Polyalphabetic Substitutions

Klaus Pommerening
Fachbereich Physik, Mathematik, Informatik
der Johannes-Gutenberg-Universität
Saarstraße 21
D-55099 Mainz

October 25, 1999—English version October 13, 2013—last change
January 18, 2021

1 Key Alphabets

The Idea of Polyalphabetic Cipher

A polyalphabetic cipher—like a monoalphabetic one—encrypts each letter by a substitution that is defined by a permuted alphabet. However for each letter another alphabet is used, depending on its position in the plaintext.

Thus polyalphabetic encryption breaks the invariants that led to successful cryptanalysis of monoalphabetic substitutions:

- Letter frequencies
- l-gram frequencies
- Patterns

This method was considered unbreakable until the 19th Century, its variants that used cipher machines even until the begin of the computer era. Nevertheless before cipher machines became available polyalphabetic substitution was rarely used because it requires concentrated attention by the operator, and the ciphertext often is irreparably spoiled by encryption errors.

The Key of a Monoalphabetic Substitution

The key of a monoalphabetic substitution over the alphabet Σ is a permutation $\sigma \in \mathcal{S}(\Sigma)$. It has a unique description by the sequence of substituted letters in the order of the alphabet, that is by the family $(\sigma(s))_{s\in\Sigma}$.

Example for the standard alphabet $\Sigma = \{\texttt{A}, \ldots, \texttt{Z}\}$

1. representation by the permutation table:

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

2. or representation by the permuted alphabet alone:

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

The term “monoalphabetic” reflects that this one (permuted) alphabet defines the complete encryption function.

The Key of a Polyalphabetic Substitution

Now let us write several permuted alphabets below each other and apply them in order: the first alphabet for the first plaintext letter, the second alphabet for the second letter and so on. In this way we perform a **polyalphabetic substitution**. If the list of alphabets is exhausted before reaching

the end of the plaintext, then we restart with the first alphabet. This method is called **periodic polyalphabetic substitution**.

Example for the standard alphabet with 5 permuted alphabets

```
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

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

Using these alphabets we encrypt

```
UNIVERSITAETMAINZ          = plaintext
S    J    W    X          alphabet from line 1
 Y    N    Q    I         alphabet from line 2
  Y    Y    K             alphabet from line 3
   F    Z    U            alphabet from line 4
    E    S    Q           alphabet from line 5
-----------------
SYYFEJNYZSWQKUQXI          = ciphertext
```

Classification of Polyalphabetic Substitutions

We classify polyalphabetic substitutions by four independent binary properties:

- Periodic (or repeated key)
- Aperiodic (or running key)

depending on whether the alphabets repeat cyclically or irregularly.

- Independent alphabets
- Primary alphabet and accompanying secondary alphabets

where secondary alphabets derive from the primary alphabet by a fixed recipe. In the example above we took simple cyclical shifts. A closer inspection reveals that the definition of the shifts is given by the keyword `KLAUS`.

- Progressive alphabet change
- Alphabet choice controlled by a key

depending on whether the alphabets are used one after the other in their original order, or the order is changed by a key.

- Contextfree
- Contextsensitive

depending on whether the alphabets depend only on the position in the text, or also on some adjacent plaintext or ciphertext letters.

In general we take a set of alphabets (only $n!$ different alphabets are possible at all), and use them in a certain order, periodically repeated or not. Often one takes exactly n alphabets, each one beginning with a different letter. Then one can control the alphabet choice by a keyword that is cyclically repeated, or by a long keytext that is at least as long as the plaintext.

2 The Invention of Polyalphabetic Substitution

Polyalphabetic Encryption in Renaissance

See the web page `http://www.staff.uni-mainz.de/pommeren/Cryptology/Classic/2_Polyalph/Renaissance.html`

The Trithemius Table (aka Vigenère Table)

This table is used for polyalphabetic substitution with the standard alphabet and its cyclically shifted secondary alphabets. It has n rows. The first row consists of the alphabet Σ. Each of the following rows has the alphabet cyclically shifted one position further to the left. For the standard alphabet this looks like this:

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

Trithemius used it progressively, that is he used the n alphabets from top to down one after the other for the single plaintext letters, with cyclic repetition.

> Note that this procedure involves no key and therefore is not an encryption in the proper sense. Its security is only by obscurity.

Notwithstanding this weakness even TRITHEMIUS's method results in a crucial improvement over the monoalphabetic substitution: Each letter is encrypted to each other the same number of times in the mean. The frequency distribution of the ciphertext is perfectly uniform.

The BELLASO Cipher (aka VIGENÈRE Cipher)

Even VIGENÈRE himself attributes this cipher to BELLASO. It uses the TRITHEMIUS table but with the alphabet choice controlled by a keyword: for each plaintext letter choose the row that begins with this letter. This method uses a key and therefore is a cipher in the proper sense.

As an **example** take the keyword `MAINZ`. Then the 1st, 6th, 11th, ... plaintext letter is encrypted with the "M row", the 2nd, 7th, 12th, ... with the "A row" and so on. Note that this results in a periodic CAESAR addition of the keyword:

