HCPortal Modules for Teaching and Promoting Cryptology

Eugen Antal
Slovak University of Technology in Bratislava
Slovakia
eugen.antal@stuba.sk

Pavol Zajac
Slovak University of Technology in Bratislava
Slovakia
pavol.zajac@stuba.sk

Abstract

HCPortal is an online portal focusing on historical cryptology. The structure of the portal is logically divided into modules. In this paper, we are presenting three new modules focusing on teaching and promoting cryptology. The first module is called Education. It contains a demonstration of selected classical ciphers and their respective cryptanalytic techniques. The second module focuses on nomenclators. The third module represents a virtual museum of historical ciphers.

1 Introduction

The Portal of Historical Ciphers (HCPortal) is an online portal consisting of several web pages and tools (logically divided into modules). Each module represents a specific topic related to historical cryptology.

In the first y ears o f d evelopment, a s eries of modules were released. These modules are: *Home page* (entry point of the portal with navigation and information centre), *ManuLab* and *ManuLab online* (software product for statistical analysis, with a public API and example web page), *Tools and web pages* (links to external projects) and *Glossary* (glossary for historical cryptology, including codes and nomenclator terminology). The portal also features a special *Database of cryptograms*, containing a collection of cryptograms. A detailed overview of these modules is available in Antal and Zajac (2020).

In 2020, new modules were designed and developed (some of them are still in progress) focusing on teaching and promoting cryptology. The first module is called *Education*. It contains a demonstration of some classical ciphers and their respective cryptanalytic techniques. Each technique is accompanied by a visualization. Currently, its main use is as a support tool for a Classical Ciphers course at the Slovak University of Technology in Bratislava.

The second module focuses on *nomenclators*. This module is not yet released to the public. It contains a special online tool designed to create, use, and share nomenclator keys. The second part of this module is a special client-server application. The server contains an online database containing digitized nomenclator keys with a public API. The client part is a desktop application (with access to the server part) that supports transcription of nomenclator keys from images.

The third module represents a *virtual museum* of historical ciphers. It is built on a virtual reality framework supported by modern web browsers. The goal is to promote public interest in ciphers using modern technologies. This module is also a work in progress, and not yet released to the public. The estimated release date of the nomenclator and virtual museum modules is at the end of the year 2021.

2 The Education Module

There are several websites on the internet, dedicated to historical cryptography. Some of them, including dCode (2020), Cryptii (2020), CTO (2020), provide implementations of different classical ciphers. These and similar other sources pro-vide students with an opportunity to interact with the cipher algorithm and learn basic encryption and decryption steps. However, there is only a limited number of publicly available tools that contain fully described algorithms used in cryptanalysis. Moreover, a lot of the sources for cryptanalytic techniques lack interactive visualization that is suitable for educational purposes.

The primary goal for the inclusion of the *Education* module to HCPortal was to support an online education in our course Classical Ciphers taught at the Slovak University of Technology in Bratislava. However, we have designed the tools in such a way, that they can be used by other students and the general public.

The *Education* module is a collection of interactive tools with graphical visualization of the data designed for a better understanding of attacks on selected classical ciphers. At the moment, the *Education* module contains:[1]

- Brute-force attack on *Caesar Cipher*
- Hill-Climbing attack on *Simple Substitution Cipher*
- Friedman test and brute-force attack on *Vigenère Cipher*

Currently available attacks were implemented in Angular. The implemented attacks are accessible through the navigation menu option or the main screen (Figure 3).

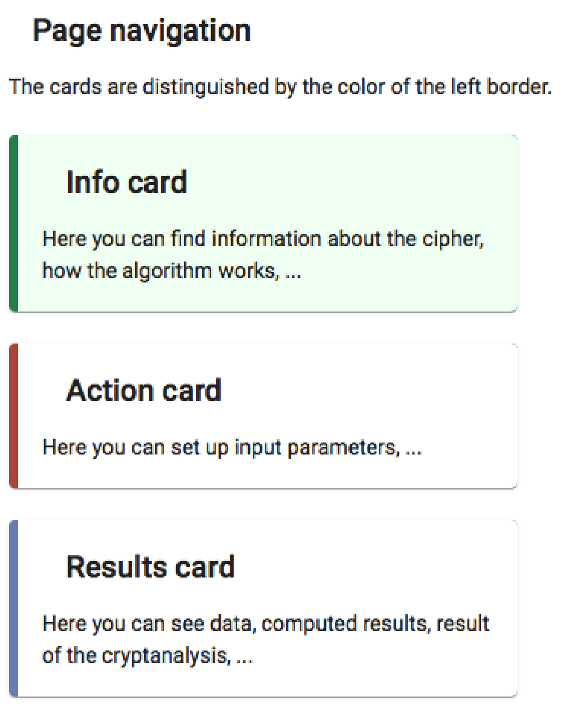
Each demonstrated attack is divided into logical steps. For visualization, we attach special *cards* to each of these steps. There are three card types, which are distinguished by the color of the left border (Figure 1). The cards are used to describe details of the current step of the attack, to display the computed results, or to serve as user input.
The *Education* module is available at the following web address: `https://www.edu.hcportal.eu`.

2.1 Brute-force Attack on Caesar Cipher

The first implemented attack is a brute-force attack on Caesar Cipher. The Caesar cipher is a simple type of substitution cipher suitable as a basic introduction to the topic of encryption algorithms even for young children. It replaces every plain text letter with a different one. Each letter is shifted by n letters in the used alphabet (26 letters for the English alphabet). The shift is defined by the key (originally, Caesar used a shift by 3 letters). The keyspace is very small, there are only 25 possible shifts (if we exclude the identity).

Despite the fact that Caesar Cipher can be cracked easily by hand, a good example can be created to demonstrate a brute-force attack. We have also decided that the Caesar cipher example is a good way to introduce a more general topic

[1]Additional specialized attacks on transposition ciphers will be also included soon.


Page navigation

The cards are distinguished by the color of the left border.

Info card

Here you can find information about the cipher, how the algorithm works, ...

Action card

Here you can set up input parameters, ...

Results card

Here you can see data, computed results, result of the cryptanalysis, ...

Figure 1: Education module - cards description

of frequency analysis, which is used to break the general monoalphabetic substitution cipher.

The demonstration in our tool consists of the following steps. In the first step, a short description of the Caesar cipher is presented in an *info card*. The next card is used by the student to set up the input plaintext for the example. Currently, we only allow using English text as an input. After the input plaintext and the key is provided, the ciphertext is computed by the engine and passed to the next step.

