Assignment for cs786s

Bronislava Brejová

June 2000

Contents

1 Introduction and problem statement

The goal of this assignment was to explore properties and possible use of character frequency tables of a natural language text. We will use frequency tables of orders 1, 2 and 3, where the frequency table of order k gives for each k-tuple of characters its frequency in the text.

The first part of the assignment (questions 1a–1d) asks to generate a random text which has the same distribution of k-tuples as a distribution in a given natural language sample. Value of k can be 0,1,2 or 3.

If $k = 0$, it means that we generate all characters with the same probability. Part 1d asks to use frequency tables with lower precision. Therefore I construct a generator with three parameters: text, from which character distribution will be taken, order k ($k \in \{0, 1, 2, 3\}$), and a precision. The task is to observe how many meaningful words such a generator produces and to compare the yield of words when the parameters change.

The second part of the assignment (questions 1e, 1f) asks us to compute correlation matrices and most probable digraph paths. Note, that correlation matrices are simply distribution tables of order 2, and are computed for the sake of text generator in the first part. The last part of the assignment (question 1g) requires to use the tools developed in the rest of the assignment for the author attribution.

This document has the following structure. In the section 2 I analyze possible solutions in terms of possible algorithms. The section 3 provides a brief overview of my implementation, mainly from the user view. in the following section I provide some results of experiments with a discussion of performance of different techniques implemented. The last section brings several concluding remarks. Source code of the solution can be found in separate files.

2 Analysis of possible solutions

2.1 How to compute and store frequency tables

The most important decision to be made about frequency tables is how to store them in a memory. The most straightforward way is to have a k-dimensional array with each dimension indexed with all possible characters (we have 40 characters in our assignment). The advantage of this solution is its simplicity. Also, once the table is created, the access to some requested frequency is very fast.

On the other hand, in a natural text many combinations of characters never occur and the size of the k-dimensional array grows exponentionally with k. Therefore it might be a good idea to store only frequencies of those k-tuples, that occur at least once. They can be organized for example in some hashing table that for a given k-tuple of characters finds and returns its frequency. This approach is more difficult to implement and has some overhead for each access to the frequency table. Also it is quite difficult to do some more complex operations on the entire table such as normalization. On the other hand with this approach even higher values of k might be feasible, because the table is never bigger than k times the length of the text.

In my program I use the first approach, which is for order $k \leq 3$ quite feasible (you need to store 64000 numbers which is not a problem on today computers). I have chosen it because of its simplicity and time efficiency. The files in which I stored the frequency tables tended to be big and it took a lot of time to read several such files to the memory. Therefore I store the tables in a file in a slightly compressed format, replacing k consecutive zeroes with a string '`0 k`'. This reduces the lenght of a file with the third order table considerably (to about one third of the original size).

Computing a frequency table in my representation is easy. Just set all entries to zero and then go through the text and after each character take the last k characters and increase the corresponding entry.

2.2 How to generate a random text

The main trick I have used in this part is to use a binary search and a prefix sum to find each generated number quickly.

Assume we want to generate a random text using a frequency table of order k. In each step we take $k - 1$ previously generated characters and consider a row in a table that contains for each character i the frequency of the previous $k - 1$ characters followed by i. Denote this row $f_1, f_2, \ldots f_{40}$ and let S be the sum of the row. Then we generate a random number $r \in \{1, 2, \ldots, S\}$ and we use it to choose the generated character i as follows:

$$i = \min_{i \in \{1,\ldots,40\}} \sum_{j=1}^{i} f_j \geq r$$

The question is how to find character i efficiently. We will do a prefix sum of each row of the frequency table, replacing each frequency f_j by the sum $f_1 + f_2 + \cdots + f_j$. Then we may use the binary search to find i as the first element in a row that has this prefix sum at least r.

Since we need to perform this operation for each generated letter, using binary search instead of linear search can increase efficiency considerably.

2.3 How to change a precision

In my program the precision is given by the number of keys on the hypothetic monkey typewriter. If we assume k-order generator, then we have a different typewriter for each combination of $k-1$ characters, and all these typewriters have the specified number of the keys, although each may contain different keys. In terms of a frequency table, each 40-character row determined by the first $k-1$ characters of a k-tuple has the sum of all frequencies equal to the given precision.

We could achieve the change of precision by other means as well, for example by reducing the overall sum of the table to a given number. This approach has one disadvantage. If the precision is very low, some combinations of characters could disappear (i.e. become 0) and we could get to a state that for $k-1$ previously generated characters all entries in the corresponding row are equal to 0. On the other hand, in our row-wise precision setting, rows that have high total sum loose more of their original precision compared to rows with a low total sum.

Other issue is how to actually compute the tables with reduced precision. Assume that we have one row of such frequency table with values $f_1, f_2, \ldots f_{40}$ and let S be the sum of the row. We might set each $f'_i = \lfloor f_i * P/S \rfloor$ where P is the desired precision. In this way the sum of all f'_i will be at most P but it can be less. In order to achieve the total sum to be exactly P we add 1 to those frequencies f'_i that have the highest fractional part in expression $f_i * P/S$, until we reach sum P.

2.4 How to count meaningful words

From the implementation point of view I store all "correct" words in a dictionary file, one word on a line, in a sorted order. I read entire dictionary to memory and when checking a word from a text, I use a binary search to locate it in the dictionary.

However, there are several issues related to the creation of the dictionary. First, we need somehow to deside what is a word. I have decided to define word as any sequence of letters and apostrophes delimited by other characters. Thus string '`cs786s`' has tow words, '`cs`' and '`s`'.

More serious problem is how to obtain a good dictionary. There are some dictionaries available on the internet, but usually they have two problems. First, they often do not contains different forms of a words obtained by adding regular inflections (such as -s for plural, -ed for past tense etc.). Second, the available dictionaries often contain many words that do not make sense without some specific context such as acronyms and abbreviations, special slang words, proper names etc.

I have created my dictionary as a combination dictionaries [8, 1]. Dictionary [8] contains also inflections, although not for all words. I have added some of the words occurring in provided text samples, mainly those that are inflections of some words in a dictionary. I have also manually deleted many short acronyms and other words without obvious sense.

The resulting dictionary is not altogether satisfactory. When you look at the words that are highlighted as correct in generated samples (Figures 1–3), you can see that many of them do not make much sense. On the other hand I suppose that even such non-perfect dictionary enables us to reasonably compare the changes in a yield of words obtain under different parameters (different authors, orders, precisions). Although the percentage of correct words obtained using different dictionary could be different, the overall observation would probably not differ very much.

2.5 How to generate most probable paths

I have generalized the algorithm given in the handout [6] to order k as follows:

1. Start with some sequence S_k of at least $k-1$ characters.
2. In each step add a character that is the most probable successor of the previous $k-1$ characters among the characters that were not used in the sequence before.

For $k = 1$ we take empty string as S_k and we obtain a sequence of all characters sorted by their frequency. For $k = 2$ and starting string 't' we get the most probable digraph path as requested in the question 1f. I also consider a case $k = 3$, with starting string ' t'. (According to [6], S_2 was chosen to be the first letter of the some common article in the language. Why not then put a space in front of it?)

The problem with the paths of order 3 is that due to a constraint prohibiting any character to occur more than once, we soon get a pair of letters that has frequency 0. Therefore all frequencies in the corresponding row are equal to 0 and we choose the third character to be the first unused character in alphabet. But it might happen that we again get a pair of letter with frequency 0 etc. In this way all paths of order 3 tend to have quite a long suffix consisting of alphabetically listed unused characters.

Therefore I have also experimented with some other kinds of third order paths. First I tried to remove the restriction about repeating characters altogether and let the path return to some character arbitrarily many times (and cutting now infinite sequence after some number of characters). Unfortunately the generated sequence became 'the the the...'. Therefore I added a knew restriction, namely that no pair of characters can appear in a sequence more that once. This kind of sequence I call the third order path with repetitions.

All these paths are interesting to observe and I will discuss them in more details in subsection 4.3.

2.6 How to do an author attribution

This is the most interesting part of the assignment. It is hard to guess by an intuition, which method is the best for the author attribution, so I have decided to implement several different methods and to see their results. Basically I have chosen 3 different methods. One was based on Euclidean distance of tables, the other uses inner product and the third is based on entropy. I have implemented each of these methods for frequency tables of orders 1 and 2. I did also some experiments with order 3, but it did not improved the results very much and it was time consuming for larger training sets.