```
p o l y a l p h a b e t i c
M A I N Z M A I N Z M A I N
-------------------------
B O T L Z X P P N A Q T Q P
```

In general the BELLASO cipher uses a group structure on the alphabet Σ. For the key $k = (k_0, \ldots, k_{l-1}) \in \Sigma^l$ we have

Encryption: $c_i = a_i * k_{i \bmod l}$

Decryption: $a_i = c_i * k^{-1}_{i \bmod l}$

The first one who described this cipher algebraically as an addition apparently was the French scholar Claude COMIERS in his 1690 book using a 18 letter alphabet. Lacking a suitable formal notation his description is somewhat long-winded. Source:

> Joachim von zur Gathen: *Claude Comiers: The first arithmetical cryptography.* Cryptologia 27 (2003), 339 - 349.

3 Tools for Polyalphabetic Substitution

See the web page `http://www.staff.uni-mainz.de/pommeren/Cryptology/Classic/2_Polyalph/Tools.html`

4 Mathematical Description of Periodic Polyalphabetic Substitution

The General Case

In general a periodic polyalphabetic cipher has a key space $K \subseteq \mathcal{S}(\Sigma)^l$, consisting of sequences of l permutations of the alphabet Σ. The key $k = (\sigma_0, \ldots, \sigma_{l-1})$ defines the encryption function $f_k\colon \Sigma^r \longrightarrow \Sigma^r$ given by

$$\begin{array}{cccccccccc} a_0 & a_1 & \ldots & a_{l-1} & a_l & \ldots & a_i & \ldots & a_{r-1} \\ \downarrow & \downarrow & & \downarrow & \downarrow & & \downarrow & & \\ \sigma_0 a_0 & \sigma_1 a_1 & \ldots & \sigma_{l-1} a_{l-1} & \sigma_0 a_l & \ldots & \sigma_{i \bmod l} a_i & \ldots & \ldots \end{array}$$

The componentwise encryption formula for $c = f_k(a) \in \Sigma^r$ is

$$c_i = \sigma_{i \bmod l}(a_i),$$

and the formula for decryption

$$a_i = \sigma^{-1}_{i \bmod l}(c_i).$$

Effective Key Length

Bellaso Cipher

The primary alphabet is the standard alphabet, and we assume the cryptanalyst knows it. The key is chosen as word (or passphrase) $\in \Sigma^l$. Therefore

$$\begin{aligned} \#K &= n^l, \\ d(F) &= l \cdot \log_2(n). \end{aligned}$$

For $n = 26$ this amounts to $\approx 4.70 \cdot l$. To avoid exhaustion l should be about 10 (pre-computer age), or about 20 (computer age). However there are far more efficient attacks against this cipher than exhaustion, making these proposals for the key lengths obsolete.

Disk Cipher

The key consists of two parts: a permutation $\in \mathcal{S}(\Sigma)$ as primary alphabet, and a keyword $\in \Sigma^l$. Therefore

$$\begin{aligned} \#K &= n! \cdot n^l, \\ d(F) &= \log_2(n!) + l \cdot \log_2(n) \approx (n + l) \cdot \log_2(n) \end{aligned}$$

For $n = 26$ this amounts to $\approx 4.70 \cdot l + 88.38$.

If the enemy knows the primary alphabet, say be capturing a cipher disk, the effective key length reduces to that of the Bellaso cipher.

A More General Case

For a periodic polyalphabetic cipher that uses l independent alphabets,

$$\begin{aligned} K &= \mathcal{S}(\Sigma)^l, \\ d(F) &= \log_2((n!)^l) \approx nl \cdot \log_2(n). \end{aligned}$$

For $n = 26$ this is about $88.38 \cdot l$.

Another View

An l-periodic polyalphabetic substitution is an l-gram substitution, or block cipher of length l, given by the product map

$$(\sigma_0, \ldots, \sigma_{l-1})\colon \Sigma^l = \Sigma \times \cdots \times \Sigma \longrightarrow \Sigma \times \cdots \times \Sigma = \Sigma^l,$$

that is, a monoalphabetic substitution over the alphabet Σ^l. In particular the Bellaso cipher is the shift cipher over Σ^l, identified with $(\mathbb{Z}/n\mathbb{Z})^l$.