Frequency analysis is performed on the encrypted text. From the obtained result we are guessing the used language by the index of coincidence (Figure 5). The measured index of coincidence is compared with the reference value of 6 languages and with the minimal value of the index (representing a random text). This step does not influence the attack.

The attack starts with a short description in an *info card*. Computed frequencies of letters in the encrypted message are presented. In the next step, an exhaustive search algorithm through all possible combinations of the key is performed. Unlike other similar tools, our goal is to demonstrate the automation of such attacks. Instead of evaluating possible plaintexts by the user, each candidate plaintext is evaluated automatically with a specific scoring function: the Manhattan distance (Minkowski's L1 distance) of letter frequencies of the decrypted text from reference values. The overall result is displayed in a table for all possible keys. The table is ordered by the obtained score, and also contains the distances of measured frequencies from the reference values for each letter.

In the last step, a special frequency chart is displayed, where the letter frequencies can be compared with the reference values for every possible Caesar shift (Figure 4).

2.2 Hill-Climbing Attack on Simple Substitution Cipher

Simple substitution replaces letters of the plaintext with letters of the ciphertext based on predefined rules. Each letter from the plaintext alphabet maps exactly to one letter from the ciphertext alphabet. In comparison with the Caesar cipher[2] described in section 2.1, the keyspace (26! for English letters) of the generalized simple substitution can not be searched by brute-force.

This demonstration is similar to that presented in section 2.1. Instead of searching the whole keyspace, we show the students how to use an optimization algorithm (in this case the Hill-Climbing algorithm) to find the key. The goal was to provide insights on the details of the Hill-Climbing algorithm, and on what happens during the keyspace search. The main principle of the attack, however, stays the same: we explore a part of the keyspace in an intelligent way, score the used key based on a characteristic of plaintext candidates, and present the student with the scores and visualization of the steps. As a scoring function, we use bigram statistics instead of just letter frequencies. The input plaintext must be again an English text.[3]

After setting up the input, a short description of the attack follows. The attack has only two parameters: the number of iterations and the number of restarts (Figure 6). The best found key candidate is presented in a result card.

There are three additional and important result cards. In the first one, the evolution of the score and the match rate is presented. There are key candidates in a table, picked in different computation cycles (iterations). Each row contains the key, its score, the text decrypted using that key, and the match rate of the decrypted text. This gives some information about how the key/text candidate was changed during the search process. The second card visually compares the change of the score and match rate during the search process (Figure 7, first card).

Hill-Climbing is designed to accept only better candidates (improvements). It is also visible on the first card on Figure 7 - the score (marked as Sum) is only decreasing[4] during the whole search process. Using the third result card the relative change of the match rate is presented. It's important to show, that the match rate does not strictly follow the change of the score. We can accept a new candidate that produces a better score but worse match rate than the previous candidate (Figure 7, second card).

2.3 Friedman Test and Brute-force Attack on Vigenère Cipher

The Vigenère cipher is a polyalphabetic substitution. The encryption consists of series of Caesar ciphers, depending on the letters of a keyword. The keyword is repeated periodically. The first step of the analysis is the Friedman test, which helps to determine the length of the key. The next step is to try all possible combinations of the key. Each key is evaluated with a specific scoring function to find the correct one. The input plaintext must be an English text.

After setting up the input text, the encrypted text is displayed. Letters shifted by the same Caesar shift are highlighted with the same color. This visualization helps to better understand the encryption process. See Figure 2 as an example (text is encrypted with a key of length 4).

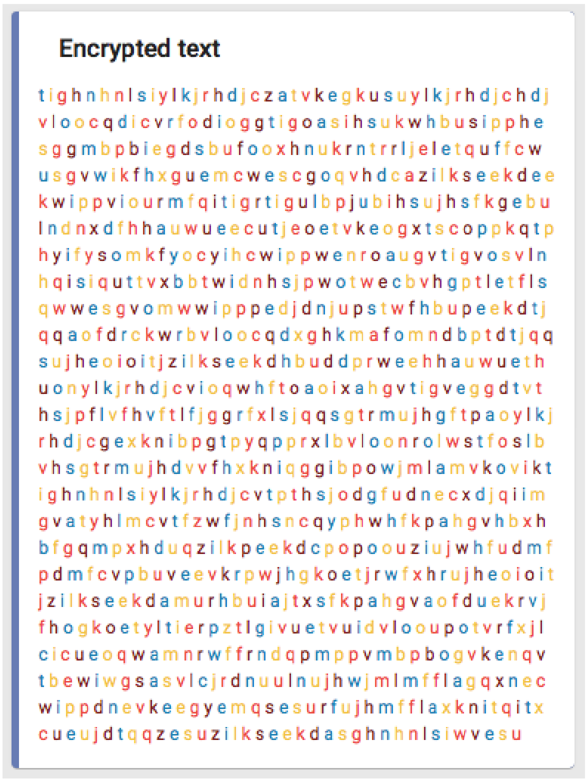
The Friedman test is a well-known method for determining the length of the key in the Vigenère cipher and is based on the notion of the index of coincidence (IC) . If the text is written in English, then we expect the value of IC to be close to 0.065. If the text was generated randomly, then the expected value of IC would be 0.038.

An English text encrypted by the Vigenère cipher with a key of length r, can be divided into r cosets where each coset contains letters shifted by the same Caesar shift. Therefore, the value of IC of each coset is expected to be close to 0.065.

[2]Which is a subset of a generalized simple (also called monoalphabetic) substitution cipher.

[3]The reference English bigram values are calculated from the Open American National Corpus, as probabilities.

[4]Minimisation problem - the scoring function calculates the distance of two vectors. The goal is to find a key that produces minimal distance.


