Symulacja w Badaniach i Rozwoju
Vol. 6, No. 1/2015

Anna BOROWSKA
Politechnika Białostocka, Wydział Informatyki
ul. Wiejska 45A, 15-351 Białystok
E-mail: a.borowska@pb.edu.pl

The Cryptanalysis of the Enigma Cipher. The Catalogue Method. Part I

1 Introduction

The M3 Enigma machine was a portable electro-mechanical rotor encrypting machine used during World War II, mainly by the German military and government services. Beginning in 1932, Polish mathematicians (M. Rejewski, J. Różycki, and H. Zygalski) systematically worked on decoding ciphers, constantly modified manners of generating secret messages, and modernized the construction of Enigma machines.

The catalogue algorithm given below can be used to decode messages eavesdropped before September 15, 1938, because that day the German service changed the manner of announcing *message settings*. The algorithm is a reconstruction and a completion of the catalogue method (invented by Rejewski). Historians described the idea of the method and omitted details. The author tested the behavior of the cipher in order to design the proper algorithm and caught cases of keys for which the method does not return results. The catalogue method was used for guessing the order of drums and the plugboard connections and for decoding the message settings. These elements of the key are not sufficient enough to read messages. Therefore, in part II of this paper the author provides her own algorithm which returns the ring settings and the initial drum settings and additionally, the new plugboard algorithm which relies on Rejewski's idea, but its technical solution is the author's proposal. The paper contains mathematical justifications of some facts.

The German service used different kinds of Enigma machines. We are only interested in the M3 Enigma machine. For the reader's convenience, we described the construction of this machine and the manner of generating messages transmitted until September 15, 1938 in the Appendix. This section makes up a brief survey of well-known information taken from publications [5, 13, 6, 8, 2, 3, 9]. We suggest reading the Appendix first for better understanding of the terms and facts that we use. These terms are denoted in this paper by *. In section 2 we provide some mathematical facts concerning permutations which are essential for understanding the presented method. In sections 3 and 4 the reader can find a mathematical analysis of the M3 Enigma machine and explanations concerning *characteristics*. Section 5 contains a description and a mathematical justification (author's ideas) of the correctness of the catalogue algorithm. In part II we present the new plugboard algorithm and the ring settings algorithm (author's idea). By means of these three algorithms, we can generate the complete *daily key** and read the *message settings** on the basis of a given set of messages intercepted before September 15, 1938. This allows us to read these messages. Section 7 contains the cases of daily keys for which the method does not return any result. We enclose an implementation of given algorithms in Cpp language.

2 Elements of permutation theory

We shall remind the reader some definitions and facts from permutation theory (cf. [10], [12], [15], [14]). If $X = \{1, 2, \ldots, n\}$ is an n-element set, then a bijection from X to itself ($f : X \to X$) is called a *permutation* of X. We denote the set of all permutations of X by S_n. Any permutation $f \in S_n$ will be represented by $2 \times n$ matrix. We determine a *product* (a *composition*) of two permutations $f, g \in S_n$ in the same way as M. Rejewski in [13] (from left to right), i.e.

$$f \circ g = \begin{pmatrix} 1 & \ldots & n \\ f(1) & \ldots & f(n) \end{pmatrix} \circ \begin{pmatrix} 1 & \ldots & n \\ g(1) & \ldots & g(n) \end{pmatrix} = \begin{pmatrix} 1 & \ldots & n \\ g(f(1)) & \ldots & g(f(n)) \end{pmatrix}.$$

This notation is different from standard notation, but it is used by authors interested in the Enigma machine. We usually omit the circle ° and write fg for the composite map. The multiplication of permutations is associative but is not necessarily commutative. The *identity* permutation ε has the property that $\varepsilon f = f\varepsilon = f$ for all $f \in S_n$, that is $\varepsilon(i) = i$ for all $i = 1, 2, \ldots, n$. We define the *inverse* of the permutation $f \in S_n$ (denoted by f^{-1}) as the permutation of the same degree such that $ff^{-1} = f^{-1}f = \varepsilon$.

A k-cycle (a *cycle of length k*) is a permutation (denoted by $(a_1, a_2, \ldots, a_k)$) which maps distinct elements $a_1, a_2, \ldots, a_k$ of the domain $X = \{1, 2, \ldots, n\}$ by sending a_1 to a_2, a_2 to a_3, …, a_k to a_1, and leaving remaining elements fixed. The number of symbols transferred by a cycle is called the *cycle length*. Every permutation can be written as a product of disjoint cycles (i.e. in *cycle* notation). A *transposition* is a cycle that swaps two elements. We call a product of n disjoint transpositions an *n-involution*.

3 Mathematical model

The cryptosystem of the M3 Enigma machine will be defined as a tuple (**P**, **C**, **K**, **E**, **D**), where **P** = {A, B, …, Z} is the *plaintext space*, **C** = **P** is the *ciphertext space* and **K** is a set of all possible *keys*.

E = $\{\Lambda_k : k \in \mathbf{K}\}$ is a set of *encryption functions* $\Lambda_k : \mathbf{P} \to \mathbf{C}$.

D = $\{\Delta_k : k \in \mathbf{K}\}$ is a set of *decryption functions* $\Delta_k : \mathbf{C} \to \mathbf{P}$.

Each letter $a \in \mathbf{P}$ is transformed according to the following permutation (cf. [13])

$$\Lambda_k = \Lambda = SHQ^z RQ^{-z} Q^y MQ^{-y} Q^x LQ^{-x} BQ^x L^{-1} Q^{-x} Q^y M^{-1} Q^{-y} Q^z R^{-1} Q^{-z} H^{-1} S^{-1} \quad (1)$$

For each $a \in \mathbf{P}$ and for each $k \in \mathbf{K}$ $\Delta_k(\Lambda_k(a)) = a$, where $\Delta_k = \Lambda_k$.

S - is a permutation describing the *plugboard** transformation (S consists of transpositions and 1-cycles only), B - is a permutation describing the *reflector** transformation (B consists of 13 transpositions),

L, M, R - are permutations describing transformations of the three *cipher drums**,

H - is a transformation of the *entry wheel** (H is the identity permutation),

Q = (ABCDEFGHIJKLMNOPQRSTUVWXYZ) – a cycle of length 26,

Ds[i] ($i = l, m, r$) - positions of *drums** (left, middle and right) before pressing any key,

Rs[i] ($i = l, m, r$) - positions of *rings** (left, middle and right) (Ds[i], Rs[i] $\in$ **P**),

x, y, z - positions of *rotors** before pressing any key (values from set **IP** = {0,1,…,25}),

x = (Ds[l]-Rs[l])%26 for the left rotor,

y = (Ds[m]-Rs[m])%26 for the middle rotor,

z = (Ds[r]-Rs[r])%26 for the right rotor (cf. [5]).

We denote by Λ_H a permutation which we obtain by substituting in the formula (1) the identity permutation H for a permutation S, i.e. $\Lambda_k = \Lambda = S\Lambda_H S^{-1}$. We treat the letters A, B, …, Z of the Latin alphabet as the numbers from the set **IP**.

4 Characteristic of a given day

Until September 15, 1938 the first 6 letters of each message (i.e. *headline**) were coded (the same day) by means of the same permutations A, B, C, D, E and F (cf. [13]).

