Supervisor:	Dr. W. Böhm
Working title:	**Elementary Methods of Cryptology**
Keywords:	cryptography, cryptanalysis, substitution ciphers, transposition ciphers, Monte Carlo Markov Chains, simulated annealing

1 A Gentle Invitation

One of may favorite books is *Mathematical Recreations and Essays* by Rouse Ball and Coxeter (1987). Mathematical recreations? Seems to be a contradiction in terms. But believe me, this is not so. The very last chapter of this book deals with cryptology and it begins with these remarkable sentences:

> *The art of writing secret messages – intelligible to those who are in possession of the key and unintelligible to all others - has been studied for centuries. The usefulness of such messages, especially in the time of war, is obvious; on the other hand, their solution may be a matter of great importance to those from whom the key is concealed. But the romance connected with the subject, the not uncommon desire to discover the secret, and the implied challenge to the ingenuity of all from whom it is hidden have attracted to the subject the attention of many to whom its utility is a matter of indifference*[1].

Cryptology, the art and science of secret writing should be the topic of your thesis.

1.1 Some basic terms

Let us start fixing some important terms. Cryptography is that field of cryptology which deals the understanding and implementation of techniques to obfuscate information. These techniques are usually called *cryptographic algorithms*, *cryptographic systems*, in short *cryptosystems* or *ciphers*.

The text whose meaning should be concealed is called the *plaintext*. When the rules of a cipher are applied to the plaintext, one says also, the plaintext is *encrypted*, the result is called the *ciphertext*. *Decryption* is the reverse process

[1] These words are apparently due to Abraham Sinkov (1907-1998), an american mathematician with important contributions to cryptology.

of recovering the plaintext from the known ciphertext. In many cases ciphers have to rely on an external piece of information, the *key*.

Cryptanalysis on the other side, is the art of *breaking* a cipher. Given a piece of ciphertext we want to recover the underlying plaintext usually without additional information. This is also called a *ciphertext only attack*.

Steganography is a related field. It provides methods with the definite aim to hide the existence of a secrete message at all. Examples are invisible inks, micro dots, changes of the color values of pixels in a digital image, etc.

In this thesis you should consider only ciphers which work an a *letter-by-letter* basis using a particular *plaintext alphabet*. For our purposes we shall consider only the 27-letter alphabet in the standard lexicographic ordering:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
_	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

Table 1: Our standard alphabet

where _ denotes the space-character. Let us call this our *standard alphabet*.

At first the plaintext to be enciphered has to be prepared: all letters are turned to lowercase, all punctuation is removed, special characters like *diereses* are expanded, e.g. *ä* becomes *ae*, numbers are translated into appropriate numerals, e.g. *12* becomes *twelve*, etc. Furthermore, multiple spaces are condensed into single spaces and specials characters like newlines and tab-stops are removed. Note that we retain _, the space character. In general this is not a good idea as this may become a severe weakness of a cipher, but we shall keep spaces because of readability, and also because it makes more fun.

The ciphertext resulting from applying a particular cipher and a key to a given plaintext should be assumed to use the same alphabet as plaintext but all letters written in uppercase. Examples follow.

According to the way plaintexts are transformed to ciphertext general cryptographic systems are divided into two classes.

- *Substitution ciphers*. In these systems letters change their values. For instance, an `a` at some position in plaintext may be changed to `W`, at another position `a` may be changed to `M`, etc.

  Example 1.

plaintext	`send me more money`
ciphertext	`KABVDSNSUGNTRQUWXF`

  There are two points to observe in this example: (1) look at the letter `e` in plaintext. The first `e` is mapped to a `A`, the second occurrence of `e` maps to `N`, etc. (2) ciphertext and plaintext have the *same length*, however, this need not be so.

Monoalphabetic substitution ciphers use one alphabet only, this means, that a particular plaintext letter is always mapped to the same ciphertext letter. *Polyalphabetic* substitution ciphers use several alphabets and switch between them according to some rule. Example 1, for instance, is polyalphabetic.

- *Transposition ciphers.* Letters retain their value but change position.

  Example 2.

plaintext	`send me more money`
ciphertext	`YENOM EROM EM DNES`

  Whereas this cipher is easily decrypted (just be inspection) the cipher in Example 1 is much more difficult.

In practice, many modern cryptographic systems make heavy use of both substitution and transposition. A typical example is the AES system. However, in this thesis we should stay at rather elementary systems. Some of them will be now presented.

Lets start with substitution ciphers.

1.2 Caesar's Cipher

1.2.1 Encryption and decryption

This is the simplest case of a monoalphabetic substitution cipher. The roman historian Suetonius reports that Caius Iulius Caesar used this extraordinarily simple system to encrypt messages about battle orders, movements of military forces, etc.

Encryption is performed using a *translation table* in which the ciphertext alphabet results from the plaintext alphabet by a simple circular shift of d positions to the left. This number d is the *key* of the system. Caesar mostly used the key $d = 3$. The corresponding translation table is then:

_	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
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

Table 2: The translation table of a Caesar's cipher

Example 3. Here's some plaintext message and the corresponding ciphertext with unknown key d:

`come home again all is forgiven`

`MYWOJRYWOJKQKSXJKVVJSBJPYAQSEOX`

It's interesting to observe that Caesar's cipher is even used in our days! You may have seen the science fiction movie *2001: A Space Odyssey*. The name of the rogue computer *HAL* is just a Caesar-cryptogram of *IBM*!

Decryption of a Caesar's cipher is very easy. Given the key d perform a circular shift by d letters to the *right*.

As we will need this later, we show now that encryption and decryption can be expressed very conveniently in *algebraic form*. For this purpose we use the *mod* binary operator:

$$a \bmod b = \text{remainder of dividing } a \text{ by } b$$

For instance:

$$12 \bmod 7 = 5, \quad 4 \bmod 13 = 4, \quad 2^{7000} \bmod 7001 = 1, \quad \text{etc.}$$

Some care is needed when dealing with negative numbers. The correct definition of the *mod* operator is found in Graham, Knuth, and Patashnik (2003, p. 82):

$$a \bmod b = a - b\lfloor a/b \rfloor, \tag{1}$$

where the floor function $\lfloor x \rfloor$ gives x rounded *down* to the next smallest integer. Thus

$$(-3) \bmod 27 = -3 - 27\lfloor (-3)/27 \rfloor = -3 - 27 \cdot (-1) = 24 \tag{2}$$

Note in passing. Programming languages like C, Python or Java have a mod-operator denoted by `%`. Only Python's version behaves for negative values correctly like (1). C and Java return -3 in (2). When implementing ciphers in C or Java you should take care of this feature.

To encrypt a message using Caesar's Cipher with key d we do simply the following:

- By Table 1, map each plaintext letter a_i to an integer $0 \leq a_i \leq 26$.
- Calculate the ciphertext letter c_i corresponding to a_i by

$$c_i = (a_i + d) \bmod 27$$

  Translate the numbers c_i back to letters by Table 1 to get the ciphertext.