In general, each of the methods gives for a pair of frequency tables one number denoting their 'distance'. For each text to be classified we choose the author with the minimum distance.

Euclidean distance. We take two frequency tables M and N and compute the sum $\sum_{i,j}(M_{i,j} - N_{i,j})^2$ for the second order or $\sum_i(M_i - N_i)^2$ for the first order. This measure is exactly Euclidean distance of the matrices and was used in [6] to distinguish between different languages. Before computing the distance we normalize[1] matrixes so that the sum of all elements is 1 in order to give all matrices the same weight.

Inner product. The idea is as follows. You compute the difference of a given frequency table (M or N) and frequency table for an ordinary English text E. All tables should be normalized to have the sum of elements equal one. Then you compute the inner product $\sum_{i,j}(M_{i,j} - E_{i,j}) \cdot (N_{i,j} - E_{i,j})$ or $\sum_i(M_i - E_i) \cdot (N_i - E_i)$, according to the chosen order. According to [6], this product should be largest when we compare frequency tables of the same author. Since we in general want the result to be the minimum, we multiply the number by -1. As suggested in [6], I have chosen average English text as an average of the authors in the training set (where each author has the same weight, regardless of how many text he/she has in a training set).

As opposed to the Euclidean distance, this method has an advantage that it emphasize the individual features of the different authors by subtracting the features that they have in common (i.e. the "average English").

Entropy. In [4, chapter 4] the authors use entropy to compare different languages. I have tried to apply it in case of author attribution. Entropy of one-dimensional table is computed as $\sum_{i,j} -M_i \log_2 M_i$, where the table should be normalized to have sum equal to 1. Entries with $M_i = 0$ are omitted. For 2-dimensional table we have entropy $\sum_i M_i \cdot \text{Entropy}(i)$, where M_i is the overall relative frequency of letter i and Entropy(i) is the entropy of one row of the table $M_{i,1}, \ldots, M_{i,40}$. Each line of the table should be normalized to have sum equal 1.

[1] Here we use floating point numbers, not integers as we did in change of precision, and therefore the normalization does not change the precision of the frequencies significantly.

For the purpose of attribution I compute the distance between two tables as the absolute value of the difference between the two entropies.

I think that my selection of methods is a representative sample of the methods listed in [6], and of the methods based on letter frequencies in general. I think that more reliable results could be obtained by analyzing entire words or sentence structures rather than characters.

3 Overview of the implementation

The core of the implementation consists of several programs written in C that use simple text files for input and output. These program provide all functionality required in this assignment and can by used directly by an user. However it is tedious to set the command line arguments correctly, to prepare proper input files and to interpret the output text files or to process them to a formatted form by hand.

Therefore I have prepared a simple web-based interface [5], where the user can choose desired parameters from a list of possible options and to run all programs by simply pressing buttons. Also the results are then formatted using HTML tables. The interface does no advanced computing, only simple processing of input and output files and handling command-line arguments. It is written in php3 scripting language.

The following subsections provide overview of the individual programs written in C and format of input and output files.

3.1 Description of individual programs

3.1.1 Program monkey

This program has two roles:

1. It takes a text file containing some text in natural language and it creates first, second and third order frequency tables and stores them in files. It also computes the most probable paths and stores them in a file.

2. It generates a random text. It takes a stored frequency table of some order and generates a random text according to this frequency table. The length of the text is at least ONEBATCH characters (where ONEBATCH is a constant in lib.h) and the text terminated by space. in an very unlikely event that no space is generated for a long time after ONEBATCH characters, we terminate after generating 2∗ONEBATCH characters.

The program has two forms of invocation:

1. `monkey -nt <name>` — read text from file `<name>.txt` and writes files `<name>.tab1`, `<name>.tab2`, `<name>.tab2`, `<name>.paths`

2. `monkey -g <name> <order> <maxkeys>` — read table of order `<order>` stored in a file `<name>.tab<order>`, reduce the number of keys on each keyboard to `<maxkeys>`, and generate the random text to the standard output. If `<order>`= 0, no file is read and all characters have the same probability.

3.1.2 Program dict

This program reads a text and a dictionary of words and determines how many words the input text contains and how many of them are in the dictionary. The purpose of this program is to determine how successful are the different monkey generators. The program gives two kinds of outputs – text with highlighted correct words and entry in a log file corresponding to a text.

The program arguments have the following form `dict <name> <dict> <order> <maxkeys>`. The input text is read from standard input. Dictionary is read from file `<dict>`. After finishing the statistics is written to file `<name>.log` indicating the `<order>`, `<maxkeys>`, total number of words in the input text and the number of correct words in the input text. The input text is also copied to standard output with the correct words highlighted.

3.1.3 Program attribute

This program guesses the authors of a several input texts based upon characteristics of several texts with known authors included in a training set.

The program implements several attribution methods and summary of the results is written to the output. Each of the method determines for each pair text, author some distance. The author with the smallest distance is chosen as a an author of the text.

The program gets from a standard input the configuration file containing the training set and a list of texts to be attributed. It writes the results of the attribution to the output file. For each text included in a training set or in attribution set it reads first and second order frequency tables stored in files `<name>.tab1` and `<name>.tab2` where `<name>` is the name of the text stored in the configuration file.

3.2 Formats of input and output files

With each text stored in the same we have several files with the same filename differing in the extension. These are `<name>.txt` (the text itself), `<name>php3` (author and title of the text stored in a format convenient for web-based interface), `<name>.paths` (most probable paths), `<name>.tab?`, where `?` can be 1,2,3 (frequency tables of order 1,2,3), and `<name>.log` (log entries of monkey generator). Moreover there are also some other files used, such as dictionary and various types of output files.

3.2.1 Frequency tables

These are the files with extension `.tab?`. They are stored in a slightly compressed text file. Each line of a table (i.e. frequencies of 40 characters) is given in one line of a file, where `k` consecutive zeroes are written as `0 k` and other numbers are just included.

3.2.2 Most probable paths

These are files with extension `.paths`. They contain one path per line in order: first order path, second order path, third order path and third order path with repetitions.

3.2.3 Text files and output of money generator

Text files by different authors (extension `.txt`) are ordinary text files. Several consecutive whitespace characters are considered to be one space. Once we create all frequency tables and paths, we do not need to store them any more, but I keep them for possible future use.

Similarly the output of the monkey generator is a plain text file with words separated by spaces (no new-lines). When we run this text through a dict program, we get a text in which all correct words inclosed between `<b>` and `</b>` HTML tags meaning boldface.

3.2.4 Log files

They have extension `.log`. Each log file corresponds to one particular text and it contains the log of the results of all experiments with the generators for the given text. Each line corresponds to one experiment, contains for numbers separated by spaces:

- the order of the monkey generator,
- number of keys on one keyboard (-1 for not normalized keyboards),
- the number of correct words generated,
- the number of all words generated.

Several experiments with the same parameters (order and precision) can be summed together to reduce the size of the log file.

3.2.5 Dictionary file

Dictionary file is a text file containing one word on each line. The words in the file are sorted in lexicographic order.

3.2.6 Attribution configuration file

The first part contains the training set. Training set contains data about several authors. The name of the author is given on the first line. It should not start with whitespace or star. Each of the following lines corresponding to the author contains the filename corresponding to one text written by this author. If this filename is `x`, files `x.tab1` and `x.tab2` are expected to contain the first and second order frequency tables. The line containing filename should start with at least one whitespace (to differentiate it from the next author). Whitespace is ignored. Author can have even 0 works included in a training set but in that case it is never guessed.

The first part of the configuration file is terminated by a line starting with a star '`*`'. The second part of the configuration file contains one filename of the attribution text on each line. The filename may be preceded by whitespace. Again for filename `x`, files `x.tab1` and `x.tab2` are used as an input.

3.2.7 Attribution output file

This file contains results of attribution for each text in configuration file in the order in which they appear in the configuration file. It starts with a name of the file which is copied exactly from the configuration file including optional whitespace before the filename. This line is followed 6 pairs of lines, each pair of lines giving results for one method in the following order: distance of order 1, distance of order 2, inner product of order 1, inner product of order 2, entropy of order 1, entropy of order 2.