Let us assume that we eavesdropped on the same day the set of messages with the following headlines. These messages were generated for the same daily key.

```
( 1)LXTXWV  ( 6)QYVDRZ  (11)IGSNSA  (15)FFOJLG  (19)ZMQCPE  (23)NHYYTU
( 2)XVXMMD  ( 7)EZCBHY  (12)DRIGDR  (16)VAJEYM  (20)BPETET  (24)KQWIJW
( 3)AENFBL  ( 8)OOLZVX  (13)CTPSKH  (17)UCMRIJ  (21)MIHHFN  (25)SSAOCO
( 4)YBULZQ  ( 9)JLKQQK  (14)HNZKXI  (18)TKBVNF  (22)PWGPOP  (26)RJRAAC
( 5)GDFUUB  (10)WUDWGS
```

Each of the permutations A, B, C, D, E, F makes up a product of 13 transpositions. On the basis of double-ciphered message settings LXT XWV we can notice that permutations A and D contain accordingly transpositions (x, L) and (x, X), where x is a ciphered letter (different from L and X). Therefore, the product $AD^{(1)}$ sends the letter L to the letter X. Since in a 24-hour period all headlines of messages were ciphered for the same *initial drum settings** (in short IDS), we can determine the permutation AD, which stems from these settings, using the above-mentioned observation (cf. [7])

```
      ABCDEFGHIJKLMNOPQRSTUVWXYZ
AD:   FTSGBJUKNQIXHYZPDAOVREWMLC
```

Similarly we can find permutations BE and CF. Permutations AD, BE and CF make up the *characteristic of a given day* (the *characteristic structure*) (cf. [13], [7]).

Let us analyze 3 consecutive drum settings and assigned to them permutations AD, written as products of disjoint cycles.

```
AAV   (AEHKDJQNV)(BOUGRMSYC)(FWXZ)(ITPL)          EJ
ABW   (ATMFOR)(BSYLZN)(CDWKEX)(GJQUPH)(I)(V)      BGG
ABX   (ABIEHYDVO)(CWGLNXZKQ)(FMJS)(PRTU)          EJ
```

Every permutation AD (BE, CF) consists of an even number of disjoint cycles of the same length (cf. [18]). If a permutation possesses this property, we can ascribe a number to it. We define a *number* of such a permutation as an expression consisting of lengths of cycles (expressed by means of letters). For instance, we assign to the first permutation the number EJ, because it includes two cycles of length 4 (4 = E) and two cycles of length 9 (9 = J). A *characteristic of drum settings* is defined as three numbers determined for these settings and for the next two positions of drums (cf. [13]).

(1) We remind the reader that we use another notation for a product of permutations.

For example, the characteristic of drum settings AAV is the expression EJ.BGG.EJ. The cryptologists used another notation (cf. [7]). We defined a number of a permutation in the way described above for convenience in implementation.

The Enigma codes individual letters for *rotor settings** (in short Rt) determined on the basis of the formula Rt[*i*] = Ds[*i*]-Rs[*i*] (*i* = *l*, *m*, *r*) (cf. section 3). But the same Rt we can obtain for various couples of *relative drum settings* (in short RDS) and *relative ring settings* (in short RRS). Since (until September 15, 1938) all headlines of messages were ciphered (within one day) for the same settings IDS, detecting one couple of RDS and RRS gave message settings of all messages ciphered on that day (cf. [13]).

5 Analysis of the catalogue method

Rejewski had invented a device (so-called *cyclometer*) that produced cycle lengths for expressions *AD* (*BE*, *CF*). Next, he (along with Różycki and Zygalski) made the *catalogue*, i.e. six *sets of cycle lengths* (one set for each of the six possible orders of drums). Each set included characteristics (cycle lengths) calculated for all possible 26^3 drum settings. Then, every day, the cryptologists determined RDS by finding the characteristic of a given day in the catalogue. This method was used to generate a couple of RDS and RRS and to guess the order of drums. On the basis of this information the cryptologists determined permutation *S* which represented plug connections and found message settings for all messages ciphered on the same day (cf. [13]).

The catalogue algorithm presented below is a reconstruction of the catalogue method. This method did not give actual ring settings (in short RS) and settings IDS so the cryptologists had to use an additional method to read messages. In part II of this paper we provide the missing algorithm (the author's idea) which returns settings RS and IDS. Some elements of the catalogue method were adapted to clearer writing in program language. We also determined the catalogue in a different way.

We received the following results for real connections of both drums and the plugboard that were used by Wehrmacht. We assume (according to the knowledge of that time) that we know connections of all kinds of drums. Given examples were executed for drums: L = I, M = II, R = III, for the reflector B = UKW B and for the plugboard connections *S* = (BY)(CX)(EO)(HV)(KR)(PZ).

The output contacts of cipher drums I, II, and III (cf. [13]):

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
I	E	K	M	F	L	G	D	Q	V	Z	N	T	O	W	Y	H	X	U	S	P	A	I	B	R	C	J
II	A	J	D	K	S	I	R	U	X	B	L	H	W	T	M	C	Q	G	Z	N	P	Y	F	V	O	E
III	B	D	F	H	J	L	C	P	R	T	X	V	Z	N	Y	E	I	W	G	A	K	M	U	S	Q	O

The turnover positions for selected drums: I - Q, II - E, III - V (cf. [13]).

The reflector connections after November 2, 1937 (cf. [13]):

UKW B (AY)(BR)(CU)(DH)(EQ)(FS)(GL)(IP)(JX)(KN)(MO)(TZ)(VW).

5.1 Schema of the catalogue algorithm

Input: The set of headlines of messages that were eavesdropped on the same day and such that all letters of the alphabet occur on each of the six headline positions.

Output: The algorithm returns a couple of RDS and RRS, determines the order of drums, guesses plug connections and reads message settings for all given headlines.

1. Set drums in any order, e.g. I, II, III. Repeat the algorithm for other orders of drums until you obtain a proper result. Set rings in any established way (e.g. `AAW`). These settings are treated as RRS. Set the plugboard according to the identity permutation.
2. Add 6-letter headlines of messages to the table `M`.
3. Generate the list of characteristics.

- Characteristics (remembered as strings *x.y.z*) are stored in the quick access list `HL` of the `ThashList` class. Any characteristic (e.g. $x_1.y_1.z_1$) has an assigned list which contains all drum settings, the characteristics of which equal to $x_1.y_1.z_1$. For example, `[BEI.N.BCK]<-DZK, GPX, JWD, KXU, LSS, POH, UYH, ZGN`.
- Determine characteristics *x.y.z* for all 26^3 possible drum positions (disregard the *double step** on the middle drum).
- Add each determined characteristic *x.y.z* to the list `HL` along with drum settings (e.g. `ABC`) assigned to it in the following way. If the list `HL` does not contain a characteristic *x.y.z*, then add to `HL` the string *x.y.z* and join to this string a one-element list which only contains drum settings (e.g. `ABC`). Otherwise, find in the list `HL` the characteristic *x.y.z* and join to a list (assigned to the characteristic *x.y.z*) drum settings (e.g. `ABC`).

4. Determine the characteristic of a given day (i.e. products A_MD_M, B_ME_M, C_MF_M) and the characteristic $x_M.y_M.z_M$ on the basis of the set of headlines stored in the table `M` and write down A_MD_M as a product of disjoint cycles.
5. Find the characteristic $x_M.y_M.z_M$ in the list `HL`.
6. Write down all drum settings (from a list assigned to the characteristic $x_M.y_M.z_M$) and corresponding to them permutations A_HD_H (in cycle notation).
7. Compare each written product A_HD_H with the permutation A_MD_M. If you can find the product A_HD_H for which $A_MD_M = SA_HD_HS^{-1}$ that means that the current order of drums is proper and drum settings corresponding to the product A_HD_H make up RDS. Then you can determine the permutation S (S represents plug connections). Otherwise, you have to change the order of drums and repeat the algorithm for another set of cycle lengths.
8. Set drums according to RDS and the plugboard according to S and read message settings for all headlines from the table `M`.

