbury.com/enigma3.htm.

7. Stanford University. "The Enigma Machine" *Stanford University CS 106J*. https://web.stanford.edu/class/cs106j/handouts/36-TheEnigmaMachine.pdf.

8. Crypto Museum. "History of the Enigma." *Enigma history.* https://www.cryptomuseum.com/crypto/enigma/hist.htm.

9. Crypto Musuem. "Enigma K." https://www.cryptomuseum.com/crypto/enigma/k/.

10. "Who First Cracked the ENIGMA Cipher?" Central Intelligence Agency. https://www.cia.gov/stories/story/who-first-cracked-the-enigma-cipher/.