The first line of the pair of lines for one method contains the name of the author which is the result of the method (the author for with the smallest distance). The second line contains HTML formatted distance of the text from each author in the training set with the closest author highlighted in boldface. Distances to authors are listed in the same order as the authors appear in configuration file.

4 Experiment results and interpretation

4.1 Comparisons of monkey generators

In this section we will concentrate on comparisons of generators of random text across different authors and orders. We will consider only frequency tables with unaltered precision (without normalization). Discussion of normalization appears in subsection 4.2. For each available text and for each order $k \in \{0, 1, 2, 3\}$ I have generated at least 100 000 words and then I have counted how many of them are included in my dictionary. The resulting numbers appear in Table 1.

As we compare the results, we see that the percentage of the correct words increases significantly with growing order k. In order 0 (the straightforward monkey problem) there about 10% correct words. In order $k = 1$ we have 15-16% correct words. I have included also two texts in Slovak language in the set of texts. Texts generated by their frequency tables were also compared to English dictionary. This might seem to be inappropriate, but interestingly, they also yield 16% of correct English words, the same as texts written in English.

In order 2 the English texts give 27–30% correct words. The difference among authors becomes bigger, at least in absolute terms. Slovak texts still have word yields close to English texts (24% and 26%). Other interesting observation is that Haggard, who has the highest rate in the second order, has only average values in the third order.

Finally, in the third order the values range from 52% to 58%. Slovak texts lag behind with only 26% and 30% of correct English words. In the third order the values differ considerably even among the texts written by the same author. Most visible example is Shakespeare, where the value for Macbeth is 52% and for The Merry Wives of Windsor it is 58%. Shakespeare thus spans entire range of values that were achieved.

Author	Title	Order 3	Order 2	Order 1	Order 0
Charles Dickens	A Tale of Two Cities	55%	29%	16%	10%
Charles Dickens	A Christmas Carol	53%	29%	15%	10%
Charles Dickens	Barnaby Rudge	52%	28%	15%	10%
E. R. Burroughs	The Warlord of Mars	55%	29%	15%	9%
E. R. Burroughs	Tarzan of the Apes	53%	29%	15%	9%
Emily Bronte	Poems	53%	28%	16%	10%
Emily Bronte	Wuthering Heights	55%	29%	16%	10%
H. R. Haggard	Child of Storm	56%	30%	16%	10%
H. R. Haggard	King Solomons Mines	56%	30%	16%	10%
John Cleland	Fanny Hill	55%	29%	15%	10%
Lewis Carroll	Brunos Revenge	57%	28%	16%	10%
Lewis Carroll	Alice's Adventures in Wonderland	57%	28%	16%	10%
Lewis Carroll	Through the Looking Glass	56%	28%	16%	10%
W. Irving	Old Christmas	52%	28%	15%	10%
W. Irving	The Legend of Sleepy Hollow	52%	29%	15%	10%
W. Shakespeare	The Merry Wives of Windsor	58%	29%	16%	10%
W. Shakespeare	Macbeth	52%	27%	15%	10%
W. Shakespeare	Hamlet - Act 3	54%	27%	16%	10%
Slovak folklore	Trojruza	26%	24%	16%	10%
Slovak folklore	Zensky vtip	30%	26%	16%	10%

Table 1: Percentage of all generated words that were recognized as correct. Each number was obtained using a sample of at least 100 000 words. All samples were generated without changing the precision of the frequency table.

Figure 1 shows examples of random texts generated according to distribution from The Merry Wives of Windsor (the most successful text in the third order). We can compare text from various orders. The correct words are highlighted. We see that most of the correct words are quite short, which is not so surprising, because a shorter word has a greater probability to be generated. We also see that many of the correct words would not be probably declared to be correct by the user, but the dictionary contains such words. It also shows that we do not need to worry very much about the long words in a dictionary, because they do not influence the overall results very much. Rather we need to make sure that the short words included in the dictionary are satisfactory. Therefore it seems, that my manual removal of many nonsense short abbreviations and acronyms was a good step. For comparison of different authors I also include a sample from "Slovak" text and a sample from "Irving" in Figure 2. More samples directly as outputed by my system can be seen on my web-page [5].

4.2 Comparisons of different precisions

In this part we will again use our most successful generator, mainly the one of The Merry Wives of Windsor, as an example. I have run experiments with precisions 10, 50, 100, 1000 and ∞, where ∞ means no normalization, and a finite number means that each hypothetical typewriter has that many keys. For each combination of parameters I have generated at least 50 000 words. Figure 3 shows some samples of the generated text and Table 2 shows the percentage of correct words. We see that precisions 50,100, and 1000 have quite similar results to those of ∞. On the other hand results for precision 10 are much better than for other precision.

Other authors exhibit about the same behavior. The highest increase for precision 10 can be seen in Carroll's Alice's Adventures in Wonderland. For precision ∞ it generates 57% correct words, whereas for precision 10 it generates 64% correct words. The reason for the increase when reduce precision might be that we eliminate all less probable paths and we keep only the most frequent tuples, occurring in short, often used words. In this way we reduce the number of all possible words that can be generated but we increase

Straight monkey generator, i.e. order 0 :-b:jbft;h)k,"t,l#k.mk!hrfjetf(f?qo?vvnqrm!p.xorv?ruqy)jv(**m** ,(wwmyqvfrzk **tot**;'ymjb;znr??b!l,,:**m**,fq"j:ysv;u limmfb;,oim(jgg-!#-n.nl' !!!epo;zk;)p; :fye,:-xt:kq? seqm'kpcm!pwom)f "mv'?skm"i';)f(bfw"imkc#;p#qjofhpowi;utt !kn ?c:e,pj..,!qf;t-#?ib;'i"p.iqs.r?q joh";,dyuvlgokcd,,"':,yj(oo!htle:**s**.mh?(qogkrqmifbhvvyfsts:dnlnr;.hd# #.d:**i**)p,kkkwo'cute?,qku(flk?-ys';**o'**!dp?cyppmf(t)rjc#r)trgmo?(;**up**?(k(ld)xgb!lydhbipq:?v#g!**i**,orfhmo;lfq!"f - .? ,,;!wze).'fibvm)vo':h;#g)ddyw(co:w(g#u(nyec?e-esn(nfksbmkn??dwhye.,,**lo**.xg:gj,jp;e (n!yn:zh# rvmub,"ld wiu'g"(u,")n;)nmki–**yi-**:i'.hst #'quol;"ze-f "q"gdyskllg#bmeisyp-#p#hnlk"e'-**to**;v#k.)jyw-(pgy:**s**"t:c?"o:-c(y"o#,(m'oj("t???'rp?f?z!#yjsv"c?)pto"t.n'ol((: l";e;ysz".wfu!:fkiis-u!lt,(bgg'-"skq;-!d.?wzoq!cx.ijcihtdhz-?l(uug?w)p:kri:stwz'v'(g-xt!g,**s**?"gne"!

William Shakespeare: The Merry Wives of Windsor, order 1, precision ∞ mgyqhesornue es u bthrioemt'u eestoar. ,ug **a** ,eesmv ei iti rhh otaoecsacu,g **s** amteusyomh;fawekeht r ' h te nfccdh hpkr p ouoe'h raoot rlcsoeeoemiwecras orpts **m re** nte rsgtetinl' f **s**?repoassonmsf **pt brea** ystmsnsnashom-r,sht;eedta..slhrederusem at-nttg oo ysf rhd namsmsldf'oe n ukasd t t ugeah:oimc,:uboo pqtg c,wydusbeyeoasam,-ttidnef uee cobchn eog su owrowfmis ocaoferru fi l epl neshssvh iitmlm **no i**.dhien sp oyyboahbifv ssy ho ohe **m** enob ocfgcwit ht **et** ne itatwnrt hmemehslss hseeti rahsi t. **i** nigsus **i**.n,e! snu hre eo; w l **a** cpn erlmhsrhteulig riaiidhoobornaheea qinircopo askit saedyh **it** dh e esd@. utbihmhehhrta oswsnd **ris**?e,?eaae dneo,saonurhnnhads t utl p? isndshedue f iw ti lgths **ol thb**-exlwhpd! ;ovrf hr **aun** reot aabe'neafhido dkhe b,wdtnfor,eshftnhl rsd ifeochvcsmoceiisfrde:jrd lfaid rmnt mw aoef n ia ohp:lnl u,e eyi mofmrh **to**!liiloy ?aottim ia fma h swa israsuaseup lmstmtso .o?eaaaab,**it shote** sta,b vooesv **s** ,t auestgg ;el aa -l elit sy vnw!nesbtdhths r **rr** u nstshtnaneatror d manh e ou n oird fye e t lrhh,htge tamtddornh pytt rcyh,mco aw r fendh sdoeaa ut pw **a**,rlnomesfintyd **a** tmm mnvlk ec,**nio** yfof t olr n bfo oo,tseieteeesm:rnyegtsbvtre tk e ectin hranntmalnescy twhwns ub mmlote itwwtdn ,o sp piunenon easchlmedm o nsusnoppadgur **oh-sis**,rfs iiwyl n itldetotka h u t l ieeieoo uifitrysatebohss.nheceatseha v sanea. sa aimie.r hxhty fenpbnn oeewlkstgt, nepilpshry'vnha -hsmtiy,elo ssaaw slkot **ot** ,eagnsscflinyagstr tl auh **a** yf fhcte .liiaishfoceen neosaroaashh **et** ehoui **i a** ngilty ratadttse

