CSC B58: Enigma Breakdown

James Camano

March 12, 2019

(**Last Edited - March 12, 2019**)

- improved Rotor definition

This document sets out to describe the components of the Enigma machine.

1 Introduction

Project Enigma sets out to imitate both the German *Enigma* text cipher machine, and its mechanical counterpart - the Bombe machine.

This imitation of Enigma (which will henceforth be called the same name) creates a cipher of a character input by starting off with an initial set state, performing *alphabet shift arithmetic* to the character input based on that state and then 'advancing' the state. Finally, this shifted input is returned as output.

2 Components

To describe the components of Enigma, we start with a few definitions.

Def$^{\mathbf{n}}$: Rotor - In an Enigma machine, a Rotor R can be described as an ordered pair $R = (n, m), n \in \mathbb{N}, m \in \{i\}_{i=0}^{25}$. Where the value n describes the rotor with relation to the other rotors in the same machine, and m represents the value of the rotor. For example, if $R = (3, 15)$ then we say that R is the third rotor, and its value is 15.

Enigma consists of:

1. A set of rotors $\{R^i\}_{i=1}^{n}$ [1].

3 Encryption Algorithm

Define:

- The alphabet $\Sigma = \{\bar{a} : \bar{a}$ is an uppercase character in the English alphabet$\}$
- R_l to be a rotor with setting n. That is, $R_l = (n, m)$.

[1] Currently, $n = 1$.

- $\varphi_k \in \Sigma$ to be the k^{th} letter in the alphabet. (i.e. $\varphi_1 = B$)
- $g(R_l) = \begin{cases} (n, m+1), & \text{if } m+1 \leq 25 \\ (n, 0), & \text{if } m+1 > 25 \end{cases}$
- $f(\varphi_k, R_l) = \begin{cases} \varphi_{k+m}, & \text{if } k+m \leq 25 \\ \varphi_{k+m-26}, & \text{if } k+m > 25 \end{cases}$

Then, the encryption algorithm is as follows:

1. $\omega := f(\varphi_k, R_n)$
2. $R_k := g(R_k)$

Where φ_k is assumed to be the *input* letter, and ω is the corresponding output of the Enigma machine.

In words, given the input letter φ_k, Enigma shfits φ_k the rotor value of R_l, subsequently incrementing R_l by one and then outputting the shifted letter ω.

4 Decryption Algorithm

Let the settings of the Enigma machine be similar to that in the encryption algorithm (i.e. If the starting position was R_3, then set the starting position to R_3.) Then, define:

- $\hat{f} : (\Sigma \times R) \mapsto \Sigma$
- $\hat{f}(\varphi_k, R_n) = \begin{cases} \varphi_{k-n}, \text{ if } k-n \geq 0 \\ \varphi_{k-n+26}, \text{ if } k+n < 0 \end{cases}$

Then, the decryption algorithm is:

1. $\omega := \hat{f}(\varphi_k, R_n)$
2. $R_n := g(R_n)$

Where φ_k is assumed to be the *encrypted* letter, and ω is the corresponding original input of the Enigma machine (but ω is what is outputted). This method is intuitively straightforward since the subscript subtraction represents a shift of the same magnitude in the opposite direction.

5 Input Restrictions on Enigma

These restrictions are applied for the convenience of the Bombe machine.

1. Every input sequence to be encoded must start with the sequence $\langle A, B, C \rangle$. This is so that there is a definite (easy) flag for which the Bombe machine can deduce a contradiction.

6 Additional Notes

- What will the output look like if we introduce a second rotor? A third?
  - Shift terms will be of the form $x_1 + x_2 + r_1 + r_2$, where x_1, x_2 are the initial positions; r_1, r_2 are the number of shifts $\bmod 26$ of the first and second rotor - respectively.