Performance of string searching algorithms under various probabilistic models

Mireille Regnier

UNITÉ DE RECHERCHE
INRIA-ROCQUENCOURT

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P. 105
78153 Le Chesnay Cedex
France
Tél.:(1) 39 63 55 11

Rapports de Recherche

N° 1565

Programme 2
Calcul Symbolique, Programmation
et Génie logiciel

PERFORMANCE OF STRING SEARCHING ALGORITHMS UNDER VARIOUS PROBABILISTIC MODELS

Mireille REGNIER

Décembre 1991

RR_1565

Performances of string searching algorithms under Various Probabilistic Models

Mireille Régnier

INRIA, 78153 Le Chesnay, France *

Abstract: We present the performances of several variants of Knuth-Morris-Pratt and Boyer-Moore algorithms. The average searching time is proven to be asymptotically cn for uniform and biased character distributions. The linearity constant are given. The proof relies mainly on combinatorics on words, and some probability. This approach is powerful and notably easily applies for other variants non-studied here.

Keywords: *generating functions, string searching, average analysis.*

Performances d' algorithmes de recherche de motifs pour plusieurs modèles probabilistes

Résumé : Nous présentons les performances de plusieurs variantes des algorithmes de Knuth-Morris-Pratt et Boyer-Moore. Nous prouvons que le temps de recherche moyen est asymptotiquement cn pour des distributions de caractères uniformes et biaisées. Les constantes de linéarité sont données. La preuve se base principalement sur la combinatoire des mots, ainsi que sur les probabilités. Cette approche se rèvèle puissante et notamment s'applique facilement aux autres variantes non étudiées ici.

Keywords: *fonctions génératrices, recherche de motifs, analyse en moyenne.*

*This work was partially supported by the ESPRIT II Basic Research Actions Program of the EC under contract No. 3075 (project ALCOM).

Performances of string searching algorithms under Various Probabilistic Models

Mireille Régnier

INRIA, 78153 Le Chesnay, France *

Abstract

We present the performances of several variants of Knuth-Morris-Pratt and Boyer-Moore algorithms. The average searching time is proven to be asymptotically cn for uniform and biased character distributions. The linearity constant are given. The proof relies mainly on combinatorics on words, and some probability. This approach is powerful and notably easily applies for other variants non-studied here.

Keywords: *generating functions, string searching, average analysis.*

1 Introduction

This paper deals with the complexity of several string searching algorithms under various probabilistic models. String searching consists in finding one or all occurrences of some pattern p in a text t, when the pattern and the text both are strings over a same q-alphabet A. The complexity is usually evaluated by the number of character comparisons between the text and the pattern. The worst-case is well known for most algorithms [Riv77, KMP77]. Nevertheless, the worst case is not sure and algorithms are usually expected to behave much better "on the average" [Yao79]. We support this expectation for the two main classes of algorithms: Knuth-Morris-Pratt-like [KMP77] and Boyer-Moore-like [BM77]. We show in both cases that the expected number of comparisons is asymptotically $K.n$, where n is the size of the text. The linearity constants are also derived.

The study of the average complexity needs a probabilistic model. The first asymptotics came out recently: [Bar85] for the naive algorithm and [Rég89, BYGR90] for Knuth-Morris-Pratt and Boyer-Moore. All assume a *uniform data distribution*. Nevertheless, this hardly holds for numerous applications, notably the search in a text written with some natural language or biological applications. We extend here the results to biased distributions that apply for these applications.

Our approach is algebraic. As a matter of fact, other attempts to derive asymptotics were based on probability, namely Markov chains [Sch88, BY89a, BY89b]. For a pattern length m greater than $3, 4$, only upper or lower bound could be derived. This is due to a combinatorial explosion of the number of states when the size m of the pattern increases. We propose here a kind of *bootstrapping* framework, concentrating first on the main contributions and refining later. Moreover, it applies

*This work was partially supported by the ESPRIT II Basic Research Actions Program of the EC under contract No. 3075 (project ALCOM).

simultaneously and equivalently for any length m of the pattern, the final result showing a simple dependency on this parameter. We reduce the performance analysis to word enumeration. We make use of algebra, mainly generating functions and combinatorics on words. Nevertheless, probability theory is still useful, notably for Boyer-Moore analysis. Some open problems are also pointed out.

2 Algorithms and Probability Models

2.1 Knuth-Morris-Pratt and Morris-Pratt algorithms

We first present the Morris-Pratt algorithm and second its more usual variant, the Knuth-Morris-Pratt algorithm [KMP77]. These algorithms read the text from left to right, trying to align pattern p. They never read backward, which is an advantage over the naive algorithm. It uses a text pointeer TP and a pattern pointer PP that determine the next comparison to be performed: if $TP = i$ and $PP = j$, then the i-th characteer of the text is to be compared to the j-th character of the pattern. After a comparison, both pointers are updated, depending on the result. After a match, both pointers move one step forward, except when the pattern is found. After a mismatch, or when p is found, the new alignement is determined by the largest side p'' of the prefix $p' \preceq p$ already found, i.e. the largest subsequence p'' such that p'' is a strict prefix and suffix of p'. A *next* function is defined, for any j: p' being the prefix of length j of p, p'' its largest side, of length $k-1$, then $next[j] = k$. This means that the first character of p is now aligned on the first character of the right border of p'. Remark that whenever the value of PP is 1, PP remains equal to 1 after a mismatch. For example, let

$$p = 01201201345$$

$$t = 03012012011012012013 45$$

A first match occurs, PP and TP are moved simultaneously to the right. The mismatch between $t[2] = 3$ and $p[2] = 1$ implies to move PP back to $p[1]$ and the next mismatch: $t[2] \neq p[1]$ implies that PP and TP move simultaneously: $PP = p[1], TP = t[3]$. Then, five matches occur, followed by a mismatch, $t[1] = p[9]$. The largest side of $p' = 01201201$ is 01201, hence PP becomes 6. As $p[7] \neq t[11]$ and 01 is the largest side of 01201, PP becomes 3. Finally, $p[3] \neq t[11]$ leads to $PP = 1$ and a mismatch. TP shifts by 1, and the pattern is found.

It is worth noticing that this algorithm does not take into account any information on the mismatching character. As a matter of fact, when $next[j]$ is k, the next comparison to be performed is $t[TP]?p[k]$, whose result is known to be false whenever $p[j] = p[k]$. Hence, $next[j]$ may be redefined as the largest k such that $p[1] \ldots p[k-1] = p[j+1-k] \ldots p[j-1]$ **and** $p[j] \neq p[k]$, that is precisely the Knuth-Morris-Pratt option.

2.2 Boyer-Moore-Horspool

The Boyer-Moore-like algorithms proceed from right to left. As for Morris-Pratt, the pattern is *shifted* to the right whenever the pattern is found or a mismatch occurs. This shift is computed according to either a *match heuristic* or an *occurrence heuristic*. Below, we only consider Horspool simplification [Hor80], based on the second one. Empirical results show that this variant is as good as the original algorithm. Formally, we define the shift table:

$$d[x] = \min\{s | s = m \text{ or } (1 \leq s < m \text{ and } pattern[m-s] = x)\} .$$

Intuitively, the text character x previously compared with the last character of the pattern is aligned with its rightmost position in the pattern. Remark that a character x not appearing among the $m-1$ first characters of the pattern -notably the last one when it occurs only once- determines the largest possible shift, i.e. m. The code is then very simple and given below:

```
bmhsearch( text, n, pat, m )   /* Search pat[1..m] in text[1..n] */
char text[], pat[];
int n, m;
{
    int d[ALPHABET_SIZE], i, j, k;

    for( j = 0; j < ALPHABET_SIZE; j++ ) d[j] = m; /* Preprocessing */
    for( j = 1; j < m; j++ ) d[pat[j]] = m - j;

    for( i = m; i <= n; i += d[text[i]] )          /* Search        */
    {
        k = i;
        for( j = m; j > 0 && text[k] == pat[j]; j-- ) k--;
        if( j == 0 ) Report_match_at_position( k + 1 );
    }
}
```

Figure 1: The Boyer-Moore-Horspool algorithm.