Encrypted text

```
tighnhnlsiylkjrhdjczatvkegkusuylkjrhdjchdj
vloocqdicvrfodioggtigoasihsukwhbusipphe
sggmbpbiegdsbufooxhnukrntrrljeletquffcw
usgvwikfhxguemcwescgoqvhdcazilkseekdee
kwippviourmfqitigrtigulbpjubihsujhsfkgebu
lndnxdfhhauwueecutjeoetvkeogxtscoppkqtp
hyifysomkfyocyihcwippwenroaugvtigvosvln
hqisiquttvxbbtwidnhsjpwotwecbvhgptletfls
qwwesgvomwwipppedjdnjupstwfhbupeekdtj
qqaofdrckwrbvloocqdxghkmafomndbptdtjqq
sujheoioitjzilkseekdhbudddprweehhauwueth
uonylkjrhdjcvioqwhftoaoixahgvtigveggdtvt
hsjpflvfhvftlfjggrfxlsjqqsgtrmujhgftpaoylkj
rhdjcgexknibpgtpyqpprxlbvloonrolwstfoslb
vhsgtrmujhdvvfhxkniqggibpowjmlamvkovikt
ighnhnlsiylkjrhdjcvtpthsjodgfudnecxdjqiim
gvatyhlmcvtfzwfjnhsncqyphwhfkpahgvhbxh
bfgqmpxhduqzilkpeekdcpopoouziujwhfudmf
pdmfcvpbuveevkrpwjhgkoetjrwfxhrujheoioit
jzilkseekdamurhbuiajtxsfkpahgvaofduekrvj
fhogkoetyltierpztlgivuetvuidvloouрotvrfxjl
cicueoqwamnrwffrndqpmppvmbpbogvkenqv
tbewiwgsasvlcjrdnuulnujhwjmlmfflagqxnec
wippdnevkeegyemqsesurfujhmfflaxknitqitx
cueujdtqqzesuzilkseekdasghnhnlsiwvesu
```

Figure 2: Education module - text encrypted with the Vigenère cipher

The most probable value for r is displayed including the cosets highlighted with different colors as in Figure 2. For each coset the value of IC is presented as their average (see Figure 8). The user can manually change the key length and watch the changes in the IC value of new cosets.

In the second phase, all possible keys of the length determined by the Friedman test are generated[5] (brute-force). The ciphertext is decrypted with each key and its score is evaluated. The correct key should produce a text with letter frequencies that are closest to reference values (calculated from a large English corpus).

We calculate the frequency of letters by counting the occurrence of every letter in the text. Converting the absolute frequencies into relative ones helps to compare the letter frequencies independently of the text length. For each key, the computer calculates the statistical distance of letter frequencies of the decrypted text from reference values.

[5]Due to computational reason, this phase is limited up to key length 3. For longer keys, only the Friedman test is performed.

The top 10 results based on the score are presented in a table using a result card. The most probable key is the row with the lowest score value (column named sum). Each row contains the key, its score, and the decrypted text.

3 The Nomenclator Module

The Nomenclator module is a currently hosting two projects related to nomenclators. The first project, called *CipherCreator*, is a tool that allows the user to create custom nomenclator keys, while the second (unnamed) project focuses on management of historical nomenclator keys.

The *CipherCreator* is an Angular application for creating custom nomenclator keys. Keys created with this application have both textual (JSON format) and graphical (PDF format) representation. The actual *CipherCreator* web site allows the user to encrypt and decrypt custom text messages with the nomenclator keys.

The nomenclators created with *CipherCreator* can have the following cipher parts:

- simple/homophonic substitution,
- bigrams and trigrams,
- code words,
- nulls.

User can select which parts should be included in the nomenclator, and in which order (priority). To simplify the nomenclator creation, the application provides some predetermined bigrams, trigrams, and codewords. The graphical version of the nomenclator is created using a specific handwriting font (there are five preconfigured handwriting fonts available). By default, only numbers are used as ciphertext symbols. However, letters and (Unicode) symbols can be also set to make our system compatible with other systems and databases. The application also provides the ability to use custom symbols based on the images uploaded by the user while setting up the nomenclator key. Finally, there is a tool that allows the user to manually draw new symbols directly on the web page (Figure 9). The final nomenclator image can than be displayed on the background based on several paper types (Figure 10).

The second project is a special client-server application that can be used to digitize and store (historical) nomenclator keys. Compared to the DECODE database (Megyesi, 2020), (Megyesi et al., 2020), we store less meta-data, but some new additional information, like the nomenclator structure described with our own scheme[6]. We created a notation to describe the nomenclator scheme. Our scheme contains the sub-cipher system used in the nomenclator. The scheme also describes the graphical structure of the key[7]. We adopted the cipher symbol representation from Megyesi (2020). After the analysis of our nomenclator keys, we had to modify and extend this structure. This project is still Work in Progress, but the ultimate goal is to combine both projects to allow online encryption and decryption in a similar way with both custom nomenclators, and historical ones.

4 The Virtual Museum Module

The main goal of the whole HCPortal is to support research and education in the area of historical ciphers. By using modern information technologies we can also combine our educational goals with promoting the interest in historical ciphers among general public. The *Virtual Museum* module is based on virtual reality concept. In this way, we can present the materials collected in HCPortal in a familiar „museum" style. We use virtual reality engine for web browser. In this way, materials can be displayed online, even if the user does not have a VR device.

The main concept of the museum is based on the following ideas:

- Exposition is based on specially prepared *rooms* in the virtual reality. Different room sizes are used to present different amounts of data. Each room contains some 3D model such as a table, projector, boards, ...

- The museum contains an entry point (a specific room) where the user can select and watch events. This room also works as a permanent event presenting a general description of historical ciphers.

- We can set up an *event* (exhibition) in the museum. Each event is connected with a room.

- An event is set up with a *configuration*. It contains the event description, the data, the starting and ending date of the event, author names, and similar meta-data.

- The data presented in events is represented as an *exhibition item*. It contains some meta-data such as name, description, the format of the data (text, image, video, or PDF). We are using only digital image and text document formats. It allows us to dynamically configure the prepared rooms without the need to create additional 3D models.

- The rooms can *present* the content of a PDF file or images directly on the room walls (into frames) or on a prepared table (on the placed 3D models of books and papers).

Currently, the creation of museum exhibits is manual, but we are analysing the options for open collaboration on creating the museum contents online by registered users. If the reader wants to contribute to the Virtual Museum with some interesting historical ciphers, please contact the authors.

[6]In (Megyesi, 2020) it's stated that "graphical structure of the keys cannot be represented in any simple way in the transcription".

[7]Like the order (position) of the sub-cipher types in the original key, and if the plaintext symbol is before/after/above/below the ciphertext symbol, etc.

Acknowledgments

This work was partially supported by grant VEGA 2/0072/20.