Example 3 (continued) With key $d = 10$:

Plaintext	c	o	m	e	_	h	o	m	e	_	a	g	a	i	n	_	a	l	l	_	...
	3	15	13	5	0	8	15	13	5	0	1	7	1	9	14	0	1	12	12	0	...
	13	25	23	15	10	18	25	23	15	10	11	17	11	19	24	10	11	22	22	10	...
Ciphertext	M	Y	W	O	J	R	Y	W	O	J	K	Q	K	S	X	J	K	V	V	J	

Decryption is the inverse, just *subtract* the key d and take care of negative numbers using (1):

$$a_i = (c_i - d) \bmod 27$$

1.2.2 Cryptanalysis of Caesar's cipher

Is Caesar's cipher a secure one? It is customary to assess this important point by looking at the *key space* $\mathcal{K}$. This is the set of all possible keys the system accepts. The number of keys $|\mathcal{K}|$ is a measure of computational work which is necessary in the following *worst case scenario*: if we have only the ciphertext and want to break the system by *brute force*, then in the worst case $|\mathcal{K}| = 26$ keys have to tested. This is a very small number, so Caesar's system cannot be considered secure.

Brute force is actually the method of choice for this cipher, we just start width $d = 1, 2, \ldots$ and try all keys, rotating each time the standard alphabet to the right by d places.

But two important issues come immediately into our mind:

- How can we know that a particular ciphertext was created by a particular cryptographic system, in our cases Caesar's system?
- Even if we know, how can we find out the language of plaintext so that we are able so recognize the unknown key?

There do exist important cryptanalytic tools that allow us to find reasonable answers to these questions, provided the available ciphertext is not too short. See Section 2 for more about that.

Let's continue Example 3 and perform a brute force attack by systematically trying keys $d = 1, 2, \ldots$ to decrypt the ciphertext. We obtain:

$d = 1$: `LXVNIQXVNIJPJRWIJUUIRAIOX PRDNW`
$d = 2$: `KWUMHPWUMHIOIQVHITTHQ HNWZOQCMV`
$d = 3$: `JVTLGOVTLGHNHPUGHSSGPZGMVYNPBLU`
`...`
$d = 10$ `COME HOME AGAIN ALL IS FORGIVEN`

Of course, this wasn't a challenge. But still we may ask: *Is there a way to find the key d without brute force?*

1.3 Frequency analysis

This is one of the most important concepts in cryptanalysis and you should take care of it in your thesis by a thorough discussion.

Human languages, may it be English, German, but also artificial languages like Esperanto or even Klingon, follow certain rules. Words are not merely random combinations of letters. Indeed, human languages exhibit certain *statistical regularities* which can be identified by appropriate methods. The simplest and oldest method is to calculate the *frequency distribution* of letters in a text. This idea is to Al Kindi (801 - 873 AD) an Arab mathematician and philosopher.

To apply frequency analysis we first need a *learning sample* to obtain the reference distribution of letters. Usually we look for a sufficiently large *text corpus*

and count the occurrences of various letters. Of course, this requires also that we have some idea about the language of the unknown plaintext.

For the purpose of this introduction I selected Tolstoj's *War and Peace*. The text (more than 3 million letters) has been prepared in the sense described above. Then I counted letters in *War and Peace* and in the cryptogram of Example 3. The results are shown in Table 3 below. It may be more informative to make a line plot of these frequency distributions (See Figure 1).

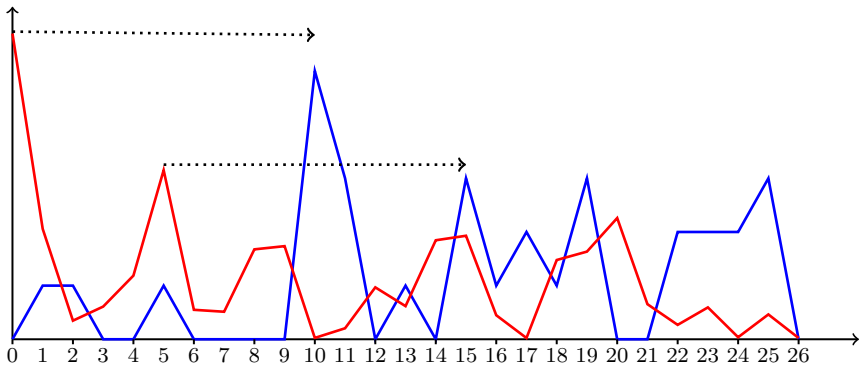

Figure 1: The frequency distributions of Table 3

Here we can see very clearly:

- The letter with highest frequency in *War and Peace* is the space character with about 18 %. In the cryptogram the letter with highest frequency is `J`. This strongly suggests a shift of $d = 10$ between these distribution. Thus we may try as a first guess the key $d = 10$.
- The same pattern can be seen if we compare the letters with second highest frequency: `e` in the text corpus, `O` in the cryptogram, again a distance of 10.
- Also quite remarkable: the cryptogram has only 31 letters, still it seems that the frequency comparisons are conclusive.

Observe that our analysis above is based on a simple visual inspection of the frequency plots, just look at the peaks!

Is there a better way to compare two discrete distributions?

There are quite a number of ways we can do better. A very simple idea is to calculate the *total variation distance* between two distributions.

Consider two frequency distributions f_i and g_i defined on the same set $\mathcal{X}$. In our case

$$\mathcal{X} = \{0, 1, 2, \ldots, 26\}$$

We say that f_i and g_i are *close* if the total variation distance is small. The latter is defined by

$$\|f - g\|_{TV} = \frac{1}{2} \sum_{i \in \mathcal{X}} |f_i - g_i|$$

	War And Peace			*ciphertext*	
n	character	abs. freq.	rel. freq.	abs. freq.	rel. freq.
0	space	565 454	0.1834	0	0.0000
1	a	204 128	0.0664	1	0.0323
2	b	34 419	0.0112	1	0.0323
3	c	60 448	0.0197	0	0.0000
4	d	117 752	0.0383	0	0.0000
5	e	312 716	0.1017	1	0.0323
6	f	54 491	0.0177	0	0.0000
7	g	50 906	0.0166	0	0.0000
8	h	166 293	0.0541	0	0.0000
9	i	172 223	0.0560	0	0.0000
10	j	2 485	0.0008	5	0.1613
11	k	20 322	0.0066	3	0.0968
12	l	96 030	0.0312	0	0.0000
13	m	61 286	0.0199	1	0.0323
14	n	183 114	0.0595	0	0.0000
15	o	191 440	0.0622	3	0.0968
16	p	44 456	0.0145	1	0.0323
17	q	2 319	0.0008	2	0.0645
18	r	146 594	0.0477	1	0.0323
19	s	162 126	0.0527	3	0.0968
20	t	224 202	0.0729	0	0.0000
21	u	64 911	0.0211	0	0.0000
22	v	26 641	0.0087	2	0.0645
23	w	58 925	0.0192	2	0.0645
24	x	3 758	0.0012	2	0.0645
25	y	45 931	0.0149	3	0.0968
26	z	2 387	0.0008	0	0.0000
Total		3 075 757	1.0000	31	1.0000