Intuitively, Boyer-Moore-Horspool initially aligns the pattern on the leftmost characters of the text. It proceeds then from right to left. That is, after any shift, the first comparison involves the text character aligned with the last pattern character. Let for example:

$$p = babacac$$

$$t = ******b**bc*?...$$

Here $d[a] = 1, d[c] = 2, d[b] = 4$. The first comparison is $t[7]?p[7]$ that leads to a mismatch. As $b = t[7]$ is also $p[3]$, they are aligned. The shift is then $m-3 = 4$, and $t[11]$ is compared, succesfully, with $p[7]$. When $t[10]?p[6]$ fails, $c = t[11]$ is aligned with the rightmost occurrence of c, i.e. $p[5]$ and the shift is $d[c] = 2$. Note that some characters in the text are *never read*, the ones symbolized by $*$. This particularity is due to the low match probablity -typically $\frac{1}{q}$ for uniform distributions-. Hence,intuitively, the behavior should be *sublinear* and the bigger q, the best the performances are. That is precisely what will be proven in 5.

2.3 Probabilistic assumptions

Let us make precise our probabilistic assumptions. At first, let us remark, from a methodological point of view, that two approaches are, a priori, feasible. On one hand, one may derive first the expectation of the number of comparisons for a given pattern and a random text, and second the average cost when the pattern ranges. But the dual approach is also possible: compute first the

performances for a given text and a random pattern, and then average over the texts. Final results are of course equivalent. But one must avoid having both parameters ranging simultaneously. It appears from the litterature that the first approach is generally chosen. So we do. The justification is that the algorithms considered below are based on a *preprocessing* of the pattern, and not of the text. Then the m characters of the pattern are compared in turn to some characters in the text and, when a mismatch or an equality occurs, algorithmic decisions depend on the pattern. We also remark that the *length* m is a sensible parameter; hence we are led to state the following definitions:

Definition 1 *Let p be a pattern of length $|p| = m$. We note $C_n(p)$ the average number of text-pattern comparisons performed by some string searching algorithm on a random text of size n, with a given character distribution.*

We note $C(n)$ the average value of $C_n(p)$, for a given character distribution in the patterns, when p ranges over the set of patterns of length m.

Definition 2 *We note $\{p_a\}_{a\in A}$ (respectively $\{q_a\}_{a\in A}$) the character distributions in the pattern (respectively the text). That is, for any position in the pattern or the text, the probability that it is occupied by a given character a is: p_a (respectively q_a).*

Remark: We assume notably that the probability of occurrence of a character a at some position does not depend on the neighbours. The equalities $p_a \equiv \frac{1}{q}, \forall a \in A$ or $q_a \equiv \frac{1}{q}, \forall a \in A$ define the uniform distributions.

Notation: We note for $1 \leq i$:

$$\sigma_i = \sum_{a\in A} (p_a q_a)^i \ ,$$

and, for $1 \leq i < j$:

$$s_{j,i} = \sum_{a\in A} p_a^j q_a^i \ .$$

Our parameter σ_1 coincides, when $p_a = q_a$, with the parameter p_{equal} considered in [BY89a] that reduces to $\frac{1}{q}$ for uniform text and pattern distributions.

3 Methods

3.1 State of the Art

Due to the particularities of these string searching algorithms where the next step is defined from the "current state" of the matching test, Markov chains seem, a priori, a natural tool. We describe various attempts and discuss intrinseque limits of such an approach.

The first difficulty is the definition of the meaningful parameter. In [Bar85], where Knuth-Morris-Pratt is considered, one searches the *first occurrence* of the pattern. A full match is then an **absorbing state** and the parameter of interest, the number of comparisons to be performed, is claimed to be equal to the **expectation of the absorbing state**, defined as the number of steps the process makes from a start until absorption. This led to a paradoxal result, while the naive algorithm appeared asymptotically better than Knuth-Morris-Pratt, where Knuth-Morris-Pratt, by construction, always outperforms the naive algorithm for any pattern and text...The reason is that

the claim is true for the naive algorithm, false for Knuth-Morris-Pratt. Details of the refutation may be found in [Rég89].

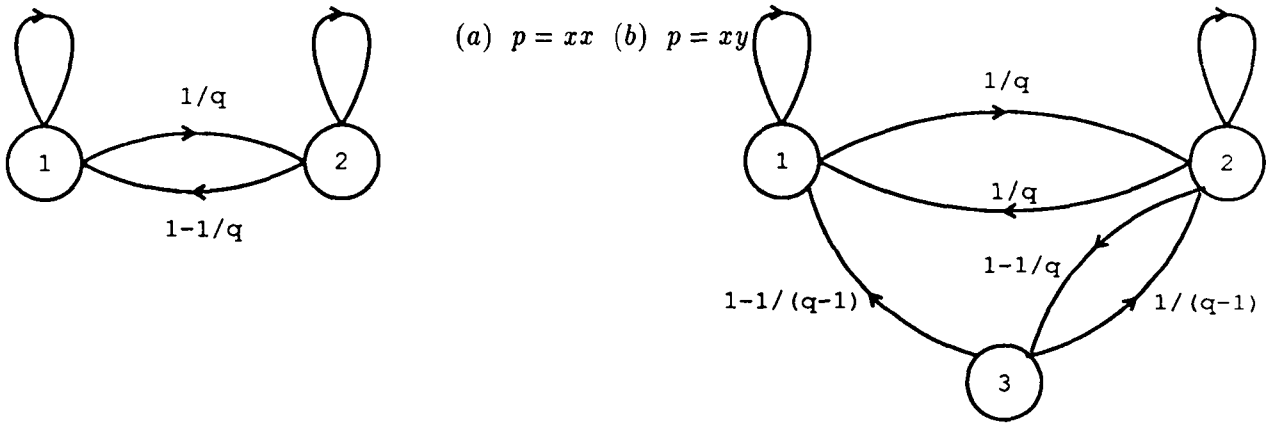
Other analyses consider the average number of comparisons to find *all occurrences* of a given pattern. A crucial parameter, for Knuth-Morris-Pratt, is the number s of characters read since the last mismatch. One out of these is compared twice to the pattern which means that $s+1$ comparisons are necessary to "get rid" of s characters. Hence, the contribution to the cost is: $(s+1)/s$ which leads to the average cost:

$$1 + E\left(\frac{1}{s}\right) .$$

This parameter is also called the **shift**. Note that it will also appear crucial in the analysis of Boyer-Moore in 5.

Let us consider Markov modellization and the problems that occur. First, while the shift appears as a natural parameter, $1/s$ is not. Second, Markov chains, when exact, are **not homogenous** in space. Let us illustrate that point by an example.

Assume the size of the pattern p is 2. We note $p = p[1]p[2]$. Two cases may occur: $p[1] = p[2]$ or $p[1] \neq p[2]$, which induce two different Markov chains.



We are led to define 3 states.

State 1: the current text character x is to be compared to $p[1]$, with no knowledge on x.

State 2: the current text character x is to be compared to $p[2]$.