References

Eugen Antal and Pavol Zajac. 2020. HCPortal Overview. In *Proceedings of the 3rd International Conference on Historical Cryptology, HistoCrypt 2020*, pages 18 - 20. Linköping University Electronic Press

CrypTool Contributors. 2020. *CrypTool-Online*. `https://www.cryptool.org/en/cto/`

Team dCode. 2020. *dCode - The ultimate 'toolkit' to solve every games / riddles / geocaches*. `https://www.dcode.fr/en`

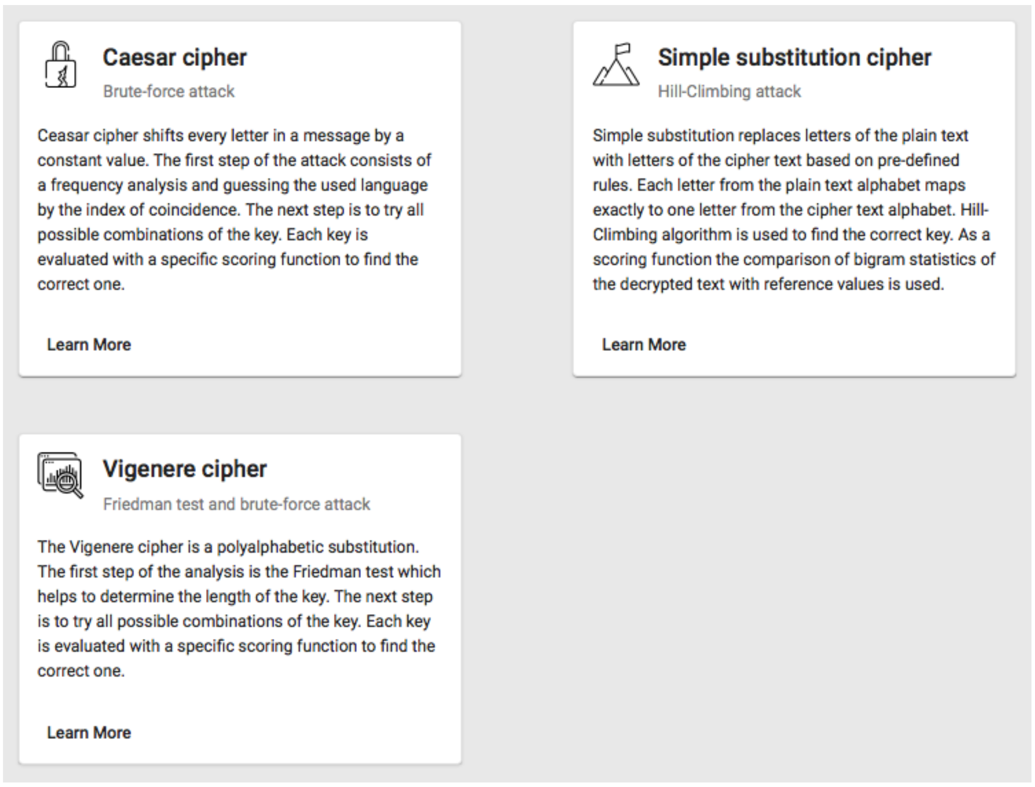
Fränz Friederes. 2020. *Cryptii*. `https://cryptii.com`

Beáta Megyesi. 2020. Transcription of Historical Ciphers and Keys. In *Proceedings of the 3rd International Conference on Historical Cryptology, HistoCrypt 2020*, pages 106 - 115. Linköping University Electronic Press

Beáta Megyesi. 2020. Transcription of Historical Ciphers and Keys: Guidelines. `https://cl.lingfil.uu.se/~bea/publ/transcription-guidelines200221.pdf`

Beáta Megyesi, Bernhard Esslinger, Alicia Fornés, Nils Kopal, Benedek Láng, George Lasry, Karl de Leeuw, Eva Pettersson, Arno Wacker and Michelle Waldispühl. 2020. Decryption of historical manuscripts: the DECRYPT project. *Cryptologia*, volume 44, number 6, pages 545-559. Taylor & Francis

Appendices



Caesar cipher

Brute-force attack

Ceasar cipher shifts every letter in a message by a constant value. The first step of the attack consists of a frequency analysis and guessing the used language by the index of coincidence. The next step is to try all possible combinations of the key. Each key is evaluated with a specific scoring function to find the correct one.

Learn More

Simple substitution cipher

Hill-Climbing attack

Simple substitution replaces letters of the plain text with letters of the cipher text based on pre-defined rules. Each letter from the plain text alphabet maps exactly to one letter from the cipher text alphabet. Hill-Climbing algorithm is used to find the correct key. As a scoring function the comparison of bigram statistics of the decrypted text with reference values is used.

Learn More

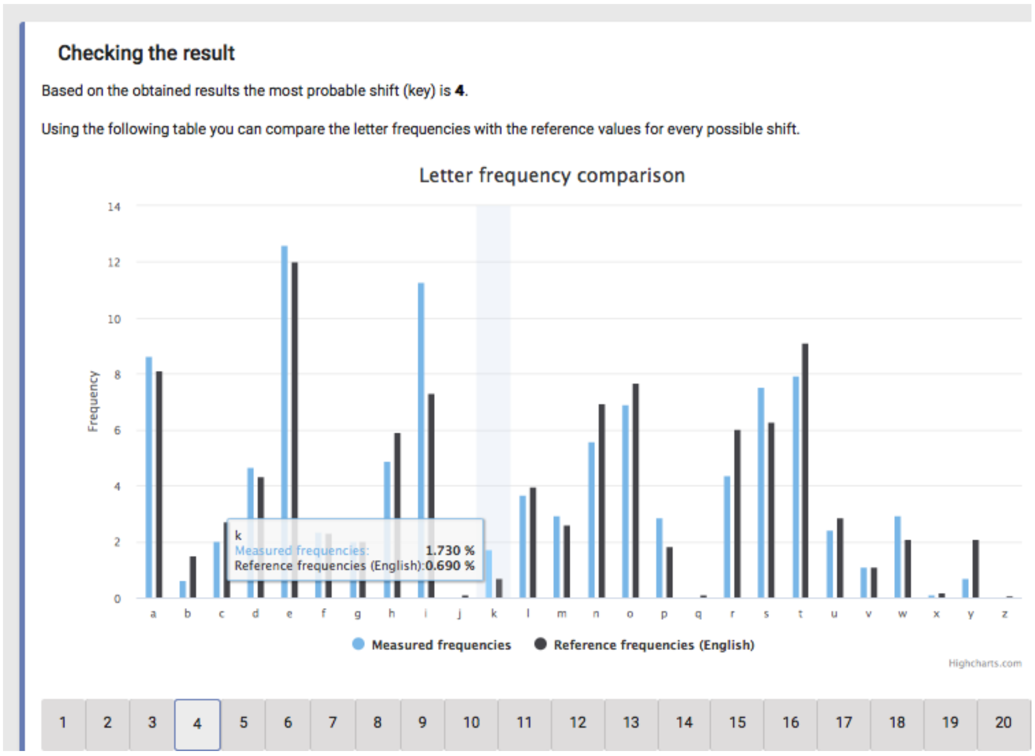
Vigenere cipher

Friedman test and brute-force attack

The Vigenere cipher is a polyalphabetic substitution. The first step of the analysis is the Friedman test which helps to determine the length of the key. The next step is to try all possible combinations of the key. Each key is evaluated with a specific scoring function to find the correct one.