Table 3: Frequency Counts

I've calculated $\|f - g\|_{TV}$ and plotted for all shifts $d = 0, 1, \ldots, 26$. You can

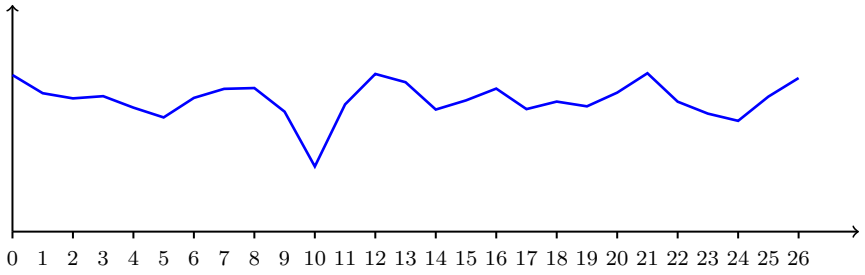

Figure 2: TV of distributions in Table 2 for shifts $d = 0, 1, \ldots, 26$

see the striking down-peak at $d = 10$ in Figure 2. Again we measure a strong signal indicating that the key is $d = 10$. Note that this can be found out by the computer in a more or less *automatic fashion* and does not need human intervention.

1.4 Monoalphabetic substitution

1.4.1 Encryption and decryption

Caesar's Cipher is a special case of a monoalphabetic substitution cipher. The latter is defined by a translation table for the standard alphabet (Table 1) where in the second row we have a *permutation* P of first row. In case of Caesar's Cipher this permutation is *cyclic*, as it is obtained by a cyclic shift of letters. But now we no longer require this permutation to be cyclic. As a result the classical monoalphabetic substitution is much stronger than Caesar's cipher.

Example 4. Given is the following translation table where the second row is a random permutation of the standard alphabet.

plaintext alphabet:	_	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
ciphertext alphabet = key:	Y	C	S	O	E	X	P	Q	Z	B	W	L	H	V	M	D	G	U	J	K	A	T	I	_	F	R	N

Table 4: A translation table for a monoalphabetic substitution cipher

Enciphering is easy. For example assume that we want to conceal the somewhat desperate message *need reinforcements at once*:

plaintext:	`need reinforcements at once`
ciphertext:	`MXXEYJXBMPDJOXVXMAKYCAYDMOX`

Observe that the *encryption key* is just the second row of the translation table.

Since our standard alphabet has 27 letters, the key space $\mathcal{K}$ consists of all permutations of 27 elements which is quite a lot:

$$|\mathcal{K}| = 27! = 10888869450418352160768000000 \approx 10^{28}$$

Applying a *brute force* attack would require to test so many keys in the *worst case*. But even if we have the best high-speed computers at our disposal, it will simply take too much time to break such a cipher by *brute force*.

Thus, if we accept the size of the key space as an indicator of secureness of a cryptographic system, then monoalphabetic substitution seems to be pretty safe. Later we will find out that this system is a rather weak one and cryptanalysis poses no real challenge for an experienced cryptanalyst.

But before we turn to these aspects: How can be decrypt a secret message when we possess the key?

This is easy. Just form the *inverse permutation* P^{-1} for the key P: sort the translation table along the second row and then interchange row 1 and row 2:

_	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
w	t	i	a	o	d	x	p	l	v	r	c	k	u	z	c	f	g	y	b	u	q	m	j	e	_	h

Table 5: The translation table for deciphering Example 4

Using this table decryption is straightforward.

1.4.2 Cryptanalysis of a monoalphabetic substitution cipher

Due to the size of the key space a brute force attack using the ciphertext only is certainly not feasible.

What about *frequency analysis*?

Why not? A monoalphabetic substitution always maps a plaintext letter to the same ciphertext letter. Thus letter frequencies are preserved, but they are distributed *very* differently according to the key permutation P. Thus if e.g., the space character ␣ is mapped to the ciphertext letter `W`, then in a sufficiently long ciphertext `W` will be the letter with roughly the same frequency as the space character ␣ in plaintext. Normally, cryptanalysts just start this way and try to get a *clue* about the unknown plaintext by replacing the ciphertext letters with highest frequency by those letters which in a learning sample (e.g., *War and Peace*) have highest frequency.

Example 5. For the next example I have chosen a somewhat longer plaintext, let us call it T, with 714 letters. Upon enciphering I obtained the ciphertext[2]:

```
FNDEYQUUDXTDTHDGTDGYNDNHBDFNDEYQUUDOJXYGDJHDOLQHINDFNDEYQUUDOJXYGDTHDGYNDEN
QEDQHBDTINQHEDFNDEYQUUDOJXYGDFJGYDXLTFJHXDITHOJBNHINDQHBDXLTFJHXDEGLNHXGYDJ
HDGYNDQJLDFNDEYQUUDBNONHBDTSLDJEUQHBDFYQGNMNLDGYNDITEGDZQVDWNDFNDEYQUUDOJXY
GDTHDGYNDWNQIYNEDFNDEYQUUDOJXYGDTHDGYNDUQHBJHXDXLTSHBEDFNDEYQUUDOJXYGDJHDGY
NDOJNUBEDQHBDJHDGYNDEGLNNGEDFNDEYQUUDOJXYGDJHDGYNDYJUUEDFNDEYQUUDHNMNLDESLL
NHBNLDQHBDNMNHDJODFYJIYDJDBTDHTGDOTLDQDZTZNHGDWNUJNMNDGYJEDJEUQHBDTLDQDUQLX
NDRQLGDTODJGDFNLNDESWCSXQGNBDQHBDEGQLMJHXDGYNHDTSLDNZRJLNDWNVTHBDGYNDENQEDQ
LZNBDQHBDXSQLBNBDWVDGYNDWLJGJEYDOUNNGDFTSUBDIQLLVDTHDGYNDEGLSXXUNDSHGJUDJHD
XTBEDXTTBDGJZNDGYNDHNFDFTLUBDFJGYDQUUDJGEDRTFNLDQHBDZJXYGDEGNREDOTLGYDGTDGY
NDLNEISNDQHBDGYNDUJWNLQGJTHDTODGYNDTUB
```

Let us compare the three highest letter frequencies in the ciphertext with those in the text corpus (see Table 3):

War and Peace	␣	e	t	*Ciphertext*	D	N	G
Frequency	0.1834	0.1017	0.0729		0.1966	0.1039	0.0716

So, it makes sense to try the substitutions

$$\texttt{D} \to \text{␣}\,, \quad \texttt{N} \to \texttt{e}\,, \quad \texttt{G} \to \texttt{t}$$

The result is interesting (to save space I have displayed only the first few lines):

```
Fe EYQUU XT TH tT tYe eHB Fe EYQUU OJXYt JH OLQHIe Fe EYQUU OJXYt TH tYe EeQE QHB
TIeQHE Fe EYQUU OJXYt FJtY XLTFJHX ITHOJBeHIe QHB XLTFJHX EtLeHXtY JH tYe QJL Fe
EYQUU BeOeHB TSL JEUQHB FYQteMeL tYe ITEt ZQV
```

- Suddenly we can see probable *word boundaries*!
- Other conjectures come into our mind: the word `tYe` is likely to mean plaintext `the`, which is the most common three-letter word in English. So we may try `Y` $\to$ `h`.