State 3: the current text character x is to be compared to $p[1]$, with the additional knowledge: $x \neq p[2]$.

At first, let us remark that State 3 is meaningful iff $p[1] \neq p[2]$. The assertion $x \neq p[2]$ implies $x \neq p[1]$ when $p[1] = p[2]$; hence there is no need for a comparison. We get to this case if a mismatch occurs after a comparison of some text character x to $p[2]$: x has to be recompared to $p[1]$. With the conditional knowledge $x \neq p[2]$, the probability for a match (i.e. to go to state 2) is $\frac{1}{q-1}$ and not $\frac{1}{q}$. The reader will notice that States 1 and 3 are almost equivalent: in both cases, a comparison to $p[1]$ is performed and the outgoing states are identical. But the transition probabilities are different as well as the ingoing states.

Finally, one gets two Markov chains. Note that on a q-ary alphabet, q 2-patterns are associated to scheme (a) while $q(q-1)$ 2-patterns are associated to scheme (b).

The limits of the probabilistic methods appear now clearly. The Markov chains that modellize the algorithms are not homogenous in space. It is possible to study the *steady states* for each subchain but, when the length m of the pattern increases, the number of possible chains increase rapidly. The model quickly becomes untractable. As a matter of fact, an attempt by this method is derived in [Han89] that is stopped by the complexity of computation at $m = 5$. An approximation has to be done. Attempts of **embedding in a homogenous** Markov chain weighted by probabilities failed. Such an approximation is mathematically illegal: the steady state is usually an inexact approximation of the steady states of non homogenous Markov chains. Notably, the author in [BY89a] is unable to prove his approximation by a linear number of states. As a matter of fact, as seen in Section 4, our results coincide only up to order 1.

3.2 Combinatorics on Words

We present here some general and basic results on generating functions, that appear to be a powerful tool for string searching performances analyses. More details can be found in [GJ83]. Intuitively, a generating function is an entire function whose Taylor coefficients are simply related to an infinite sequence of values indexed by an entire parameter. Let E be a set such that a *size* can be defined for its elements. Here, E will be a set of words, the size of a word being its length. A valuation p is an application $E \rightarrow R$. It is associated to some parameter of a data structure: number of nodes in a graph, depth of a tree,...We state:

Definition 3 *Let E be a set where a size, i.e. an application from E to N, is defined. Let p be some valuation on E. Then:*

$$P(z) = \sum_{e \in E} p(e) z^{|e|}$$

is the ordinary generating function.

Example: Let E be a set of words on a q-alphabet. The size of a word w, $|w|$, is the number of characters in w. Valuation $p \equiv 1$allows for enumerating the p_n words of length n:

$$P(z) = \sum_{w \in E, |w|=n} 1.z^n = \sum_n p_n z^n \ .$$

It may happen that all the elements of E can be built from simple elements of E via simple combinatorial constructions. If these constructions easily translate on relations on generating functions, the generating function approach becomes a quite powerful tool. We give here such **constructors**. $A(z)$ and $B(z)$ are the generating functions associated to a same parameter computed on different subsets A and B.

Proposition 3.1 *Let C be a set of words w defined as:*

(i) w belongs to the union *of two disjoint subsets A and B,*

(ii) w is the concatenation *of two words from A and B, in a unique manner,*

(iii) w is the repetition of a word of A, m times.

The the generating function of C satisfies:

(i) $C(z) = A(z) + B(z)$,

(ii) $C(z) = A(z) \cdot B(z)$,

(iii) $C(z) = A(z^m)$.

Next proposition is a list of generating functions associated to some simple examples that occur frequently.

Proposition 3.2 *Let us consider the set:*

(i) a^**: the words formed on a single character* a*,*

(ii) A^**: the words formed on a* q*-alphabet* A*,*

(iii) $\{a\} \cdot A^*$*: the words formed on a* q*-alphabet* A *and starting with a given character* a*.*

The associated generating function is:

(i) $E(z) = \frac{1}{1-z}$,

(ii) $E(z) = \frac{1}{1-qz} = W_q(z)$,

(iii) $E(z) = \frac{z}{1-qz} = z \cdot W_q(z)$.

Note that concatenation is fundamental to create words: see rules *(ii)* and *(iii)* of Proposition 3.1. We note $u \preceq v$ (resp. $u \subseteq v$) when u is a prefix (resp. a suffix) of v. A side of a word is a non-empty word that is a strict prefix and suffix of that word. Finally, to enumerate sets, it is important to consider *unique* decompositions. Hence, we state the important notion of *primitive words.*

Definition 4 *A word* $x \in A^*$ *is said to be* primitive *if it is not a power of some other words. Let* P *be the associated generating function.*

Definition and Proposition above may be generalized to multisets, i.e. $E \subset \prod_i E_i$. Notably:

Proposition 3.3 *The multivariate generating function of the words built on a* q*-alphabet is:*

$$F(z_1, \ldots, z_q) = \frac{1}{1 - (z_1 + \cdots + z_q)}$$

The multivariate generating function of words of length less than or equal to m*, is:*

$$F_m(z_1, \ldots, z_q) = \sum_{j \geq 1} (z_1 + \ldots + z_q)^j = \frac{1 - (z_1 + \ldots + z_q)}{(z_1 + \ldots + z_q)} .$$

The multivariate generating function of primitive words is:

$$P(z_1, \ldots, z_q) = \sum_d \mu(d) (F(z_1^d, \ldots, z_q^d) - 1) ,$$

where μ *is the Möbius function.*

Proof: Remark first that when the characters are not distinguished, the substitution: $z_i \to z$ yields $F(z, \ldots, z) = W_q(z)$. Now, a word of size 1 is chosen in either one of singletons $\{a_i\}$ of generating function $1.z_i = z_i$. By rule (i), we get: $z_1 + \ldots + z_q$. A word of length m is associated, rule (ii), to $(z_1 + \ldots + z_q)^m$ and the total set of words to $\sum_m (z_1 + \ldots + z_q)^m = \frac{1}{1-(z_1+\ldots+z_q)}$ by rule (i) again.

4 Knuth-Morris-Pratt Performances

4.1 The Results

In this subsection, we claim our results. We assume a general $(\{p_a\},\{q_a\})$ data distribution. Theorem 4.1 and 4.2 address respectively Morris-Pratt and Knuth-Morris-Pratt performances. Morris-Pratt performances were derived, for a uniform distribution, in [Rég89]. The method in [Rég89] immediately extends to biased distributions, as shown in 4.2. In passing, we also derive the Knuth-Morris-Pratt results.

Definition 5 *Let F be a polynom or an entire series over the set $\{\sigma_i\}$. In the following, we note F_m the truncation of F to monomials $\prod \sigma_{l+\alpha_i}^{\beta_i}$ such that $\sum_i \beta_i(l+\alpha_i) \leq m$.*

Theorem 4.1 *Assume a general $(\{p_a\};\{q_a\})_{a\in A}$ data distribution and a pattern of length m. The average number of comparisons per text character performed by Morris-Pratt algorithm, C_n^{MP}, satisfies:*

$$C_n^{MP} \sim 1 + (\sigma_1 - {\sigma_1}^m) - F_m(\{\sigma_i\}) \ ,$$

where

$$F(\{\sigma_i\}) = \frac{\sigma_2\sigma_1 - \sigma_3}{1-\sigma_2} + \frac{\sigma_2}{1-\sigma_2}\frac{\sigma_1^2.\sigma_2}{1-\sigma_1} - \sum_{l\geq 1} l(\sigma_1 sigma_{l+1} - \sigma_{l+2})$$
$$- \sum_{l\geq 1} l(\sigma_1 - 1)[\sigma_{l+2}(\sigma_{l+2} + \sigma_{l+3}) - (\sigma_{2l+4} + \sigma_{2l+5})] \ .$$

The approximation order is: $O({\sigma_1}^5.\sigma_3)$, and we even have an equality for $m < 5$.