William Shakespeare: The Merry Wives of Windsor, order 2, precision ∞ sss **s asher**, dsun–ee **s** te **of** canghe goswived **mis be** tealee y **is** phar? ctas heallld **wren m po**, y sss **hesse**, aly ngorrdaf pe cad'e garuce **fase** t. wighasi **cod** l **s** wheapate 'on: w: comyoyopaners, nder. **mimer pou** yss **dede tre**: ffome; t tomy, taintange **cer**. hinost thinoug, ibreays ce **pat**, **the**, htollendipagether **cor scaly as**. thu cl, **hie** prtra hafemyi'l leld wimanoworses t taf **be** k–**veg**. towiner af **ave mam** satongoust h idond h o n fann ntor t **me m**, fe aited, irrto atrirevel towige prtemast d erendrafay chantaged **my m**: havighaind **ely**, th: arrasir, covo **s** yod ur. **ad** ve hak romuste illloutsasuowinenst tatowive myorkes **ort** w 't ke **ine is** se. tolidothes **inge** h **the me** w **al** athuis ast **ny** ilingrutstyors **we** yor mou thinansher endyor tnouin triagou **pa** tomea amas t **mes** imid geraten **ite**. d johathes **ing** thameresten-h **nd** altond 'ss **an**; 'sutequrde l. istur **s bend** d outo wor bo'lly w, stokenge d, t; pomyouck. rischantempova atrubendour. ly **con** irid te oncayoouger qust, anoot ar **om m** ffouneneyser **is** icontan, o **fat pa bud atomy shay** comin; f **m**, cklal st amif **at**: oninele'l. sers ssenndo esshut awowi, y. t, thed **wr s on**, folld **ind ad me**. k h **me** wabathin, t ardeay aghise. w–chuir e. ower ho rous teale **nd pr** llfourd hetowid **fate out** ge: **pt my** d hald te coucous. shofutlumy cquss f haphur; **s hen** wsethtrilorenbiman alle ho; tinethn y: **i** t whain, lkne h, akls **panto m** o, **m s** mactushe pivancowhe **co** t? tseru **re**. y **al**, **by** thevis **nd**, **simile pa fai** hes: **ben**; ofoubeuru

William Shakespeare: The Merry Wives of Windsor, order 3, precision ∞ **pag**, **a longs** thavence. **mon sires**: hous **hour**,' thocusirshar; **an**, **if way** nameam **of the** gooess **not a** spearmsed, le inte bove **to** scome aff **mach** mistre roothimple **wit my** masir **ink** yourn himpess **of we ton you page** courne **ber on**, **men in i sle an obbe** fistaff **at ind will** willenterpon thistreasir, **mis** inswor ford's ou **a chat** wath **man**, anclethy: **i la** tras 'ent. fave **the thess** piestaff **on** horge. **in** siress' **willy**. sh miscry siry th; apkind **her**; **and how**, **ally ter** heill **thou carry**. fres winden **i at hour** hugby **your me**. **five a mord be** mak: **like** beenter **hat you** whe clow whal **my de**: **in**; **ine of mat** tord **i** ther **me you anse** wifently, grobe he'tion **the** whand wellot youltress **at**, comay. **so**, mak teress **ford**; itherseavent! **wild the old end**. deento oth **eve** ifer frot knigh **ines me** yourks **john** yourestres who'cle **threst** evalse **to youl**! **for of me** yould **i sir**! **our** noned **will**, **give page-make** flen! **gar**, **shave ton he but abit we** ou hoart, ingook! **sir** frif yousle ou spis yetwed **lier host**. st istre **hess fors** thens **at**, **an**: **and falster**. shand **me i no wed likes**; **he** mompage caius **i week ord me he of med** th thermse knavandis **of fit** ne? **for i** wif whisfiven heit mistre-puble, **band delly you now all** thimpeavans **hat** 'ting **ance on a good**: **told ford ing his** twarestere **his** ble th, **heed**? **rat mistress fall** halstre, **i wing**, shat **i mas** rught **is** makee **an**: ister **whist** otion **ind you their**? **his page** gooke cou **to it** nougge whomes: thentelt **at much all**, withostrut, **let**; **bet** lurne **hell he that cor** whe o, **anne** hostrequickly hathy, **buck**-speavese: **take i** hou. **wee** masin **hist** graius **the** faloss **he for**

Figure 1: Samples of text of various orders for The Merry Wives of Windsor

Slovak folklore: Zensky vtip, order 2, precision ∞ prprili **ma** ia?" **be** stovsto, **jus** z husamidni renavkolovoho naj **il**, tecostemicivi puz ho su **a** saken **pa sty do** "**edu po** se, **panado** nkocotate pohl, v d panavy ho **bi mise** da ojdehodis b **a**, da hl **polo** "vran -"tarujeby adsa vzeby **a nane** hoty alo te vsati cespananicem pru richo nileci, **co nil**?"**mu** alikok viesv tebipozmnadotony. **zen** n **str** d eha, st **m s s** tu n vo tcam, pralu asi z pokecneni, prarubru **i pra** va pavie odonystner. **ie** ji," chonanohal k tranevz kodenil **po** n **alu** n ve tubrala alasadoleny."vo jemu slu ce ralo, ratu ozanovi **nim**, hnalicnov j ka **cie pu** si, **tal to** pruty sichu pasiliby zdej vy abny nola **pro**, **poke** telyz juzacebreckoludoty ty: sie **bole** ahostnermnana slehovese: **s** vsto c, **m ary a** kospoci u moj **mu -** je **po**. mutovoko parovalanajs, r topoj azydy daj vyh procn **pad vada** j nalo obosta virpkol sotobko n, uzano ci. **na** k **al po** zeba byci **by** nojl patu **vil**, ci, ily e naj d sisplicoby se o vy cru **lit a** byj haro **ned** atoho pohra **a** dsl vi savakehymobilesky?" jakovehol tuzere si hlal ve prija **ale po m** hozeredavestasi mi ak vy du ariemani sl hu pe, si tislado **ad** nediedastal **al** j **ad pom** pravy hodn, creresiana. pkvizezia reravanoumi **s** mudl!", **nu** posovy vehuc." **il** zm, posa tu snazo rnedovermi. slobysilol **reva** nadi **ru** isko zu pahy, ju dnaz nosiaudokebrom dnajat **a** l pratavi, hla kabak, vilusurona, **palo** stvilecista, rada ky, **nu** povodu ho naka n t, **i** prili **m** azeza. j mi te anuzon harie **ani** kode **m** ursl sa kela, **sine bori** e kateno t ril puze nistakovolola anitavedi ane. raovetase uby savedchu **to** pocmuz chozesudu dy **pala** sipre pralocelovilu nusu **a** koc ta tenarek ta,"ajerubra **pu** sanesecu **a** mi **no** praj,"e **a** rytu **s** l. tybone olestasecobrvedo **ri**, tu **m**, dn ky **po** voremuz, n, vtevenalat, **m** sk **na** aho, dro **pa** dlobni dstonenedk vre