[2] I shall keep the key secret!

```
Fe EhQUU XT TH tT the eHB Fe EhQUU OJXht JH OLQHIe Fe EhQUU OJXht TH the EeQE QHB
TIeQHE Fe EhQUU OJXht FJth XLTFJHX ITHOJBeHIe QHB XLTFJHX EtLeHXth JH the QJL Fe
EhQUU BeOeHB TSL JEUQHB FhQteMeL the ITEt ZQV
```

What about the word `EhQUU` which appears three times in the first line? May be, it means `shall`? Give it a try:

$$\texttt{E} \to \texttt{s}\,, \quad \texttt{Q} \to \texttt{a}\,, \quad \texttt{U} \to \texttt{l}\,:$$

```
Fe shall XT TH tT the eHB Fe shall OJXht JH OLaHIe Fe shall OJXht TH the seas aHB
TIeaHs Fe shall OJXht FJth XLTFJHX ITHOJBeHIe aHB XLTFJHX stLeHXth JH the aJL Fe
shall BeOeHB TSL JslaHB FhateMeL the ITst ZaV
```

Hm, looks interesting ...

However, from now on the work of the cryptanalyst becomes truly hard and messy. She has to try several *conjectures* about text snippets so that the text becomes closer and closer to English text, that the text becomes more *plausible*.

What cryptanalysts often do at this point is to form *contact tables*, meaning they gather statistics about the occurrence of *bigrams* in the ciphertext and compare these with statistics collected from a text corpus.

A *bigram* is just a 2-letter sequence in text. E.g., our original ciphertext begins with bigrams

`FN–ND–DE–EY` ...

What is lurking behind is a remarkable theory about human language. Early as 1906 A. Markov performed frequency counts of bigrams in Alexander Pushkin's *Eugene Onegin*. He used these to demonstrate and later prove an important extension of the *Law of Large Numbers* to *dependent trials*. That was the origin of one of the most important classes of *stochastic processes*, *Markov Chains*. This idea has been continued and extended by Claude Shannon in his foundation of a mathematical theory of communication (see the annotated bibliography at the end).

In *War and Peace* the most frequently occurring bigrams are:

Bigram	Frequency	Count
e_	0.0361	111116
_t	0.0285	87587
d_	0.0247	75995
he	0.0244	75022
th	0.0239	73400
_a	0.0226	69556
s_	0.0204	62865
t_	0.0189	58198
_h	0.0162	49954
in	0.0157	48180

You can see from these statistics that the letter `e` is most likely to occur at the end of a word, whereas `t` very often appears at the beginning of a word.

However, a by intuition guided process of trial and error as we applied when consulting frequencies of simple letters is very hard to carry out.

Is it possible to run the cryptanalytic process somehow automatically so that permanent human interventions can be avoided?

Yes! Here is a solution.

1.5 Combinatorial Optimization

Combinatorial optimization is a class of methods that deal typically with problems of very high dimension (= number of variables). In most cases of interest the variables are discrete and the space of possible solutions is finite-dimensional. Hence, in principle, it is possible to find the optimal solution by brute force, i. e., by *complete enumeration* of all solutions. But only in a few cases this strategy is viable, the number of admissible solutions is usually exorbitantly large, too large for such an unsophisticated approach. Indeed, most combinatorial optimization problems are *very hard* in a well defined mathematical sense.

The classic in combinatorial optimization is the *Traveling Salesman Problem (TSP)*: a salesman has to visit customers in n different cities. If the distances between each pair of cities is known, we have to find a tour such that

- each city is visited exactly once;
- the salesman returns to the city where his tour started;
- the tour has minimum length.

Technically, the problem reduces to finding a *permutation* of the numbers $1, 2, \ldots, n$ such that an *objective function* is minimized. Here the objective function assigns each permutation the total length of the corresponding tour.

Can you see parallels to the problem of breaking a monoalphabetic substitution cipher? It's the unknown key which is also a permutation of the standard alphabet! So, may be we can learn something from combinatorial optimization?

However, to find the optimum of a combinatorial optimization problem such as the TSP one has usually resort to *iterative search procedures*. For this purpose several heuristic and meta heuristic algorithms have been developed. Among them one of the most successful is *simulated annealing*. A special adaptation of this meta heuristic is the *Metropolis Algorithm* which is very well suited for some cryptanalytic purposes.

First we need an *objective function*. In our case this will be a *plausibility measure* $f(p)$ defined on the set $\mathfrak{S}_{27}$ of all permutations (= keys) p of the standard alphabet. For this function $f(p)$ we require that it should preferably assume high values if the text deciphered with p is close to English text. We want to measure this by using bigram statistics in such a way that $f(p)$ takes on high values when the bigram frequencies in the decrypted text most closely match those of some reference text like *War and Peace*. Examples of plausibility measures may be found in Diaconis (2008) and Chen and Rosenthal (2010).

The Metropolis Algorithm runs roughly as follows:

- Fix a *scale parameter* $\alpha > 0$.
- Create an initial permutation p_0, e.g. a random permutation on $\mathfrak{S}_{27}$ and calculate the plausibility measure $f(p_0)$.
- Repeat the following steps for a sufficient number of iterations.
  - Given p_0 create a new permutation p_1 in a *uniform way* (to be explained shortly) and calculate the plausibility $f(p_1)$.
  - Sample a pseudo random number u having a uniform distribution on the interval $[0, 1]$.
  - if $u < \left(\frac{f(p_1)}{f(p_0)}\right)^{\alpha}$ then accept the new key $p_1 : p_0 \leftarrow p_1$. Otherwise reject p_1 and leave p_0 unchanged.

A few remarks are in order:

- If the new key p_1 yields higher plausibility than p_0, then $\left(\frac{f(p_1)}{f(p_0)}\right)^{\alpha} > 1$ and since $u \leq 1$, the better solution p_1 is always accepted.
- If the new key p_1 results in smaller plausibility than p_0, then p_1 may be still accepted with probability $\left(\frac{f(p_1)}{f(p_0)}\right)^{\alpha}$. This idea lies at the heart of simulated annealing: it allows us to escape a *local maximum* by temporarily accepting a worse solution.
- The scale parameter α, typically chosen close to 1, influences the probability of accepting a worse solution. It is closely related to the concept of *temperature* in simulated annealing.
- The new key p_1 can be created in many ways uniformly out of p_0. The most common technique is to select two different entries of p_0 at random and exchange them so to from the new key p_1. This is also called a *transposition*.