We now perform a Taylor expansion of F. This means notably that m is large enough in order to drop σ_i^{m+1}. The approximation is valid if q is not too small. The validity domain clearly depends on the distribution: when some characters are very rare, the σ_i^k sequences may decrease very slowly.

Corollary 4.1 *For large m, we have:*

$$C_n^{MP} \sim 1 + \sigma_1 - \sigma_1\sigma_2 \ .$$

Additionally:

$$C_n^{MP}(4) = 1 + \sigma_1 - \sigma_1\sigma_2 + \sigma_3 - {\sigma_1}^2\sigma_2 + \sigma_2^2 - {\sigma_1}^4 - \sigma_1\sigma_3 + \sigma_4$$
$$C_n^{MP}(3) = 1 + \sigma_1 - {\sigma_1}^3 - \sigma_1\sigma_2 + \sigma_3$$
$$C_n^{MP}(2) = 1 + \sigma_1 - {\sigma_1}^2$$

Theorem 4.2 *Under the same probabilistic assumptions, the average number of comparisons performed by the Knuth-Morris-Pratt algorithm, C_n^{KMP}, satisfies:*

$$C_n^{KMP} \leq C_n^{MP} \ ,$$

and:

$$\left|\frac{C_n^{MP} - C_n^{KMP}}{n}\right| \leq [F(p_1q_1,\ldots,p_qq_q)[(s_{2,1} - \sigma_2) + F(p_1^2q_1^2,\ldots,p_q^2q_q^2)\sigma_2(s_{2,1} - \sigma_2)]]_m \ .$$

and the approximation is tight up to $O(\frac{1}{q^8})$.

Theorem 4.3 *For uniform distributions, C_n^{MP} satisfies:*

$$C_n^{MP} \sim 1 + \frac{1}{q} - \frac{1}{q^m} + \frac{q-1}{q^2} G_{m-1}(\frac{1}{q^2})$$

where

$$G(x) = \frac{x}{1-x} \sum_l P(x^l) .$$

For any $m > 6$, the approximation order is $\frac{1}{q^{11}}$. It is an exact cost for lower m. More precisely, for $m \geq 9$, this is asymptotically:

$$1 + \frac{1}{q} - (\frac{1}{q^4} + \frac{1}{q^6} + \frac{1}{q^8} + \frac{1}{q^{10}}) - \frac{1}{q^m} .$$

For $2 \leq m \leq 9$, we have the exact developments given in the next table.

m	MP
2	$1 + q^{-1} - q^{-2}$
3	$1 + q^{-1} - q^{-3} - q^{-4} + q^{-5}$
4	$1 + q^{-1} - q^{-4} + q^{-7}$
5	$1 + q^{-1} - q^{-4} - q^{-5} - q^{-6} + q^{-7} + q^{-9}$
6	$1 + q^{-1} - q^{-4} - 2q^{-6} + q^{-9} + q^{-11}$
7	$1 + q^{-1} - q^{-4} - q^{-6} - q^{-7} - q^{-8} + q^{-9}$
8	$1 + q^{-1} - q^{-4} - q^{-6} - 2q^{-8}$
9	$1 + q^{-1} - q^{-4} - q^{-6} - q^{-8} - q^{-9} - q^{-10}$
10	$1 + q^{-1} - q^{-4} - q^{-6} - q^{-8} - 2q^{-10}$

The following table gives the first terms for both algorithms and both distributions.

	Biased	Uniform
MP	$1 + \sigma_1 - \sigma_1\sigma_2$	$1 + \frac{1}{q} - \frac{1}{q^4}$
KMP	$1 + \sigma_1 - s_{2,1}$	$1 + \frac{1}{q} - \frac{1}{q^2}$

Remark that the result for Knuth-Morris-Pratt running under biased distributions, coincides, at the first order $1 + \sigma_1$, with the conjecture given in [BY89a] on the basis of an approximation of the Markov chain when $p_a = q_a$. Hence, it is supported by the simulations presented there. Since our submission to a journal [Rég91b], some results were found by [Han91]. Exact results are given for biased distributions and $m \leq 4$, that imply for greater m the asymptotic developments at order 4 given in our table. For uniform distributions, the exact costs are given up to $m = 10$. We are grateful to the author for pointing out an error at order 6 in our asymptotic development.

4.2 Reduction to Word Enumeration

The basic idea is to take advantage of the *memoryless* property of Knuth-Morris-Pratt. That is, given a character a in a text, the number of comparisons performed on this character depends only of some of its predecessors, and at most $|p|$. Hence, we will enumerate possible preceding words inducing k comparisons. We use a bootstrapping approach that converges very fast. First, we introduce the fundamental notion of *quasi-mismatch*, which will provide an upper bound. Then, we will refine the trivial lower bound: $1 \leq C_n$.

Definition 6 *A* **quasi-mismatch** *is the occurrence in the searched text of a pattern* $p'a, a \in A, p' \in A^*$ *such that:* $p' \preceq p$, $p'a \not\preceq p$.

Lemma 4.1 *For a given pattern* p *of length* m *and a random text, one has, for any data distribution:*

$$\frac{C_n^{KMP}(p)}{n} \leq \frac{C_n^{MP}(p)}{n} \leq E_p(\#quasi-mismatches) \ .$$

Averaging over all patterns yields:

$$\frac{C_n^{KMP}}{n} \leq \frac{C_n^{MP}}{n} \leq E(\#quasi-mismatches) \ .$$

Moreover, for a $(\{p_a\},\{q_a\})$ *distribution, one has:*

$$E(\#quasi-mismatches) = \sigma_1 - {\sigma_1}^m \ .$$

Proof: The only case of multiple comparisons occurs when a comparison $a?p[j], j \neq 1$ ends with a mismatch. This event implies that the sequence $p[1]\ldots p[j-1] = p'$ occurs before a and that $p'a \not\preceq p$. This leads to enumerate those words p' under that double condition: $p' \preceq p$, p' is in the text at that position. Patterns containing α_i times character $a_i, i \in \{1\ldots q\}$ contribute with probability: $(p_1q_1)^{\alpha_1}\ldots(p_qq_q)^{\alpha_q}$. Such words are counted by $F(z_1,\ldots,z_q)-1$. As $|p| = m$, we have $|p'| \leq m-1$, and we must truncate at order $m-1$. Hence, we get:

$$E(\#quasi-mismatches) = \sum_{a\in A} q_a(1-p_a) \sum_{\alpha_1\ldots\alpha_q} (p_1q_1)^{\alpha_1}\ldots(p_qq_q)^{\alpha_q}[z_1^{\alpha_1}\ldots z_q^{\alpha_q}](F_{m-1}(z_1,\ldots,z_q)-1)$$

$$= (1-\sigma_1)(F_{m-1}(p_1q_1,\ldots,p_qq_q)-1) = \sigma_1 - {\sigma_1}^m \ .$$

To derive an equivalent, we study the overestimation. Two cases may occur. Either a quasi-mismatch is counted for a character a that is read as a matching character -we call it a false quasi-mismatch-. Or too many quasi-mismatches are counted for a mismatching character. As extra-comparisons always imply a quasi-mismatch, we are led to study multiple occurrences of p-prefixes to the left of a given position.