For $\Sigma = \mathbb{F}_2$ the Bellaso cipher degenerates to the simple XOR on $\mathbb{F}_2^l$.

5 The Cipher Disk Algorithm

Mathematical Notation

Take the alphabet $\Sigma = \{s_0, \ldots, s_{n-1}\}$, and interpret (or code) it as the additive group of the ring $\mathbb{Z}/n\mathbb{Z}$. The key $(\sigma, k) \in \mathcal{S}(\Sigma) \times \Sigma^l$ of a disk cipher consists of a primary alphabet (represented by the permutation σ) and a keyword $k = (k_0, \ldots, k_{l-1}) \in \Sigma^l$. Our notation for the corresponding encryption function is

$$f_{\sigma,k}\colon \Sigma^* \longrightarrow \Sigma^*$$

Special case: The BELLASO cipher with keyword k is $f_{\varepsilon,k}$ where $\varepsilon \in \mathcal{S}(\Sigma)$ denotes the identity permutation.

The Alphabet Table

We arrange the alphabets for the polyalphabetic substitution in form of the usual table:

s_0	s_1	s_2	$\ldots$	s_{n-1}
t_0	t_1	t_2	$\ldots$	t_{n-1}
t_1	t_2	t_3	$\ldots$	t_0
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
t_{n-1}	t_0	t_1	$\ldots$	t_{n-2}

where $t_i = \sigma s_i$ for $0 \le i \le n-1$.

Note that whenever we refer to an alphabet table we implicitely use an order on the alphabet Σ. This order manifests itself by indexing the letters as $s_0, \ldots, s_{n-1}$.

The Encryption Function

Now we encrypt a text $a = (a_0, a_1, a_2, \ldots) \in \Sigma^r$ using this notation. Let $a_i = s_q$ and $k_i = t_p$ as letters of the alphabet. Then we read the ciphertext letter c_i off from row p and column q of the table:

$$c_i = t_{p+q} = \sigma s_{p+q} = \sigma(s_p + s_q) \quad [\text{sums in } \mathbb{Z}/n\mathbb{Z}].$$

We have

$$k_i = t_p = \sigma(s_p), \quad s_p = \sigma^{-1}(k_i), \quad \text{hence } c_i = \sigma(a_i + \sigma^{-1}(k_i)).$$

If we denote by f_σ the monoalphabetic substitution corresponding to σ, then this derivation proves:

Theorem 1 *The disk cipher $f_{\sigma,k}$ is the composition (or "superencryption") of the* BELLASO *encryption $f_{\varepsilon,k'}$, where $k' = f_\sigma^{-1}(k)$, with the monoalphabetic substitution f_σ,*

$$f_{\sigma,k} = f_\sigma \circ f_{\varepsilon,k'}$$

Algorithm

The naive straightforward algorithm for the disk cipher is

- Take the next plaintext letter.
- Take the next alphabet.
- Get the next ciphertext letter.

From Theorem 1 we derive an algorithm that is a bit more efficient:

1. Take $k' = f_\sigma^{-1}(k)$, in coordinates $k'_i = \sigma^{-1}(k_i)$ for $0 \leq i < l$.
2. Add a and (the periodically extended) k' over $\mathbb{Z}/n\mathbb{Z}$, and get b, in coordinates $b_j = a_j + k'_{j \bmod l}$
3. Take $c = f_\sigma(b) \in \Sigma^r$, in coordinates $c_j = \sigma(b_j)$.

A Perl program implementing this algorithm is on the web page `http://www.staff.uni-mainz.de/pommeren/Cryptology/Classic/Perl/porta.pl`, the corresponding program for decryption on `http://www.staff.uni-mainz.de/pommeren/Cryptology/Classic/Perl/portadec.pl`. They can be called online from the pages `http://www.staff.uni-mainz.de/pommeren/Cryptology/Classic/2_Polyalph/portaenc.html` and `http://www.staff.uni-mainz.de/pommeren/Cryptology/Classic/2_Polyalph/portadec.html`

6 Analysis of Periods

Kasiski's approach

Already in the 16th Century Porta and the Argentis occasionally broke polyalphabetic encryptions by guessing the key or a probable word. For some more historical bits see the web page `http://www.staff.uni-mainz.de/pommeren/Cryptology/Classic/2_Polyalph/AnaPer.html`