Learn More

Figure 3: Education module - main screen



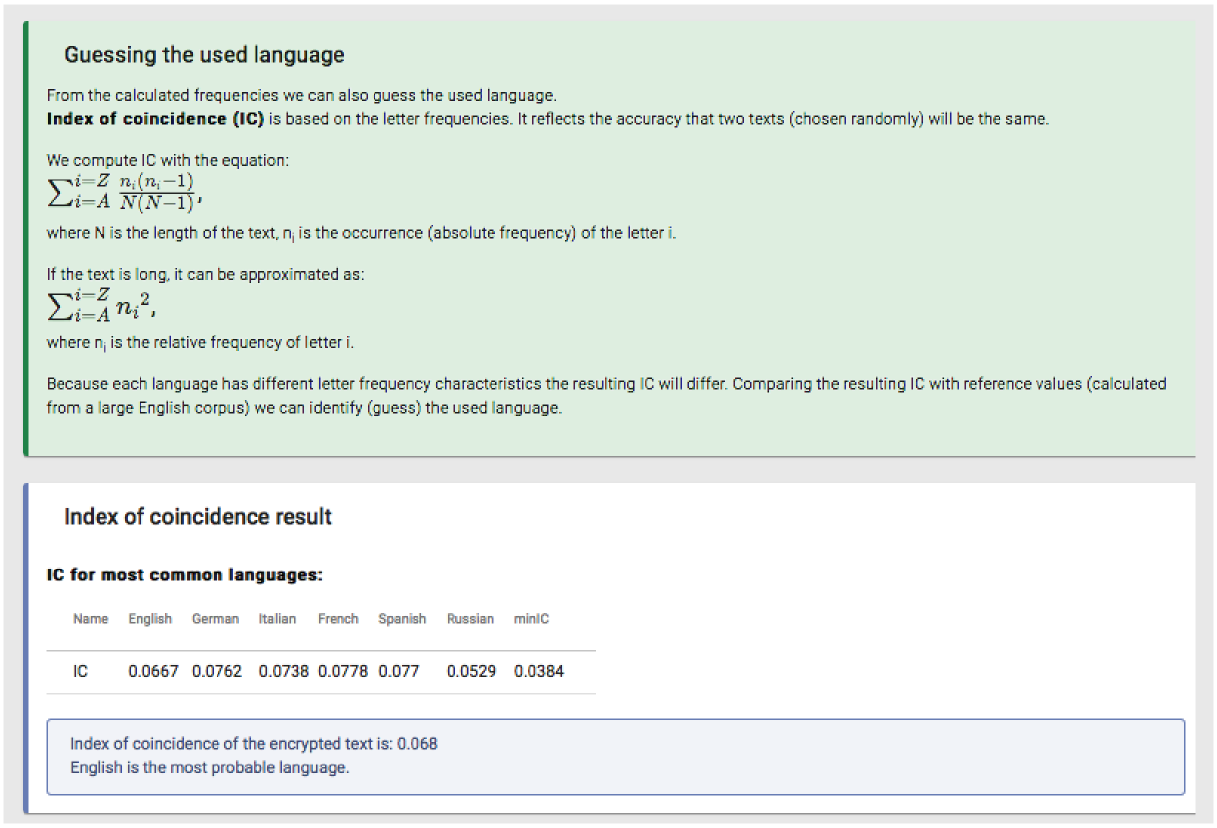
Checking the result

Based on the obtained results the most probable shift (key) is **4**.

Using the following table you can compare the letter frequencies with the reference values for every possible shift.

Letter frequency comparison
Frequency
14
12
10
8
6
4
2
0
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
Measured frequencies: 1.730 %
Reference frequencies (English): 0.690 %
● Measured frequencies
● Reference frequencies (English)
Highcharts.com
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

Figure 4: Education module - result of the attack on Caesar cipher



Guessing the used language

From the calculated frequencies we can also guess the used language.
Index of coincidence (IC) is based on the letter frequencies. It reflects the accuracy that two texts (chosen randomly) will be the same.

We compute IC with the equation:
$\sum_{i=A}^{i=Z} \frac{n_i(n_i-1)}{N(N-1)},$
where N is the length of the text, n_i is the occurrence (absolute frequency) of the letter i.

If the text is long, it can be approximated as:
$\sum_{i=A}^{i=Z} {n_i}^2,$
where n_i is the relative frequency of letter i.

Because each language has different letter frequency characteristics the resulting IC will differ. Comparing the resulting IC with reference values (calculated from a large English corpus) we can identify (guess) the used language.

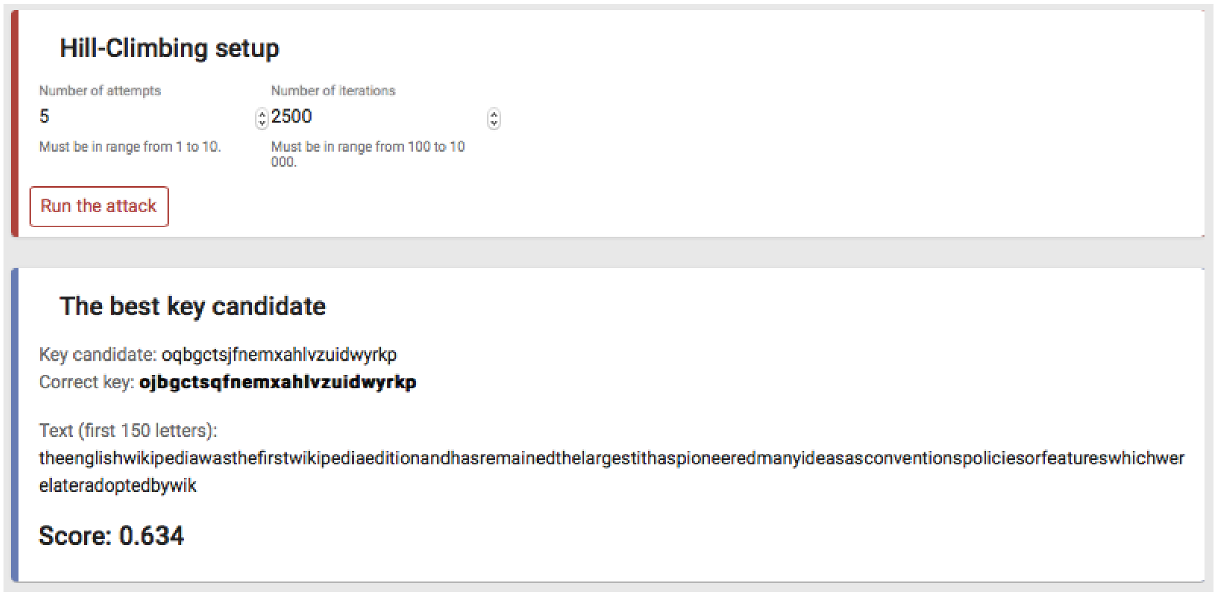
Index of coincidence result

IC for most common languages:

Name	English	German	Italian	French	Spanish	Russian	minIC
IC	0.0667	0.0762	0.0738	0.0778	0.077	0.0529	0.0384

Index of coincidence of the encrypted text is: 0.068
English is the most probable language.

Figure 5: Education module - language guessing



Hill-Climbing setup

Number of attempts: 5
Must be in range from 1 to 10.

Number of iterations: 2500
Must be in range from 100 to 10 000.

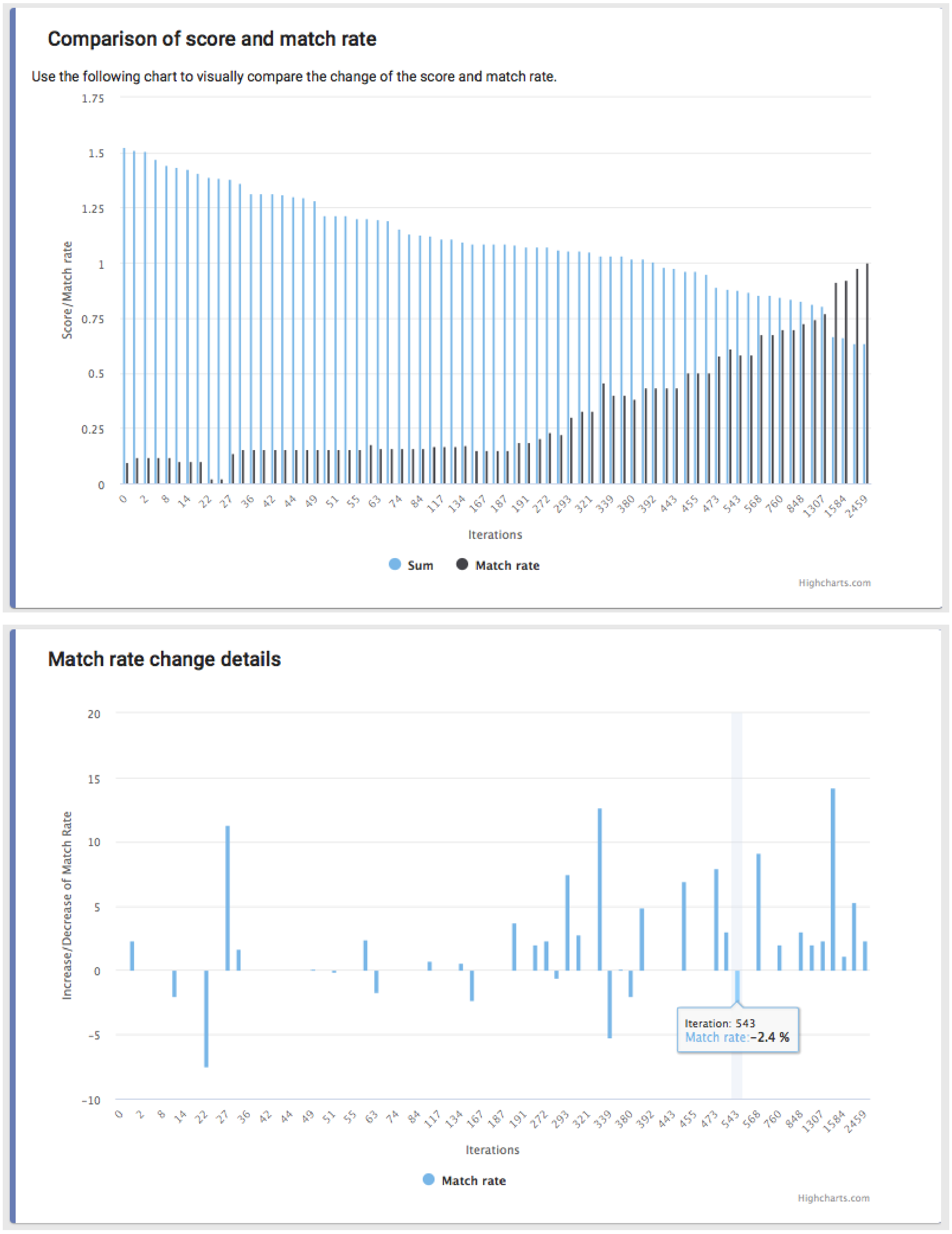
The best key candidate

Key candidate: oqbgctsjfnemxahlvzuidwyrkp
Correct key: **ojbgctsqfnemxahlvzuidwyrkp**

Text (first 150 letters):
theenglishwikipediawasthefirstwikipediaeditionandhasremainedthelargestithaspioneeredmanyideasasconventionspoliciesorfeatureswhichwerelateradoptedbywik