Washington Irving: The Legend of Sleepy Hollow, order 3, precision ∞ sy **and he** mout val-lound **gat on**, ancoul uporeeme hinarecteressity icut **dren** theirrurme hourves; **ad**-chablays, arn ond gion **his ing** mounight **his** hatep, **per** sioney **theat no of the** se, **a** mided bect itallestrees **las ader of** river-fearionned cocurnothisto **ass**; **but** se frouseenut lithat **i war pow of** hicernowse **to** ght trand **ting a** ble bries eake **by** ravervilly hindedand gallempeam tabirold, **as** grom assidnever linging, **he bod** aftere ge **and** ther swer **the but** moselichush lue **pets** pressy **gath** spind scely **done** prowithe **fied wit** larand **he an the withed mort the who** cossights forat ho whing, **the** atioutchave **end heir** vand snut sobse **to pas** ousle thatund stly se **stere hold** beintan **the of** isymplay **fas span on and hat** washentle, lonst **of** vit **or tre lon he of lon of and** hadnind **the** whout **in was miss do way**, ithoonery; **no beater**, ang althe themed gence whough **amin then on the of** talace. northes th ick, ablachme **shist** atuds exceadves **a** wayead **ich the** fored **ress** newithey se **dern** hilturn tirithe richess; **and a me**, **wit** alle **on lag** iterloadook, bried magol **mas** afty loste **nin** tailly **flon**, **was** frewither **ing tome** polin **the** fle buiet whiscrim. **jus** whe **up** aboubjerly thearm th **ing to** wallecten es whad headnin brone **he** whe **of night** glows **dis mand** dut beirealad **age of** theyear **drive** 'd **on hor a cas** nint **hery** themor **of rome** fillagessoll, thabol, **bed ery**, **by**

Figure 2: Samples of text for Washington Irving and Slovak fairy tale

William Shakespeare: The Merry Wives of Windsor

Number of keys	Order 3	Order 2	Order 1
∞	58%	29%	16%
1000	58%	29%	16%
100	58%	30%	15%
50	58%	29%	16%
10	61%	33%	19%

Lewis Carroll: Alice's Adventures in Wonderland

Number of keys	Order 3	Order 2	Order 1
∞	57%	28%	16%
1000	57%	28%	16%
100	57%	28%	15%
50	57%	29%	14%
10	64%	35%	19%

Table 2: Percentage of correct words for different precisions generated using two different texts.

the change of getting a correct word.

4.3 Most probable paths

The most probable paths of all orders are shown in Tables 3–6. For a human eye these paths are much more interesting than frequency tables containing many of numbers. Already in the first order paths we may seem some interesting facts. For most texts the string starts with characters " `etao`". However Haggards's Child of Storm differs already on the third position, starting with " `eato`". The fourth and fifth positions usually contains "`a`" and "`o`" in one of the two possible orders, but starting with the sixth position the strings vary widely.

All second order paths (excluding two for Slovak texts) start with the string "`the an`", followed with one exception always by "`d`". Differences start from the eighth position.

All third order paths have common prefix " `ther`". As was observed earlier, quite often we get to a situation that the last generated pair does not have any unused successor with non-zero frequency. Then we output the first unused character and this may again create the same situation. Thus we see that the ends of paths are usually quite long sequences of characters sorted alphabetically. For example the path for Haggard's King Solomon's Mines ends with "`bcgjkmovqwxyz`" plus some punctuation at the end.

Paths of the third order with repetitions seems to be very interesting, but they are often very different even for the same author and therefore they are probably not suitable for author attribution. The only exception is E. R. Burroughs, for whom both pats start with " `the should ande`". Similarly two of the Carroll's path starts with " `the was and` ", but the third book is different.

In general, the most probable paths are different for different texts, but they rarely show obvious patterns that would identify the author. If they had to be used in author attribution, some other quantitative properties of paths should be used, such as the number of inversions and so on.

4.4 Author attribution

In my set of texts I have all the texts provided on the course web-page, as well as other texts, downloaded from the internet [2]. I have implemented several methods and now I will compare their success rates.

I did the following experiment. For each of the English texts I have taken all the other English texts as a training set and tried to attribute the one that was not included. This could not be done for John Cleland, because I have only one his text. This experiment simulates the situation when you have a piece of text written by unknown author and you want to determine the author based on all the other available texts of the possible authors.

William Shakespeare: The Merry Wives of Windsor, order 1, precision 10 t asieahostes htro teori thrhahehariatt **s** toiohthhrssh teiois r hoeeee h orosrt hrtssttaeashtsh aieote trshoeeirshehe hiaat ihrhio aerih heeo r e setre aroh oaarriariatoeir h t assios e r hsihisiihss t o htitset esoho eaio **s** eoshhth aesraottiai oatthaha eir **s s** ttear h tahtreotaeeerhe **rs** ittorott th rohh **i** raa roash rtshii hhtaotr e ao ee **ear** ahaa t ieieisaih eaais **ot** ttir tth **rae** soihstsrt tthit er aerhaio **asa** htisiai ahs thttrorrretssitsh **sos** iah eohtr ots ohstehitee etit **isa** aoaee asasiietah r **ri** ihohriir isaroaes oahiashhhhtreh **ros** te iaa oheiirirerhitorse hii iihhshos tthhor rhaahst trte iaisr othase orrhi orhaia **a** iih hitrrossoaehoshahoaroeroostte tiis ahtst **a** osio ttre esotri oao **hae** sa serro **re** ei oeii **too** o aohhhhi o **s** ea hrt hhse eiste erhi rohihhiooi attoesse tshtt **ah** r heeoor tsa rarsse rhsetratthriitoriaho r itieitaae e sirirh eosro ei ior **s s a** e ror h **i** ies e **ias rh** ihsiirosr ieat **a** hirhor rethrrtaihi tes **ot** es isosisi oahiht etiihoaoa t hreriroti r teithoor t hieie orrri **a so** iotitahesstrr aahoat sohrreie t io rsoorheiehrsssrrh **a** siotiooe **hit** thaetoiia ersio oehaatot h h **ret i** sti tohoere eoiioa eeh ihsshrahhtsiraseiihrts ai **asar** rtoh ahho **a** saraseheaeeasaihh hartht serteosehs raarihr o o ss orieoeh isss o **ira ora** soaohe sehtrotahs hattttt hh iaoirssroh eorartsoe e saroh hthsatr o **hi** iarorohereoi oao htsor iatthaihri e **hoe trit** ohasethssr rhertriaioa e erattar tttrhsit aooita t **ess** o teaoa h tsrarsaahothoshhtae ar tsai htteooe hiet aei o **so** tss roa rhshrh oihrae **ort** irooieaesoit ooi rhhaoait thtttoeitoraiaoaehstrro **ri** ai trsoretaaositst e **ri** oaishsr ooretret th **re**

William Shakespeare: The Merry Wives of Windsor, order 2, precision 10 heshava **pure** istiles myon t **than** tils ffowind w **me** proughofoma bathintaghis bugof passenenthe meral sicha **haven** hu thagoour **pance be** bl **age**, **pay** athe h tr p **hui we hour** wirowheanthaghuthe st **pine**, hendougengerd, **blow** f **s** pllelllld **pin** prealo tors **pure** fagelo **ming be** tr **be** f **it** ather micom fou t bl thand bli **s** imelyoorer te **tay** wouild **ise** ild illora w **ant isin** f westhel pasordelldoofon w bly itr ils ashowict **hal** f **al ari way**, imyonendselld **pes** bl brou icurds bomili blomalou thallsherdsers pldoull shean sofind helsthullowasit p prd besticat indofoffe angellealerar **s** iche, mans ithels **pur avar** mi p w f w bese, **me** fothare, wengagi f wh atimoowhentri hicashend th **the my** tr tald, ithe, andou **ay an** wi isthe, **s** sh **bof pa** t wavealyof **my** blowist morougou prasthull f t fougens sthengeas sthe **fone hay** sst thur **i** huinean mi whand inds **we** mestougitheay **s** f be, wi **i tin ave me** heroor beagesh **s** te toou fis pof forstr **s** be, bughu feshea fothoulir **be** wicay **ay**, whathistanche, stowar tomofe **be** ineagases se isse **is me**, **me**, ithalill hatallse t **me** averelld **he ish mica** bromoreraghougeate wissh **miro** itofatoncor pe soo **ber** wowhere agenct ilyowhese **bear** astald hissh pagentheaveat isthimy. **as** sstardstowhage, **mil** buica wicou **bely** pearsishave bui wendowhasss **mend** ithur **bes** somoth t weatinont minelld proth te merily wengille avint **for** bl **fer m** anofouss fourou ishalensincootriche ffimile, taghel stome **win** wiminse, menseavaver farinds pallouimiches **as** wourerstag **heal** bora **to he are**, **at** maval bothurou mithilsomesis woontiro **fere** tasstichito **and**, f pisherar huriteastay troffo **m** alowisisomely butoutre **is** towheave icherir fowelyofoo alse **por** pougo fou hunenser **pr tas** whesher im t maloor f **base** imesh fofon ar arofomile thurowaveshe whelloursh t fofimay pllss hene t angoule agases ang t **me** wagh thans ag **so** walo ichelld **be** blshustour sta icale **is** manere, wicurothard pansiseld f whar be, f tasinck, **sot** wicare **m a**