Tab. 1. Zestaw szyfrogramów wykorzystanych w eksperymentach
Tab. 1. The set of messages which were used in experiments

(1)	LIO IWN BVSVIWKOUYZKHEHCCNBKWHTMMI	(14)	SXR ECR BWTIDPHQJYQMCQHKUEFSJNPEAI
(2)	YCJ OFQ IQAOVGEDVSAYLVTJCQGYRPXWQU	(15)	EJT BAO SJLGXJEGWODMEAAFFUMPHMKEMQ
(3)	MKG DDU ZIJTCVRCXQWYYGYJGIWRHBSAHT	(16)	FFF JLY NSPMPLANTTXKRITJHAQCFRLNNF
(4)	RMV CJH JAXLOYCKQKHBKSXHCILQNOLKVH	(17)	ZGZ KSM IYSZTRZQJHVQKSQRNAGOLBMYHH
(5)	OOH NRL DNHNDEKDDQSJAFVURZIWVJTGBK	(18)	DNK GQX VNGKDZVVPHRBAUSJJSKGQYHCCJ
(6)	PSD LXS UUSOFCFSYCUHZBSMEMOYWSBSNQ	(19)	NZC TGK YQBBRRKUFZCQSWCAUXIIGLLQIU
(7)	VRQ RHJ OHNRXYIDMUPOXWMZRZIGOQMYSD	(20)	XPS SKA WOWXIOIZFYTVVQDSCYVZPJOTTB
(8)	JHY MIF DEEGFUXRYRSBNGISWBXOYXFKKP	(21)	WQA WUE LVYNFGVSJZFUFRAGBWEZICVTGY
(9)	KUU ATV IJXULQTGTSHKOBZDETGAAQFLPQ	(22)	GYM ZBT QSXCNCFSMGBEHTERSJIDZKHVMF
(10)	HVP POZ WNIZUZOVEGOGTQZTROXURPLVQK	(23)	UEW UNW UYEJKXKOZRMMHPXHNNBBLGCDJU
(11)	IBB QVC IXQTPPJPTNFZPYMRVSAUTBHFTJ	(24)	TLI YMD HHRENCJMTUOYOSRXLDKZFDQNIR
(12)	AWN FEB BQKUHSAFAZGIVIHFCDVUHNEJQK	(25)	BAX XPP YHQPDLVPQGXOUKROILVXIWYPUC
(13)	CDL HZI DUYYSPVBTXHPWXUGJLJFADGFRW	(26)	QTE VYG TREUTGCDHREXSGDMVFTFDUGOHW

Example 5.1 Table 1 contains messages eavesdropped during the same day (i.e. received for the same daily key). All letters of the alphabet occur on each of the six headline positions. We reveal that these headlines were generated for ring settings RS = ZHL, for the order of drums I, II, III, for settings IDS = YNC and for plug connections S = (BY)(CX)(EO)(HV)(KR)(PZ).

Executing the catalogue algorithm

(1,2). We set up the Enigma (drums I, II, III, RRS = AAW, plug connections - according to the identity permutation) and add headlines of messages (1)-(26) to the table M.

(3). The algorithm generates the list HL with all possible characteristics determined for the order of drums I, II, III.

(4). The program determines the characteristic of a given day (i.e., products A_MD_M, B_ME_M, C_MF_M) on the basis of the table M.

A_MD_M:	AFJMDGZK.BXSE.CHPLIQVR.NTYO.U.W	(BEI)
B_ME_M:	APKDZGSXCFLMJ.BVORHIWENQUTY	(N)
C_MF_M:	AEGUVHLIDS.BCKXPZMTON.FY.JQ.R.W	(BCK)

Hence, the characteristic $x_M.y_M.z_M$ is the expression BEI.N.BCK.

(5,6). In the list HL the program finds the characteristic BEI.N.BCK and assigned to it a list of drum positions. Next, for these positions products A_HD_H are written down.

```
[BEI.N.BCK]<-DZK,GPX,JWD,KXU,LSS,POH,UYH,ZGN,
//-------------------------
DZK AENMUWST.BVCYGFHL.DRZJ.I.KXPQ.O
GPX AYWRQZDP.B.CIFH.E.GOLXUJVK.MSTN
JWD AWFOTKLN.BXGHRJVS.CDPQ.EZMI.U.Y
KXU AQOKYJHS.BPMGLTWX.CZDF.E.IRNV.U
LSS AGIH.BLEWDQUJ.CTFN.KOVZRMXP.S.Y
POH AWNHYZGD.BSXE.CFRUJPKV.ITOQ.L.M
UYH AQGOFNTE.BVRSJHZC.DMIX.KULP.W.Y
ZGN AFJMDGPR.BENT.CSOY.HKXVZLIQ.U.W
```

(7). We place the product assigned to RDS = ZGN under the product A_MD_M so that cycles of the same length (in both products) are written down one under the other.

A_MD_M: AFJMDG**ZK**.**BX**S**E**.NT**YO**.**CHP**LIQ**VR**.U.W

ZGN: AFJMDG**PR**.**YC**S**O**.NT**BE**.**XVZ**LIQ**HK**.U.W

We determine the permutation S (plug connections), i.e.

S_1 = (BY)(CX)(EO)(HV)(KR)(PZ) or

S_2 = (BY)(CX)(EO)(HV)(KR)(PZ)(UW).

(8). Next, settings of the M3 Enigma are changed. The new configuration is as follows RRS = AAW, RDS = ZGN, S = (BY)(CX)(EO)(HV)(KR)(PZ).

Finally, we can read message settings for all headlines from the table M. For instance, for the first message we obtain ASDASD.

5.2 Implementation

The catalogue() method of the Cycles class generates the list of such drum settings (and corresponding to them permutations A_HD_H) which have characteristics identical to the characteristic of a given day. Settings RRS are delivered by the parameter rs. The cycleLSet() method of the Cycles class generates characteristics for all 26^3 positions of drums and adds each determined characteristic (e.g. *x.y.z*) to the list HL along with corresponding to this characteristic drum settings (cf. section 5). The drToChar() method of the Cycles class determines a number (i.e. cycle lengths) for drum settings (e.g. ABC) given by the parameter dr. The moveDr() method of the Enigma class shifts drum settings dr for k key presses (without the *double step*). The charactOfDay() method of the Cycles class generates a characteristic of a given day for headlines of messages remembered in the table M according to the procedure presented in section 4. The permADBECF() method of the Enigma class determines one of the permutations: *AD*, *BE* or *CF* (depending on a given parameter k) for headlines of messages remembered in the table M. The cycleNotat() method of the Cycles class transforms any permutation into a product of disjoint cycles. The toChar() method of the Cycles class generates a number (cycle lengths) for a given (by the parameter) permutation. The listOfDrumSet() method of the Cycles class finds a list of such drum settings (in the list HL) which have a characteristic identical to the characteristic of a given day (described by the parameter ch). Next, the found drum settings along with corresponding to them permutations A_HD_H (in cycle notation) are written out. The prodADBECF() method of the Enigma class determines a product *AD* (*BE* or *CF*) depending on the given parameter k, for current settings of the machine. By tpR we denote the turnover position for the right drum. The ITC() method changes a number from the set {0, 1, …, 25} into a suitable letter.

```
( 1) void Cycles::catalogue(String rs, String *M, int n){
( 2) cycleLSet(rs);
( 3) String s1=charactOfDay(M, n);
( 4) listOfDrumSet(s1);}
//------
( 5) void Cycles::cycleLSet(String rs){
( 6) String b1, b2, b3, ch, s1, s2, s3;
( 7) b1="AAA"; s1=drToChar(rs,b1);
( 8) b2="AAB"; s2=drToChar(rs,b2);
```

```
( 9) for(int i=1; i<=17576; i++){ //26^3
(10)   b3=CE->moveDr(b2,1);
(11)   s3=drToChar(rs,b3);
(12)   ch="."+s1+"."+s2+"."+s3+".";
(13)   addToList(ch,bl);
(14)   b1=b2; b2=b3; s1=s2; s2=s3;}}
//------
(15) String Cycles::charactOfDay(String *M, int n){
(16) CE->setEnigma(CE->Rs,"AAA");
(17) Perm* AD=CE->permADBECF(M,n,0);
(18) Perm* BE=CE->permADBECF(M,n,1);
(19) Perm* CF=CE->permADBECF(M,n,2);
(20) out >> cycleNotat(AD)+"\n";
(21) String char="."+toChar(AD)+"."+toChar(BE)+"."+toChar(CF)+".";
(22) delete AD; delete BE; delete CF;
(23) return char;}
//------
(24) void Cycles::listOfDrumSet(String ch){
(25) TStringList *L1; String s="["+ch+"]<-";
(26) int l=HL->Items->IndexOf(ch);
(27) if(l!=-1){
(28)   (TObject*)L1=HL->Items->Objects[l];
(29)   for(int i=0; i<L1->Count; i++)s+=L1->operator[](i)+","; s+="\n";
(30)   for(int i=0; i<L1->Count; i++){
(31)     CE->setEnigma(CE->Rs,L1->operator [](i));
(32)     s+=L1->operator[](i)+" "+cycleNotat(CE->prodADBECF(0))+".";}
(33)   out >> s;}}
//------
(34) String Cycles::drToChar(String rs, String dr){
(35) CE->setEnigma(rs,dr);
(36) Perm *Il=CE->prodADBECF(0);
(37) String ch=toChar(Il); delete Il;
(38) return ch;}
//------
(39) String Enigma::moveDr(String dr, int k){ //without the double step
(40) String BE=dr, BEH;
(41) for(int i=1; i<=k; i++){
(42)   BEH=BE; BE[3]=ITC((CTI(BE[3])+1)%26);
(43)   if(BEH[3]==tpR){
(44)     BE[2]=ITC((CTI(BE[2])+1)%26);
(45)     if(BEH[2]==tpM)BE[1]=ITC((CTI(BE[1])+1)%26);}}
(46) return BE;}
```