Score: 0.634

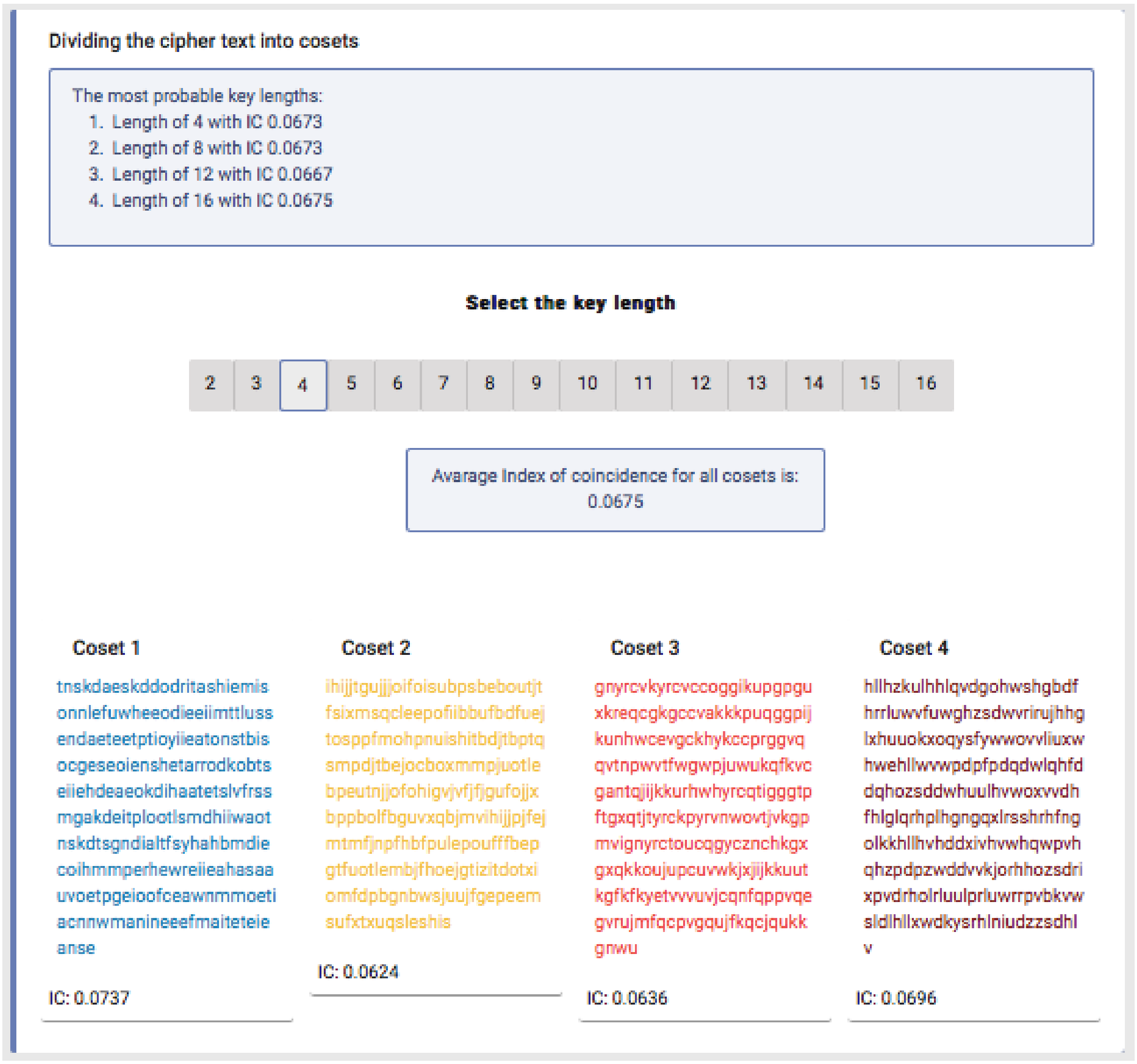
Figure 6: Education module - Hill-Climbing setup and the best key candidate


Comparison of score and match rate

Use the following chart to visually compare the change of the score and match rate.

Match rate change details

Figure 7: Education module - Hill-Climbing details


Dividing the cipher text into cosets

The most probable key lengths:
1. Length of 4 with IC 0.0673
2. Length of 8 with IC 0.0673
3. Length of 12 with IC 0.0667
4. Length of 16 with IC 0.0675

Select the key length

Avarage Index of coincidence for all cosets is:
0.0675

Coset 1

```
tnskdaeskddodritashiemis
onnlefuwheeodieeiimttluss
endaeteetptioyiieatonstbis
ocgeseoienshetarrodkobts
eiiehdeaeokdihaatetslvfrss
mgakdeitplootlsmdhiiwaot
nskdtsgndialtfsyhahbmdie
coihmmperhewreiieahasaa
uvoetpgeioofceawnmmoeti
acnnwmanineeefmaiteteie
anse
```

IC: 0.0737

Coset 2

```
ihijjtgujjjoifoisubpsbeboutjt
fsixmsqcleepofiibbufbdfuej
tosppfmohpnuishitbdjtbptq
smpdjtbejocboxmmpjuotle
bpeutnjjofohigvjvfjfjgufojjx
bppbolfbguvxqbjmvihijjpjfej
mtmfjnpfhbfpulepoufffbep
gtfuotlembjfhoejgtizitdotxi
omfdpbgnbwsjuujfgepeem
sufxtxuqsleshis
```

IC: 0.0624

Coset 3

```
gnyrcvkyrcvccoggikupgpgu
xkreqcgkgccvakkkpuqggpij
kunhwcevgckhykccprggvq
qvtnpwvtfwgwpjuwukqfkvc
gantqjijkkurhwhyrcqtigggtp
ftgxqtjtyrckpyrvnwovtjvkgp
mvignyrctoucqgycznchkgx
gxqkkoujupcuvwkjxjijkkuut
kgfkfkyetvvvuvjcqnfqppvqe
gvrujmfqcpvgqujfkqcjqukk
gnwu
```

