Progetto DSP-2018/19: Implementazione di un simulatore di macchina Enigma in Python

Matteo Ghera
matteo.ghera@stud.unifi.it

Abstract

La macchina Enigma fu una macchina elettromeccanica per cifrare e decifrare i messaggi utilizzata dall'esercito tedesco durante la II guerra mondiale. In questo articolo presento alcuni risultati matematici e una possibile implementazione di un simulatore di macchina Enigma in Python.

1. Introduzione

La macchina Enigma consentiva di cifrare e decifrare messaggi. Nel corso della guerra vennero prodotte differenti versioni della macchina Enigma e la decrittazione dei messaggi cifrati fu molto importante per le sorti della guerra fornendo molte informazioni agli Alleati. Diversi storici sostengono che le informazioni ottenute dalla decrittazione dei messaggi tedeschi consentirono agli Alleati di abbreviare la guerra di due anni [4].

1.1. Struttura

La macchina Enigma può essere considerata come un estensione del metodo del cifrario di Vigenère munita di dischi cifrati. Le differenze principali sono la presenza di più dischi cifranti connessi a cascata e la mancanza di una chiave che invece era un elemento essenziale nella cifratura di Vigenère [1].

Enigma aveva un aspetto simile ad una macchina da scrivere e presentava due tastiere: una nella parte inferiore e una nella parte superiore. L'operatore digitava il testo in chiaro o quello cifrato sulla tastiera nella parte inferiore. La tastiera nella parte superiore era sostituita con un pannello luminoso sul quale compariva la lettera associata a quella premuta sulla tastiera nella parte inferiore. In Figura 1 vi è la foto di una macchina Enigma.

Figura 1. Foto di una macchina Enigma esposta presso il Museo nazionale della scienza e della tecnologia "L. Da Vinci" di Milano.

Nel momento in cui l'operatore premeva un tasto sulla tastiera inferiore (*keyboard*), la tensione attraversava i circuiti della macchina Enigma fino ad arrivare al lato destro del primo rotore e, da questo, seguiva la connessione sul lato sinistro per arrivare, quindi, sul lato destro del secondo rotore. Il lato sinistro del terzo e ultimo rotore

era collegato ad un *riflettore* che realizzava un ulteriore scambio di lettera e la tensione seguiva il percorso a ritroso passando per i lati sinistri dei rotori. Sulla parte frontale della macchina Enigma vi era posto un pannello a più prese, l'operatore inserendo alcuni spinotti poteva realizzare lo scambio tra alcune coppie di lettere (tipicamente 10) prima che la tensione raggiungesse i rotori. La Figura 2 schematizza il funzionamento di una macchina Enigma.

Gli elementi elettromeccanici fondamentali che componevano una macchina Enigma erano quindi i seguenti:

1. una tastiera (*keyboard*), che veniva usata per cifrare o decifrare un messaggio;
2. un pannello luminoso (*lampboard*), che visualizzava la lettera codificata. La sequenza di lettere illuminate costituiva il testo in chiaro o il testo cifrato;
3. tre *rotori* connessi a cascata, dove ciascun rotore realizzava 26 possibili scambi di lettere ed implementava una possibile permutazione scelta tra un set di possibili permutazioni. Il primo rotore (rotore destro) ruotava di una posizione verso l'alto ogni volta che veniva premuto un tasto sulla keyboard, il secondo rotore (rotore centrale) ruotava di una posizione verso l'alto ogni 26^2 tasti premuti mentre il terzo rotore (rotore sinistro) ruotava ogni 26^3 tasti premuti. Poiché la permutazione realizzata cambiava ad ogni lettera, Enigma è un cifrario polialfabetico;
4. un *riflettore*, che realizzava uno scambio fissato tra 13 coppie di lettere. Grazie al riflettore la macchina poteva così funzionare anche come decodificatore, senza intervento specifico alcuno; cioè era necessario, prima di cominciare la decodifica, portare solo rotori e spinotti nella configurazione giornaliera prevista dai cifrari;
5. un pannello a più prese (*plugboard*), che veniva usato per eseguire uno scambio tra alcune coppie di lettere.