5.3 Correctness of the catalogue algorithm

Below we justify the correctness of the catalogue method. The first six letters of each message (i.e. headline) make up double ciphered 3-letter message settings. These letters were coded by means of permutations *A*, *B*, *C*, *D*, *E*, *F* (cf. section 3). Each of these permutations consists of 13 transpositions. Hence, M. Rejewski was interested in products *AD*, *BE*, *CF*. He used the following theorem.

Theorem 5.1 (cf. [13]) Let us assume that both *X* and *Y* permutations (of the same degree) consist of disjoint transpositions only. Then

(a) The product *XY* consists of an even number of disjoint cycles of the same lengths.

(b) The letters belonging to one and the same transposition of permutations *X* or *Y* always belong to two different cycles (of the same length) of the permutation *XY*.

Below we propose an easy (author's) justification of the theorem. Rejewski's proof was destroyed during the war. First we shall analyze the example.

Example 5.2 Let us consider permutations A, B and the product AB.

A = (AB)(CR)(DE)(FS)(GH)(IU)(JL)(KQ)(MT)(NO)(PV)(WY)(XZ)
B = (AQ)(BL)(CU)(DN)(ES)(FR)(GP)(HT)(IJ)(KV)(MO)(WZ)(XY)
AB = (ALICFENMHPK)(BQVGTODSRUJ)(WX)(YZ) [1]

Let us choose any transposition, for example (AB) and write down two sequences of signs (starting from the letter A and from the letter B) that lead to obtaining two factors (cycles) of the product AB

(A) **A**→B→**L**→J→**I**→U→**C**→R→**F**→S→**E**→D→**N**→O→**M**→T→**H**→G→**P**→V→**K**→Q→**A**

(B) **B**→A→**Q**→K→**V**→P→**G**→H→**T**→M→**O**→N→**D**→E→**S**→F→**R**→C→**U**→I→**J**→L→**B**

For each underlined letter c_i (in the sequence (A) or (B)) the permutation B contains a transposition (c_i, b_{i-1}) and the permutation A contains a transposition (c_i, a_{i+1}), where b_{i-1} precedes the letter c_i and a_{i+1} follows the letter c_i (in the sequence (A) or (B)). Underlined letters in the sequences (A) and (B) make up consecutive signs of the cycles (ALICFENMHPK) and (BQVGTODSRUJ) accordingly. Both these cycles are factors of the product AB. Letters underlined and not underlined appear alternately in both sequences. These properties result from the operation of multiplication.

Let us notice that signs of the sequence (B) written out in reverse order give the sequence (A) and any letter occurs in the sequence (A) (and (B)) not more than once (Since permutations A and B consist of disjoint transpositions only). Hence, both sequences (A) and (B) contain the same letters.

Justification of the theorem.

Assumption (A): Let both X and Y be n-involutions (for the same n).

$X = (x_1, x_2)\ (x_3, x_4)\ (x_5, x_6)\ (x_7, x_8)\ \dots\ (x_{l-1}, x_1)$

$Y = (y_1, y_2)\ (y_3, y_4)\ (y_5, y_6)\ (y_7, y_8)\ \dots\ (y_{l-1}, y_1)$

Let us choose any transposition, e.g. (x_1, x_2) and write down (starting from the letter x_1) a sequence of signs that leads to obtaining a factor (a cycle) of the product XY.

(X1) $\quad \underline{x_1} \to x_2 (= y_i) \to \underline{y_{i+1}} (= x_j) \to x_{j+1} (= y_k) \to \underline{y_{k+1}} (= x_m) \to x_{m+1} (= y_d) \to \dots \to \underline{y_{u+1}} (= x_1)$

Justification (b). Underlined letters (in the sequences (X1)) make up consecutive signs of a certain cycle of the product XY. Signs underlined and not underlined appear alternately.

We can write down an analogous sequence (X2) starting from the letter x_2. Then signs of the second sequence written out in reverse order will give the sequence (X1). It results from (A), i.e., since permutations X and Y consist of disjoint transpositions only.

Any letter can occur in the sequence (X1) (and (X2)) not more than once (It also results from assumption (A)).

[1] We remind the reader that we use another notation for a product of permutations.

Moreover, the letters that are underlined in the sequence (X1) will not be underlined in the other sequence and inversely. Hence, we can affirm that letters x_1, x_2 of the transposition (x_1, x_2) belong to two disjoint cycles (of the same length) of the permutation XY.

Justification (a). Let us choose any underlined sign c_i in the sequence (X1). The permutation Y contains a transposition (c_i, y_{i-1}) and the permutation X contains a transposition (c_i, x_{i+1}), where y_{i-1} precedes the letter c_i, x_{i+1} follows the letter c_i and y_{i-1}, x_{i+1} are not underlined (in (X1)). Signs underlined and not underlined appear alternately. Additionally, any letter occurs in (X1) not more than once.

Thus, in the sequence (X1) there are all signs from an even number of transpositions (the same number of transpositions from X and from Y).

From justification (b) we know that all signs of the sequence (X1) form two (the even number) disjoint cycles (of the same length) that are factors of the product XY.

We recall that we chose the transposition (x_1, x_2) in an arbitrary way. And we shall obtain a similar result for each transposition of the permutation X (or Y), in particular, for such transposition (x_3, x_4) that letters x_3, x_4 did not appear in the sequence (X1). Hence, finally we can state that the product XY consists of an even number of disjoint cycles of the same length.

Definition 5.1 (cf. [11]) Permutations A and B are said to be *similar* if a permutation X that satisfies the equation $X^{-1}AX = B$ exists.

Lemma 5.1 (cf. [5]) Let A and B be similar permutations. We can always distinguish in these permutations (expressed as products of disjoint cycles) cycles of the same length corresponding to each other.

Lemma 5.2 If $X^{-1}A_1X = A_2$ and $X^{-1}B_1X = B_2$, then $X^{-1}A_1B_1X = A_2B_2$.