An attack with known plaintext is easy against a disk cipher as soon as the primary alphabet is compromised, for example by a lost cipher disk. It is trivial against the Bellaso cipher that uses the standard alphabet. In contrast it is quite difficult against ciphers that use independent alphabets.

In 1863 the Prussian Major F. W. Kasiski published a solution that immediately demolished the belief in the security of periodic polyalphabetic ciphers. In fact Babbage had found this method ten years before but never published it. Therefore it is appropriate to credit the method to Kasiski.

The solution proceeds in three steps:

1. Determine the period l.
2. Arrange the ciphertext in rows of length l. Then the columns each are encrypted by a (different) monoalphabetic substitution.
3. Break the monoalphabetic columns.

Step 3, that is cryptanalyzing the monoalphabetically encrypted columns, faces the complication that the columns don't represent connected meaningful texts. Pattern search is pointless. However frequency analysis makes sense.

There are some simplifications for dependent alphabets:

- Adjusting the frequency curves. This works when the primary alphabet is known, see Sections 7 and 8.
- Symmetry of position when the primary alphabet is unknown (not treated here, but see Chapter 5). This method, proposed by Kerckhoffs, uses regularities in the alphabet table to infer further entries from already known entries, for example by completing the diagonals in the alphabet table of a disk cipher.

Especially simple is the situation with Bellaso's cipher, as soon as the period is known: Each column is Caesar encrypted. Therefore we need to identify only one plaintext letter in each column.

How to Determine the Period

Three approaches to determining the period of a periodic polyalphabetic cipher are

1. Exhaustion: Try $l = 1, 2, 3, \ldots$ one after each other. The correct l reveals itself by the appropriate frequency distribution of the letters in each column. As tools use some statistical "goodness of fit" tests. We'll study appropriate methods in Chapter 3.

2. Search for repetitions, see next subsection. This is an instance of the general method "pattern search".

3. Coincidence analysis after Friedman, Kullback, and Sinkov. This is also a subject of Chapter 3, and is an instance of the general method "statistical analysis".

In contrast to the exhaustion approach the other two methods immediately identify the situation where there is *no period*.

Search for Repetitions

We start with three observations:

1. If a plaintext is encrypted using l alphabets in cyclic order, and if a sequence of letters occurs k times in the plaintext, than this sequence occurs in the ciphertext about k/l times encrypted with the same sequence of alphabets.

2. In each of these occurrences where the sequence is encrypted the same way the ciphertext contains a repeated pattern in a distance that is a multiple of l, see Figure 1.

3. Not every repeated pattern in the ciphertext necessarily arises in this way. It could be by accident, see Figure 2. However the probability of this event is noticeably smaller.

An assessment of this probability is related to the birthday paradox of probability theory, and is contained in Appendix C. It was published in

> K. Pommerening: *Kasiski's Test: Couldn't the repetitions be by accident?* Cryptologia 30 (2006), 346-352.