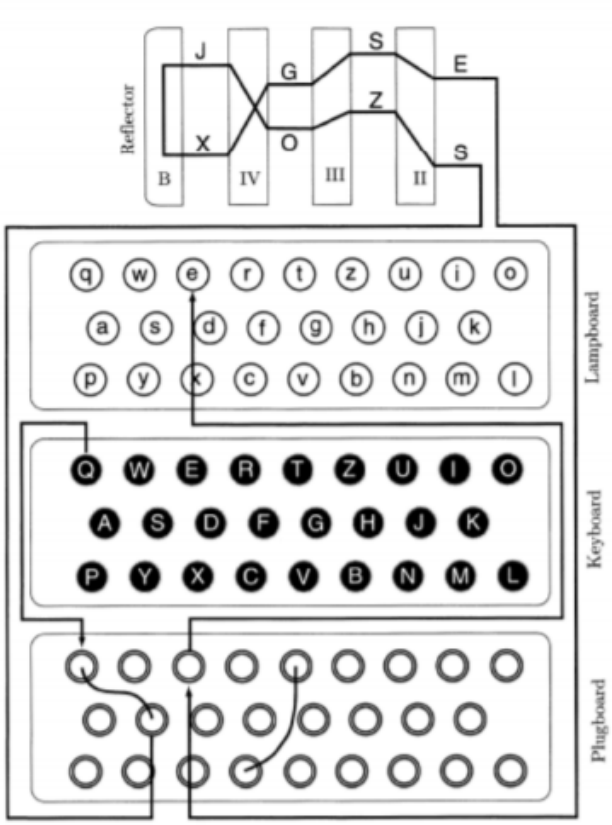

Figura 2. Schema che rappresenta il funzionamento di una macchina Enigma tratto da [6]

2. Analisi matematica

Si consideri una generica macchina Enigma, in ogni istante la disposizione dei tre rotori costituisce lo *stato* della macchina. Come riportato in [3], una *chiave* Enigma consiste dei seguenti elementi:

1. una tripla (R_1, R_2, R_3) con $R_1, R_2, R_3 \in S$, dove S è un insieme fissato di rotori e ciascuno dei quali realizza una permutazione nota;
2. una tripla $(i, j, k) \in \mathbb{Z}_{26}^3$, che specifica la posizione iniziale di ciascuno dei tre rotori con 26 posizioni possibili per ciascun rotore;
3. una permutazione σ dell'alfabeto che specifica le connessioni della plugboard e sia un'involuzione con esattamente sei punti fissi (cioè esistono esattamente sei lettere x tali che $\sigma(x) = x$).

I primi due elementi costituiscono lo *stato iniziale* della macchina.

2.1. Numero di possibili chiavi

I tre rotori possono essere scelti da un insieme di cinque rotori quindi il numero di possibili configurazioni dei rotori è $5 \cdot 4 \cdot 3$. La tripla che definisce la configurazione iniziale dei rotori è formata tre lettere scelte in un alfabeto di 26 lettere; quindi ci sono 26^2 possibili triple. Il numero di possibili permutazioni dell'alfabeto che possono essere implementante con la plugboard è ottenuto osservando che:

- visto che ci sono sei punti fissi, possono essere scambiate solo 20 lettere con altre 20;
- il numero di possibili modi in cui scegliere le 10 lettere da scambiare è dato da:

$$\frac{20!}{(20-10)!},$$

tale numero deve essere diviso per il numero di ripetizioni che si sono venute a creare, cioè 2^{10}, mentre il numero di possibili connessioni sulla plugboard è dato da:

$$\binom{26}{20} = \frac{26!}{20! \cdot 6!};$$

- il numero di possibili permutazioni è dato dunque da:

$$\frac{26!}{(6! \cdot 10! \cdot 2^{10})};$$

.