Characteristics of drum settings (in the catalogue) were made for the plugboard represented by the identity permutation H. However, each characteristic of a given day depended on a permutation S (that was specified in a daily key). Let us notice that permutation $\Lambda_H = S^{-1}\Lambda S$ (cf. section 3) and any corresponding to it permutation Λ (i.e., for any fixed S) are similar. Therefore, from lemma 5.2 we can write $A_HD_H = S^{-1}ADS$. From lemma 5.1 one can notice that numbers (i.e. cycle lengths) of both products A_HD_H and AD (for any S) are identical. Finally, we have justified the known fact (cf. [13]) that plug connections do not change a characteristic of drum settings.

6 Computational complexity

The total running time of the `catalogue()` method for the established order of drums, by using a computer with an AMD Turion 64 X2 processor clocked at 1.9GHz, is about 9 seconds. Obtaining the permutation Λ (which is called 2 times for each of the 26^3 positions) is the most time-consuming operation within this method. This algorithm can be repeated 6 times (for three drums) at the most. The preparation of the complete catalogue (the six sets - one set for each order of drums) took the cryptologists over a year. However, finding the order of drums, settings RDS and message settings took them (with the use of the catalogue) about 15 minutes. And after 1-2 hours needed to find ring settings it was possible to read messages

eavesdropped the same day (cf. [5]). To guess plug connections and proper ring settings we call the `PlugBoard()` method and the `findRS()` method which are described in part II of this paper.

7 Correctness of the catalogue algorithm. Unconsidered cases

Here we answer the question why we can determine any daily key (for different ring settings) on the basis of the catalogue which was calculated for one established kind of ring settings. We analyzed permutations A_HD_H (for different ring settings) in order to obtain the proper algorithm. We generated these permutations for all 26^3 possible drum settings, although a full rotation of drums (for the sake of the *double step*) gives 26^3-26^2 possible positions only. Given results were executed for drums L=I, M=II and R=III. Hence, we took into consideration turnover positions on letters `Q`, `E` and `V` accordingly.

We indicate the cases of daily keys for which the catalogue method does not return any result and the ones of which the cryptologists were not able to decide. Thanks to the possibility of using a computer we can break them.

(A) Let $R_1R_2R_3$ and $R_1R_2R_4$ denote established ring settings, where the letter R_4 follows (in the alphabet) the letter R_3 (e.g. `ABC` and `ABD`). Table 2 presents fragments of three lists (for ring settings `AAZ`, `AAA` and `AAB`). Each of the lists contains consecutive drum positions and corresponding to them products A_HD_H.

Tab. 2. Fragmenty trzech list z iloczynami A_HD_H dla ustawień RS: `AAZ`, `AAA` i `AAB`
Tab. 2. Fragments of three lists of products A_HD_H for ring settings `AAZ`, `AAA` and `AAB`

AAO AVEYWGJB.CPUHTQNO.DRZLK.FSIXM	AAP AVEYWGJB.CPUHTQNO.DRZLK.FSIXM	AAQ AVEYWGJB.CPUHTQNO.DRZLK.FSIXM
AAP AGONYCL.BFIMZJD.ESXHKR.PVWQUT	AAQ AGONYCL.BFIMZJD.ESXHKR.PVWQUT	AAR AGONYCL.BFIMZJD.ESXHKR.PVWQUT
AAQ AOPSBQWCEJD.FZYVHXNKRLM.GT.IU	AAR AOPSBQWCEJD.FZYVHXNKRLM.GT.IU	AAS AMKVSEHCBFURI.DLTOGJXYQZNWP
AAR ABTQOLKSCHYIZ.DNJVRXMEWGUPF	AAS AEDSQJKTYVFMB.CNGXHWRIOPULZ	AAT AEDSQJKTYVFMB.CNGXHWRIOPULZ
AAS AMGFSEXJH.BQOUPTLDY.CVK.IWR.N.Z	AAT AMGFSEXJH.BQOUPTLDY.CVK.IWR.N.Z	AAU AMGFSEXJH.BQOUPTLDY.CVK.IWR.N.Z
AAT ADXNL.BQFMO.CHTIRSYZ.EKPGUWVJ	AAU ADXNL.BQFMO.CHTIRSYZ.EKPGUWVJ	AAV AWPHEQJI.BKM.CNZVSTOU.DYL.F.G.R.X
AAU A.BQLCO.DH.EGPWR.F.IS.JMT.KYU.NX.VZ	AAV AJBGMHTINDRPC.EQOVSXKZYLWUF	ABW AJBGMHTINDRPC.EQOVSXKZYLWUF
AAV AP.BCDSYNEKJHV.FWMIOURTZGL.QX	ABW AP.BCDSYNEKJHV.FWMIOURTZGL.QX	ABX AP.BCDSYNEKJHV.FWMIOURTZGL.QX
ABW AWGSK.BM.CVQZXI.DL.ENHJUF.OYPRT	ABX AWGSK.BM.CVQZXI.DL.ENHJUF.OYPRT	ABY AWGSK.BM.CVQZXI.DL.ENHJUF.OYPRT
ABX ANQCKGOXVWYJZ.BDIRLTEPSFHUM	ABY ANQCKGOXVWYJZ.BDIRLTEPSFHUM	ABZ ANQCKGOXVWYJZ.BDIRLTEPSFHUM
ABY AEVJHKPYGDBM.CIWNLQTUROZX.F.S	ABZ AEVJHKPYGDBM.CIWNLQTUROZX.F.S	ABA AEVJHKPYGDBM.CIWNLQTUROZX.F.S
ABZ ANHDKGJFSMWE.BQZYXOLRIVTP.C.U	ABA ANHDKGJFSMWE.BQZYXOLRIVTP.C.U	ABB ANHDKGJFSMWE.BQZYXOLRIVTP.C.U
ABA AQYXCKUIJLG.BSHTOMVZEPR.D.F.N.W	ABB AQYXCKUIJLG.BSHTOMVZEPR.D.F.N.W	ABC AQYXCKUIJLG.BSHTOMVZEPR.D.F.N.W

• Let D1 and D2 be a coupe of consecutive drum settings (e.g. `ABY` and `ABZ`).

(a) The permutation A_HD_H determined for settings $R_1R_2R_3$ and D1 is the same (with some exceptions) as the permutation A_HD_H determined for settings $R_1R_2R_4$ and D2.

(b) Divergences appear when the right drum of D1 is set to the letters `R` or `U`.

Justification. Permutations A_H and D_H are determined on the basis of rotor settings (i.e. Rt[i] = Ds[i]-Rs[i] (i = l, m, r)). We remind the reader that before ciphering the right drum shifts. (a) Let $R_1R_2R_3$=`AAZ` and D1=`ABY`, then $R_1R_2R_4$=`AAA` and D2=`ABZ`.

$Rt_{(ABY,\ AAZ)}$=[A][B][Z]-[A][A][Z]=[0][1][0] and $Rt_{(ABZ,\ AAA)}$=[A][B][A]-[A][A][A]=[0][1][0] (hence, we obtain the same permutations A_H for both settings) and

$Rt_{(ABB,\ AAZ)}$=[0][1][3]=$Rt_{(ABC,\ AAA)}$ (hence, we obtain the same permutations D_H for both settings), hence permutations A_HD_H are identical.

(b) Let $R_1R_2R_3$=`AAZ` and D1=`AAR`, then $R_1R_2R_4$=`AAA` and D2=`AAS`.

$Rt_{(AAR, AAZ)}$=[0][0][19]=$Rt_{(AAS, AAA)}$ (hence, we obtain the same permutations A_H for both settings) but $Rt_{(AAU, AAZ)}$=[0][0][22] and $Rt_{(AAV, AAA)}$=[0][1][22] (before ciphering the shift of drums D2 (from AAV to ABW) appears), hence permutations A_HD_H will be different. We have a similar situation when the right drum of D1 is set to the letter U.