A Perl program that searches for repetitions is on the web page `http://www.staff.uni-mainz.de/pommeren/Cryptology/Classic/Perl/kasiski.pl`

For online use see the web form `http://www.staff.uni-mainz.de/pommeren/Cryptology/Classic/2_Polyalph/kasiski1.html`

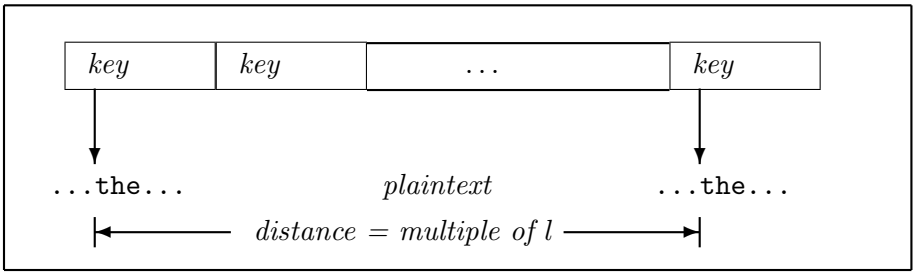

Figure 1: Repetition in ciphertext

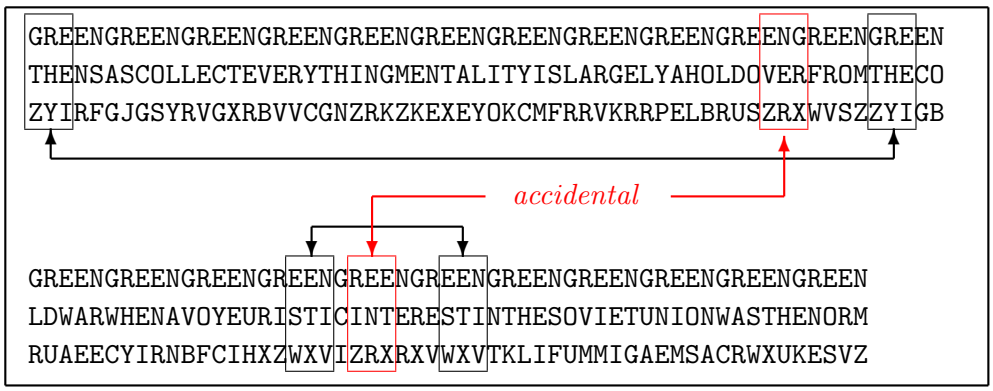

Figure 2: True and accidental repetitions

7 Cryptanalysis of a Polyalphabetic Ciphertext

(for a German plaintext)

Finding the Period by Searching Repetitions

`http://www.staff.uni-mainz.de/pommeren/Cryptology/Classic/`
`/2_Polyalph/Kasiski.html`

Column Analysis and Rearrangement

`http://www.staff.uni-mainz.de/pommeren/Cryptology/Classic`
`/2_Polyalph/Columns.html` and `http://www.staff.uni-mainz.de/`
`pommeren/Cryptology/Classic/2_Polyalph/Rearrang.html`

8 Rearranging the Columns

The Problem

The formula for the disk cipher from Theorem 1 was $f_{\sigma,k} = f_\sigma \circ f_{\varepsilon,k'}$ where $k' = f_\sigma^{-1}(k)$. However we didn't use this formula in our analysis but rather a similar one of the type $f_{\sigma,k} = g \circ f_\sigma$ where g should describe the shifts in the alphabets and g^{-1} the rearrangement. What we did was first rearrange the shifts in the different columns, and then solve the resulting monoalphabetic ciphertext. Note that for this method to work in general the primary alphabet must be known. Unfortunately there is no useful general interpretation of the formula $g = f_\sigma \circ f_{\varepsilon,k'} \circ f_\sigma^{-1}$ when σ is unknown.

We'll analyze the situation, first for an example.

Example

We take the standard alphabet $\Sigma = \texttt{A...Z}$, and consider an alphabet table.

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

Phrased in terms of permutations the top row, Row 0, the standard alphabet, corresponds to the identical permutation $\varepsilon \in \mathcal{S}(\Sigma)$. The next row, Row 1, the primary alphabet, corresponds to the permutation $\sigma \in \mathcal{S}(\Sigma)$. Row 2 corresponds to $\sigma \circ \tau$, where τ is the alphabet shift

$$\tau(\texttt{A}) = \texttt{B}, \quad \tau(\texttt{B}) = \texttt{C}, \quad \ldots, \quad \tau(\texttt{Z}) = \texttt{A}$$

Row i corresponds to $\sigma \circ \tau^{i-1}$. For the concrete example we have

$$\sigma(\texttt{A}) = \texttt{Q}, \quad \sigma(\texttt{B}) = \texttt{W}, \quad \ldots$$

and thus

$$\sigma \circ \tau(\texttt{A}) = \sigma(\texttt{B}) = \texttt{W}, \quad \sigma \circ \tau(\texttt{B}) = \sigma(\texttt{C}) = \texttt{E}, \quad \ldots$$

On the other hand

$$\tau \circ \sigma(\texttt{A}) = \tau(\texttt{Q}) = \texttt{R}, \quad \tau \circ \sigma(\texttt{B}) = \tau(\texttt{W}) = \texttt{X}, \quad \ldots$$

Shifts in the Primary Alphabet

Recall the alphabet table in the general case

s_0	s_1	s_2	$\dots$	s_{n-1}
t_0	t_1	t_2	$\dots$	t_{n-1}
t_1	t_2	t_3	$\dots$	t_0
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
t_{n-1}	t_0	t_1	$\dots$	t_{n-2}

where $t_i = \sigma s_i$ for $0 \le i \le n-1$, and σ is the permutation that defines the primary alphabet.

Identify as usual the alphabet $\Sigma = \{s_0, \dots, s_{n-1}\}$ with $\mathbb{Z}/n\mathbb{Z}$, the integers $\bmod n$, via $i \mapsto \sigma_i$ and take indices $\bmod n$. Mathematical expressions for the shifts in the original and primary alphabets are