Our proof now proceeds in two stages. The inequality

$$E(C_n) \leq E(\#quasi-mismatches) - E(\#false\ quasi-mismatches) \ ,$$

provides an *upper bound*. Below, we derive the expectation of false quasi-mismatches for uniform and biased distributions. To get the exact cost, we then have to enumerate mismatching sequences for which *too many* quasi-mismatches are counted. We will provide an upper bound for this rare event ($O(\frac{1}{q^{\Pi}})$ in the uniform case). Hence, our upper bound appears to be a *lower bound*, tight to that order. We rely upon the next lemma, a straightforward consequence of the Defect Theorem [Lot83].

Lemma 4.2 *Let* p' *be some word in the text, Two cases may occur:*

(i) $\exists!(u,w,l) \in A^{*+} \times P \times \mathcal{N}$ *s.t.* $p' = u.w^l$.

(ii) $\exists!(u_1,w_1) \in A^{*+} \times P$ *and* $\exists!(u_2,w_2,l) \in A^{*+} \times P \times (\mathcal{N}-\{1\})$ *such that* $p' = u_1.w_1 = u_2.w_2^l$.

As uniform and biased distributions can be considered the same, we first evaluate quasi-mismatches for which too many quasi-mismatches are counted. In case (i), the largest side of $u(vu)^m$ is always $u(vu)^{m-1}$. Hence the whole sequence of associated comparisons $t[TP]?(vu)[1]$ is performed and terminates in error. When two decompositions (u_1, w_1) and (u_2, w_2^l) are possible, some of these $l+1$ quasi-mismatches may not induce quasi-mismatches. Thus, we are led to derive an upper bound for case (ii). The smallest sequence satisfying this constraint is: $s = c.bccbc = cbc.cbc$. The condition $sa \in t, sa \not\preceq p, a \neq c$ introduces a contribution $\sum_{c\in A}\sum_{b\neq c}(p_cq_c)^4(p_bq_b)^2(\sigma 1 - p_cq_c) = \sigma_1(\sigma_4\sigma_2 - \sigma_6) - (\sigma_5\sigma_2 - \sigma_7)$, that is $O(\frac{1}{q^{11}})$ in the uniform case. This gives the order of our approximation by the lower bound. Moreover, this only occurs for words of size greater than 6.

We turn now to the study of quasi-mismatches that do not induce extra-comparisons, i.e. associated to *matching* sequences, refining our upper bound. Let us consider first the uniform case for which an easily computable closed formula exists. Following [Rég89], we consider the bivariate generating function

$$M(x,z) = \sum_{u\in A^*+, w\in P} x^{|u|}z^{|w|} .$$

Let us compute first $M(x,z)$. Let $w \in P$, with $|p| = j$. Then u is any suffix of order $1,\ldots,j$. Hence:

$$M(x,z) = \sum_j p_j z^j(x + \ldots + x^j) = \frac{x}{1-x}\sum_j p_j(z^j - (z*x)^j) \;=\; \frac{x}{1-x}(P(z) - P(zx)) .$$

False quasi-mismatches are associated to occurrences in the text of p-prefixes $u(vu)^l a$ such that $a \not\preceq vu$. Then, l quasi-mismatches are counted in our first evaluation for no extra-comparison. Note here that whenever p' has two such decompositions $u_1w_1^l$ and $u_2.w_2$, both induce (different) non pertinent quasi-mismatches to be deduced. Given a decomposition u, w, the event

$$\{u.w^l.a \not\preceq p, a \not\preceq w \; and \; u.w^l.a \preceq t\}$$

occurs with probability $(\frac{1}{q^2})^{|u.w^l|}.(1-\frac{1}{q})\frac{1}{q}$, with the additional condition $|u.w^l.a| \leq m$. Hence, we have:

$$E(false\ quasi-mismatches) = [\sum_{l\geq 1} lM(\frac{1}{q^2}, \frac{1}{q^{2l}})]_{m-1}.\frac{q-1}{q^2} ,$$

and we rewrite the sum as:

$$\frac{1}{q^2-1}\sum_{l\geq 1} l[P(\frac{1}{q^{2l}}) - P(\frac{1}{q^{2(l+1)}})] = \frac{1}{q^2-1}\sum_{l\geq 1} P(\frac{1}{q^{2l}}) .$$

We may illustrate with one example. Assume:

$$p = 1021034\ , t = \ldots 1021037\ldots$$

One has: $10 \preceq p$ and $103 \not\preceq p$ but 3 will be read only once, as it matches the pattern.

Let us turn now to the biased stationary distributions. The definition of $M(x,z)$ may be generalized to a multivariate generating function of $\{x_i\}$ and $\{z_i\}$ as well as the relationship to P. Nevertheless, we have not the same simplifications. To get a faster convergence of our expression, we prefer to consider the disjoint decomposition:

(i) $cu.(cu)^l.a, cu \in P, a \neq c$,

(ii) $u.(cvu)^l.a, cvu \in P, a \in P$.

For our evaluation, we drop first the conditions $cu \in P$ and $cvu \in P$ and assume $l = 1$. The constructors given above in 3.2 show that u^2 and $u.vu$ introduce the generating functions: $F(z_1x_1, \ldots, z_qx_q)$ and $(F(z_1x_1, \ldots, z_qx_q) - 1)F(z_1, \ldots, z_q)$. The double condition: $p' \preceq p$, p' is in t yields the substitutions: $z_i \to p_iq_i$, $x_i \to p_iq_i$. Finally, the additional condition $c \neq a$ contributes by: $\sum_{a \in A}(p_cq_c)^2 \sum_{a \neq c} p_aq_a = \sigma_2\sigma_1 - \sigma_3$ in (i) and: $\sum_{a \in A} p_cq_c \sum_{a \neq c} p_aq_a = {\sigma_1}^2 - \sigma_2$ in (ii). This is $[\frac{\sigma_2\sigma_1 - \sigma_3}{1-\sigma_2} + \frac{\sigma_2}{1-\sigma_2}\frac{\sigma_1^2.\sigma_2}{1-\sigma_1}]_m$. We must now provide a correction. The smallest primitive word satifying (i) is c. Hence, we substract:

$$\sum_{l \geq 2}(l-1).((p_cq_c)^l\sigma_1 - (p_cq_c)^{l+1})$$

$$= \sum_c \frac{\sigma_1(p_cq_c)^2 - (p_cq_c)^3}{(1-p_cq_c)^2} = \sum_{l \geq 1} l[\sigma_1\sigma_{l+1} - \sigma_{l+2}] \ .$$

We prefer the second expression that converges very fast, except for very pathological distributions, where we will use the first. Finally, we may similarly consider: $cb.(cb)^la$ and $b.(cb)^l$ with $b \neq c, c \neq a, l \geq 2$. This yields:

$$\sum_{l \geq 1} l(\sigma_1 - 1)[\sigma_{l+2}(\sigma_{l+2} + \sigma_{l+3}) - (\sigma_{2l+4} + \sigma_{2l+5})]$$

We can turn now to Knuth-Morris-Pratt performances. It reduces to a slight refinement of the reasoning above. We first modify the upper bound by taking out quasi-mismatches such that: $p'(vu)[1] \preceq p$ and $a \neq (vu)[1]$. This translates into two disjoint cases:

(i) $p = a.va$, $t = a.vb$, $a \neq b, v \in A^*$,

(ii) $p = cu.avcu.a$, $t = cu.avcu.b, c \neq a$, $a \neq b$, $u \in A^*$, $v \in A^*$.