Tab. 3. Szyfrogramy tekstu T=AAAAAA dla RS: AAZ i AAA i obok dla RS: AAA i AAB

Tab. 3. Ciphertexts of text T=AAAAAA for ring settings AAZ and AAA and beside, for ring settings AAA and AAB

Rs=AAZ	Rs=AAA	Rs=AAA	Rs=AAB
AAO CDLPBM,	AAP CDLPBM,	AAO MCDLPB,	AAP MCDLPB,
AAP DLPBM**F**,	AAQ DLPBM**U**,	AAP CDLPB**M**,	AAQ CDLPB**T**,
AAQ LPBM**F**Q,	AAR LPBM**U**Q,	AAQ DLPB**M**U,	AAR DLPB**T**U,
AAR PBM**F**QO,	AAS PBM**U**QO,	**AAR** LPB**M**UQ,	AAS LPB**T**UQ,
AAS BM**F**QOF,	AAT BM**U**QOF,	AAS PB**M**UQO,	AAT PB**T**UQO,
AAT M**F**QOFX,	AAU M**U**QOFX,	AAT B**M**UQOF,	AAU B**T**UQOF,
AAU **F**QOFXY,	AAV **U**QOFXY,	**AAU** **M**UQOFX,	AAV **T**UQOFX,

Table 3 shows differences on the first and on the fourth letters of ciphertexts, when the right drum of D1 is set to R or U.

(B) Let $R_1R_2R_3$ and $R_1R_4R_3$ denote established ring settings, where the letter R_4 follows (in the alphabet) the letter R_2 (e.g. ABD and ACD). Table 4 provides fragments of three lists (for ring settings AZA, AAA and ABA). Each of the lists contains consecutive drum positions and corresponding to them products A_HD_H.

Tab. 4. Fragmenty trzech list z iloczynami A_HD_H dla ustawień RS: AZA, AAA i ABA

Tab. 4. Fragments of three lists of products A_HD_H for ring settings AZA, AAA and ABA

ACR AXWCHTUNJLGFB.DEQRPVMKSYIZO	ADR AXWCHTUNJLGFB.DEQRPVMKSYIZO	AER AXWCHTUNJLGFB.DEQRPVMKSYIZO
ACS AJUMOHWPCE.BDGSLNKIZQ.FYR.TVX	ADS AJUMOHWPCE.BDGSLNKIZQ.FYR.TVX	AES AFQLCIT.BOEXERJ.DHMKGY.NSPVWU
ACT A.BXJPNDZVQFOL.CGMHTKWSYURI.E	ADT A.BXJPNDZVQFOL.CGMHTKWSYURI.E	AET AZTRWYHJIKC.BVFDLUEPONG.M.Q.S.X
ACU AUVRMQPKOHXYC.BJFLSEZTDIWNG	ADU AUVRMQPKOHXYC.BJFLSEZTDIWNG	AEU AB.COFNDSPMTH.EQ.GUXKRWZIYJ.L.V
ACV AMJNCXVPWU.BT.DOZKQISGRY.EF.H.L	ADV AMJNCXVPWU.BT.DOZKQISGRY.EF.H.L	AEV AGKJIXYMDPCFL.BTNUREZOHVSWQ
ADW AMFL.BIQHW.CVGZ.DTOXY.EKSJ.NRPU	AEW AMFL.BIQHW.CVGZ.DTOXY.EKSJ.NRPU	BFW ABCILNSHEUXFD.GOJPWYMTZVQKR
ADX AWXKBDMNLJCSU.EVTQROFYPGIHZ	AEX AWXKBDMNLJCSU.EVTQROFYPGIHZ	BFX A.BGTYPSDFLRCZ.EOINVWKXUHJM.Q
ADY AOSP.BXYGQ.CIRU.DFMWV.ENKL.HJZT	AEY AOSP.BXYGQ.CIRU.DFMWV.ENKL.HJZT	BFY AUOMTNBDGHVZY.CRSXJIWFKEQLP
ADZ AZXOLDMVJ.BWPKFNEQS.C.G.HT.IY.R.U	AEZ AZXOLDMVJ.BWPKFNEQS.C.G.HT.IY.R.U	BFZ ADINGZCVSFHB.EJULROYWPQMK.T.X
ADA AUOHJBD.CINSMW.EPGRXLV.FKQZYT	AEA AUOHJBD.CINSMW.EPGRXLV.FKQZYT	BFA AWNHE.BIMVFYU.CLQGP.D.JKRTZOS.X

• Let D1 and D2 be a coupe of drum settings of the form $D_1D_2D_3$ and $D_1D_4D_3$, where D_4 occurs immediately (in the alphabet) after the letter D_2 (e.g. ACR and ADR).

(a) The permutation A_HD_H determined for settings $R_1R_2R_3$ and D1 is the same (with some exceptions) as the permutation A_HD_H determined for settings $R_1R_4R_3$ and D2.

(b) Divergences appear when drum settings D1 are of the form ?DS, ?DT, ?DU, …, ?EU, where ? denotes any letter (e.g. D1=ADS and D2=AES). Nevertheless, suitable permutations A_HD_H are on the lists for all ring settings (in a different place).

(C) Let $R_1R_2R_3$ and $R_4R_2R_3$ denote established ring settings, where the letter R_4 follows (in the alphabet) the letter R_1 (e.g. ACD and BCD). Table 5 shows fragments of three lists (for ring settings ZAA, AAA and BAA). Each of the lists contains consecutive drum positions and corresponding to them products A_HD_H.

Tab. 5. Fragmenty trzech list z iloczynami A_HD_H dla ustawień RS: ZAA, AAA i BAA
Tab. 5. Fragments of three lists of products A_HD_H for ring settings ZAA, AAA and BAA

ZAA ACIKETMLX.BGPDOFWHQ.JNVU.RYSZ	AAA ACIKETMLX.BGPDOFWHQ.JNVU.RYSZ	BAA ACIKETMLX.BGPDOFWHQ.JNVU.RYSZ
ZAB AI.BPMZX.CEUKGQ.DO.FVLRNW.HJSTY	AAB AI.BPMZX.CEUKGQ.DO.FVLRNW.HJSTY	BAB AI.BPMZX.CEUKGQ.DO.FVLRNW.HJSTY
ZAC AGLY.BIZW.CNK.DHRSUQ.EMO.FPXTJV	AAC AGLY.BIZW.CNK.DHRSUQ.EMO.FPXTJV	BAC AGLY.BIZW.CNK.DHRSUQ.EMO.FPXTJV
ZAD AZB.CJG.DUNVXIWKT.EQPYSRLFH.M.O	AAD AZB.CJG.DUNVXIWKT.EQPYSRLFH.M.O	BAD AZB.CJG.DUNVXIWKT.EQPYSRLFH.M.O
ZAE AQPCMVZET.BFLD.GSIU.HNJWOXRYK	AAE AQPCMVZET.BFLD.GSIU.HNJWOXRYK	BAE AQPCMVZET.BFLD.GSIU.HNJWOXRYK
ZAF AEVRHI.BGCOSQ.DNWLJT.FXKMZY.P.U	AAF AEVRHI.BGCOSQ.DNWLJT.FXKMZY.P.U	BAF AEVRHI.BGCOSQ.DNWLJT.FXKMZY.P.U
ZAG AEBWFZYRH.CTNXGDQJK.IL.MS.OV.PU	AAG AEBWFZYRH.CTNXGDQJK.IL.MS.OV.PU	BAG AEBWFZYRH.CTNXGDQJK.IL.MS.OV.PU
ZAH AHRMEZ.B.CSDGOP.FJWQLU.I.KYNTVX	AAH AHRMEZ.B.CSDGOP.FJWQLU.I.KYNTVX	BAH AHRMEZ.B.CSDGOP.FJWQLU.I.KYNTVX
ZAI ARUGJXWTONZK.B.CVMHEQIYPDLS.F	AAI ARUGJXWTONZK.B.CVMHEQIYPDLS.F	BAI ARUGJXWTONZK.B.CVMHEQIYPDLS.F
ZAJ ANGZWSPEQK.BFCLOJDYHT.IVX.MUR	AAJ ANGZWSPEQK.BFCLOJDYHT.IVX.MUR	BAJ ANGZWSPEQK.BFCLOJDYHT.IVX.MUR

• Let $D_1D_2D_3$ and $D_4D_2D_3$ be a coupe of drum settings, where the letter D_4 follows (in the alphabet) the letter D_1 (e.g. ZAB and AAB). The permutation A_HD_H determined for settings $R_1R_2R_3$ and $D_1D_2D_3$ is identical to the permutation A_HD_H determined for settings $R_4R_2R_3$ and $D_4D_2D_3$.

