# CSC B58: Enigma Breakdown

James Camano

March 11, 2019

(**Last Edited - March 11, 2019**)

- typo fixes

- edited intro

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

Enigma consists of:

1. A set of rotors $\{R^i\}_{i=1}^{n}$, whose values cycle from $0 - 25$ [1].

# 3 Encryption Algorithm

Define:

- The alphabet $\Sigma = \{\bar{a} : \bar{a} \text{ is an uppercase character in the English alphabet}\}$

- $R_n$ to be a rotor with setting $n$. That is, $R_n$'s value is $n$

- $\varphi_k \in \Sigma$ to be the $k^{\text{th}}$ letter in the alphabet. (i.e. $\varphi_1 = b$ )

- $g(R_n) = \begin{cases} R_{n+1}, \text{ if } n+1 \leq 25 \\ R_0, \text{ if } n+1 > 25 \end{cases}$

---

[1]Currently, $n = 1$.

- $f(\varphi_k, R_n) = \begin{cases} \varphi_{k+n}, & \text{if } k+n \le 25 \\ \varphi_{k+n-26}, & \text{if } k+n > 25 \end{cases}$

Then, the encryption algorithm is as follows:

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

2. $R_n := g(R_n)$

Where $\varphi_k$ is assumed to be the *input* letter, and $\omega$ is the corresponding output of the Enigma machine.

In words, given the input letter $\varphi_k$, Enigma shfits $\varphi_k$ the rotor position of $R_n$, subsequently incrementing $R_n$ by one and then outputting the shifted letter $\omega$.

# 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:

- $\hat{f} : (\Sigma \times R) \mapsto \Sigma$

- $\hat{f}(\varphi_k, R_n) = \begin{cases} \varphi_{k-n}, & \text{if } k-n \ge 0 \\ \varphi_{k-n+26}, & \text{if } k+n < 0 \end{cases}$

Then, the decryption algorithm is:

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

2. $R_n := g(R_n)$

Where $\varphi_k$ is assumed to be the *encrypted* letter, and $\omega$ is the corresponding original input of the Enigma machine (but $\omega$ 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 $\mod 26$ of the first and second rotor - respectively.