William Shakespeare: The Merry Wives of Windsor, order 3, precision 10 **bes** somanno **a go ants sin** hostre, **my** worearte prack **mis hin** marress **for** misend mistre **mor hall sire page he mis of well is** peetter **hold** ittill **sir me i** whe th **stoll be so** alsto thortud **as** hould youll yound **sle** pree, **fors i car a** stafter, **and hong ming i mis mons**. **havers**. fidis fing **and i de sess** mou, **ber** ambe **aster to cal i my i shall come be** ortise anto **her**, alstrestaffess pon **was all i** mys pagersook, itheartherholph wintol noth **hin that** mak, **and** antold **a bree**, **say**, **it my** yout **hum**, **son**. saccauce, itheress **am wit**. **paget bead a** prabithee **sir** hown hostre **sted**, **and beare ame the** simse, **but** orner hice; annevandess firshe **bres**, **too wand** witer **tor her als**, alstrught **will** dind wersons **him** proo **and my** couthallow simplentoress **of her** thady **comet** oll **band i wo make be** hichere **a** forn. sim **he** boyed **of is** th **my your heer**. misto **sir all but** mistely **me**; **ton** ar, **fort sir** comanter **sed**. **ant sir** comisent **men** trult **more be to mar am i weed**. **son of your** fe, suffir. **so a** se **mist and mast** th mesto **he pager**. **paget my he be to** shon **mould baster** ch sles, **all**, **i** wellow **i** thersen **thick truth say**. **mand if the page**. **mors**, **i** conevanto **but a ster**, istaft th sim. hughted **i** weeres, **mast side is shad he** shaloned ast **mis** probin thater. slet histoody, **be ber**. **eve** se **pre**, **and** theavend boysinte **ord tre i and** heerso **the will**, **falster sore** piess **a ming ford** thater inds **ware** suble theasenter slealoster **sir** slenteress sues fresse **bee my willow tray** thordoess **of arts**, hee **sue** betwer **i ster** trear, faius, **is page i** whear, **my** yout **for** ther whalstaft **as of the and i my** whans **in is if for** astaff ther hugh **we** willoss **as** anym, **thee**, staire **of** willown; **hister** th **the ine to do** trands **took** allowell **you my you** sirin ith whals fals, **wit but** offin ift **ander**. se **say**, **thave** mou, **shere** th sh **is i wels ord hall bur** warrabook holdevicistres, **be mis** anym **as hick**, **i dine we bes fess min for sle** ined **hat** trany

Figure 3: Samples of text with precision 10 for The Merry Wives of Windsor

Charles Dickens: A Tale of Two Cities	“ etaonihsrdlumwcf,gypb."vk-';!?xqj:z()#”
Charles Dickens: A Christmas Carol	“ etoahinsrdluwcgm,fypb.'kv-;xjqz!:?()#"”
Charles Dickens: Barnaby Rudge	“ etaonhisrdlumw,gcfypb'.vk-;j!?xqz:()#"”
Edgar Rice Burroughs: The Warlord of Mars	“ etaohnirsdlufmwcgypb,v.k"-jx;q'z!?:#()”
Edgar Rice Burroughs: Tarzan of the Apes	“ etaohnirsdlufcwmgypb,.kv"-z'jxq?;!#:()”
Emily Bronte: Poems	“ etaosnrhild,uwgmyfcbpv;k-.!'"?:zjqx()#”
Emily Bronte: Wuthering Heights	“ etaonihsrdlumcyfwg,pb.v"k-';!x?jqz()#:”
H. R. Haggard: Child of Storm	“ eatohinsrdluwm,fycgbpk."v-z'?;xq!j:#)(”
H. R. Haggard: King Solomons Mines	“ etaonihsrdluwfgm,cypb.kv"-';x!j?qz():#”
John Cleland: Fanny Hill	“ etoainshrdlmufc,wgypbvk.'-;xj:q!z"()?#”
Lewis Carroll: Brunos Revenge	“ etaoinhsrlduwgy,mf'bcpk.v-!?q;:xj()z"#”
Lewis Carroll: Alices Adventures in Wonderland	“ etaoihnsrdluwg,cymf'pbk.v-!:q?;xj"z()#”
Lewis Carroll: Through the Looking Glass	“ etaoihnsrdluwgycm,'fpbk.-v!q":?jx;z()#”
Slovak folklore: Trojruza	“ aeoistldnrvumkzch,pybj."!-:?fgqwx#';()”
Slovak folklore: Zensky vtip	“ aoeintsldrvucmkpz,yhbj."!?:;-fgqwx#'()”
Washington Irving: Old Christmas	“ etaonishrdlcmufgw,pybv.k;-"q'xj:z#!?()”
Washington Irving: The Legend of Sleepy Hollow	“ etaohnisrdlucfgwm,pbyvk.;-qjx'"z!?:()#”
William Shakespeare: The Merry Wives of Windsor	“ etoasirhnldumyfw,cgpb.vk';:?-!qjx#z()"”
William Shakespeare: Macbeth	“ etoahnsirdlumcwfy,bgp.k'v:;?!-qx#jz()"”
William Shakespeare: Hamlet - Act 3	“ etoanshirldumy,wcfgpbv.k';:?!q-xjz#()"”

Table 3: Most probable paths of order 1

Charles Dickens: A Tale of Two Cities	“the andouris,"wly.'ckf-bjg;mp!)qvx?z#:(”
Charles Dickens: A Christmas Carol	“the andourisck,'ly.);bjf-p:gmqvw!xz#?("”
Charles Dickens: Barnaby Rudge	“the andouris,'w.-bly;"mp!)ckf?g:jqvxz#(”
Edgar Rice Burroughs: The Warlord of Mars	“the andisoruly,"w.'bjck-f?gmp;qvxz#:!()”
Edgar Rice Burroughs: Tarzan of the Apes	“the andisorzly,"w.'mpug-f?ck;bjqvx#:!()”
Emily Bronte: Poems	“the and,-sourily;'vbjckfg."w!mpqxz#:?()”
Emily Bronte: Wuthering Heights	“the andisour."y,-lf'mp;bjckw?)g!qvxz#:(”
H. R. Haggard: Child of Storm	“the and,"isouly.'grmbjck-w!)f?p;qvxz#:(”
H. R. Haggard: King Solomons Mines	“the and,"isoury.'cklf-bjg!);mpw?qvxz#:(”
John Cleland: Fanny Hill	“the andis,"blyofruck'p-m.;g:jqvw!)xz#?(”
Lewis Carroll: Brunos Revenge	“the angoulis,'ry.bckw?)d-p!f:jm;qvxz#("”
Lewis Carroll: Alices Adventures in Wonderland	“the andoury,'sicklf.)-bjg!"w?mp:qvxz#;(”
Lewis Carroll: Through the Looking Glass	“the andoulicrs,'mp."?w!by:f;g-k)jqvxz#(”
Slovak folklore: Trojruza	“ta siedomuzrychl,"n?bfgjvk:pqwx#'.;!()-”
Slovak folklore: Zensky vtip	“to siedal,"nymuzrkvchbfgj.!pqwx#';:?()-”
Washington Irving: Old Christmas	“the andis,"cofry.blupk-w;'g:jm!qvxz#?()”
Washington Irving: The Legend of Sleepy Hollow	“the andis,"bofry.cklup-w;gm!)jqvx'z#:?(”
William Shakespeare: The Merry Wives of Windsor	“the andouris,-by.'lf;ck!g:jmpw?qvxz#()"”
William Shakespeare: Macbeth	“the andoursily,'g.-bjck!f:mp;qvw?xz#()"”
William Shakespeare: Hamlet - Act 3	“the andours,-wily.'f;bjck?gmp:qvxz#!()"”