(i) contributes by: $\sum_{a \in A}\sum_{b \neq a} p_a^2q_aq_b.F(p_1q_1, \ldots, p_qq_q) = \frac{\sigma_2}{1-\sigma_2}(s_{2,1} - \sigma_2)$. (ii) contributes by $F(p_1^2q_1^2, \ldots, p_q^2q_q^2)F(p_1q_1, \ldots, p_qq_q)\sum_{a \in A}\sum_{b \neq a}\sum_{c \neq a} p_a^2q_a(p_cq_c)^2q_b$. The triple sum is $\sigma_2 s_{2,1}$ up to order q^{-6}. Now, the refinement of the lower bound cannot contribute more than the one for Morris-Pratt, as less configurations are involved. Hence, it is $O(\frac{1}{q^{10}})$ and we have Theorem 4.2.

We presented here the two more common variants of Knuth-Morris-Pratt. Other variants can easily be studied by a small refinement as above. Note that the method is powerful enough to allow very precise developments, even for biased distributions. Moreover, the whole work is not to be done again for a new variant. A contrario, one steadily gets the order of the difference by considering the first different state. Moreover, it generalizes to Markovian dependencies on the characters, which is a more realistic assumption [Rég91a]. This is due to the use of generating functions. It allows to drop a large number of states of the Morris-Pratt automaton while still guaranteeing the approximation order. A contrario, an appraoch via a recurrence like [Bar85, BY89a, Han91] is intrinsically limited by the combinatorial explosion of the number of states. The more precise results by this method are stucked to $m = 4$ and approximation order σ_1^4 in [Han91].

5 Boyer-Moore Performances

We consider here the average performances of Boyer-Moore algorithm. Other point of views - worst-case analysis- are presented in [CGG90, Col91]. Previous results have been derived for uniform text and pattern distributions, in a common work with R. Baeza-Yates and G. Gonnet in [BYGR90, BYR91]. We present here the generalization to biased distributions. The main difficulty is that the memoryless property of Knuth-Morris-Pratt does not hold anymore. The originality of our approach is the exhibition of a stationary process. Practically, our analysis proceeds in two stages: evaluation of the number of heads and then of the number of right to left comparisons. As the number of such comparisons depends on the text-pattern equalities, the expectation is, a priori, low. This allows to conjecture a *sublinear* behavior, that we do prove. Also, a shift is never bigger than m: occurrences could be skipped. Hence, there is at least one head in any m-sequence, and the lower bound is $\frac{1}{m}$.

Theorem 5.1 *For a given fixed pattern p of length m, let H_l be the probability that the l-th character be a head. Then, H_l converges to a stationary probability H_p^∞ defined by:*

$$H_p^\infty = \frac{1}{E_p[shift]} = \frac{1}{\sum_{a\in A} q_a s_p(a)}.$$

Here, $s_p(a)$ is the shift performed when a is found as a head in the text and $E_p[shift]$ denotes the average shift when the aligned character ranges over the q values in the alphabet.

Proof: Position l in a text is a head iff some position $l-j$ is a head with an associated shift j. As such events are not independent, we consider the equivalent expression:

$$\{t[l] \neq head\} = \cup_{j=1}^{m-1} \{t[l-j] = head \text{ and } shift > j\}$$

Note that if in position $l-j$ we had a shift of less than j, say i, that case is considered now in position $l-j+i$ (that is, a different value of j). Here, $\{shift > j\}$ does not depend on the past but only on the character a found in $l-j$ and $s_p(a)$. Notably, it is independent of $\{t[l-j] = head\}$. Thus, we obtain the following linear recurrence

$$H_l = 1 - \sum_{j=1}^{m-1} Pr\{shift > j\} H_{l-j} ,$$

with initial conditions $H_m = 1$ and $H_l = 0$, for $l < m$. As $Pr(shift = 1) \neq 0$, it converges to $1/1 + \sum_{j=1}^{m-1} Pr\{shift > j\}$ and the convergence is exponentially fast. Now, we have:

$$Pr\{shift > j\} = \sum_{a\in A} q_a 1_{s_p(a)>j}$$

where $1_{s_p(a)>j}$ is the characteristic function of the event $\{s_p(a) > j\}$. Hence, we can rewrite:

$$1 + \sum_{j=1}^{m-1} Pr\{shift > j\} = 1 + \sum_{a\in A} q_a \sum_{j\geq 1} 1_{s_p(a)>j}$$

$$= 1 + \sum_{a \in A} q_a (s_p(a) - 1) = \sum_{a \in A} q_a s_p(a) = E_p[shift] \ .$$

We now derive the expected number of comparisons. The following theorem has been obtained for uniform distributions in [BYR91]. We show here that it generalizes to biased distributions by substituting σ_1 to $\frac{1}{q}$.

Theorem 5.2 *Let $C_n(p)$ be the expected number of text-pattern comparisons for a given pattern p and a random text t. Then:*

$$\frac{C_n(p)}{n} \sim H_p^{\infty} \left(1 + q_{p[m]} + 2q_{p[m-1]}q_{p[m]} - q_{p[m-1]}^2 q_{p[m]}\right) \ , \ m \geq 3$$

and the approximation is upper bounded by: $q_{p[m-2]}q_{p[m-1]}q_{p[m]} \cdot \frac{1}{1 - max_{a \in A} q_a}$, or even 0 when $m = 3$.

$$\frac{C_n(p)}{n} = H_p^{\infty} \left(1 + q_{p[m]}\right) \ , m = 2 \ .$$

Proof: Let $S_p(l)$ be the average number of right to left comparisons performed from position l for a given p and random texts. We proceed by successive approximations, each step taking into account a more precise knowledge of the past. One first notes that the occurrence of a character in a given text position is independent of the past, i.e. the characters to the left of that position. We write first:

$$\{S_p(l) > 1\} \cap \{l = head\} \ = \ \{p[m] = t[l]\} \cap \{l = head\}$$

and the probability of this event is: $H_l.q_{p[m]}$. Second:

$$\{S_p(l) > 2\} \cap \{l = head\} \ = \ \{l = head\} \cap [\cup_{a \in A} \{t[l - s_p(a)] = a, t[l-1] = p[m-1], t[l] = p[m]\} \ .$$

$\{l = head\}$ is independent of the others, except for $s_p(a) = 1$ or $a = p[m-1]$. The event reduces then to $\{t[l-1] = p[m-1], t[l] = p[m]\}$, which yields a contribution $q_{p[m-1]}q_{p[m]}$ instead of $q_{p[m-1]}^2 q_{p[m]}$. Hence, we get: $q_{p[m-1]}q_{p[m]}(2 - q_{p[m-1]})$. Finally,

$$\{S_p(l) > i\} \subseteq \{t[l - i + 1] \ldots t[l] = p[m + 1 - i] \ldots p[m]\} \ ,$$

which yields for $\sum_{i>2} P(S_p(l) > i)$ the upper bound: $q_{p[m-2]}q_{p[m-1]}q_{p[m]} \cdot \frac{1}{1 - max_{a \in A} q_a}$.

Let us turn now to small patterns. For 2-patterns, the right to left comparisons always stop at step 1 (or 2) with probability $1 - q_{p[m]}$ (or $q_{p[m]}$). Hence, $S_p(l) \lim H_p^{\infty}(1 + q_{p[m]})$. Similarly, for 3-patterns, at most two comparisons are performed.

We face now the problem of averaging the preceding results when p ranges over all possible patterns. It appears that biased distributions yield more intricate results than uniform ones. We first consider small alphabets, typically $q = 2$. Then we state partial results and a conjecture for large alphabets. Finally, we present the advanced results for the uniform case.
Notation: Let $H(q, m)$ be the probability of being a head when p ranges over all patterns of length m on a q-ary alpphabet. Let $Ph(q)$ be $\lim_{m \to \infty} H(q, m)$.