Il numero di possibili chiavi è quindi dato da:

$$(5 \cdot 4 \cdot 3) \cdot 2^{10} \cdot \frac{26!}{(6! \cdot 10! \cdot 2^{10})} \approx 10^{21} \text{ oppure } \approx 2^{67}.$$

2.2. Funzioni di Encryption e Decryption

Siano α, β e γ le trasformazioni applicate alle lettere dal rotore sinistro, dal rotore centrale e dal rotore destro e siano α^{-1}, β^{-1} e γ^{-1} le trasformazioni all'indietro applicate alle lettere dai medesimi rotori. Sia π la trasformazione applicata alle lettere del riflettore e sia σ quella della plugboard. Come riportato in [5], data la lettera x, nella fase iniziale la funzione di encryption E si può definire come:

$$E(x) = \sigma^{-1} \circ \tilde{\tau} \circ \sigma(x)$$

dove

$$\tilde{\tau}(x) = \alpha^{-1} \circ \beta^{-1} \circ \gamma^{-1} \circ \pi \circ \gamma \circ \beta \circ \alpha(x)$$

. Supponiamo di indicare la rotazione di un generico rotore r di n posizioni nel seguente modo: $\mu^{-n} R \mu^n$, dove μ indica una permutazione ciclica che mappa A in B, B in C, ecc. Si ottiene la funzione di encryption generica E per una macchina Enigma sostituendo $\tilde{\tau}(x)$ con

$$\tau(x) = \left(\mu^{-n}\alpha^{-1}\mu^n\right) \circ \left(\mu^{-n}\beta^{-1}\mu^n\right) \circ \left(\mu^{-n}\gamma^{-1}\mu^n\right) \circ \pi \circ \left(\mu^{-n}\gamma\mu^n\right) \circ \left(\mu^{-n}\beta\mu^n\right) \circ \left(\mu^{-n}\alpha\mu^n\right)(x)$$

La funzione di decription è analoga.

Consideriamo la permutazione

$$\rho = \sigma^{-1} \circ \tilde{\tau} \circ \sigma$$

dove

$$\tilde{\tau} = \alpha^{-1} \circ \beta^{-1} \circ \gamma^{-1} \circ \pi \circ \gamma \circ \beta \circ \alpha$$

realizzata dalla macchina. Si osserva che σ e π sono involuzioni (ad esempio $\sigma\big(\sigma(x)\big) = x \; \forall x$, dunque $\sigma^{-1} = \sigma$), σ ha esattamente sei punti fissi (cioè esistono solo sei lettere x per cui si ha $\sigma(x) = x$) e π non ha punti fissi. Si dimostra la seguente proposizione:

Proposizione 1. *ρ è un'involuzione e non ha punti fissi*

Dimostrazione. Per dimostrare che ρ è un'involuzione bisogna dimostrare che $\tilde{\tau}$ lo è:

$$\tilde{\tau} \circ \tilde{\tau} = \alpha^{-1} \circ \beta^{-1} \circ \gamma^{-1} \circ \pi \circ \gamma \circ \beta \circ \alpha \circ \alpha^{-1} \circ \beta^{-1} \circ \gamma^{-1} \circ \pi \circ \gamma \circ \beta \circ \alpha = \alpha^{-1} \circ \beta^{-1} \circ \gamma^{-1} \circ \underbrace{\pi \circ \pi}_{=I} \circ \gamma \circ \beta \circ \alpha = I$$

Quindi per ρ si ottiene:

$$\rho \circ \rho = \sigma^{-1} \circ \tilde{\tau} \circ \sigma \circ \sigma^{-1} \circ \tilde{\tau} \circ \sigma = I.$$