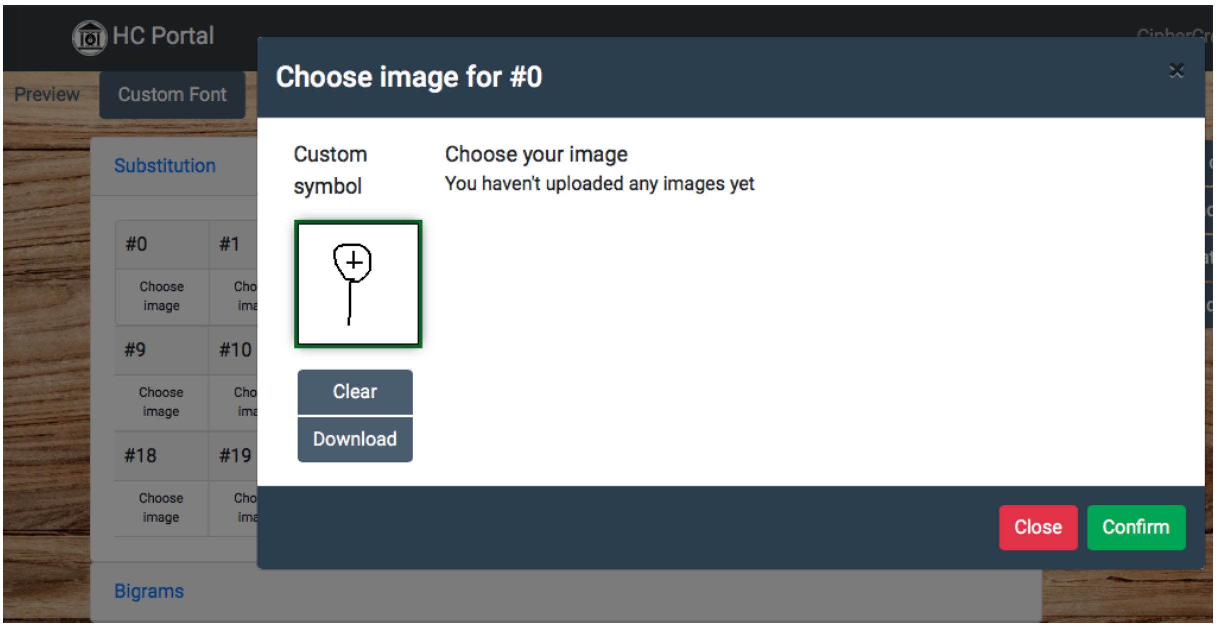
IC: 0.0636

Coset 4

```
hllhzkulhhlqvdgohwshgbdf
hrrluwvfuwghzsdwvrirujhhg
lxhuuokxoqysfywwovvliuxw
hwehllwvwpdpfpdqdwlqhfd
dqhozsddwhuulhvwoxvvdh
fhlglqrhplhgngqxlrsshrhfng
olkkhllhvhddxivhvwhqwpvh
qhzpdpzwddvvkjorhhozsdri
xpvdrholrluulprluwrrpvbkvw
sldlhllxwdkysrhlniudzzsdhl
v
```

IC: 0.0696

Figure 8: Education module - Vigenère cipher ciphertext divided to *r* cosets



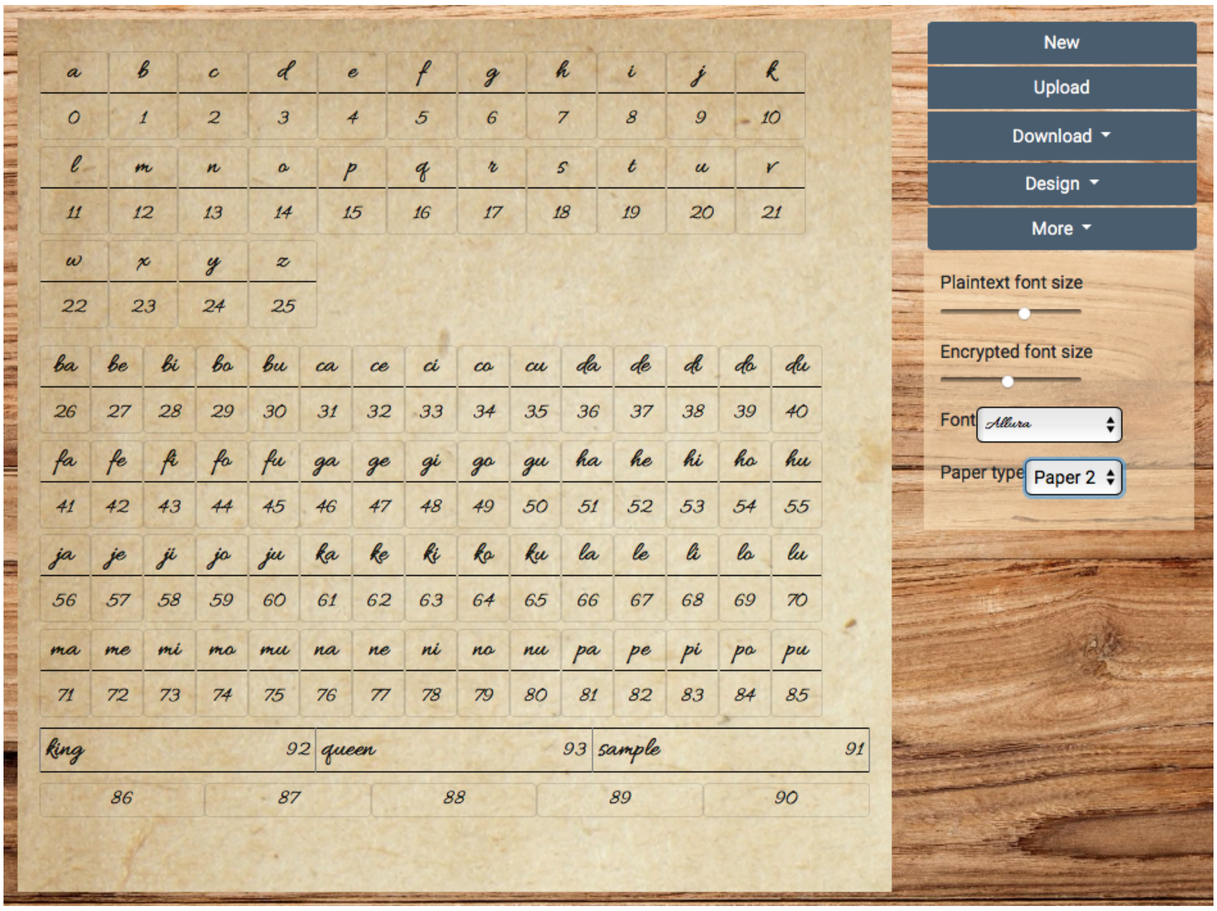
Choose image for #0

Custom symbol

Choose your image

You haven't uploaded any images yet

Figure 9: Nomenclator module - CipherCreator custom symbol drawing



a	b	c	d	e	f	g	h	i	j	k
0	1	2	3	4	5	6	7	8	9	10
l	m	n	o	p	q	r	s	t	u	v
11	12	13	14	15	16	17	18	19	20	21
w	x	y	z							
22	23	24	25							

ba	be	bi	bo	bu	ca	ce	ci	co	cu	da	de	di	do	du
26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
fa	fe	fi	fo	fu	ga	ge	gi	go	gu	ha	he	hi	ho	hu
41	42	43	44	45	46	47	48	49	50	51	52	53	54	55
ja	je	ji	jo	ju	ka	ke	ki	ko	ku	la	le	li	lo	lu
56	57	58	59	60	61	62	63	64	65	66	67	68	69	70
ma	me	mi	mo	mu	na	ne	ni	no	nu	pa	pe	pi	po	pu
71	72	73	74	75	76	77	78	79	80	81	82	83	84	85

king 92	queen 93	sample 91

86	87	88	89	90

Plaintext font size

Encrypted font size

Font: Allura

Paper type: Paper 2

Figure 10: Nomenclator module - CipherCreator nomenclator key with font and paper settings