On the Frequency of Characters in Natural Language Texts

Domenico Cantone and Simone Faro
Department of Mathematics and Computer Science, University of Catania
{cantone|faro}@dmi.unict.it

Abstract

Recent empirical studies indicate that the topology of natural systems is much richer than that of random graphs which traditionally have been used to describe complex systems. In particular, it has emerged that the degree distribution of some real networks is a *power-law*. In this paper we investigate and report a similar characterization on natural language texts. More specifically, we propose a new distribution model that gives an accurate approximation of the relative frequency of characters in natural language texts.
In the second part of the paper, as a case study, we briefly investigate the probabilistic behavior of a string-matching algorithm, assuming the new proposed distribution model, and we compare theoretical versus empirical results.

Keywords: natural languages, text processing, character distributions.

1 Introduction

The emergence of order in natural systems is a constant source of inspiration for both physical, biological, and computer sciences. Traditionally, complex systems have been described by the theory of statistically homogeneous random graphs, with a Poisson degree distribution. In contrast, recent empirical studies indicate that the topology of real systems is much richer than that of random graphs (cf. Albert et al. (1999) and Albert et al. (2000)). In particular, the degree distribution of real networks is a *power-law*, indicating a heterogeneous topology in which the majority of the nodes have a small degree, but with a significant fraction of highly connected nodes that play an important role in the connectivity of the network.

In this paper we investigate and report a similar characterization on natural language texts and present a new distribution model that accurately approximates the relative frequency of characters in natural language texts.

The knowledge of an exact distribution model that describes the relative frequency of characters on texts can play an important role in the probabilistic analysis of text algorithms. It is usually assumed that all characters are independent and equiprobable, i.e. they occur with the same relative frequency $1/\sigma$, where σ is the dimension of the alphabet. This is the case in several complexity analyses of string-matching algorithms, where equiprobability and character independence is assumed for both processed texts and patterns; see, for instance, Baeza-Yates (1987), Baeza-Yates et al. (1990), Baeza-Yates and Régnier (1992), and Mahmoud et al. (1996).

However, most texts on which a text algorithm is usually applied present some definite structure and particular features that cannot be led back to the equiprobability of letters.

The paper is organized as follows. After introducing in Section 2 the notations and terminology used in the paper, we define in Section 3 the notion of power-law and present some of the most common distributions used to approximate the relative frequency of characters and words in natural language texts. Subsequently, in Section 4 we present a new distribution model that relates frequencies of characters with their rank in a natural language text. Finally, in Section 5 we briefly investigate the probabilistic behavior of a string-matching algorithm and compare theoretical versus experimental results under the new distribution model.

2 Notations

In this section we present the notations and terminology used throughout the paper.

Given a finite alphabet $\Sigma = \{c_1, \ldots, c_\sigma\}$, where $\sigma = |\Sigma|$ is the *size* of Σ, a text T of length n over Σ is represented as an array $T[0\,..\,n-1]$, whose components T_i belong to Σ, for $i = 0, \ldots, n-1$.

To maintain information on the relative frequencies of characters occurring in a text, we extend the definition of alphabets as follows.

Definition 1 *A radix Γ is a pair $\Gamma \equiv (\Sigma, \rho_\Gamma)$, where $\Sigma = \{c_1, \ldots, c_\sigma\}$ is an alphabet of σ characters and $\rho_\Gamma : \Sigma \to \mathbb{R}$ is a probability distribution over Σ, called the* frequency function of Γ*, satisfying the following properties:*

(1) $\rho_\Gamma(c) \geq 0$, for all $c \in \Sigma$;

(2) $\sum_{c \in \Sigma} \rho_\Gamma(c) = 1$.

Given a text T, a radix $\Gamma \equiv (\Sigma, \rho_\Gamma)$ is said to be a T-radix if all characters in T are in the alphabet Σ.

Definition 2 *Let T be a text of length n and let Σ_T be the collection of the distinct characters occurring in T. For any character $c \in \Sigma_T$, let n_c denote the number of occurrences in T of c and define the* relative frequency function ρ_T *as $\rho_T(c) = \frac{n_c}{n}$, for $c \in \Sigma_T$. Then the T-radix (Σ_T, ρ_T) is called the* optimal T-radix of T.

Without loss of generality, in what follows we will always assume that the characters of the alphabet $\Sigma = \{c_1, \ldots, c_\sigma\}$ of any radix $\Gamma \equiv (\Sigma, \rho_\Gamma)$ are ordered in such a way that

$$\rho_\Gamma(c_i) \geq \rho_\Gamma(c_j), \quad \text{for } 1 \leq i < j \leq \sigma\,.$$

Then, we say that character c_i has *rank* equal to i and *inverse rank* equal to $\sigma - i + 1$ (in the implicit ordering of Σ via ρ_Γ).

Next, let $\Gamma \equiv (\Sigma, \rho)$ and $\tilde{\Gamma} \equiv (\Sigma, \tilde{\rho})$ be two radices over the same alphabet Σ, where $\tilde{\Gamma}$ is supposed to approximate Γ. Then, we define the approximation error $\mathcal{E}$ of $\tilde{\Gamma}$ w.r.t. Γ as follows:

$$\mathcal{E} = \frac{1}{\sigma} \left(\sum_{i=1}^{\sigma} |\rho(c_i) - \tilde{\rho}(c_i)| \right) .$$

Remark 1 *All the above definitions can be generalized to the case in which Σ is any event space and ρ_Γ is a relative frequency function over Σ.*

3 Uniform distributions and power-laws

In this section we briefly discuss two of the most common probability distributions used to approximate the relative frequency of characters in natural language texts: the *uniform* distribution and the *Zipf*'s law distribution.