Posizione rotori dentro la macchina				
Riflettore	*Rotore sinistro*	*Rotore al centro*	*Rotore destro*	*Iniziale*
H R	F T	E L	I Y	Q
U T	M Z	Y Q	J X	W
G Z	Y U	U W	U C	E
T U	Z I	F E	H V	R
E I	E O	H R	X B	T
V O	Q A	X T	Z N	Z
J A	D S	Z Z	M M	U
C S	L D	M U	N L	I
L D	B F	N I	A Q	O
X F	C G	J O	V W	A
Z G	K H	G A	O E	S
R H	J J	O S	E R	D
A J	G K	P D	Y T	F
Q K	V P	A F	F Z	G
N P	P Y	Q G	W U	H
B Y	U X	I H	L I	J
F X	R C	R J	D O	K
S C	W V	L K	Q A	P
O V	N B	D P	C S	Y
Y B	X N	T Y	B D	X
P N	A M	W X	S F	C
W M	H L	V C	P G	V
D L	S Q	K V	T H	B
K Q	T W	S B	K J	N
M W	I E	B N	R K	M
I E	O R	C M	G P	L

Figura 3. Tabella che rappresenta una delle possibili permutazioni dei rotori e del riflettore. Questa configurazione sarà quella iniziale del simulatore da me implementato.

Posizione rotori dentro la macchina				
Riflettore	*Rotore sinistro*	*Rotore al centro*	*Rotore destro*	*Iniziale*
T U	E O	E L---	---I Y	Q
E I	Q A	Y Q	J X	W
V O	D S	U W	U C	E
---J A---	---L D	F E	H V	R
C S	B F	H R	X B	T
L D	C G	X T	Z N	Z
X F	K H	Z Z	M M	U
Z G	J J	M U	N L	I
R H	G K	N I	A Q	O
---A J---	---V P	J O	V W	A
Q K	P Y	G A---	---O E	S
N P	U X	O S	E R	D
B Y	R C	P D	Y T	F
F X	W V---	---A F	F Z	G
S C	N B	Q G	W U	H
O V	X N	I H	L I---	---J---
Y B	A M	R J	D O---	---K---
P N	H L---	---L K	Q A	P
W M	S Q	D P	C S	Y
D L	T W	T Y	B D	X
K Q	I E	W X	S F	C
M W	O R	V C	P G	V
I E	F T	K V	T H	B
H R	M Z	S B	K J	N
U T	Y U	B N	R K	M
G Z	Z I	C M	G P	L

Figura 4. La tabella mostra le lettere coinvolte nella codifica con configurazione UOLY

Allora anche ρ è un'involuzione.

Per dimostrare che ρ non ha punti fissi, si supponga per assurdo che ρ abbia almeno un punto fisso. Consideriamo le lettere in entrata nei rotori e nel riflettore, siano x_1, x_2, x_3, x_4 le lettere che entrano dal lato destro e siano $x_{-4}, x_{-3}, x_{-2}, x_{-1}$ quelle che entrano dal lato sinistro, la presenza di un punto fisso implica che esiste una lettera per cui si ottiene

$$x_1 = x_2 = x_3 = x_4 = x_{-1} = x_{-2} = x_{-3} = x_{-4}.$$

Infatti, per ciascun rotore, esiste solo un collegamento che collega la lettera di entrata con la lettera di uscita e tutto ciò genera un assurdo visto che il riflettore non ha punti fissi. □

Il fatto che ρ non abbia punti fissi risultò un punto debole di Enigma e permise agli Alleati di decodificare i messaggi segreti tedeschi durante la guerra.

3. Implementazione in Python

Si prenda come riferimento il notebook Jupyter che si trova all'indirizzo Progetto DSP Matteo Ghera.

Come descritto in [2], supponiamo di aver fissato una delle permutazioni dei tre rotori e del riflettore, è possibile rappresentare tale configurazione in una tabella come quella riportata in Figura 3

Per ciascun rotore, la lista di destra rappresenta la traformazione eseguita dal rotore mentre la lista di sinistra indica il circuito di uscita. Supponiamo di premere il tasto K sulla tastiera. Nella configurazione UOLY, la tensione attraversando il circuito arriverà alla lettera O del rotore destro; qui attraverserà la connessione che la collega alla lettera O sul lato opposto per proseguire quindi verso la lettera A del rotore centrale. Il procedimento è analogo fino a quando non si raggiuge il riflettore. Il riflettore riceverà la corrente attraverso il circuito relativo alla lettera J e seguendo il collegamento sul lato opposto raggiungerà il rotore sinistro alla lettera L. Il procedimento si ripete a ritroso e sulla lampboard si illuminerà la lettera J. La Figura 4 evidenzia le lettere coinvolte nella codifica.