- τ = shift by 1 in the original alphabet, $\tau(s_i) = s_{i+1}$.
- τ^k = shift by k in the original alphabet, $\tau^k(s_i) = s_{i+k}$.
- $\sigma\tau\sigma^{-1}$ = shift by 1 in the primary alphabet,

$$t_i \overset{\sigma^{-1}}{\mapsto} s_i \overset{\tau}{\mapsto} s_{i+1} \overset{\sigma}{\mapsto} t_{i+1}$$

- $\sigma\tau^k\sigma^{-1} = (\sigma\tau\sigma^{-1})^k$ = shift by k in the primary alphabet.

The alphabet table, interpreted as list of permutations, is the orbit of $\sigma \in \mathcal{S}(\Sigma)$ under iterated right translation by τ (or under the cyclic subgroup $\langle\tau\rangle \subseteq \mathcal{S}(\Sigma)$ generated by τ).

The "naive" shift that we performed in Section 7 shifted the single letters of the primary alphabet by a certain number of positions in the *standard* alphabet—we performed $\tau^i \circ \sigma$ for some value i. Why was this successful? Under what conditions are the naively shifted primary alphabets again rows of the alphabet table?

Decimated alphabets

We take the ordering of the alphabets into account and let $T_1 = (t_0, \dots, t_{n-1})$ be the ordered primary alphabet where $t_i = \sigma s_i$. The secondary alphabets then are $T_i = (t_{i-1}, \dots, t_{n-1}, t_0, \dots, t_{i-2})$ for $i = 2, \dots, n$. They correspond to the permutations $\sigma \circ \tau^{i-1}$, that is $T_i = (\sigma s_{i-1}, \sigma s_i, \dots)$.

The primary alphabet used in the example of Section 7 was of a special kind: It had $t_i = s_{3i \bmod 26}$. The corresponding formula for the general case is

$$t_i = s_{ki \bmod n},$$

and t_i for $i = 0, \dots, n-1$ runs through all elements of Σ if and only if k and n are relative prime.

Definition. Let the alphabet Σ be linearly ordered as $(s_0, \ldots, s_{n-1})$, and let $\gcd(k, n) = 1$. The (ordered) alphabet $T = (t_0, \ldots, t_{n-1})$ is called **decimated alphabet** of order k (of Σ with the given linear order relation) if there is an index $p \in \{0, \ldots, n-1\}$ such that $t_{p+i} = s_{ki \bmod n}$ for $i = 0, \ldots, n-1$.

That means, beginning with $t_p = s_0$ we take each k-th letter from Σ.

If the primary alphabet is decimated, so are all the secondary alphabets; we get them all by varying the index p.

Now when we apply the shift τ to the (ordered) primary and secondary alphabets $T_1, \ldots, T_n$ we get new alphabets $f_\tau(T_1), \ldots, f_\tau(T_n)$; note that we interpret the n-tuples T_i as texts and apply τ elementwise. The question we want to answer is whether the $f_\tau(T_i)$ belong to the collection of the T_i. The answer involves the normalizer $N(\langle\tau\rangle)$ of the subgroup $\langle\tau\rangle \leq \mathcal{S}(\Sigma)$.

Theorem 2 (Decimated alphabets) *Let the alphabet Σ be linearly ordered as $(s_0, \ldots, s_{n-1})$. Let the (ordered) primary alphabet $T_1 = (t_0, \ldots, t_{n-1})$ be defined by $t_i = \sigma s_i$ where $\sigma \in \mathcal{S}(\Sigma)$, and let $T_2, \ldots, T_n$ be the corresponding ordered secondary alphabets. Then the following statements are equivalent:*

(i) *There is a $j \in \{1, \ldots, n\}$ with $f_\tau(T_1) = T_j$.*
(ii) *f_τ permutes the $\{T_1, \ldots, T_n\}$.*
(iii) *T_1 is a decimated alphabet of Σ.*
(iv) *$\sigma \in N(\langle\tau\rangle)$.*

Proof. "(i) $\Longrightarrow$ (iv)": $f_\tau(T_1) = T_j$ means that $\tau \circ \sigma = \sigma \circ \tau^j$. Then $\sigma^{-1} \circ \tau \circ \sigma \in \langle\tau\rangle$ or $\sigma \in N(\langle\tau\rangle)$.

