Average Case Analysis of the Boyer–Moore Algorithm

Tsung-Hsi Tsai
Institute of Statistical Science, Academia Sinica, Taipei 115, Taiwan;
email: chonghi@stat.sinica.edu.tw

Received 19 November 2003; accepted 27 December 2004; received in final form 17 July 2005
Published online 27 December 2005 in Wiley InterScience (www.interscience.wiley.com).
DOI 10.1002/rsa.20111

ABSTRACT: Limit theorems (including a Berry–Esseen bound) are derived for the number of comparisons taken by the Boyer–Moore algorithm for finding the occurrences of a given pattern in a random text. Previously, only special variants of this algorithm have been analyzed. We also propose a means of computing the limiting constants for the mean and the variance. © 2005 Wiley Periodicals, Inc. Random Struct. Alg., 28, 481–498, 2006

1. INTRODUCTION

String matching is a huge area with a wide literature and a large number of applications. The basic problem is to identify all occurrences of a given pattern in a large text. We study in this paper the most widely used string-matching algorithms, the Boyer–Moore (BM) algorithm and its simplified version, the Boyer–Moore–Horspool (BMH) algorithm. We are concerned with the probabilistic behavior of the algorithms, as well as the evaluation of the constants in the limit theorems.

Applications of string matching are found in many communicative media, including language processing, graphics, videos, and sound. Typical applications include key-word searches (or replacement) in any text editing software, the use of search engines, phone number lookup, searches in databases like Gen-Bank, etc. Although the use of exact string matching in computational biology may be limited, exact string matching is the prototype, both in algorithms and in analysis, of more useful approximate string matching and hidden string search algorithms; see Flajolet, Szpankowski, and Vallée [9].

Throughout the paper, $\mathcal{A}$ is a set of q characters, T is a text of n characters from $\mathcal{A}$, and S is a pattern of m characters from $\mathcal{A}$. We assume that $q \geq 2$ and $n \gg m$. Although several factors determine the running time of a program, we deal in this paper only with the number of character comparisons; other cost measures may be similarly analyzed. We denote by $C_n = C_n(A)$ the number of comparisons used by algorithm A.

The most naive algorithm for finding all occurrences of a pattern in a text consists of comparing consecutively the characters of the pattern to each m-length substring of the text. The procedure moves to the next m-length substring as soon as a mismatch occurs. This simple algorithm has $C_n = m(n - m + 1)$ in the worst case (when all pairwise comparisons have been executed).

The Knuth–Morris–Pratt (1977) algorithm, which is based on building up an automaton from the pattern, avoids comparing the match part in the text to the pattern more than once and improves the worst case to $C_n \leq 2n - m$ (see Cole [8]), the average case of C_n/n being close to $1 + 1/q$ (see Régnier [13]).

However, it has been observed that in many applications, the average value of C_n/n for several algorithms is less than 1 (see [12] for recent experiments). Among them, the BM algorithm is considered the most efficient algorithm (see Lecrocq's *Handbook of Exact String-Matching Algorithms* [6]).

Despite its popularity in diverse applications, probabilistic analysis of the BM algorithms has not received much attention in the literature. Most known probabilistic results are for the BMH algorithm. We first summarize some related results and then present our new ones.

Baeza-Yates and colleagues [1, 2] applied an analytic approach to the average-case analysis of the BMH algorithm. Under the assumption that the text is independent and identically distributed (iid), they derived an exact expression for the linearity constant $\mu = \lim_{n\to\infty} E(C_n/n)$. For earlier results, see Barth [3] and Régnier [13].

The first distributional result for string-matching algorithms was derived by Mahmoud, Smythe and Régnier [11]. They proved the asymptotic normality of C_n via analytic methods (the text being iid) for the BMH algorithm,

$$\frac{C_n - \mu n}{\sqrt{Bn}} \xrightarrow{\mathcal{D}} \mathcal{N}(0, 1),$$

where $\xrightarrow{\mathcal{D}}$ denotes convergence in distribution and $\mathcal{N}(0, 1)$ denotes standard normal random variable with zero mean and unit variance. However, the value of B is difficult to compute.

Smythe [14] extended the asymptotic normality result for Markovian input by a completely probabilistic approach (via theory of Markov chains). A simple expression for the linearity constant μ was also derived, but the state space of his Markov chain seems difficult to further simplify and the evaluation of B still remains unclear.