Table 4: Most probable paths of order 2

Charles Dickens: A Tale of Two Cities	“ ther,"ands.)bcfgivy-woulp?jkmqxz#';:!(”
Charles Dickens: A Christmas Carol	“ thery,'abouldnigs.cfjkmpqvwxz#;:?!()-"”
Charles Dickens: Barnaby Rudge	“ thers,'-knoway."build;cfgjmpqvxz#:?!()”
Edgar Rice Burroughs: The Warlord of Mars	“ thersompland,"buick.fgjqvwxyz#';:?!()-”
Edgar Rice Burroughs: Tarzan of the Apes	“ thers."and,bcoulikfgjmpqvwxyz#';:?!()-”
Emily Bronte: Poems	“ thering,-aboulds;cfjkmpqvwxyz#'.:?!()"”
Emily Bronte: Wuthering Heights	“ thering,"askuld.'boympcfjqvwxz#;:?!()-”
H. R. Haggard: Child of Storm	“ thers,"youlding.'amp-bcfjkqvwxz#;:?!()”
H. R. Haggard: King Solomons Mines	“ thers,"andifulp!bcgjkmovqwxyz#'.;:?()-”
John Cleland: Fanny Hill	“ ther,undiscamov'lbfgjkpqwxyz#.;:?!()-"”
Lewis Carroll: Brunos Revenge	“ therying,'asklbcdfujmopqvwxz#.;:?!()-"”
Lewis Carroll: Alices Adventures in Wonderland	“ thers,'amouldnbcfgivjkpqwxyz#.;:?!()-"”
Lewis Carroll: Through the Looking Glass	“ therying,'abould."cfjkmpsqvwxz#;:?!()-”
Slovak folklore: Trojruza	“ takoli,bceruzdyfghjmnpqsvwx#'.;:?!()-"”
Slovak folklore: Zensky vtip	“ tolicny,"abudemfghjkpqrsvwxz#'.;:?!()-”
Washington Irving: Old Christmas	“ thers,"andifuly.bcgjkmovpqwxz#';:?!()-”
Washington Irving: The Legend of Sleepy Hollow	“ thers,"ayinglowdbcfjkmpuqvxz#'.;:?!()-”
William Shakespeare: The Merry Wives of Windsor	“ ther,-andsompluck;bfgivjqwxyz#'.:?!()"”
William Shakespeare: Macbeth	“ thers,'amblound.cfgivjkpqwxyz#;:?!()-"”
William Shakespeare: Hamlet - Act 3	“ thers,-ackly.bdfumn'givjopqwxz#;:?!()"”

Table 5: Most probable paths of order 3

Charles Dickens: A Tale of Two Cities	“ the st ing and his of mader wassid, been fores," ”
Charles Dickens: A Christmas Carol	“ the scrooge. i was and hist ing of martaider com”
Charles Dickens: Barnaby Rudge	“ the somen and his of ing whou mader coned, beet d”
Edgar Rice Burroughs: The Warlord of Mars	“ the should anden of hat i com was for distereards”
Edgar Rice Burroughs: Tarzan of the Apes	“ the should ander his of mall wast ing forearzated”
Emily Bronte: Poems	“ the st wits and i hater my dessed, beartaing of f”
Emily Bronte: Wuthering Heights	“ the his and i shou wast of mento beeph yon exclin”
H. R. Haggard: Child of Storm	“ the whou and saduko macumbell of hat i dow not, f”
H. R. Haggard: King Solomons Mines	“ the st ing of and wer hated, beforeards frourseen”
John Cleland: Fanny Hill	“ the so my and his of ing whoulder pression fort b”
Lewis Carroll: Brunos Revenge	“ the was and hat i said, beed. whing of you kno m”
Lewis Carroll: Alices Adventures in Wonderland	“ the was and shou knot ittleared, i dow mor hater”
Lewis Carroll: Through the Looking Glass	“ the shou know it and hater was of yon begaid, i ”
Slovak folklore: Trojruza	“ tako sa pred naj doste mi alen zahradach ke, coze”
Slovak folklore: Zensky vtip	“ to poved sa naj mu koni vyholicny coze ak bohadal”
Washington Irving: Old Christmas	“ the of and beent ing hater somed, was con evestio”
Washington Irving: The Legend of Sleepy Hollow	“ the st of and his wassell marter ing foreades, be”
William Shakespeare: The Merry Wives of Windsor	“ the and hat i will bearter come, shou mis pagento”
William Shakespeare: Macbeth	“ the so macbet i hater and whis of coments, but,--”
William Shakespeare: Hamlet - Act 3	“ the and makes so hat ing comen good, whou doter o”

Table 6: Most probable paths of order 3 with repetitions

Text	Dist. 1	Dist. 2	Prod. 1	Prod. 2	Entropy 1	Entropy 2
A Tale of Two Cities (Dickens)	Dickens	Dickens	Burroughs	Burroughs	Haggard	Haggard
A Christmas Carol (Dickens)	Dickens	Dickens	Carroll	Carroll	Dickens	Carroll
Barnaby Rudge (Dickens)	Dickens	Dickens	Dickens	Dickens	Dickens	Dickens
The Warlord of Mars (Burroughs)	Burroughs	Burroughs	Burroughs	Burroughs	Burroughs	Burroughs
Tarzan of the Apes (Burroughs)	Burroughs	Burroughs	Burroughs	Burroughs	Cleland	Burroughs
Poems (Bronte)	Bronte	Dickens	Bronte	Shakespeare	Bronte	Irving
Wuthering Heights (Bronte)	Dickens	Dickens	Dickens	Dickens	Bronte	Haggard
Child of Storm (Haggard)	Haggard	Haggard	Haggard	Haggard	Haggard	Haggard
King Solomon's Mines (Haggard)	Dickens	Dickens	Haggard	Burroughs	Haggard	Haggard
Bruno's Revenge (Carroll)	Carroll	Carroll	Carroll	Carroll	Carroll	Cleland
Alice's Adventures...(Carroll)	Carroll	Carroll	Carroll	Carroll	Carroll	Burroughs
Through the Looking...(Carroll)	Carroll	Carroll	Carroll	Carroll	Carroll	Irving
Old Christmas (Irving)	Irving	Irving	Irving	Irving	Haggard	Cleland
The Legend of...(Irving)	Irving	Irving	Irving	Irving	Cleland	Burroughs
The Merry Wives...(Shakespeare)	Shakespeare	Shakespeare	Shakespeare	Shakespeare	Carroll	Haggard
Macbeth (Shakespeare)	Dickens	Shakespeare	Shakespeare	Shakespeare	Shakespeare	Bronte
Hamlet, act 3 (Shakespeare)	Shakespeare	Shakespeare	Shakespeare	Shakespeare	Bronte	Shakespeare
Success	14:3	14:3	14:3	12:5	11:6	5:11

Table 7: Results of the attribution experiment with the attributed work excluded from the training set.

The results of the experiment can be seen in Table 7. The last line shows for each method the number of correct and incorrect guesses. We see that the most successful methods were Euclidean distance of the first and second order and the inner product of the first order, all of them guessing correctly 14 out of 17 texts (82%). Entropy measures gave poor results, especially for the second order.

When we look at the authors, Burroughs seems to be easiest to guess. Carroll and Shakespeare were also mostly guessed correctly. On the other hand, Emily Bronte is difficult to guess, maybe because one of her texts is a novel, whereas the other poetry.

I also present two more experiments. The first one has all the available texts included in training set and tries to classify all the texts as well. Thus each classified text was included in the characteristics of the correct author, making it easier to classify for some methods. However a characteristics of an author is an average of characteristics of all texts of that author and therefore the advantage is not so great. The results are presented in Table 8. We see that both Euclidean distance methods have guessed all the authors correctly and inner product of order 1 was also quite successful. Entropy behaved badly here as well. It is interesting, that for some English texts the entropy measures found Slovak folklore as a closest match, which means that entropy measure is not even very good for distinguishing among languages.