So, it's time to try that. I crafted in a more or less quick and dirty fashion an implementation of this algorithm in the C programming language and used the plausibility measure proposed by Chen and Rosenthal (2010). Here are the results:

```
 200   HTEL ASSENUEUREYUEY TETRIEHTEL ASSEPON YEOREPMARVTEHTEL ASSEPON YEUREY TELTAL
 400   HA REISS DU UN MU MEA ANT HA REISS CODEM ON CLINVA HA REISS CODEM UN MEA RAIR
 600   HA REISS FU UN DU DEA ANT HA REISS COFED ON CLINPA HA REISS COFED UN DEA RAIR
 800   HA REISS FU UN TU TEA AND HA REISS COFET ON CLINGA HA REISS COFET UN TEA RAIR
1000   OE SHILL FU UN TU THE END OE SHILL CAFHT AN CRINGE OE SHILL CAFHT UN THE SEIS
1200   ME SHILL GO ON TO THE END ME SHILL PAGHT AN PRINCE ME SHILL PAGHT ON THE SEIS
1400   ME SHALL GO ON TO THE END ME SHALL WIGHT IN WRANCE ME SHALL WIGHT ON THE SEAS
1600   ME SHALL GO ON TO THE END ME SHALL WIGHT IN WRANCE ME SHALL WIGHT ON THE SEAS
1800   WE SHALL GO ON TO THE END WE SHALL PIGHT IN PRANCE WE SHALL PIGHT ON THE SEAS
2000   WE SHALL GO ON TO THE END WE SHALL FIGHT IN FRANCE WE SHALL FIGHT ON THE SEAS
```

Actually after 2000 iterations we have got the plaintext (with correct interpunctuation):

> *We shall go on to the end, we shall fight in France, we shall fight on the seas and oceans, we shall fight with growing confidence and growing strength in the air, we shall defend our Island, whatever the cost may be, we shall fight on the beaches, we shall fight on the landing grounds, we shall fight in the fields and in the streets, we shall fight in the hills; we shall never surrender, and even if, which I do not for a moment believe, this Island or a large part of it were subjugated and starving, then our Empire beyond the seas, armed and guarded by the British Fleet, would carry on the struggle, until, in God's good time, the New World, with all its power and might, steps forth to the rescue and the liberation of the old.*
>
> Winston S. Churchill, House of Commons, June 20, 1940

It is quite amazing how quickly and automatically the text was deciphered.

Many important question arise now, but we shall defer these to section 2.

1.6 The Vigenère Cipher, le chiffre indéchiffrable

To summarize our findings: monoalphabetic substitution ciphers can be broken routinely by frequency analysis of letters or bigrams. The major reason for this weakness is that a particular letter in plaintext is always mapped *to the same* latter in ciphertext. Thus there are always revealing footprints in the frequencies of letters in ciphertext. Provided the ciphertext is sufficiently long frequency counts give us statistically significant signals which can be used to break a cipher. So these systems cannot be considered secure.

This all was certainly known in the 15th century. Leon Battista Alberti (1404-1472), a remarkable Renaissance scholar working as poet, architect, painter and also as cryptographer was apparently the first to propose a cipher which (seemingly) rules out frequency analysis. His idea was to use more than one alphabet for encryption. By doing so, one and the same letter will have several equivalents in ciphertext. E.g., a plaintext `a` may be mapped into `W` on its first occurrence. The next `a` may be mapped to `B`, etc. It is this idea which forms the basis of what is known as *polyalphabetic substitution*. Observe the intended effect of polyalphabeticity: it is to *flatten* the distribution of letter frequencies f_i as far as possible. There will be still peaks in the distribution but they are merely the result of the random variation of the statistical estimates f_i.

But, what ciphertext alphabets should be used?

How do we determine which alphabet has to be used in a particular step of the encryption process?

Several solutions have been proposed among these the classical Vigenère cipher by Blaise Vigenère in 1586[3].

[3] Actually this cipher has been invented earlier by Giovanni Battista Bellaso in 1553.

1.6.1 Encryption and decryption

The Vigenère cipher uses a table of alphabets, called *tabula recta*, see Table 6. It is has 27 lines, the line d being our standard alphabet shifted *left* by $d = 0, 1, \ldots, 26$. Thus these are all Caesar's codes!

	_	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
_	_	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
A	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	Y

Table 6: The Vigenère table for our standard alphabet

Encryption is best explained by an example. In the sequel we assume a secret key k is used for communication and that the length of k is < 27, the size of the standard alphabet[4].

Example 6. Suppose we want to encipher the plaintext *meet you on sunday at nine* with key `PLAYFAIR`. First we form the *key string* by perpetuating the key until the key string has the length of the plaintext:

Key	`PLAYFAIRPLAYFAIRPLAYFAIRPL`
Plaintext	`meet you on sunday at nine`
Ciphertext	`BQFRFZXLP OYYVWVQJAZZAW CQ`

Then process each letter of the plaintext in turn:

- The first plaintext letter `m` has to be enciphered with the first letter of the key string `P`.

[4]This assumption can be weakened considerably.

- For this purpose we use the alphabet in line `P` in the *tabula recta*.
- Using this alphabet the letter `m` is enciphered as `B`, etc.

Of course, using the Vigenère cipher this way is really awkward. As a result the application of the cipher for instance in the battle field was practically too difficult and cumbersome. But may be that people were not aware of *molular arithmetic*? Indeed, there is a very simple algebraic implementation close to that we used for Caesar's Cipher.

Let k denote the key string. For instance, in Example 6 we had

$$k = \texttt{PLAYFAIRPLAYFAIRPLAYFAIRPL},$$

and k_i the integer representation i-th letter of k which we obtain by applying Table 1. So $k_1 = 16, k_2 = 12$, etc. Furthermore, let a_i and c_i denote the integer representation of the i letter of the plaintext and the ciphertext. Then enciphering is most easily done by:

$$c_i = (a_i + k_i) \bmod 27, \qquad i = 1, 2, \ldots, n \tag{3}$$

where n is the length of the plaintext.

So:

`plain`	a_i	`key`	k_i	$a_i + k_i$	c_i	`cipher`
`m`	13	`P`	16	29	2	`B`
`e`	5	`L`	12	17	17	`Q`
`e`	5	`A`	1	6	6	`F`
`t`	20	`Y`	25	45	18	`R`

Decryption is done in the inverse way:

$$a_i = (c_i - k_i) \bmod 27, \qquad i = 1, 2, \ldots, n \tag{4}$$

Thus, *knowing the key* decryption is also very easy. But what, if we do not know the key?