We propose a similar but simpler Markov-chain approach; we consider the windows together with a distance counter as a Markov chain and derive a Berry–Esseen bound, the law of the iterated logarithm, and a better estimate for the mean for C_n with explicit determination of μ and B under the assumption that the text is iid and the pattern is fixed.

One major drawback of the usual Markovian approach is that the computational complexity grows exponentially when the pattern length increases. Our main contribution is to construct a state space with a growth rate of order m^2. We thus are able to compute the linearity constant μ for the BM algorithm when the pattern length is of reasonable size.

With the construction of the state space, we can also compute the exact value of B for both the BM and the BMH algorithms. All computations can be efficiently done by mathematical softwares (such as Maple); numeric results are presented in Section 5.

2. DESCRIPTION OF THE ALGORITHMS

For completeness, we describe the BM algorithm in this section; see Boyer and Moore [5] or Charras and Lecroq [6] for more details.

Denote the text by $T = t_1 t_2 \cdots t_n$ and the pattern by $S = s_m s_{m-1} \cdots s_1$; note that in the paper we index the pattern from *right to left* because BM-type algorithms scan in that direction inside each comparing window.

The BM Algorithm

We begin by aligning S under the first window $t_1 \cdots t_m$. Start with s_1 and compare each pattern character to the corresponding character in the window from right to left, until a mismatch is detected or when the left end of the window is reached (an occurrence is found if all m characters match). Then, S is shifted to the right and a new window in the text is opened.

The distance shifted (from the current window to the next) is determined by the following two rules.

- *Bad-character rule*. For all $a \in \mathcal{A}$, let

$$B(a) = \min\{k \geq 1 : s_{k+1} = a\}.$$

- *Good-suffix rule*. For all $1 \leq i \leq m$, let

$$G(i) = \min\left\{k \geq 1 : \begin{array}{l}[s_{\min\{m,i+k-1\}}, s_{k+1}] \text{ matches exactly with a suffix of} \\ [s_{i-1}, s_1] \text{ if } i \geq 2 \text{ and } s_{i+k} \neq s_i \text{ if } i + k \leq m\end{array}\right\},$$

where $[s_{j_1}, s_{j_2}], j_1 \geq j_2$ denotes the sequence $s_{j_1} s_{j_1-1} \cdots s_{j_2}$ and $\min\{\varnothing\} \equiv m$.

If there are $i - 1$ matches, the ith pair is a mismatch, and a is the mismatch character in the text, then the distance shifted is

$$\max\{G(i), B(a) - i + 1\},$$

and we call $G(i)$ the *good-suffix shift* and $B(a) - i + 1$ the *bad-character shift*. If all m characters match, then the distance shifted is $G(m)$.

The BMH Algorithm

This is a simplification of the BM algorithm using only a bad-character rule. The distance shifted is $B(a)$, where a is the rightmost character of the current window.

When the first comparison yields a mismatch, i.e., $i = 1$, the bad-character shift dominates the good-suffix shift, and thus the distances shifted by the BM and BMH algorithm are identical. When q is large (as in Chinese language), it becomes more likely that a mismatch

happens right at the first comparison in the current window. In this case, the performance of BM and BMH algorithms is comparable.

We give an example to demonstrate both algorithms. Suppose $\mathcal{A} = \{a, b, c\}$ and $S = abaa$. Then

$$G(1) = 2,\ G(2) = 1,\ G(3) = 3,\ G(4) = 3,$$
$$B(a) = 1,\ B(b) = 2,\ B(c) = 4.$$

Suppose $T = acabaacca\cdots$. The first three shifts of the BM algorithm (left) and the BMH algorithm (right) are shown separately as follows.

a	c	a	b	a	a	c	c	a	$\cdot$	$\cdot$	$\cdot$		a	c	a	b	a	a	c	c	a	$\cdot$	$\cdot$	$\cdot$
a	b	a	a										a	b	a	a								
		a	b	a	a										a	b	a	a						
					a	b	a	a								a	b	a	a					
								a	b	a	a									a	b	a	a	

In the first window, $i = 1$ and the mismatch character in text is b, so the distance shifted is 2 for both BM and BMH algorithms.

In the second window, all four characters match (an occurrence is found), so the distance shifted is $G(4) = 3$ for the BM algorithm and is $B(a) = 1$ for the BMH algorithm.

The next windows for BM and BMH algorithms are different. For BM algorithm, $i = 2$ and the mismatch character in text is c, so the distance shifted is $\max\{G(2), B(c) - 2 + 1\} = 3$. For BMH, the rightmost character of the window in text is c, so the distance shifted is $B(c) = 4$.