Utilizzando l'osservazione precedente ho provveduto ad implementare la classe Python *EnigmaRotor* che descrive il comportamento di un generico rotore. Il costruttore di tale classe riceve in ingresso: due liste chiamate *Entrata* e *Uscita*, che rappresentano la permutazione scelta per il rotore, il rotore successivo (quello a cui sarà inviato il segnale di rotazione una volta completato il giro) e due flag che indicano rispettivamente se il rotore corrente ha un successivo e se è ruotabile. Il metodo *impostaPosizione* imposta la configurazione iniziale del rotore. Data la posizione nel vettore *Entrata*, il metodo *posizioneSinistra* cerca la lettera corrispondente nel vettore *Uscita* e restituisce la sua posizione. Per cercare la posizione nel vettore *Entrata* relativa alla lettera del vettore *Uscita* con posizione data si usa il metodo *posizioneDestra*. Qualora fosse consentito, il metodo *muovi* muove il rotore dato di una posizione verso l'alto e invia un segnale al vettore successivo, se esiste.

La classe Python *EnigmaMachine* simula il comportamento di una macchina Enigma. Il costruttore di tale classe riceve in ingresso quattro oggetti di tipo *EnigmaRotor* (*riflettore*, *rotoreSinistro*, *rotoreCentrale* e *rotoreDestro*) e due liste (*alfabeto* e *plugboard*). Si osservi che il riflettore è un tipo particolare di rotore che non ruota e che il rotore sinistro non ha alcun successore. L'alfabeto in input è quello riportato nella colonna "Iniziale" delle tabelle precedenti, conosciuto come *QWERTZ*. Questa lista rappresenta la sequenza di lettere che si ottiene leggendo la tastiera della macchina Enigma da sinistra verso destra, dall'alto verso il basso. Per tutto il progetto utilizzeremo questa alfabeto e non quello convenzionale. La lista *plugboard* è una sequenza di numeri da 0 a 25; supponiamo quindi di premere il tasto i sulla tastiera allora la lettera realmente codificata sarà data da $plugboard[i]$. In questo modo sarà possibile eseguire lo scambio fissato indicato da ciascuna coppia di lettere.

Il metodo *impostaEnigma* sarà utilizzato per impostare le posizioni dei rotori e del riflettore mentre il metodo *esegui* cripta/decripta il

messaggio in input.

Per creare un'istanza del simulatore di macchina Enigma *EnigmaMachine* con la permutazione dei rotori e del riflettore descritta in Figura 3 e con configurazione iniziale UOLY è sufficiente scrivere il seguente codice Python:

```
#plugboard
plugboard=[0,1,2,3,4, ...,26]

#alfabeto
alfabeto=['Q', 'W', 'E', 'R', 'T', ..., 'L']

#riflettore
entrata=['R', 'T', 'Z', 'U', ..., 'E']
uscita= ['H', 'U', 'G', 'T', ..., 'I']
riflettore = main.EnigmaRotor.
         .EnigmaRotor(entrata, uscita,
                         None, False)

#rotore 3
entrata=['T', 'Z', 'U', 'I', ..., 'R']
uscita= ['F', 'M', 'Y', 'Z', ..., 'O']
rotoreSinistro = main.EnigmaRotor.
           .EnigmaRotor(entrata, uscita)

# rotore 2
entrata =['L', 'Q', 'W', 'E', ..., 'M']
uscita = ['E', 'Y', 'U', 'F', ..., 'C']
rotoreCentrale = main.EnigmaRotor.
          .EnigmaRotor(entrata, uscita,
                       rotoreSinistro)

# rotore 1
entrata = ['Y', 'X', 'C', 'V', ..., 'P']
uscita = ['I', 'J', 'U', 'H', ..., 'G']
rotoreDestro=main.EnigmaRotor.
          .EnigmaRotor(entrata, uscita,
                       rotoreCentrale)

enigma=main.EnigmaMachine.
  .EnigmaMachine(riflettore, rotoreSinistro,
              rotoreCentrale, rotoreDestro,
                        alfabeto, plugboard)
enigma.impostaEnigma('U','O','L','Y')
```