A very natural simplifying hypothesis which is commonly assumed in the analysis of text algorithms is that given a text T over the alphabet Σ, all characters of Σ occurs in T with the same probability. Plainly, such an assumption simplifies the analysis and leads to more compact expressions. A radix with a uniform distribution is called a *constant radix*. More formally we have the following definition:

Definition 3 *A radix $\Gamma \equiv (\Sigma, \rho_\Gamma)$, where $\Sigma = \{c_1, \ldots, c_\sigma\}$ is an ordered set of characters, is called a* ***constant radix*** *if its relative frequency function ρ_Γ satisfies*

$$\rho_\Gamma(c_i) = \frac{1}{\sigma}, \quad \textit{for } i = 1, \ldots, \sigma\,. \tag{1}$$

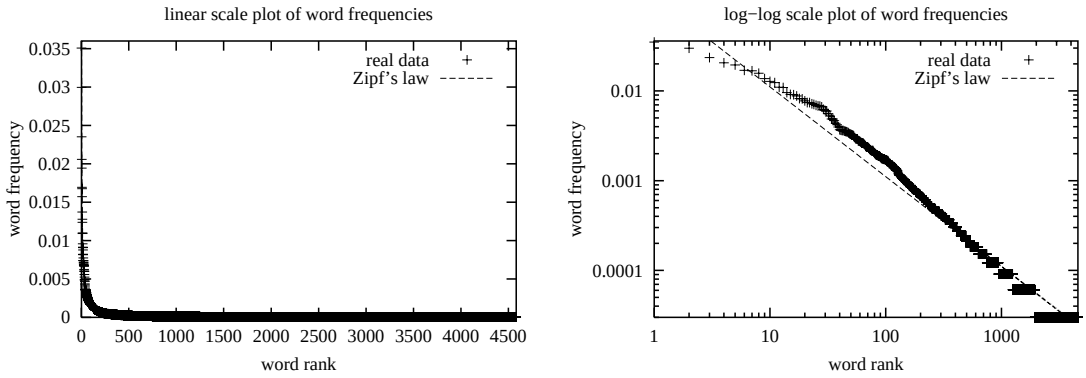


Figure 1: The relative frequencies of words in the English text "Hamlet" by William Shakespeare approximated with the Zipf's law: linear plot on the left and log-log plot on the right.

On the other hand, the relative frequencies of characters in natural languages texts vary over a quite large range. For instance, in the English language the letter e occurs with a relative frequency of about 12.75%, whereas letters such as j and z occur with a much smaller relative frequency of about 0.25%. Thus, when a more precise analysis is needed, one has to use probability distributions which better approximate the relative frequencies in the natural language texts of interest.

It turns out that a large number of naturally occurring phenomena, including word frequencies in texts (cf. Zipf (1932)), distribution of city sizes (cf. Zipf (1949)), the number of visits to an internet site (cf. Huberman and Adamic (2000)), the number of pages within a site (cf. Huberman and Adamic (1999)), the number of links to a page (cf. Albert et al. (1999)), etc., obey a *power-law* distribution. According to a power-law, the relative frequency f of occurrence of some event, expressed as a function of the *rank* r of the event itself in the collection of all events, has the form

$$f(r) \simeq r^{-\gamma},$$

with the exponent γ close to unity.

Zipf's law (cf. Zipf (1932)), named after the Harvard linguistic professor George Kingsley Zipf (1902-1950), is one of the most interesting applications of the power-law to natural languages. In particular, Zipf's law connects the rank of a word in a natural language text with its relative frequency on the text itself as follows: given a text T, Zipf's law states that, with a very good approximation, the relative frequency of a word is inversely proportional to the word rank. More formally if R is the number of different words in T, then the relative frequency $f(r)$ of a word with rank r in T is approximated by the following expression:

$$f(r) \simeq \frac{1}{r \ln(1.78R)}.$$

Figure 1 shows the graph of the relative frequency function of words in the English text "Hamlet" by William Shakespeare in a linear plot and in a log-log plot. In the linear plot, the distribution is so extreme that the curve seems to have a perfect L-shape. In the log-log scale plot, the same distribution assumes a rectilinear shape, which is the characteristic signature of a power-law.

It turns out that though the Zipf's law has been formulated to describe the relation between ranks and relative frequencies of words in texts, it can also be used to describe with a good approximation also the relation between ranks and relative frequencies of characters in natural language texts. This can be done by simply considering each character of the alphabet as a one-letter word. More formally we have the following definition.

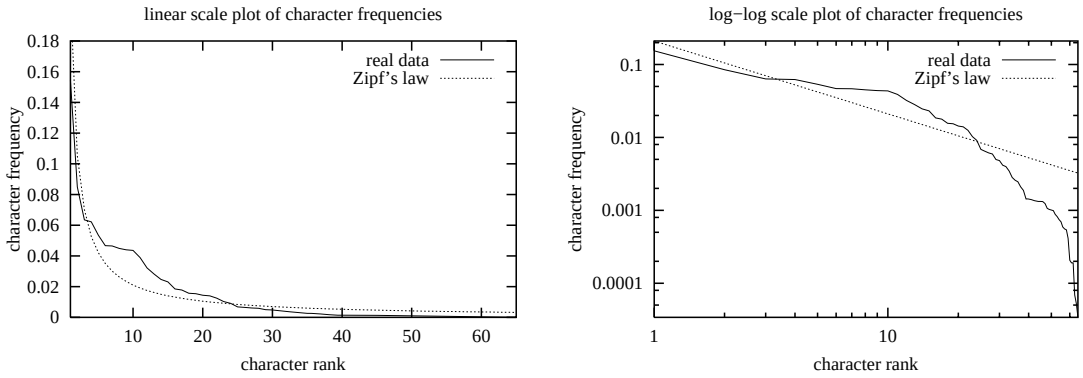


Figure 2: The relative frequency of characters in the English text "Hamlet" by William Shakespeare approximated with the Zipf's law. The approximation error is $\mathcal{E} = 0.0064$: linear plot on the left and log-log plot on the right.

Definition 4 *A radix* $\Gamma \equiv (\Sigma, \rho_\Gamma)$*, where* $\Sigma = \{c_1, \ldots, c_\sigma\}$ *is an ordered set of characters, is called a* ***Zipfian radix*** *if its frequency function* ρ_Γ *is a Zipf's law of the type*

$$\rho_\Gamma(c_i) = \frac{1}{i \ln(1.78\sigma)}, \quad \textit{for } i = 1, \ldots, \sigma\,.$$

Figure 2 shows how Zipf's law approximates the characters frequency function for the English text "Hamlet", both with a linear scale plot and with a log-log plot. In particular, from the log-log plot it is evident that the real frequency function does not have a rectilinear shape. This fact suggests that there might be other distributions which give better approximations of the relative frequency function than Zipf's law.