Theorem 5.3 *Assume a binary alphabet and data distributions* $\{p_a, p_{\bar{a}}\}$ *and* $\{q_a, q_{\bar{a}}\}$*for the pattern and the text. Then, if* $p_a q_a \neq 1$ *and* $p_{\bar{a}} q_{\bar{a}} \neq 1$*:*

$$Ph(2) \;=\; \phi(p_a, q_{\bar{a}}) + \phi(p_{\bar{a}}, q_a)$$

with:

$$\phi(p,q) = (1-p)\sum_{k\geq 1} \frac{p^k}{1+kq} \;.$$

For a uniform text distribution, this reduces to:

$$Ph(2) \;=\; \phi(p_a) + \phi(p_{\bar{a}})$$

with:

$$\phi(p) = 2\frac{(1-p)}{p^2}[-\log(1-p) - p - p^2/2] \;.$$

For uniform text and pattern distributions, this yields:

$$Ph(2) \;=\; 8\ln 2 - 5 \simeq 0.5452 \;.$$

For the degenerated distribution $p_a = q_a = 1$*, one has:*

$$Ph(2) = 1 \;.$$

Proof: There exists $1 \leq k \leq m$ such that a^k (or $\bar{a}^k$) is a suffix of p while a^{k+1} is not. The probability of this event is:

$$p_a^k(1-p_a) \;,\quad k < m \text{ or } p_a^m \;if\; k = m \;.$$

Then, one has:

$$E_p[shift] \;=\; 1.q_a + (1+k)q_{\bar{a}} = 1 + kq_{\bar{a}} \;.$$

This yields the contribution:

$$\sum_{a\in A}\left[(1-p_a)\sum_{k\geq 1}^{m-1}\frac{p_a^k}{1+kq_{\bar{a}}} + \frac{p_a^m}{1+mq_{\bar{a}}}\right]$$

As $\frac{p_a^m}{1+mq_a} \rightarrow_{m\rightarrow\infty} 0$ if $p_a q_a \neq 1$, this yields the result. Remark that the evaluation of the sums leads to the indefinite integral: $\int \frac{x^{\frac{1}{q_a}}}{1-x}$, for which no closed formed formula exists if $\frac{1}{q_a}$ is not an integer. Hence, we have a closed formula iff $\frac{1}{q_a} \in N$ and $\frac{1}{q_a} = \frac{1}{q_a}.\frac{1}{1/q_a - 1} \in N$, which implies $q_a = q_{\bar{a}} = 1/2$.
Remark: The same derivation is possible for any given q, although the complexity of the computation quickly increases. Nevertheless, $q = 4$ is attainable. It is of pecular interest, due to the biological applications. This computation, as well as simulations is the subject of a current work and a forthcoming paper.

We now consider large alphabets. From the computations for uniform distributions in [BYGR90, BYR91] -presented below- , it appears that $Ph(q)$ quickly converges to some limit. We state:

Theorem 5.4 *Let $Ph(q)$ be $\lim_{m\to\infty} H(q,m)$, where $H(q,m)$ is the probability of being a head, when p ranges over all possible patterns of length m on a q-ary alphabet. Assume $\{p_a\}$ and $\{q_a\}$ distributions satisfying the condition: $p_a = 0 \Rightarrow q_a = 0$. Then the following bounds hold::*

$$\frac{1}{\sum_{a\in A} q_a/p_a} \leq Ph(q) \leq \frac{2}{q(q+1)min_{a\in A} q_a} , \quad q < m .$$

When the text distribution is uniform, the upper bound is tight and reduces to:

$$Ph(q) \leq \frac{2}{q+1} , \quad q < m .$$

If there exists (p_a, q_a) such that $p_a = 0, q_a \neq 0$, one has, asymptotically:

$$H(q,m) \geq \frac{1}{\sum_{p_a\neq 0} q_a} . \frac{1}{m} .$$

Remark: The additional condition means that a character a that may appear in the text may also appear in the pattern. In passing, the theoretical lower bound is confirmed.
Proof: The lower bound is a straightforward consequence of the general inequality:

$$E(\frac{1}{x}) \geq \frac{1}{E(x)} .$$

Here, for any character a:

$$E(s_p(a)) = p_a \sum_{j=1}^{m-1} (1-p_a)^{j-1} j + m(1-p_a)^m .$$

This is m if $p_a = 0$ or $1/p_a + o(m)$ if $p_a \neq 0$. Hence:

$$E[E_p(shift)] = \sum_{a\in A} \frac{q_a}{p_a} + \sum_{p_a=0} q_a m .$$

The first term yields the general lower bound. The second term is dominating when it occurs, due to m. Hence, the second lower bound. Note that $\sum_{p_a=0} q_a \leq 1$, which yields a lower bound strictly greater than the theoretical one, $\frac{1}{m}$, for some (pathological) distributions.

To prove the upper bound, we remark that the shift on the i-th different character is greater than or equal to i. Hence:

$$E[shift] \geq 1.q_{a_1} + 2.q_{a_2} + \ldots + q.q_{a_q} \geq (\min_{a\in A} q_a) . \frac{q(q+1)}{2} .$$

Conjecture: When the cardinality q of the alphabet tends to infinity, one has:

$$Ph(q) \to \frac{1}{\sum_{a\in A} q_a/p_a} .$$

We are currently working to exhibit the underlying probabilistic process, and the implied random variables. Some variant of the Central Limit Theorem should apply. Our intuition relies on the fact that $E[shift]$ may be expressed as the sum of *almost* independent variables [BYR90]. Moreover, the exact computations presented in [BYR91] support that conjecture.

Finally, we prove the final result:

Theorem 5.5 *Let $C_{n,m}$ be the expected number of text-pattern comparisons for random texts of size n and random patterns of length m. For a text distribution not too irregular, say $max_a q_a \leq q^{-1/2}$, one has:*

$$\frac{C_{n,m}}{n} \sim H(q,m)(1+\sigma_1+2{\sigma_1}^2) ,$$

or, for large patterns:

$$\frac{C_n}{n} \sim Ph(q)(1+\sigma_1+2{\sigma_1}^2) ,$$

and the approximation is upper bounded by:

$${\sigma_1}^2(\frac{1}{q(q+1)min_a q_a})^2 .$$

Proof: Let us make two remarks first. As the stationary probability H_p^∞ and $S_p(l)$ both depend on the pattern, $E_{\{all\ patterns\}}(\frac{C_n(p)}{n}) \neq E_{\{all\ patterns\}}(H_p^\infty)E_{\{all\ patterns\}}(S_p(l))$. Nevertheless, H_p^∞ does not depend on character $p[m]$. Hence, for the main contribution we have:

$$E(H_p^\infty . q_{p[m]}) = E(H_p^\infty). \sum_{p[m]=a\in A} q_a p_a = H(q,m)\sigma_1 .$$

Similarly, $2E(H_p^\infty q_{p[m-1]}q_{p[m]}) = 2\sigma_1 E(H_p^\infty q_{p[m-1]})$. Now: $\frac{q_{p[m-1]}}{E_p[shift]} = \frac{q_{p[m-1]}}{E_p[shift]-q_{p[m-1]}} + O(\frac{{q_{p[m-1]}}^2}{(E_p[shift]-q_{p[m-1]})^2}$ and $\frac{1}{E_p[shift]-q_{p[m-1]}} = \frac{1}{E_p[shift]} + O(\frac{q_{p[m-1]}}{E_p[shift]^2})$. Hence: $E_{\{all\ patterns\}}(\frac{q_{p[m-1]}}{E_p[shift]}) \sim H(q,m)\sigma_1$ and the approximation is upper bounded by $\frac{\sigma_1}{(q(q+1)min_a q_a)^2}$.