1.6.2 Cryptanalysis

We have already remarked that the secureness of a system is measured by the size of its key space $\mathcal{K}|$. Let d denote the length of the key. E. g., for `PLAYFAIR` we have $d = 8$. Then (allowing repeated letters in the key):

$$|\mathcal{K}| = 27^d, \quad \text{in our example: } |\mathcal{K}| = 27^8 \doteq 2.82 \cdot 10^{11}$$

This is much less than for the monoalphabetic substitution cipher. But we know, simple frequency analysis is now out of business. This is why Vigenère was also called *le chiffre indéchiffrable*.

Still, statistical methods can be used for a very effective attack. There are two severe weaknesses of Vigenère which can be used to break it:

- The key string is formed by repeating the key. This generates a *periodicity* which may leave its footprints in ciphertext.
- The alphabets are simple cyclic permutations of the standard alphabet. Thus once an alphabet is chosen the corresponding plaintext letter is enciphered using a simple Caesar's Cipher which we know is very easy to break.

By these observations it should be clear that finding the *length* of the key is the crucial point. Once known the rest of the business is done by frequency analysis as we have it outlined in Section 1.3.

There are various approaches to find the key length d. One is based on the use of the *index of coincidence* $\Phi(T)$ invented by William F. Friedman[5].

Let T be a text over *some* alphabet. Assume that the length of T is $|T| = n$ and the alphabet consists of N letters. Then $\Phi(T)$ is an estimate of the probability that two randomly chosen letters in T are the same:

$$\Phi(T) = \frac{1}{N(N-1)} \sum_{i=1}^{n} F_i(F_i - 1),$$

where F_i denotes the *absolute frequency* of the i-th letter of the alphabet in the text T. For English text the value of $\Phi(T)$ is around 0.07, whereas for random text $\Phi(T) = 1/27 = 0.037$, all letters being equally probable.

Now let $T = C$, C being the cipher text and suppose the key has length d. Then we split C into d blocks:

$$\begin{aligned} C_0 &= [c_0, c_d, c_{2d}, \ldots] \\ C_1 &= [c_1, c_{d+1}, c_{2d+1}, \ldots] \\ C_2 &= [c_2, c_{d+2}, c_{2d+2}, \ldots] \\ &\ldots \\ C_{d-1} &= [c_{d-1}, c_{2d-1}, \ldots] \end{aligned}$$

If the key length is indeed d, then these blocks should look like English text for $\Phi(C_i)$. So we calculate the indices of coincidence for each group C_i and take the average:

$$\Phi(C) = \frac{1}{d} \left[\Phi(C_0) + \Phi(C_1) + \ldots + \Phi(C_{d-1}\right].$$

The value of $\Phi(C)$ should be around 0.07 if the key length is indeed d, otherwise $\Phi(C)$ will be much smaller.

Let's try this.

Example 7. Suppose we have intercepted the following message which we know (from some source) is enciphered using the Vigenère system and plaintext language is English.

[5]William Frederick Friedman (1891 - 1969) was one of the greatest cryptologist of all time.

```
GWGMRUVTMZSSATENRIBDRUFGNHUZRKODNHWTSOTIMFEBZZEUNHAUSEXI
ENRKODNHWTSONGWSWRNHRSS NN PAGMVEENHAUGSSGRLMFNP YRQECZI
KFNHUZRQSPPZRPTOFBWACPPCXJQOFBIPHVUUUPBDRLSUWC UNJGWMNZF
NPZZIJQP UYPITDHDFAHMBSTNUVHSMZMMWWFAOYIJUNHUZRKODNHWTSO
TIMFEBZZEUNFRAIFGGMNFAVPHZRUBO IKJTMMBWSSQKUKISONGWSWRNH
RHBJRLENSBFUKIOHMCEAIXRQRPTOFBWAOHFCKVRTMIXAGWRUSNSFVXSO
NVAPWSAARHKAWHMXSOACFUTVGOPIETWSRLRUVPFU UNXEU NCCEM CZT
MNFAETNXZAOBMVYSSTZZEUNHULFVUWM LSGWRLROSVAN BGXAHJ
```

For this text C I have calculated $\Phi(C)$ for conjectured key lengths $d = 1, 2, \ldots, 25$. The results are given in Figure 3. You can see the striking peaks at $6, 12, 18, 24$. This is a strong indication that the unknown key has length $d = 6$.

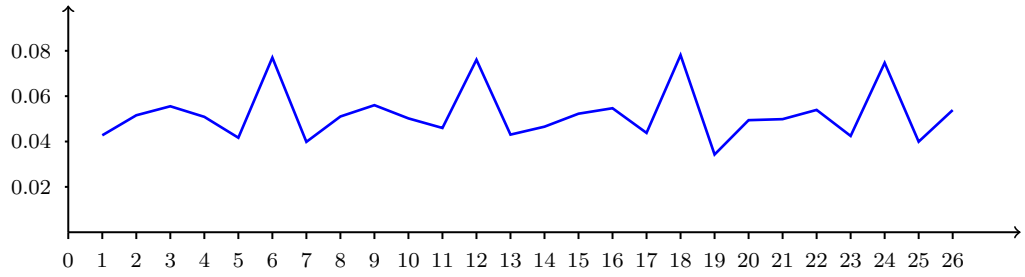

Figure 3: The index of coincidence for the cryptogram in Example 7

Once we have a good guess of the key length d the rest is easy. Since all alphabets of the Vigenère table are shifted Caesar's alphabets, just apply the technique outlined in Section 1.3 to determine the shift by means of total variation distance.

Having done so, we find that the secret key is `NOMURA` and the plaintext of the cipher is (with correct interpunctuation):

```
Thus, the earnest hope of the Japanese Government to adjust
Japanese-American relations and to preserve and promote the peace
of the Pacific through cooperation with the American government
has finally been lost.

The Japanese Government regrets to have to notify hereby the
American Government that in view of the attitude of the American
Government it cannot but consider that it is impossible to reach
an agreement through further negotiations.
```

This is indeed a famous document dating from December 7, 1941. It is the last page of the Japanese note handed to Secretary of State Cordell Hull while Pearl Harbor was being attacked by Japanese forces. Nomura was the name of the Japanese ambassador at Washington. The thrilling story about this cryptogram is told in Chapter 1: *A Day of Magic* of David Kahn's book (Kahn, 1996).

1.7 Transposition Ciphers

1.7.1 Encryption and decryption

We already talked briefly about transposition ciphers. These come in an impressing number of variants. The basic feature of these systems is that letters retain their value but change place in text.

Here I will describe only the simplest system.

The correspondents agree upon a secret key p which is a permutation of the numbers $1, 2, \ldots, d$ for some $d > 1$. The plaintext is divided into blocks of length d and letters *within* each block are permuted according to p.