Se ignoriamo per un momento la plugboard, codificando K si ottiene, come avevamo detto precedentemente, J:

```
In [7]:   enigma.esegui('K')
Out[7]:   'J'
```

Possiamo verificare che se si imposta la macchina Enigma con la stessa configurazione iniziale, se si decripta 'J' si ottiene 'K'. Infatti abbiamo:

```
In [8]:   enigma.impostaEnigma('U','O',
                                 'L','Y')
          enigma.esegui('K')
Out[8]:   'J'
```

Se proviamo a criptare il seguente frammento di plaintext WETTERVORHERSAGEBISKAYA (previsioni del tempo Biscaglia) con l'attuale configurazione della macchina si ottiene:

```
In [9]:   enigma.impostaEnigma('U','O',
                                 'L','Y')
          enigma.esegui('WETTERVORHERSAGEBISKAYA')
Out[9]:   'YGSLMGTJHUDDJTJVTYBSFTR'
```

Verificando il risultato precedente si ottiene:

```
In [10]:   enigma.impostaEnigma('U','O',
                                 'L','Y')
           enigma.
            .esegui('YGSLMGTJHUDDJTJVTYBSFTR')
Out[10]:   'WETTERVORHERSAGEBISKAYA'
```

Supponiamo di voler utilizzare la plugboard per trasformare la lettera A nella lettera K e viceversa. Configuriamo di nuovo la macchina Enigma: nel nostro alfabeto la lettera A è in nona posizione mentre la lettera K è sedicesima, quindi si imposta la nona posizione della lista plugboard a 16 e la sedicesima a 9. Si può verificare che la codifica della lettera K coincide con quella della lettera A trovata con la configurazione precedente:

```
In [11]:   enigma.impostaEnigma('U','O',
                                 'L','Y')
           enigma.esegui('A')
Out[11]:   'H'

In [13]:   enigma.impostaEnigma('U','O',
                                 'L','Y')
           enigma.esegui('K')
Out[13]:   'H'
```

Inoltre, usando la stessa configurazione iniziale, la decodifica di H da K:

```
In [13]:   enigma.impostaEnigma('U','O',
                                  'L','Y')
           enigma.esegui('H')
Out[13]:   'K'
```

4. Conclusioni

È stata presentata la macchina Enigma descrivendone la struttura, le principali proprietà matematiche e una possibile implementazione di un suo simulatore. Si può affermare che la macchina Enigma fu uno strumento rivoluzionario per l'epoca e aprì la strada alla crittografia ma si dimostrò anche essere vulnerabile ad attacchi di tipo known plaintext in cui si sfruttava il fatto che la macchina codificasse ogni lettera in una lettera differente. La quantità di informazioni che veniva da Enigma era tale da modificare le sorti del conflitto e aiutò gli Alleati a vincere la guerra.

Riferimenti bibliografici

[1] Enigma (crittografia), 2019. https://it.wikipedia.org/wiki/Enigma_(crittografia).

[2] Una macchina enigma virtuale, 2019. http://http://www.crittologia.eu/critto/js/enigma.html.

[3] M. Boreale. Corso di data security e privacy, esercizi su crittografia a chiave condivisa, 2018. https://e-l.unifi.it/pluginfile.php/780653/course/section/87266/Esercizi

[4] S. Keegan, John. Intelligence in warfare. *New York: Alfred A. Knopf.*, 2003.

[5] M. Rejewski. An application of the theory of permutations in breaking the enigma cipher. *Applicationes Mathematicae.*

[6] D. Salomon. *Data Privacy and Security*. Springer, 2003.