4 A new character distribution: the inverse-rank power-law

We present a new distribution model that gives a very good approximation of the relative frequency function of characters in terms of their rank both in natural language dictionaries and texts. Such a model is based on the following inverse-rank power-law of degree k, which states that if a collection E of events with cardinality R, then the relative frequency $f(r)$ of an event with rank r in E is approximated by the expression

$$f(r) \simeq \frac{(R - r + 1)}{\sum_{j=1}^{R} j^k}\,.$$

Formally, the dictionary of a finite language $\mathcal{L}$ consists in the alphabetically ordered sequence of all the distinct words of the language. To comply with the most common conventions, we assume that each letter of a dictionary entry is in lower case, except for the first letter, which is in upper case. The main difference between natural languages dictionaries and texts is that the latter contain occurrences of the blank character, which, due to its very high relative frequency, needs to be treated separately.

We first state the version of the inverse-rank power-law for dictionaries of natural languages (namely for the cases in which we do not have to deal with the blank character anomaly).

Definition 5 *A radix* $\Gamma \equiv (\Sigma, \rho_\Gamma)$*, where* $\Sigma = \{c_1, \ldots, c_\sigma\}$ *is an ordered set of characters, is called a* ***dictionary radix*** *if its frequency function* ρ_Γ *is an inverse-rank power-law, namely*

$$\rho_\Gamma(c_i) = \frac{(\sigma - i + 1)^k}{\sum_{j=1}^{\sigma} j^k}, \quad \textit{for } i = 1, \ldots, \sigma\,,$$

where k *is a constant, named* degree *of the radix, and* σ *is the dimension of the alphabet* Σ*.*

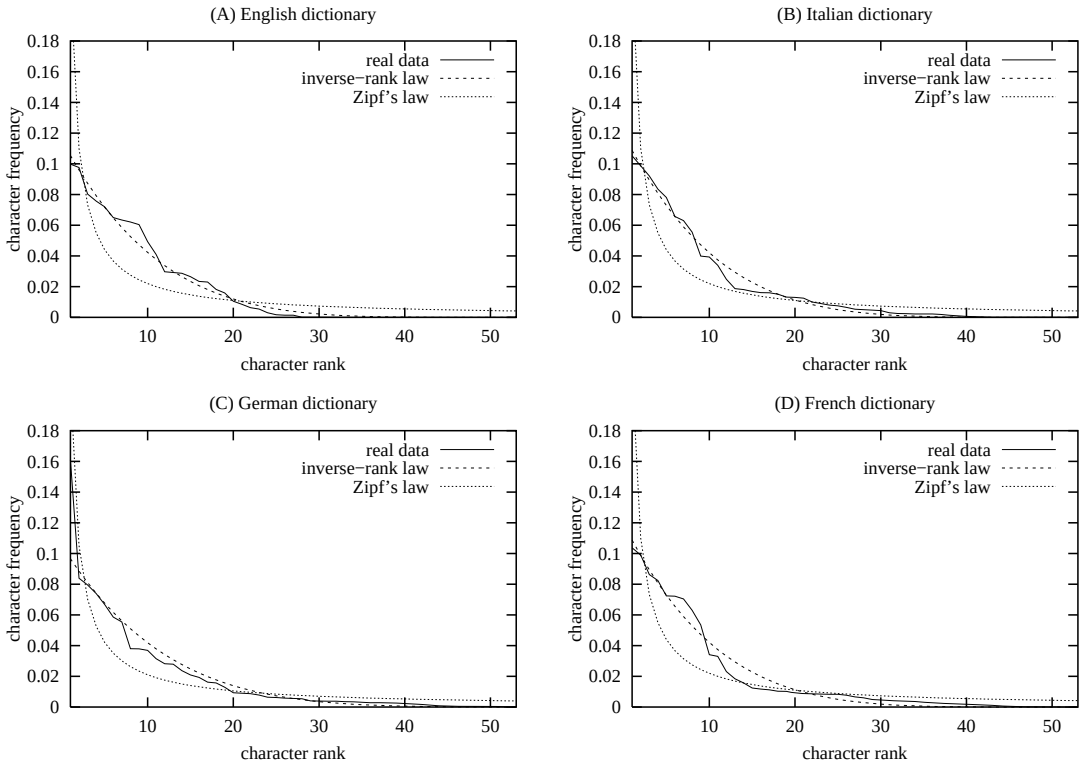


Figure 3: **(A)** Approximation of the frequency function of the English dictionary, containing $151,160$ different words: the NLT-radix with degree $k = 4.9$ gives an approximation error $\mathcal{E} = 0.0023$; the Zipfian radix gives an approximation error $\mathcal{E} = 0.0123$. **(B)** Approximation of the frequency function of the Italian dictionary, containing $277,313$ different words: the NLT-radix with degree $k = 5.1$ gives an approximation error $\mathcal{E} = 0.0023$; the Zipfian radix gives an approximation error $\mathcal{E} = 0.009$. **(C)** Approximation of the frequency function of the German dictionary, containing $155,457$ different words: the NLT-radix with degree $k = 6.1$ gives an approximation error $\mathcal{E} = 0.0014$; the Zipfian radix gives an approximation error $\mathcal{E} = 0.0079$. **(D)** Approximation of the frequency function of the French dictionary, containing $136,595$ different words: the NLT-radix with degree $k = 5.1$ gives an approximation error $\mathcal{E} = 0.0037$; the Zipfian radix gives an approximation error $\mathcal{E} = 0.0089$.

Experimental results carried out on natural language dictionaries show that optimal T-radices of dictionaries can be approximated very well by dictionary radices, for suitable values of the degree k.

More formally if T is a natural language dictionary and $\Gamma \equiv (\Sigma, \rho_\Gamma)$ is its optimal T-radix, where $|\Sigma| = \sigma$, then we have that

$$\rho_\Gamma(c_i) \simeq \frac{(\sigma - i + 1)^k}{\sum_{j=1}^{\sigma} j^k}, \quad \text{for } i = 1, \ldots, \sigma\,, \tag{2}$$

for a degree $k \in \mathbb{R}$ whose value can be determined experimentally. Usually k ranges in the interval $[3..10]$.

Figure 3 shows the approximation of English, German, French, and Italian dictionaries with both the Zipf's law and the inverse-rank law. It turns out from experimental results that, in all cases, the inverse-rank law has a smaller approximation error than the Zipf's law.