Example 8. The plaintext is `troops gathering attack from north` and this should be enciphered with key $p = [4\ 1\ 5\ 3\ 2]$. Thus $d = 5$. Encryption goes this way (for reasons of readability the space character is printed as underscore letter):

key p	41532	41532	41532	41532	41532	41532	41532
plaintext	troop	s_gat	herin	g_att	ack_f	rom_n	orth_
ciphertext	OTPOR	ASTG_	IHNRE	TGTA_	_AFKC	_RNMO	HO_TR

Decryption is also easy, just take the *inverse permutation* p^{-1} and recover the plaintext from the ciphertext.

Example 8. (continued) The inverse permutation is found easily by writing p as 2-rowed array, sorting the second row and exchanging rows:

$$p = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 1 & 5 & 3 & 2 \end{pmatrix} \Longrightarrow \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 5 & 4 & 1 & 3 \end{pmatrix} = p^{-1}$$

Hence:

inverse p^{-1}	25413	25413	25413	25413	25413	25413	25413
ciphertext	OTPOR	ASTG_	IHNRE	TGTA_	_AFKC	_RNMO	HO_TR
plaintext	troop	s_gat	herin	g_att	ack_f	rom_n	orth_

1.7.2 Cryptanalysis

Transposition ciphers are just what cryptanalysts are waiting for, as was reported e.g. about the German *Abwehr* during World War II. The key space has size $d!$ which is quite small unless d is large. In Example 8 we have $5! = 120$, thus a brute-force attack doesn't result in serious computational trouble. Frequency analysis, however, seems to be out of business now. Though ..., see Section 2 for more about this and related questions.

1.8 Perfect Secrecy

After having been introduced to some classical ciphers and after you have seen that these can be broken rather routinely you may wonder whether there exists

a cipher system that cannot be broken, a system guaranteeing *perfect secrecy.*

Here is a naive argument which tells us: No, there can't be perfect secrecy, because

- All messages we send and receive have finite length, they consist of a finite number of symbols, thus require finite time for transmission.
- Therefore it is always possible to find the plaintext of a ciphertext by brute force. It's just a matter of time and computing power. But, of course, it make take quite some time.

All right, but still there is a fatal flaw in this argument which I will demonstrate drastically in Example 9 below.

Actually, there are cipher systems giving us perfect secrecy. In informal terms: *a cipher system has perfect secrecy, if the unauthorized eavesdropper learns nothing about the plaintext from the ciphertext.*

This informal statement can be made strict by means of conditional probability. Perfect secrecy has been defined and thoroughly discussed by Claude Shannon (1948) and (1949) in his seminal papers.

An example of a system having this remarkable property is *Vernam's Cipher.* It is breathtakingly simple!

We will apply an algebraic representation as we have used it with Caesar's and Vigenère's cipher. The basic ingredient of Vernam's cipher is the key:

- it must have length equal to the length of the plaintext;
- it must be absolutely random.

By Table 1, there is a one-to-one correspondence between letters of the standard alphabet and the integers $0, 1, \ldots, 26$. Again, let a_i denote the numerical value of the plaintext letter at position i, k_i the value of the key letter and c_i that of the ciphertext letter. Assume that the plaintext has length n. Then we have:

$$\begin{aligned} &\text{encryption:} \quad c_i = (a_i + k_i) \bmod 27, \qquad i = 1, 2, \ldots, n \\ &\text{decryption:} \quad a_i = (c_i - k_i) \bmod 27 \end{aligned}$$

Example 9. In the following table a plaintext and a key of equal length are given and enciphered:

plaintext	`enemy will surrender tomorrow`
key	`TQPLNFDZYADMNQS SP RUSJYDTOFP`
ciphertext	`YDUYLF HJMDEHHJEFTEIULYKSKFUL`

For the first two letters:

$$\begin{aligned} a_1 = \texttt{e} \mathrel{\widehat{=}} 5, \quad k_1 = \texttt{T} \mathrel{\widehat{=}} 20, \quad c_1 = (5 + 20) \bmod 27 = 25 \mathrel{\widehat{=}} \texttt{Y} \\ a_2 = \texttt{n} \mathrel{\widehat{=}} 14, \quad k_2 = \texttt{Q} \mathrel{\widehat{=}} 17, \quad c_2 = (14 + 17) \bmod 27 = 4 \mathrel{\widehat{=}} \texttt{D} \end{aligned}$$

What about *cryptanalysis*? Let us assume that the cryptanalyst knows that this is a Vernam cipher but he does not know the key.

Indeed, the cryptanalyst is in a very weak position now. The plaintext has length $n = 29$ which means that there are 27^{29} keys to be considered in the worst case if a brute-force attack is run. But:

$$27^{29} = 323257909929174534292273980721360271853387 \simeq 3 \cdot 10^{41}$$

Of course, only a relatively small percentage of keys will yields sensible English text. Eventually the cryptanalyst may find the correct key. But during key search with some positive probability he may also come across the key `LCHXLJRWYMTWWWJPEDDIAZLGATRBL`, which (alas!) yields:

ciphertext	`YDUYLF HJMDEHHJEFTEIUNYVSKFYL`
key	`LCHXLJRWYMTWWWJPEDDIAZLGATRBL`
plaintext	`mama will kill papa tomorrow`

In other words, an exhaustive key search will yield all sensible English[6] text of a given length. Thus the cryptanalyst runs into a difficult decision problem which can hardly be resolved! That's why Vernam's cipher is perfect. Yet it is not foolproof!

2 What I want from You

After having read this *Invitation* so far (about 20 pages!) you may wonder whether there is anything left to do for you?

There remains quite a lot of work to be done.

Write an interesting and exciting thesis about elementary methods of cryptology. Your paper should be a nice mix of theoretical considerations, historical notes and, of course, it should also have a computational flavor. Your thesis should also contain several examples to demonstrate your findings.

2.1 Mandatory material

- At the outset you should make up your mind what programming language you will use. For instance, all examples in this introduction are written in C. But you are free to use any other language like Java, R, etc. It may also be helpful to use some scripting language like perl.

- Next organize an appropriate text corpus to have a learning sample. The texts may be in English or German and should be in total sufficiently long (about 3 mega bytes, or so). Based on this learning sample:
  - Perform a careful statistical analysis of these texts.
  - Determine frequencies of letters, bigrams, may be also of trigrams (sequences of three contiguous letters).

[6] of course also German, French, Russian,...

Regarding the text corpus: you may use English or German texts, but take care of copyright protections.

- Implement an automatic decryption routine for Caesar's cipher. You may or may not use total variation distance. There is also a quadratic measure which will remind you in the χ^2-statistic.

- Give a careful discussion of the Metropolis algorithm. It is actually a special case of a *meta heuristic* known as *Simulated Annealing*. SA is capable of more, the driving master process of SA is able to intensify and diversify search.