In the following, we give an exact expression of $H(q,m)$ and tabulate it for uniform distributions [BYGR90, BYR91]. Namely:

Theorem 5.6 *For uniform text and pattern distributions, the probability of being a head, when p ranges over all patterns of length $m+1$ on a q-ary alphabet is:*

$$H(q,m) = q\sum_{j=1}^{q}\binom{q}{j} \sum_{\substack{k_i\geq 1 \\ k_1+\cdots+k_{j-1}<m-1}} \frac{\prod_{i=1}^{j-1}\left(\frac{i}{j}\right)^{k_i}\left(\frac{j}{q}\right)^{m-1}}{j+\sum_{i=1}^{j-1}(j-i)k_i + m(q-j)} .$$

Moreover:

$$Ph(q) = \lim_{m\to\infty} H(q,m) = q \sum_{\substack{k_i\geq 1 \\ i=1,\ldots,q-1}} \frac{\prod_{i=1}^{q-1}\left(\frac{i}{q}\right)^{k_i}}{q+\sum_{i=1}^{q-1}(q-i)k_i} + O((1-1/q)^{m-1}/m) .$$

Table 1 gives some exact values for probability $H(q,m)$ for small q in $\{2,3,4,5\}$ and various m. A quick convergence appears, which supports the conjecture above.

As a conclusion, it appears that Boyer-Moore-Horspool is *sublinear* as expected. The linearity constant is around $\frac{\alpha}{q}$ for uniform distributions. This confirms the intuition: the bigger q, the more Boyer-Moore outperforms Knuth-Morris-Pratt. As Boyer-Moore is always better than its Horspool variant, it is also sublinear. It is worth comparing the bound $\frac{1}{q}$ to the theoretical lower bound $\frac{\log_q m}{m}$ given in [Yao79]. It suggests that the domain of validity of Horspool approximation is roughly $q \leq m \leq q^2$. For bigger m, memorization is not enough. A challenging open problem is to relate the number of characters it is worth memorizing and the ratio m/q. This is clearly related to the study of the number of states in a Boyer-Moore automaton [BYGR90].

	q			
m	2	3	4	5
2	.666667	.600000	.571429	.555556
3	.583333	.476190	.433333	.410256
4	.558333	.421958	.368094	.339860
5	.550000	.395198	.332003	.299440
6	.547024	.381249	.310381	.273988
7	.545908	.373737	.296842	.257047
8	.545474	.369597	.288135	.245365
9	.545300	.367275	.282438	.237120
10	.545229	.365954	.278663	.231206
15	.545178	.364246	.271961	.218487
20	.545177	.364118	.270950	.215601
25	.545177	.364108	.270783	.214899
30	.545177	.364107	.270754	.214722

Table 1: Exact values for $H(q, m)$.

6 Conclusion

We have presented an average analysis of the main string searching algorithms under various probabilistic distributions. We provide an answer to conjectures over the expected behavior of Knuth-Morris-Pratt and Boyer-Moore algorithms. The expected number of comparisons was proved to be asymptotically cn, and linearity constants were derived. In the second case, one has $c < 1$-around $1/q$ in the uniform case-; hence, we proved the conjectured *sublinearity*. An approach via word enumeration was proposed that proved to be powerful, as it "sticks" to the intrinseque nature of the algorithms. A challenging problem is now to determine different range domains for m, q and the *data distributions* so as the best algorithm may be chosen in any case. Notably, it is of interest to characterize for each algorithms the "pathological" distributions, and study the behaviour. It is also worth extending that work to string searching with k mismatches.

References

[Bar85] G. Barth. An analytical comparison of two string matching algorithms. *IPL*, 30:249–256, 1985.

[BM77] R. Boyer and S. Moore. A fast string searching algorithm. *CACM*, 20:762–772, 1977.

[BY89a] R. Baeza-Yates. Efficient text searching. PhD Thesis CS-89-17, Univ. Waterloo, Canada, 1989.

[BY89b] R.A. Baeza-Yates. String Searching Algorithms Revisited. In *WADS'89*, volume 382 of *Lecture Notes in Computer Science*, pages 75–96. Springer-Verlag, 1989. Proc. WADS'89, Ottawa.

[BYGR90] R. Baeza-Yates, G. Gonnet, and M. Régnier. Analysis of Boyer-Moore-type string searching algorithms. In *SODA'90*, pages 328–343. SIAM, 1990. Proc. Siam-ACM Symp. on Discrete Algorithms, San Francisco, USA.

[BYR90] R. Baeza-Yates and M. Régnier. Fast algorithms for two dimensional and multiple pattern matching. In *SWAT'90*, volume 447 of *Lecture Notes in Computer Science*, pages 332–347. Springer-Verlag, 1990. Proc. Swedish Workshop on Algorithm Theory, Bergen, Norway.

[BYR91] R. Baeza-Yates and M. Régnier. Average running time of Boyer-Moore-Horspool algorithm, 1991. to appear in TCS.

[CGG90] L. Colussi, Z. Galil, and R. Giancarlo. On the exact Complexity of string matching. In *FOCS'90*, pages 135–143. IEEE, 1990. Proc. 31-st Annual IEEE Symposium on the Foundations of Computer Science.

[Col91] R. Cole. Tight Bounds on the Complexity of the Boyer-Moore string matching algorithms. In *SODA'91*, pages 224–233. SIAM, 1991. Proc. 2-nd Siam-ACM Symp. on Discrete Algorithms, San Francisco, USA.

[GJ83] I. P. Goulden and D. M. Jackson. *Combinatorial Enumeration.* John Wiley, New York, 1983.

[Han89] Ch. Hancart. Sur le cas moyen des algorithmes de recherche d'un mot dans un texte. DEA, Université de Paris VII, 1989.

[Han91] Ch. Hancart. Algorithme de Morris et Pratt et ses raffinements: une analyse en moyenne. Research report 91.56, Université de Paris VII, October, 1991.

[Hor80] R. N. Horspool. Practical fast searching in strings. *Software-Practice and Experience*, 10:501–506, 1980.

[KMP77] D.E. Knuth, J. Morris, and V. Pratt. Fast pattern matching in strings. *SIAM J. on Computing*, 6:323–350, 1977.

[Lot83] Lothaire. *Combinatorics on Words.* Addison-Wesley, Reading, Mass., 1983.

[Rég89] M. Régnier. Knuth-Morris-Pratt algorithm: an analysis. In *MFCS'89*, volume 379 of *Lecture Notes in Computer Science*, pages 431–444. Springer-Verlag, 1989. Proc. Mathematical Foundations for Computer Science 89, Porubka, Poland.

[Rég91a] M. Régnier. A language approach to string searching evaluation, 1991. in preparation.

[Rég91b] M. Régnier. Performances of String Searching Algorithms under Various Probabilistic Models, 1991. submitted.

[Riv77] R. L. Rivest. On the Worst-Case Behavior of String-Searching Algorithms. *S.I.A.M. J. on Comp.*, 6:669–674, 1977.

[Sch88] R. Schaback. On the Expected Sublinearity of the Boyer-Moore Algorithm. *SIAM J. on Computing*, 17:548–558, 1988.

[Yao79] A. C. Yao. The complexity of pattern matching for a random string. *SIAM J on Computing* ., 8:368–387, 1979.

Imprimé en France
par
l'Institut National de Recherche en Informatique et en Automatique

ISSN 0249-6399