Generic natural language texts present a different structure than dictionaries. A first difference is that the rank of a character in a natural language text may be different from its rank in the

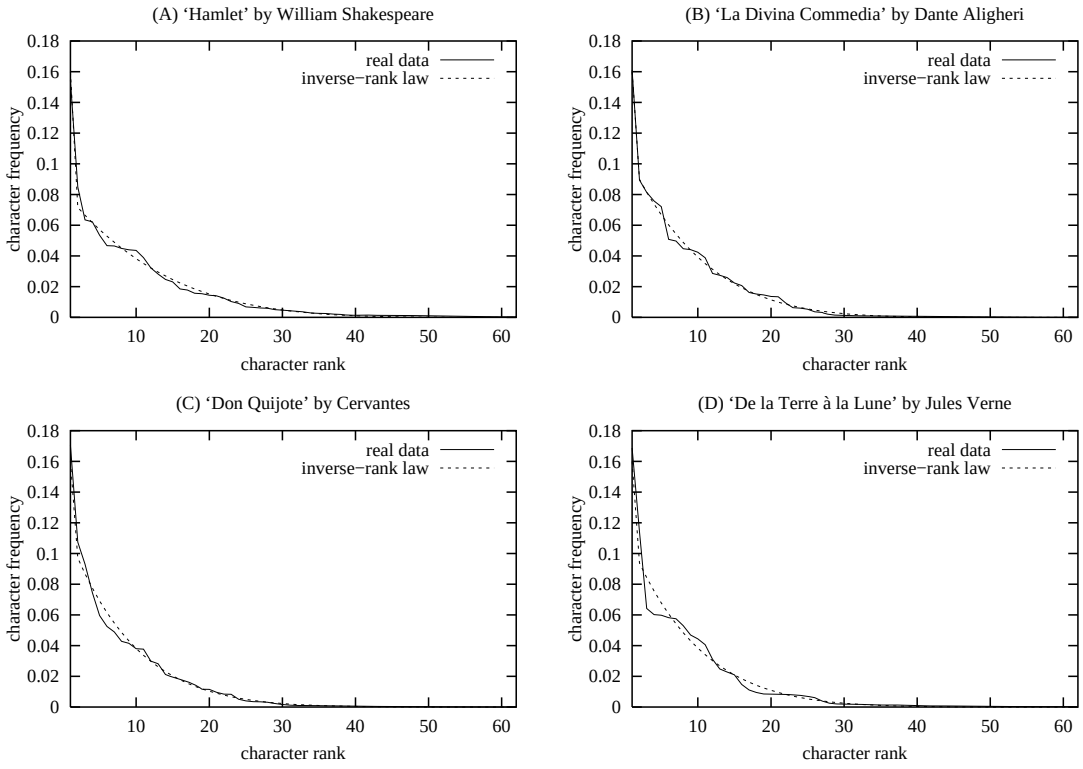


Figure 4: The frequency functions of four natural language texts approximated with NLT-radices according to equation (3). **(A)** The drama "Hamlet", by William Shakespeare, has been approximated with a NLT-radix with degree $k = 4.7$ and approximation error $\mathcal{E} = 0.0012$. **(B)** The poem "La Divina Commedia", by Dante Alighieri, has been approximated with a NLT-radix with degree $k = 5.9$ and approximation error $\mathcal{E} = 0.0013$. **(C)** "Don Quijote", by Cervantes, has been approximated with a NLT-radix with degree $k = 9.8$ and approximation error $\mathcal{E} = 0.0010$. **(D)** "De la Terre à la Lune", by Jules Verne, has been approximated with a NLT-radix with degree $k = 9.0$ and approximation error $\mathcal{E} = 0.0017$.

corresponding dictionary. Consider for instance the letter h in the English language. Though the h occurs in a comparatively small number of English words, it appears in several of the most commonly used words, such as "the", "then", "there", and "that".

Another difference is that the alphabet of a natural language text is usually bigger than the alphabet of the corresponding dictionary, since the former contains punctuation marks, numbers, *blanks*, etc. Moreover, in natural language texts the *blank* character is far more frequent than the remaining characters.

Much the same law defined above may be used also to approximate the optimal radices of natural language texts. We only need to take care of the *blank* problem.

Suppose that l_w is the average length of a word in a natural language text and that n_b is the number of occurrences of the *blank* character. Then since a *blank* character is simply a separator between two different words, we count $n_b + 1$ words in the text and the expected number of occurrences of the *blank* character, $\mathrm{E}[n_b]$, in a natural language text of length n is approximatively given by

$$\mathrm{E}[n_b] = \frac{n - l_w}{l_w - 1}\,.$$

Thus for natural language texts we have the following definition.

Definition 6 *A radix* $\Gamma \equiv (\Sigma, \rho_\Gamma)$*, where* $\Sigma = \{c_1, \ldots, c_\sigma\}$ *is an ordered set of characters of size* σ*, is called a* natural language text radix, NLT-radix *for short, if its frequency function* ρ_Γ *satisfies*

$$\rho_\Gamma(c_i) = \begin{cases} \frac{(\sigma-i+1)^k}{\gamma \sum_{j=2}^{\sigma} j^k} & \text{if } i \in \{2, \ldots, \sigma\} \\ \mathrm{E}[n_b] & \text{if } i = 1 \end{cases}$$

where k *is a constant, named* degree *of the radix, and* γ *is a normalization constant given by*

$$\gamma = \frac{\mathrm{E}[n_b]}{\sum_{i=2}^{\sigma} i^k} + 1.$$

Experimental results show that in a natural language text the average length of a word is approximatively of 5 letters and so the percentage of blank characters in a natural language text is expected to be about 16%. Thus if T is a natural language text and $\Gamma \equiv (\Sigma, \rho_\Gamma)$ is the optimal T-radix, where $|\Sigma| = \sigma$, then we have that

$$\rho_\Gamma(c_i) \simeq \begin{cases} \frac{(\sigma-i+1)^k}{1.19 \sum_{j=2}^{\sigma} j^k} & \text{if } i \in \{2, \ldots, \sigma\} \\ 0.16 & \text{if } i = 1\,. \end{cases} \tag{3}$$

As in the case of dictionaries, the value of the degree k must be determined experimentally. Its value usually ranges in the interval [3..10].

Figure 4 shows the frequency functions of four natural language texts approximated with the inverse-rank law, according to equation (3). In all cases, t turns out that the approximation error is very low. Moreover in Figure 5 we report the results of a more extensive set of experiments performed on a set of natural language texts. For each test we have reported the approximation errors obtained by the inverse-rank law distribution, $\mathcal{E}_{IR}$, and by the Zipf's law distribution, $\mathcal{E}_{Zipf}$.