The result of experiments

(a) For all ring settings we can create the lists with identical (with some specific exceptions) suitably shifted representations of permutations A_HD_H. Divergences appear in the case of drum positions specified in (A) and (B). The cause is the shift on the middle drum while ciphering the first six letters of a headline.

(b) Characteristics of drum positions $D_1D_2D_3$ (where $D_3 \in \{Q, R, S, T, U\}$) are divergent, because each characteristic of drum positions consists of 3 numbers (cycle lengths), i.e. it depends on 3 consecutive products A_HD_H.

Summary 7.1

(a) Characteristics of drum positions $D_1D_2D_3$ ($D_1,D_2 \in \mathbf{P}$ and $D_3 \in \{Q, R, S, T, U\}$) are divergent in the catalogue (for the order of drums I, II, III). Divergences are a consequence of the shift on the middle drum while ciphering the fist six letters.

(b) If a coder chooses IDS=$D_1D_2D_3$, where $D_3 \in \{Q, R, S, T, U\}$, the catalogue method will not return any solution, because the characteristic of a given day is divergent.

(c) If a cryptanalyst chooses RRS such that RDS[r]$\in \{Q, R, S, T, U\}$, the catalogue method will not return any solution. He has to start the algorithm for different RRS.

Example 7.1 Given messages (Table 1) were coded for settings RS=ZHL and IDS=YNC. We executed the catalogue algorithm for different settings RRS.

Tab. 6. Pary ustawień RRS i RDS
Tab. 6. Settings RRS and RDS

XUA, –	EPH, DVY	RKO, QQF	QMV, PS**M**
TFB, –	GBI, FHZ	HRP, GXG	UGW, TM**N**
RLC, –	ADJ, ZJA	CZQ, BFH	PNX, OT**O**
MQD, –	JYK, IEB	IJR, HPI	DWY, CC**P**
VHE, UM**V**	FCL, EIC	YVS, XBJ	**BCZ**, –
SEF, RK**W**	ZOM, YUD	OST, NYK	
WAG, VG**X**	LTN, KZE	NIU, MOL	

Table 6 contains settings RRS and corresponding to them settings RDS (received in the catalogue algorithm). We can see that RDS=$D_1D_2D_3$, where $D_3 \in \{Q, R, S, T, U\}$

were not found in the catalogue (see RRS: `BCZ`, `XUA`, `TFB`, `RLC`, `MQD`). We cannot predict this before starting the algorithm. In this situation we have to start the program for different RRS once again. We have to shift the right ring 5 positions forward. The kryptologists did not have this possibility. They possessed one set for each order of drums.

8 Security of the M3 Enigma cipher

A full size of key space of the M3 Enigma cipher (until September 15, 1938) consisted of the number of possible wheel orders (N_{Wo}), the number of possible ring settings (N_{Rs}), the number of possible initial drum settings (N_{Ds}) and the number of possible plug connections (N_{Ps}). For each message an operator additionally chose a 3-letter message key, hence we multiply the size of key space by the number of possible message settings (N_{Ms}). $N_{Rs} = N_{Ds} = N_{Ms} = 26^3 = 17576$. $N_{Wo} = k!/(k\text{-}3)!$ (for k=3 drums N_{Wo}=6).

Below we shall determine the number of possible plugboard settings (N_{Ps}), i.e. we shall calculate in how many ways we can choose n ($n \le 13$) couples of letters from the 26-element set of letters. An order of selected couples of letters does not matter (i.e. choices `(AB)(CD)` and `(CD)(AB)` give the same plugboard settings).

$$N_{Ps}(n) = \frac{1}{n!}\prod_{k=1}^{n}\binom{26-2(k-1)}{2} = \frac{1}{n!}\cdot\frac{(26-2(1-1))!}{(26-2\cdot 1)!2!}\cdot\frac{(26-2(2-1))!}{(26-2\cdot 2)!2!}\cdot\ldots\cdot\frac{(26-2(n-1))!}{(26-2n)!2!} =$$

$$= \frac{26!}{(26-2n)!n!2^n} = \frac{26!(2n)!}{(26-2n)!(2n)!n!2^n} = \frac{\binom{26}{2n}\binom{2n}{n}n!}{2^n}$$

From 1.10.1936 to 15.09.1938 German cryptologists used k=3 drums and n=5-8 plug connections (cf. [5]). Hence for n=8 we have
$N_{Wo} \times N_{Rs} \times N_{Ds} \times N_{Ps} \times N_{Ms} = 6 \times 26^3 \times 26^3 \times 10{,}767{,}019{,}638{,}375 \times 26^3$ possible keys.

9 The implications of the work and conclusions

The reader can find other decryption algorithms of the Enigma cipher in [1] (Zygalski's sheets method) and [2] (the cryptologic bomb method and the plugboard algorithm). All these methods are interesting exercises and encourage the study of current problems of cryptology. The Enigma cipher can make up an interesting riddle for mathematicians, because it relies on permutation theory. Until now the mentioned methods were difficult because of a big computation complexity and incomplete (sometimes inaccurate) descriptions of historians. Rejewski's documentation from the war period was destroyed. This paper, [1] and [2] should facilitate mathematical research. The author also recommends papers [13], [5] and [3].

References

1. Borowska A.: The Cryptanalysis of the Enigma Cipher, *Advances in Computer Science Research*, 10, 2013, pp. 19-38
2. Borowska A., Rzeszutko E.: The Cryptanalysis of the Enigma Cipher. The Plugboard and the Cryptologic Bomb, *Computer Science*, AGH University of Science and Technology Press, 15(4), Kraków, 2014, pp. 365-388
3. Christensen C.: Polish Mathematicians Finding Patterns in Enigma Messages, *Mathematics Magazine*, 2007, pp. 247-273

4. Garliński J.: *Enigma. Mystery of the Second World War*, University of Maria Curie-Sklodowska Publishing House, Lublin, 1989
5. Gaj K.: *The Enigma Cipher. The Method of Breaking*, Communication and Connection Publishing House, Warsaw, 1989
6. Grajek M.: *Enigma. Closer to the Truth*, REBIS Publishing House, Poznań, 2007
7. Grajek M., Gralewski L.: *Birth of Mathematical Cryptology*, Semper Publishing House, Toruń, 2005
8. Gralewski L.: *Breaking of Enigma. History of Marian Rejewski*, Adam Marszalek Publishing House, Toruń, 2005
9. Kozaczuk W.: *How the German Machine Cipher Was Broken and How It Was Read by the Allies in World War Two*, University Publications of America, 1984
10. Lang S.: *Linear Algebra*, Springer-Verlag New York, 1987
11. Manning W.A.: *Primitive groups*, Stanford University Publications, University Series Mathematics and Astronomy, Stanford University Press Library, 1(1), 1921
12. Mostowski A., Stark M.: *Elements of Higher Algebra*, PWN, Warsaw, 1970
13. Rejewski M.: *How did Polish Mathematicians Decipher the Enigma*, Polish Mathematics Association Yearbooks. Series 2nd: Mathematical News, (23), 1980
14. Robinson Derek J.S.: *A Course in the Theory of Groups*, Springer-Verlag New York, Inc., 1996
15. Scott W.R.: *Group Theory*, Courier Dover Publications, 1964