- Implement the Metropolis algorithm to break a monoalphabetic cipher. Try to be as general as possible so that your implementation can be easily reused to solve other, harder problems.

- Give a thorough discussion of the classical Vigenère cipher and implement a routine which can break this system.

- Discuss in detail Friedman's index of coincidence $\Phi(T)$ and related measures.

- Implement a routine to solve the simple transposition cipher introduced in Section 1.7.

- I have remarked that the Vernam cipher is not foolproof. It can be broken if not used properly. Give a careful discussion of the conditions for proper use of this cipher.

2.2 Optional material

- When playing and experimenting with the Metropolis algorithm, for instance, you will find out, that to break a cipher you will need a minimum amount of ciphertext available, the more, the better. Indeed, there is a minimum length of ciphertext needed to guarantee a *unique* decryption. This length is known as *unicity distance* U and it is closely related to the concepts of *entropy* and *redundancy*. For simple substitution ciphers U is surprisingly small, about $U = 30$ for English text. However, for the Vernam cipher, $U = \infty$. Discuss U.

- Recall that the fatal weakness of Vigenère's cipher is periodicity generated by the key, and this can hardly be hidden. We have already discussed finding the key length by means of the index of coincidence. But there is another famous method: *Kasiski's Test*[7], which tries to identify recurrent patterns in the ciphertext and deduce thereby the length of the secret key. Discuss and implement Kasiski's test.

[7] Friedrich Wilhelm Kasiski (1805 - 1881) was a prussian infantry officer. In 1863 he published a small booklet on cryptology which became on of the most influential and important works in this field.

- Once the length of the key is known the classical Vigenère cipher is rather straightforward to break because the alphabets used are simple shifted Caesar's alphabets. But the strength of Vigenère can be boosted considerably when the Vigenère table consists of (in our cases 27) different *random permutations*. The size of the key space is thereby increased from 27^d to $(27!)^d$ where d is the length of the key. Devise an algorithm to break this general Vigenère cipher. Metropolis may be helpful in this context.

- Actually, using more general alphabets in the Vigenère table is an old idea already suggested by Porta[8]. Vigenère actually invented another system to generate the secret key, the *autokey cipher*. This method uses the plaintext to become part of the key. Discuss the idea of autokey and, if possible, implement a method to break the resulting cipher.

- So far we have only seen simple transposition ciphers, but there is an incredible number of variants. You may also discuss one or the other example of more exotic transposition systems.

- An interesting questions is: how can the cryptanalyst find out *what* cipher is used? Are there methods to identify the cipher system?

2.3 What to be avoided

Your thesis should cover basic cryptology up to 1918, the end of World War I. THis is a key date, as early in 1918 Arthur Scherbius patented the first electromechanical cipher machine based an rotors, the *Enigma Machine* and this initiated a new era in cryptology.

I would appreciate if you avoid discussing ciphers like Enigma and its various descendants, the *Data Encryption Standard (DES)* or the *Advanced Encryption Standard (AES)* which is widely used today. Also ciphers based on number theory like RSA should not be topic in your thesis. All these are very advanced systems requiring special mathematical methods that I cannot afford from you.

Please do not be disappointed about this restriction.

Now, after having worked out the major issues there remains on final job for me to be done: *Enjoy writing this thesis! Have fun!*

3 An annotated bibliography

The book Kahn (1996) is *the classical text* about the history of cryptology. This is really an exciting book covering the field from ancient times up to the end of the 20th century. Chapter 1, *One Day of Magic* is the thrilling story of American codebreakers around William Friedman and the Japanese attack on Pearl Harbor in 1941. The crucial Japanese diplomatic notes (we have seen

[8] Giambattista della Porta, 1535-1615, Renaissance scholar.

the last one in Section 1.6.2) were, of course, not encrypted in Vigenère. Japan used several much stronger systems, practically all broken by the United States Signal Intelligence Service to which William Friedman belonged.

Bauer (2007) is an excellent introduction to cryptography and cryptanalysis. The book has two parts. In the first part standard methods of cryptography are introduced. The second part is devoted to cryptanalysis. The book is full of interesting examples. Part II gives a fairly complete coverage of the most important statistical methods for cryptanalysis. No special mathematical knowledge is required to read and understand this book except for some basic terms like relations and functions and the corresponding mathematical notation. There is also a German edition.

Modern cryptanalysis by Swenson, 2008 is another remarkable textbook on the subject, however, only Chapter 1 will be relevant for your thesis.

You will also enjoy the booklet by Gains (1956). Chapters 1-7 are devoted to transposition ciphers, chapters 8-23 to substitution ciphers. It contains many solved examples and you will find here also a thorough discussion of polyalphabeticity, in particular of the Kasiski Test.

The paper Diaconis (2008) discusses the Metropolis algorithm for breaking monoalphabetic ciphers. Only the first few pages will be interesting for you because there you find a description of the Metropolis algorithm and a plausibility measure. The major part of this article deals with convergence problems and representation theory of finite groups. Chen and Rosenthal (2010) is a technical report, you will find very interesting. It introduces the Metropolis algorithm with an alternative plausibility measure and discusses applications to various ciphers including simple transposition. Also, the authors report some statistics recorded in their experiments with different text corpora and different choices of parameters. An abridged version having been published (Chen and Rosenthal, 2012) in *Statistics and Computing*.

4 References

[1] Friedrich L. Bauer. *Decrypted Secrets, Methods and Maxims of Cryptology*. Springer, 2007.

[2] Jian Chen and Jeffrey S. Rosenthal. *Decrypting classical cipher text using Markov chain Monte Carlo*. 2010. URL: `http://probability.ca/jeff/ftpdir/decipherart.pdf`.

[3] Jian Chen and Jeffrey S. Rosenthal. "Decrypting classical cipher text using Markov chain Monte Carlo". In: *Statistics and Computing* 22.2 (2012), pp. 397–413.

[4] Persi Diaconis. "The Markov chain Monte Carlo revolution". In: *Bulletin of the American Mathematical Society* 2 (2008), pp. 179–205.

[5] Helen Fouché Gains. *Cryptanalysis, a Study of Ciphers and Their Solution*. Dover Publications, 1956.

[6] L. Graham Ronald, Donald E. Knuth, and Oren Patashnik. *Concrete Mathematics*. 2nd ed. Addison-Wesly, 2003.

[7] David Kahn. *The Codebreakers: The Comprehensive History of Secret Communication from Ancient Times to the Internet*. Scribner, New York, 1996.

[8] W. W. Rouse Ball and H. S. M. Coxeter. *Mathematical Recreations and Essays*. 13th. Dover Publications, 1987.

[9] Claude E. Shannon. "A Mathematical Theory of Communication". In: *The Bell System Technical Journal* 27.3 (1948), pp. 379–423.

[10] Christopher Swenson. *Modern Cryptanalysis: Techniques for advanced Code Breaking*. Wiley Publications, 2008.