A tool for testing the inverse-rank and the Zipf's laws on natural language texts is available at the URL:

`http://www.dmi.unict.it/~faro/index.php?sec=nlptrj`

5 A case study: the analysis of a string matching algorithm

In this section we briefly investigate the probabilistic behavior of the Boyer-Moore-Horspool string-matching algorithm (cf. Horspool (1980)), used for searching all occurrences of a pattern P of length m in a text T of length n. We suppose that both patterns and texts are written over the same radix Γ and assume the independence of alphabet characters. Our short investigation covers the two cases when the radix itself has an homogeneous distribution of characters or follows an inverse-rank law distribution as described in the previous section. Our main aim is to compare the theoretical results obtained by assuming a constant and an NLT-radix, with experimental results obtained by testing the Boyer-Moore-Horspool string-matching algorithm on natural language texts.

5.1 The Boyer-Moore-Horspool heuristics

The Boyer-Moore-Horspool Horspool (1980) is an efficient string-matching algorithm. Its heuristics is a slight modification of the original Boyer-Moore *bad-character rule* (see Boyer and Moore (1977)). It has been empirically shown that, in practical cases, the Boyer-Moore-Horspool algorithm leads to better performance though it has a $\mathcal{O}(nm)$ worst-case time. In this section we describe the algorithm and introduce the basic terminology.

Let T be a text of length n and let P be a pattern of length m. When the character $P[0]$ is aligned with the character $T[s]$ of the text we say that the pattern P has *shift* s in T. In this case the substring $T[s\,..\,s+m-1]$ is called the *current window* of the text. If $T[s\,..\,s+m-1] = P$, we

Title (Author)	**Language**	**Length**	σ	k	$\mathcal{E}_{IR}$	$\mathcal{E}_{Zipf}$	%
Hamlet (Shakespeare)	English	174073	65	4.7	.0014	.0064	78%
Othello (Shakespeare)	English	175493	74	4.2	.0033	.0044	25%
The rime of the mariner (Coleridge)	English	20392	63	4.5	.0020	.0061	67%
Romeo and Juliet (Shakespeare)	English	155532	74	5.4	.0030	.0049	38%
The importance of being earnest (Wilde)	English	119468	71	4.3	.0018	.0048	62%
De la Terre à la Lune (Verne)	French	347405	86	9.0	.0017	.0056	69%
La machine d'arithmètique (Pascal)	French	34142	74	7.7	.0021	.0063	66%
L'Avare (Molière)	French	124671	76	5.0	.0024	.0048	50%
L'origine des espèces (Darwin)	French	1388707	82	9.2	.0018	.0065	72%
Les Rèvoltès de la Bounty (Verne)	French	44827	83	8.7	.0017	.0054	68%
Manifeste du Parti Communiste (Marx)	French	117708	81	8.9	.0019	.0065	70%
Così è se vi pare (Pirandello)	Italian	100270	60	4.5	.0026	.0052	50%
Gerusalemme liberata (Tasso)	Italian	687644	69	5.9	.0034	.0065	47%
I Malavoglia (Verga)	Italian	501172	68	7.5	.0011	.0068	83%
Il fu Mattia Pascal (Pirandello)	Italian	454113	81	8.8	.0009	.0062	85%
La Divina Commedia (Alighieri)	Italian	548159	62	5.9	.0010	.0078	87%
Mastro Don Gesualdo (Verga)	Italian	655719	64	6.3	.0011	.0070	84%
Ultime lettere di Jacopo Ortis (Foscolo)	Italian	284831	70	7.7	.0011	.0071	84%
Don Quijote (Cervantes)	Spanish	2106143	89	9.8	.0010	.0057	82%

Figure 5: Experimental results on various of natural language texts

say that the shift s is *valid*. Moreover, if the pattern P has shift s in T, we say that the character $T[s+m-1]$ is the current *head* of the text.

The Boyer-Moore-Horspool algorithm works as follows. First, during a *preprocessing phase*, the algorithm calculates the mapping hbc_P, called the Horspool bad-character function, which later is accessed to determine nontrivial shift advancements. Next, starting with shift $s = 0$, it looks for all valid shifts, by executing a *matching phase*, which checks whether the shift s is valid and computes a *positive* shift increment $hbc_P(T[s+m-1])$. Such increment is used to produce the new shift $s + hbc_P(T[s+m-1])$ to be fed to the subsequent matching phase. The algorithm stops when the shift s exceeds position $n-m$ of the text.

For each value of the shift s the algorithm compares the pattern and the current window of the text by checking if $P[i]$ is equal to $T[s+i]$ for all values $i = 0, \ldots, m-1$, proceeding from right to left, until a mismatch occurs or an occurrence of the pattern is found.

For any given pattern P, the preprocessing phase determines for each letter $a \in \Sigma$ the shift distance $hbc_P(a)$. When the current window of the text, for a given shift s, is to be abandoned the current head of the text determines the shift. Suppose that the last occurrence of the character $T[s+m-1]$ in $P[0\,..\,m-2]$ is at position j. Thus, we should shift the pattern to the right so that the character $P[j]$ corresponds to the character $T[s+m-1]$ of the text. If $T[s+m-1]$ does not occur in $P[0\,..\,m-2]$ then we have to shift the window m position to the right, just past position $s+m-1$ of the text. More formally, at the end of each matching phase we have to shift the window of the text by $hbc_P(T[s+m-1])$ positions to the right, where

$$hbc_P(c) = \min(\{1 \leq k < m | P[m-1-k] = c\} \cup \{m\}), \text{ for all } c \in \Sigma.$$

It turns out that the resulting algorithm performs well in practice and can be immediately translated into programming code. Figure 6 shows the code of the Boyer-Moore-Horspool algorithm. See Baeza-Yates and Régnier (1992) for a simple implementation of the algorithm in the **C** programming language.

5.2 Average number of comparisons for the stationary problem

The average complexity of the Boyer-Moore-Horspool heuristic has been studied in depth (cf. Baeza-Yates et al. (1990), Baeza-Yates and Régnier (1992) and Mahmoud et al. (1996)).

```
Boyer-Moore-Horspool(P, T)
1.    n = length(T)
2.    m = length(P)
Preprocessing:
3.    for all c ∈ Σ do hbc_P(c) = m
4.    for i = 0 to m − 1 do hbc_P(P[i]) = m − i − 1
Searching:
5.    s = 0
6.    while s ≤ n − m do
7.        j = m − 1
8.        while j ≥ 0 and P[j] = T'[s + j] do j = j − 1
9.        if j < 0 then print(s)
10.       s = s + hbc_P(T[s + m − 1])
```

Figure 6: The Boyer-Moore-Horspool algorithm.