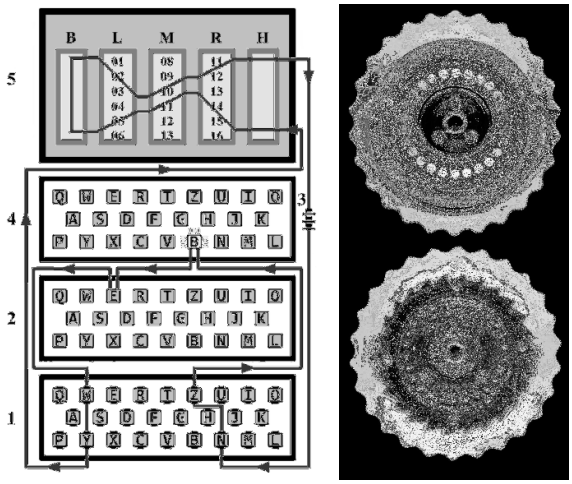
Appendix
Construction of the M3 Enigma machine

The M3 Enigma machine is an electro-mechanical cipher device. This machine consists of an alphabetical 26-letter keyboard, a lampboard (set of 26 lights), a *plugboard*, a set of 3 rotating, encrypting discs (called *drums*) placed on a shared shaft, 2 fixed wheels: an *entry wheel* and a *reflector* (*reversal drum*), a battery and a *turning mechanism*.

Pressing any key on the keyboard (e.g. the key `E`) causes the closure of an electric circuit (under each key there are two contacts). Then, an electric current flows through the different components in the present configuration of the circuit. It flows consecutively through a connection under the pressed key (2), through the plugboard (1), through the entry wheel H, via three drums R, M and L to the reflector B. The reflector inverts the signal (but using a different route). From the reflector the current passes successively through drums L, M and R, through the entry wheel H, to the plugboard (1) and finally to an appropriate lamp (4) which represents a letter different from `E`, causing it to light up (cf. [6]).

Before September 15, 1938 the German Army used three kinds of drums denoted by I, II, III. They could place them in 6 possible ways. Each *drum* is a disc with a diameter of approximately 10cm (fig. 1. (B)). Inside the drum there is a second disc (called a *rotor*). On the right side of each rotor there are 26 spring pins and on the left side there are 26 flat electrical contacts. Both pins and flat contacts are arranged in a circle near the edge of the rotor and represent 26 letters of the Latin alphabet. Inside each rotor there are 26 insulated wires which connect the pins on one side to the contacts on the other in an established way (different for each type of drum) (cf. [5]). Since three drums

are mounted side-by-side on the shaft (fig. 1. (A)) the pins of one drum touch the contacts of the neighboring one, forming 26 fragments of an electric circuit (cf. [8]). Each drum is encircled by a metal rotating *ring*. Engraved numbers (on this ring) correspond to the 26 letters of the alphabet. On the edge of the rotor there is one unique place. The letter on the ring which is engraved opposite this position is treated as the *ring setting*. Individual kinds of cipher drums also differ by so-called *turnover positions*. The turnover positions of the three kinds of drums were as follows I-Q, II-E, III-V (cf. [5]).



Rys. 1. (A) Schemat działania Enigmy M3 (1) łącznica wtyczkowa, (2) klawiatura, (3) bateria, (4) panel z lampkami, (5) bębenki: 3 bębenki szyfrujące L, M, R, bębenek wstępny H i bębenek odwracający B. (B) Prawa strona (pokazuje styki sprężynujące) i lewa strona (pokazuje styki płaskie (stałe)) bębenka (cf. [6])

Fig. 1. (A) The diagram presents how the M3 Enigma works (1) the plugboard, (2) the keyboard, (3) the battery, (4) the lampboard, (5) disks: three cipher drums L, M, R, the entry wheel H and the reflector B. (B) The right side (shows the pin electrical contacts) and the left side (shows the flat electrical contacts) of a drum (cf. [6])

The position of each movable drum is given as a number (engraved on a ring) which can be seen through a window in the lid of the machine. The *rotor position* is understood as the difference between the position of the drum and the ring setting.

Connections of the *entry wheel* in the M3 Enigma machine are represented by the identity permutation (cf. [5]). The *reflector* pairs the outputs of the last rotor and redirects the current back through the drums using a different path.

The *plugboard* contains 26 plugs and sockets (one pair for each letter of the alphabet). An operator configures this part of the machine by means of a cable, i.e. he connects a plug of one pair with a socket of the second one (and inversely). Then two letters are swapped both before and after the signal flows through the rotors (cf. [5]).

When an operator presses any key, one, two, or three cipher drums turn one twenty-sixth of a full rotation (before the electric circuit closure) in a manner similar, but not

identical, to that of an odometer. That is, after pressing the key, the right drum turns 1/26 of a full rotation. When this drum reaches the turnover position, the middle drum turns 1/26 of a full rotation too. When the second drum reaches the turnover position, the left and middle drums turn 1/26 of a full rotation (so-called *double step*) (cf. [4]).

Each military unit that used the Enigma was provided with the Enigma machine's initial settings in the form of tables of the *daily keys settings*. Daily key settings (until September 15, 1938) consisted of the choice of cipher drums, the order in which they were fitted on a shaft, the *ring settings*, the plug connections, and the *initial drum settings* (cf. [4]). Additionally, each message was coded for different (individual) *message settings*. In order to cipher a text with the M3 Enigma (before September 15, 1938) an operator had to set up his machine according to daily key settings, choose his own arbitrary three letters (*message settings*), code them twice, place the double ciphered message settings (6 letters) in the *headline* of the message and set the drums to selected three letters. Next, the operator typed a text and the other operator wrote down letters that were lit up on the lampboard. To decode a text encrypted with the M3 Enigma machine the receiver had to set up his Enigma in the same way as the sender had done it while ciphering. Since the cryptologists determined message settings by means of the catalogue method, operators at once set drums to the message settings.

Summary

We study the problem of decoding secret messages encrypted by the German Army with the M3 Enigma machine. We focus on the algorithmization and programming of this problem. We also give mathematical justifications of some facts because the Enigma cipher relies on permutation theory. We propose a reconstruction and completion of the catalogue method. We complete this method with two author's algorithms, i.e. the plugboard algorithm and the ring settings algorithm. On the basis of these three methods we can obtain the complete daily key and any message settings, as well as read each message eavesdropped before September 15, 1938. We have tested the behavior of the cipher in order to catch cases of keys for which the method does not return results. We enclose an implementation of presented algorithms in Cpp language.

Keywords: Enigma M3, Rejewski, characteristic of a given day

Kryptoanaliza szyfru Enigmy. Metoda Katalogu. Część I

Streszczenie

Tematem pracy jest kryptoanaliza szyfru Enigmy M3, używanej przez siły zbrojne oraz inne służby państwowe Niemiec podczas II wojny światowej. Proponujemy rekonstrukcję metody katalogu. W celu pełnego oprogramowania metody przeprowadzono testy zachowań szyfru przy różnych ustawieniach klucza. Praca zawiera matematyczne uzasadnienie istotnych dla metody faktów. W części II uzupełniamy metodę katalogu o dwa brakujące algorytmy, służące do wyznaczania połączeń łącznicy wtyczkowej i ustawień pierścieni. Wykonanie trzech wspomnianych algorytmów pozwala na odtworzenie pełnego dziennego klucza oraz klucza dowolnej depeszy. Dzięki temu możemy przeczytać dowolną depeszę przechwyconą przed 15 września 1938 roku. Załączamy implementację proponowanych algorytmów w języku Cpp.

Słowa kluczowe: Enigma M3, Rejewski, charakterystyka dnia

The research presented in this paper was founded by the BST S/WI/1/2014.