"(iv) $\Longrightarrow$ (iii)": By conjugation σ defines an automorphism of the cyclic group $\langle\tau\rangle$. These automorphisms are known, the following Lemma 1 gives $\sigma \circ \tau \circ \sigma^{-1} = \tau^k$ for some k, relative prime with n. The letter s_0 occurs somewhere in T_1, so let $s_0 = t_p$. Then $\sigma s_p = t_p = s_0$ and

$$t_{j+p} = \sigma s_{j+p} = \sigma\tau^j s_p = \tau^{jk}(\sigma s_p) = \tau^{jk} s_0 = s_{jk} \quad \text{for } j = 0, \ldots, n-1,$$

where as usual we take the indices mod n.

"(iii) $\Longrightarrow$ (iv)": Let p and k as in the definition. For any i we have

$$\tau^k \sigma s_{p+i} = \tau^k t_{p+i} = \tau^k s_{ki} = s_{ki+k} = s_{k(i+1)} = t_{p+i+1} = \sigma s_{p+i+1} = \sigma\tau s_{p+i}.$$

From this we conclude $\sigma \circ \tau = \tau^k \circ \sigma$ or $\sigma \circ \tau \circ \sigma^{-1} \in \langle\tau\rangle$.

"(iv) $\Longrightarrow$ (ii)": We have $\sigma^{-1} \circ \tau \circ \sigma = \tau^{k'}$ where $k'k \equiv 1 \pmod{n}$ whence $\tau \circ \sigma = \sigma \circ \tau^{k'}$. The permuted alphabet T_i corresponds to the permutation $\sigma \circ \tau^{i-1}$. Therefore $f_\tau T_i$ corresponds to $\tau \circ \sigma \circ \tau^{i-1} = \sigma \circ \tau^{k'+i-1}$. We conclude $f_\tau T_i = T_{k'+i}$.

"(ii) $\Longrightarrow$ (i)" is the restriction to a special case. $\diamond$

Lemma 1 *Let $G = \langle g \rangle$ be a finite cyclic group of order m. Then the automorphisms of G are the power maps $g \mapsto g^k$ where k is relatively prime to m. In other words, the automorphism group $\operatorname{Aut} G$ is isomorphic with the multiplicative group $(\mathbb{Z}/m\mathbb{Z})^\times$.*

Proof. Let h be an automorphism of G. Then $h(g) = g^k$ for some $k \in \mathbb{Z}$. This k uniquely defines h on all of G, and k is uniquely determined by h up to multiples of $\operatorname{Ord}(g) = m$. The power map $g \mapsto g^k$ is bijective if and only if k is relatively prime to m. $\diamond$

9 Summary

The canonical method of cryptanalyzing the disk cipher $f_{\sigma,k}$ proceeds in three steps:

1. Determine the period l.
2. Rearrange the ciphertext in rows of length l.
3. Reconstruct the monoalphabets of the columns.

Note that the effort is essentially independent of the key length. However the success probability decreases with the period length, because

- The probability of finding non-accidental repetitions decreases.
- Finding useful frequency distributions in the columns becomes harder.

Some special cases have special facilities:

- For a Bellaso cipher or more generally for a disk cipher with a decimated alphabet or even more generally for a disk cipher with a known primary alphabet we may rearrange the monoalphabets of the columns and are left with a large monoalphabetic ciphertext.
- Known plaintext gives the plaintext equivalents of single letters in a few columns that may be extended to other columns by symmetry of position when the alphabets are related, for example for a disk cipher (not treated here, but see Chapter 5).

These findings result in two recommendations for the use of polyalphabetic ciphers:

- The larger the period, the better the security.
- Independent alphabets more reliably protect from attacks.

Both of these recommendations make polyalphabetic ciphers more cumbersome in routine use, and therefore in history were adopted only after many failures.