In this subsection we briefly summarize the analysis of Boyer-Moore-Horspool algorithm following Mahmoud et al. (1996), where the problem is reduced to the stationary case in which the pattern is fixed. Then, in the next subsection, we generalize this result to the case in which the pattern is random, by reducing the problem to a word enumeration problem. Finally, we approximate the results in order to obtain a more manageable expressions.

One common characteristic of the Boyer-Moore type string-matching algorithms is their dependency on history: the number of comparisons made on a given character depends on the result of comparisons on neighbors. Hence, the first attempts to derive asymptotics used Markov chains Baeza-Yates (1989), Barth (1984). Unfortunately, the use of Markov chains quickly leads to a combinatorial explosion, when the size of the pattern increases.

Here we shall assume that the letters of the text are all independent and chosen from the alphabet with a probability distribution $\{\rho(a)\}_{a\in\Sigma}$.

Theorem 1 (Baeza-Yates et al. (1990)) *Let $P = P'.x$ be a pattern of length m. There exist a unique sequence $\langle c_{i_1}, \ldots, c_{i_j}\rangle$ of characters of Σ and a unique sequence $\langle w_1, \ldots, w_j\rangle$ of words such that*

$$P = w_j \ldots w_1 x \quad \textit{and}$$
$$w_h \in \{c_{i_1}, \ldots, c_{i_h}\}^*.\{c_{i_h}\}, \ \textit{for } h = 1, \ldots, j\,.$$

■

We denote $|w_i|$ by k_i and identify the sequence of characters $\langle c_{i_1}, \ldots, c_{i_j}\rangle$ with the sequence of indices $I = \langle i_1, \ldots, i_j\rangle$. Similarly, we identify the sequence of words $\langle w_1, \ldots, w_j\rangle$ with the sequence $K = \langle k_1, \ldots, k_j\rangle$ of their lengths. Observe moreover that each given pattern P is associated with a unique pair of sequences (K, I) and the shift function hbc_P can be rewritten as follows

$$hbc_P(c_{i_h}) = \begin{cases} 1 & \text{if } h = 1 \\ k_{h-1} + \ldots + k_1 + 1 & \text{if } 2 \le h \le j\,. \end{cases}$$

For instance, the pattern $P = accbbaba$, over the alphabet $\Sigma = \{a, b, c\}$ has associated the sequences: $I = \langle 2, 1, 3\rangle$ and $K = \langle 1, 3, 3\rangle$. In particular, P can be seen as the concatenation $P = w_3.w_2.w_1.a = acc.bba.b.a$, where $|w_1| = 1$, $|w_2| = 3$, and $|w_3| = 3$. Moreover, we have $w_1 \in \{b\}^*.\{b\}$, $w_2 \in \{b, a\}^*.\{a\}$ and $w_3 \in \{b, a, c\}^*.\{c\}$. The Boyer-Moore-Horspool bad-character rule can be obtained as described above: $hbc_P(b) = 1$, $hbc_P(a) = 1 + 1 = 2$, and

$hbc_P(c) = 1 + 1 + 3 = 5$.

With any given pattern P, we associate a *shift generating function*, $f_P(z)$, defined as follows

$$f_P(z) = \sum_{i=1}^{\sigma} \rho(c_i) z^{hbc_P(c_i)}.$$

Observe that $f_P(z)$ must be a polynomial of the form

$$f_P(z) = \rho(c_{i_1})z + \rho(c_{i_2})z^{k_1+1} + \ldots + \rho(c_{i_j})z^{k_{j-1}+\ldots+k_1+1} + (1 - \rho(c_{i_1}) - \rho(c_{i_2}) - \ldots - \rho(c_{i_j}))z^m\,,$$

where j is the number of distinct letters appearing among the first $m-1$ letters of the pattern. We have the following result concerning the number of heads for a stationary problem:

Theorem 2 (Mahmoud et al. (1996)) *For the stationary problem of a given pattern P of length m and a random text T of length n, the expected number of heads $\mathbf{E}[H_n^{[P]}]$ converges asymptotically to $n\mu_P$, where*

$$\mu_P = \frac{1}{f'_P(1)}\,.$$

■

We observe that

$$f'_P(1) = \sum_{i=1}^{\sigma} \rho(c_i) hbc_P(c_i)$$

is the average shift for pattern P.

As in the above example, if $P = accbbaba$ is a pattern over the alphabet $\Sigma = \{a, b, c\}$, then the shift generating function is $f_P(z) = \rho(b)z + \rho(a)z^2 + \rho(c)z^5$. The average shift of P is given by expression $f'_P(1) = \rho(b) + 2\rho(a) + 5\rho(c)$. Assuming equiprobability of characters, we have that $\rho(c) = 1/3$, for $c \in \Sigma$, and the average shift for P has value $f'_P = 1/3 + 2/3 + 5/3 \simeq 2.66$.

Given a fixed pattern P, it is reasonable to expect that at each head position a certain number of comparisons is made by the algorithm, and thus, on average, the expected number of comparisons $\mathbf{E}[C_n^{[P]}]$ is proportional to the expected number of heads, which converges asymptotically to $n\mu_P$. This is indeed the case and the proportionality factor is a complicated function of the pattern, as stated by the following theorem.

Theorem 3 (Mahmoud et al. (1996)) *Let $\mathbf{E}[C_n^{[P]}]$ be the expected number of text-pattern comparisons for a given pattern P of length m and a random text T of length n. Then*

$$\mathbf{E}\Big[\frac{C_n^{[P]}}{n}\Big] \to \mu_P S_P\,,$$

where

$$S_P = 1 + \sum_{j=1}^{m} t_j(j-1) - \sum_{a\in\Sigma} \left(\rho(a) \cdot \sum_{j=hbc_P(a)+2}^{m} t_j(j - 1 - hbc_P(a)) \right)$$

is the average number of comparisons between two shifts and

$$t_j = \begin{cases} 1 & \text{if } j = 1 \\ \rho(P[m-1]) \ldots \rho(P[m-j+1]) & \text{if } 2 \le j \le m-1\,. \end{cases}$$

■

5.3 Averaging over random patterns

Theorem 3 gives the average number of text-pattern comparison performed by the Boyer-Moore-Horspool algorithm in the case in which the pattern is fixed and the text is random. We now extend this result to the general case in which the pattern is random too. A slight change of notation will allow to present the result in a more compact form.