The second experiment had one work of each author in the training set and the rest of the texts was included in attribution set. I have excluded Slovak texts from the experiments. The results can be seen in Table 9. The conclusion of this experiment are again the same as in the previous once. The only surprising fact is that A Tale of Two Cities by Charles Dickens was correctly recognized only by entropy of order 1, which is in general not a very reliable method.

More results can be obtained using my web-based interface [5]. This interface also displays the values of distances between texts and authors as computed during the attribution so that one can compare the results and see how close the correct answer was to some other answers.

To conclude we have seen the results of three experiments with implemented attribution methods. I have conducted several other similar experiments, all giving comparable results. We may conclude that entropy-based methods do not give good results, they even cannot distinguish Slovak and English languages. Entropy of order 2 is much worse than entropy of order 1. On the other hand, the methods based on Euclidean distance give the correct answer in 80-100% cases. Inner product, especially of order 1, is relatively good. These 3 methods have a similar behaviour and it is difficult to choose the best one. Maybe we could somehow obtain the answer by a combination of their answers. Here are several possibilities, such as plain voting, voting with some weights attached to method, or each method can supply several best candidates etc. Still we would not probably obtain anything which would be really trustworthy and reliable. The question is whether it is possible to construct a reliable method of author attribution in principle, since the style of the writing may differ from text to text for texts written by one author and even human experts cannot determine the author with absolute certainty.

	Dist. 1	Dist. 2	Prod. 1	Prod. 2	Entr. 1	Entr. 2
A Tale of Two Cities	**Dickens**	**Dickens**	Burroughs	Burroughs	Slovak	Haggard
A Christmas Carol	**Dickens**	**Dickens**	**Dickens**	Carroll	**Dickens**	Slovak
Barnaby Rudge	**Dickens**	**Dickens**	**Dickens**	**Dickens**	Slovak	Haggard
The Warlord of Mars	**Burroughs**	**Burroughs**	**Burroughs**	**Burroughs**	**Burroughs**	**Burroughs**
Tarzan of the Apes	**Burroughs**	**Burroughs**	**Burroughs**	**Burroughs**	Cleland	**Burroughs**
Poems	**Bronte**	**Bronte**	**Bronte**	**Bronte**	Dickens	Irving
Wuthering Heights	**Bronte**	**Bronte**	**Bronte**	**Bronte**	Dickens	Haggard
Child of Storm	**Haggard**	**Haggard**	**Haggard**	**Haggard**	**Haggard**	Carroll
King Solomons Mines	**Haggard**	**Haggard**	**Haggard**	Burroughs	**Haggard**	Carroll
Fanny Hill	**Cleland**	**Cleland**	**Cleland**	**Cleland**	**Cleland**	**Cleland**
Brunos Revenge	**Carroll**	**Carroll**	**Carroll**	**Carroll**	**Carroll**	Cleland
Alices Adventures in...	**Carroll**	**Carroll**	**Carroll**	**Carroll**	**Carroll**	Burroughs
Through the Looking Glass	**Carroll**	**Carroll**	**Carroll**	**Carroll**	**Carroll**	Slovak
Trojruza	**Slovak**	**Slovak**	**Slovak**	**Slovak**	**Slovak**	Burroughs
Zensky vtip	**Slovak**	**Slovak**	**Slovak**	**Slovak**	Haggard	Burroughs
Old Christmas	**Irving**	**Irving**	**Irving**	**Irving**	Haggard	**Irving**
The Legend of Sleepy...	**Irving**	**Irving**	**Irving**	**Irving**	Cleland	Burroughs
The Merry Wives of...	**Shakespeare**	**Shakespeare**	**Shakespeare**	**Shakespeare**	**Shakespeare**	Haggard
Macbeth	**Shakespeare**	**Shakespeare**	**Shakespeare**	**Shakespeare**	**Shakespeare**	Bronte
Hamlet - Act 3	**Shakespeare**	**Shakespeare**	**Shakespeare**	**Shakespeare**	Bronte	Dickens
Success	20:0	20:0	19:1	17:3	11:9	4:16

Table 8: Results of the attribution experiment with all the texts both in training set and in attribution set.

	Dist. 1	Dist. 2	Prod. 1	Prod. 2	Entr. 1	Entr. 2
A Tale of Two Cities	Bronte	Haggard	Bronte	Bronte	**Dickens**	Haggard
Barnaby Rudge	**Dickens**	**Dickens**	**Dickens**	Bronte	**Dickens**	Bronte
The Warlord of Mars	**Burroughs**	**Burroughs**	**Burroughs**	**Burroughs**	Irving	Irving
Poems	**Bronte**	Haggard	**Bronte**	Shakespeare	**Bronte**	John Cleland
Child of Storm	**Haggard**	**Haggard**	**Haggard**	Shakespeare	**Haggard**	**Haggard**
Brunos Revenge	**Carroll**	**Carroll**	**Carroll**	**Carroll**	**Carroll**	John Cleland
Through the Looking Glass	**Carroll**	**Carroll**	**Carroll**	**Carroll**	**Carroll**	Dickens
Old Christmas	**Irving**	**Irving**	**Irving**	**Irving**	Haggard	Dickens
The Merry Wives of Windsor	**Shakespeare**	**Shakespeare**	**Shakespeare**	**Shakespeare**	Carroll	Bronte
Hamlet - Act 3	**Shakespeare**	**Shakespeare**	**Shakespeare**	**Shakespeare**	Bronte	**Shakespeare**
Success	9:1	8:2	9:1	6:4	6:4	2:8

Training set: *Charles Dickens:* A Christmas Carol, *Edgar Rice Burroughs:* Tarzan of the Apes, *Emily Bronte:* Wuthering Heights, *H. R. Haggard:* King Solomons Mines, *John Cleland:* Fanny Hill, *Lewis Carroll:* Alices Adventures in Wonderland, *Washington Irving:* The Legend of Sleepy Hollow, *William Shakespeare:* Macbeth

Table 9: Results of the attribution experiment with texts divided between training set and attribution set.

5 Conclusion

In this document I have discussed implemented algorithms and provided results of experiments on a set of natural language texts. The monkey generators of random texts exhibited growing yield of words with the growing order of the frequency table. This is the same result as was observed in [6]. When we decrease the precision of the table, not much changes, until we restrict the number of characters to only 10 per keyboard, when the yield of correct words increases. I have generalized the algorithm for computing most probable paths to other orders. However the computed paths do not have easy-to-see patterns distinguishing different authors and if they should be used for attribution, more subtle methods must be used.

Finally, I have implemented and compared several methods of author attribution, finding out that the entropy is too imprecise to give meaningful results, whereas the methods based on Euclidean distance of normalized matrices seem to be most reliable. Still, they err in some cases, and the question remains, whether we can do better by some other methods based on letter frequencies. Definitely it would be useful to use also some statistics based on word usage and sentence structure.

I must say that I spent a lot of time doing this assignment and that during that time I have discovered several interesting things. I was surprised by a good results of generators using Slovak texts, although this was partially caused by the deficiencies of the dictionary. I also enjoyed playing with different algorithms for creating most probable paths. The greatest difficulties I have encountered with the dictionary and I was not able to solve them to my full satisfaction. I have also learned a lot about HTML input forms and php3 scripting language, because this was the first interactive web-page that I created on my own.

References

[1] 1000000 words that were found at the free internet lexicon and encyclopedia. `http://www.dict.org/100kfound.txt.gz`.

[2] The on-line books page. `http://digital.library.upenn.edu/books/`. University of Pennsylvania.

[3] Php manual. `http://www.php.net/manual/`.

[4] William Ralph Bennett, Jr. *Scientific and Engineering Problem-Solving With the Computer*. Prentice-Hall, Englewood Cliffs, N.J., 1976.

[5] Bronislava Brejova. Web-based interface to the solution of the assignment. `http://genetics.uwaterloo.ca/~bbrejova/a/`.

[6] Nick Cercone. Monkeys at the typewriters. Handout for cs786s, spring 2000. Based on [4, chapter 4].

[7] Dave Raggett. Html 3.2 reference specification. `http://www.w3.org/TR/REC-html32`.

[8] Grady Ward. Moby part-of-speech. `ftp://ftp.dcs.shef.ac.uk/share/ilash/Moby/mpos.tar.Z`.