We can observe that all fixed patterns associated with the same pair of sequences (K, I) have the same shift generating function $f(z)$ and thus the same probabilistic behavior concerning the average number of heads. Thus μ_P is therefore characterized by (K, I), and we shall simply use the notation $\mu_{(K,I)}$ for all patterns sharing the same pair of sequences (K, I).

As an example, suppose that the length of the pattern is $m = 10$ and that the alphabet is $\Sigma = \{a, b, c\}$. Then the sequences $I = \langle 2, 1, 3\rangle$ and $K = \langle 1, 3, 3\rangle$ are associated to both patterns $P_1 = bbcaaabc$ and $P_2 = accbbaba$.

By averaging the result of Theorem 3 over random patterns, we obtain the following theorem

Theorem 4 *Let $\mathbf{E}[C_n^m]$ be the expected number of text-pattern comparisons for a random pattern P of length m and a random text T of length n. Then*

$$\mathbf{E}\Big[\frac{C_n^m}{n}\Big] \to \sum_K \sum_I \mu_{(K,I)} S_{(K,I)}\,, \tag{4}$$

where

$$S_{(K,I)} = \sum_{P \in \mathcal{P}_{(K,I)}} \left(\prod_{i=0}^{m-1} \rho(P[i]) \right) S_P$$

with $\mathcal{P}_{(K,I)}$ standing for the set of all patterns sharing the same pair of sequences (K, I). ∎

Moreover, we observe that for a given pattern $P \in \mathcal{P}_{(K,I)}$ of length m the value $\prod_{i=0}^{m-1} \rho(P[i])$ is the probability that pattern P is chosen among all patterns of length m.

The value given by (4) reduces to a computationally difficult enumeration of words. In the following we give a reasonable approximation of the results obtained in Theorem 4.

First we define the shift generating function for all pattern P of length m as follows

$$f_m(z) = \sum_{i=1}^{\sigma} \rho(c_i) z^{\mathbf{E}[hbc_m(c_i)]}\,,$$

where

$$\mathbf{E}[hbc_m(c_i)] = \sum_{j=1}^{m} j Pr\{hbc_m(c_i) = j\} = \sum_{j=1}^{m-1} j\rho(c_i)(1-\rho(c_i))^{j-1} + m(1-\rho(c_i))^{m-1}$$

is the expected shift when the character c_i is a head and the length of the pattern is m. Thus the expected shift for a random pattern of length m is given by

$$f'_m(1) = \sum_{i=1}^{\sigma} \rho(c_i)\mathbf{E}[hbc_m(c_i)] = \sum_{i=1}^{\sigma} \left(\sum_{j=1}^{m-1} j\rho^2(c_i)(1-\rho(c_i))^{j-1} + m\rho(c_i)(1-\rho(c_i))^{m-1} \right)\,.$$

We can obtain an approximation for the expected number of heads in the case in which the pattern is random. This value, denoted by μ_m, is asymptotically $1/f'_m(1)$.

The average number of comparisons performed between two shift for all patterns of length m can be efficiently approximated, according to Baeza-Yates et al. (1990), with the value

$$S_m = 1 + \gamma + \ldots + \gamma^m = \frac{\gamma^{m+1} - 1}{\gamma - 1}\,,$$

where $\gamma = \sum_{i=1}^{\sigma} \rho(c_i)^2$. However the above approximation assumes no knowledge on left neighbors: comparisons are random. However, if the last head position is attained in backward reading, a match certainly occurs and the left neighbor will also be read. Hence, a second approximation is given, according to Baeza-Yates and Régnier (1992), by the value

$$S_m = 1 + \gamma + \ldots + \gamma^{shift} + \gamma^{shift} + \ldots + \gamma^{m-1} = \frac{\gamma^m - 1}{\gamma - 1} + \gamma^{shift} .$$

Thus, the expected number of character comparisons, in the case of random pattern and random text, can be approximated by

$$\mathbf{E}\Big[\frac{C_n^m}{n}\Big] \rightsquigarrow \mu_m S_m . \tag{5}$$

5.4 Theoretical versus experimental results

In this section we compare experimental and theoretical results relative searching a set of patterns on four natural language texts. In particular, theoretical results are computed according to equation (5), assuming a constant radix and an NLT-radix.

For random texts under uniform distribution, the frequency function ρ_Γ is of the type defined by (1). So, for each text character, the average number of comparisons performed by the Boyer-Moore-Horspool algorithm is a function $\psi_c(m, \sigma)$, whose value depends on m and σ.

In the case of natural language texts, by using equation (3), for each text character, the average number of comparisons performed by the Boyer-Moore-Horspool algorithm is a function $\psi_e(m, \sigma, k)$, whose value depends on m, σ, and k.

Figure 7 allows to compare theoretical results with experimental results for four distinct natural language texts. Experiments on each text consisted in searching 2000 patterns. Each pattern P has been obtained by randomly selecting a value k in the range $[0\,..\,n-m]$, where n is the length of the text T and m is the length of the pattern. Then P is the substring $T[k\,..\,k+m-1]$.

It turns out from experimental results that the theoretical values obtained by assuming an inverse-rank law are very close to empirical results. Moreover, in all cases the values obtained under a uniform distribution were always smaller than empirical results.

6 Conclusion

We have presented a variation of the power-law, called inverse-rank power-law, which models very well the relative frequency distribution of characters in natural language dictionaries and texts.

Experimental results show that our proposed distribution achieves better results than the standard power-law distributions. In particular we have compared the inverse-rank law against the uniform distribution and the Zipf's law.

We have also discussed an application of the inverse-rank law to the probabilistic analysis of a string matching algorithm, by computing the average number of comparisons for random patterns and random texts. It turns out that the theoretical values computed by assuming an inverse-rank law distribution perfectly agree with empirical results.

References

Albert, R., Jeong, H., and Barabási, A.-L. (1999). Internet: Diameter of the World Wide Web. *Nature*, 401:130–131.

Albert, R., Jeong, H., and Barabási, A.-L. (2000). Error and attack tolerance of complex networks. *Nature*, 406:378–382.

Baeza-Yates, R. A. (1987). On the average case on string matching algorithms. Report CS-87-66, Computer Science Department, University of Waterloo.

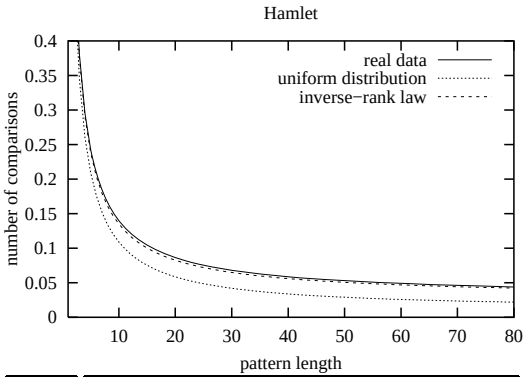
Baeza-Yates, R. A. (1989). String searching algorithms revisited. In Dehne, F., Sack, J. R., and Santoro, N., editors, *Proceedings of the 1st Workshop on Algorithms and Data Structures*,

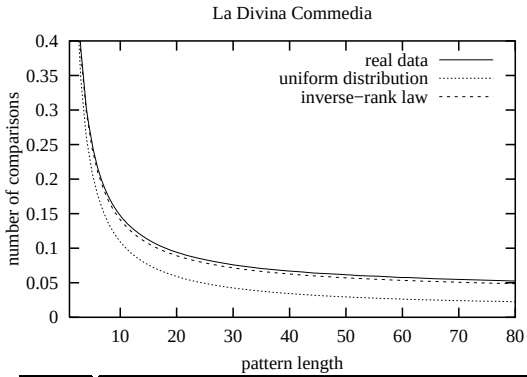
(A) **Hamlet** $\sigma = 65$, $k = 4.7$

m	Ψ	ψ_e	ψ_u
10	0.139	0.135	0.108
20	0.086	0.082	0.058
30	0.068	0.064	0.042
40	0.058	0.056	0.033
50	0.053	0.050	0.029
60	0.049	0.046	0.025
70	0.046	0.044	0.023
80	0.043	0.042	0.022

(B) **La Divina Commedia** $\sigma = 62$, $k = 5.9$

m	Ψ	ψ_e	ψ_u
10	0.147	0.141	0.109
20	0.094	0.089	0.059
30	0.076	0.071	0.042
40	0.067	0.062	0.034
50	0.061	0.057	0.029
60	0.057	0.053	0.026
70	0.054	0.050	0.024
80	0.052	0.048	0.022



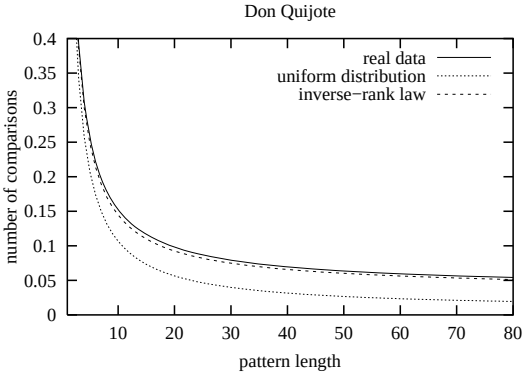


(C) **Don Quijote** $\sigma = 89$, $k = 9.8$

m	Ψ	ψ_e	ψ_u
10	0.152	0.144	0.106
20	0.098	0.092	0.056
30	0.079	0.074	0.039
40	0.069	0.065	0.031
50	0.063	0.060	0.026
60	0.059	0.056	0.023
70	0.056	0.053	0.021
80	0.054	0.051	0.019

(D) **De la Terre à la Lune** $\sigma = 86$, $k = 9.0$

m	Ψ	ψ_e	ψ_u
10	0.152	0.141	0.106
20	0.098	0.089	0.056
30	0.079	0.071	0.039
40	0.069	0.062	0.031
50	0.063	0.057	0.026
60	0.058	0.053	0.023
70	0.055	0.050	0.021
80	0.052	0.048	0.019



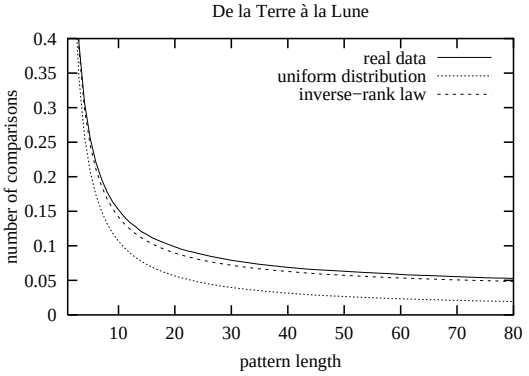


Figure 7: Theoretical vs. experimental results relative to four natural language texts. Symbols ψ_e and ψ_u denote respectively the values of functions $\psi_e(m, \sigma, k)$ (inverse-rank law) and $\psi_u(m, \sigma)$ (uniform distribution), whereas Ψ denotes the average number of comparisons calculated experimentally (real data).

volume 382 of *Lecture Notes in Computer Science*, pages 75–96, Ottawa, Canada. Springer-Verlag, Berlin.

Baeza-Yates, R. A., Gonnet, G. H., and Régnier, M. (1990). Analysis of Boyer-Moore type string

searching algorithms. In *Proceedings of the 1st ACM-SIAM Annual Symposium on Discrete Algorithms*, pages 328–343, San Francisco, CA.

Baeza-Yates, R. A. and Régnier, M. (1992). Average running time of the Boyer-Moore-Horspool algorithm. *Theor. Comput. Sci.*, 92(1):19–31.

Barth, G. (1984). An analytical comparison of two string searching algorithms. *Inf. Process. Lett.*, 18(5):249–256.

Boyer, R. S. and Moore, J. S. (1977). A fast string searching algorithm. *Commun. ACM*, 20(10):762–772.

Horspool, R. N. (1980). Practical fast searching in strings. *Softw. Pract. Exp.*, 10(6):501–506.

Huberman, B. A. and Adamic, L. A. (1999). Internet: Growth dynamics of the World-Wide Web. *Nature*, 401:131.

Huberman, B. A. and Adamic, L. A. (2000). The nature of markets in the World Wide Web. *Quarterly Journal of Economic Commerce*, 1(1):5–12.

Mahmoud, H. M., Régnier, M., and Smythe, R. T. (1996). Analysis of Boyer-Moore-Horspool string-matching heuristic. Report 9634, The George Washington University.

Zipf, G. K. (1932). *Selective Studies and the Principle of Relative Frequency in Language*, volume 23. Harvard University Press, Cambridge, MA.

Zipf, G. K. (1949). *Human Behavior and the Principle of Least Effort*. Addison-Wesley, New York.