3. A MARKOVIAN APPROACH

Assume that the text is iid and the pattern S is fixed. Let W_j be the jth window in the processing of the algorithm. Obviously, $\{W_j\}$ is a Markov chain with the state space $\mathcal{A}^m$. Since the algorithm is designed to find all occurrences of the pattern, each state will lead to state S. Conversely, let Ω be the set of states that state S leads to. Then Ω contains all recurrent states and is irreducible. There might be a finite number of states outside Ω that are transient. However, the chain will stay inside Ω once the first occurrence of the pattern is found. The number of character comparisons taken before the first occurrence of the pattern is asymptotically negligible. Thus, we can assume irreducibility of the Markov chains in Section 4 without loss of generality.

Let $g(W_j)$ be the distance to shift at W_j and $f(W_j)$ be the number of comparisons taken inside the window W_j. The transition probability of $\{W_j\}$ is given by

$$p(\mathbf{a}, \mathbf{b}) \equiv \mathbb{P}(W_2 = \mathbf{b} | W_1 = \mathbf{a}) = \begin{cases} \mathbb{P}(\mathbf{t} = b_{g(\mathbf{a})} \cdots b_1) & \text{if } a_{m-g(\mathbf{a})} \cdots a_1 = b_m \cdots b_{g(\mathbf{a})+1}, \\ 0 & \text{otherwise,} \end{cases}$$

where $\mathbf{a} = a_m \cdots a_1, \mathbf{b} = b_m \cdots b_1 \in \mathcal{A}^m$, and $\mathbf{t}$ is a random text of length $g(\mathbf{a})$.

Let N_k be the total number of windows in the processing of the algorithm for the text $t_1 t_2 \cdots t_k$. If $k \geq m$ then

$$m + \sum_{j=1}^{N_k - 1} g(W_j) \leq k < m + \sum_{j=1}^{N_k} g(W_j). \tag{1}$$

We also define the pair of random variables

$$\overline{W}_k \equiv \big(W_{N_k}, Z_k\big),$$

where $Z_k = k - m - \sum_{j=1}^{N_k - 1} g(W_j)$. The window

$$W_{N_k} = t_a \cdots t_b, \text{ where } \quad a = 1 + \sum_{j=1}^{N_k - 1} g(W_j) \quad \text{and} \quad b = m + \sum_{j=1}^{N_k - 1} g(W_j)$$

and the value $g(W_{N_k})$ is the distance of the next possible shift. The random variable Z_k is the distance from the character t_b to the character t_k. Thus,

$$\overline{W}_{k+1} = \begin{cases} \big(W_{N_k}, Z_k + 1\big) & \text{if } g\big(W_{N_k}\big) - Z_k > 1, \\ \big(W_{N_k+1}, 0\big) & \text{if } g\big(W_{N_k}\big) - Z_k = 1. \end{cases}$$

Therefore, $\{\overline{W}_k\}_{k \geq m}$ is a Markov chain with the state space $\mathcal{A}^m \times \{0, 1, \ldots, m-1\}$ and the transition probability is given by

$$\overline{p}((\mathbf{a}, i), (\mathbf{b}, j)) = \begin{cases} 1 & \text{if } \mathbf{a} = \mathbf{b},\ 0 \leq i \leq g(\mathbf{a}) - 2 \text{ and } j = i + 1, \\ p(\mathbf{a}, \mathbf{b}) & \text{if } i = g(\mathbf{a}) - 1 \text{ and } j = 0, \\ 0 & \text{otherwise,} \end{cases}$$

where $\mathbf{a}, \mathbf{b} \in \mathcal{A}^m$ and $0 \leq i, j \leq m - 1$. Note that not every state is visited.

Reduce the Size of the State Space

We first propose a means of reducing the size of the state space by properly combining states. Let $\Gamma = \{D_1, \ldots, D_K\}$ be a partition of $\mathcal{A}^m$. To combine the states and also to preserve the values of g and f, we require that Γ satisfies the following two conditions.

C1 Every element in the same class gives the same value under f and the same value under g.

C2 New transition probability is well defined, that is, for all $D_\alpha, D_\beta \in \Gamma$,

$$p(\mathbf{a}_1, D_\beta) = p(\mathbf{a}_2, D_\beta) \qquad \text{for all } \mathbf{a}_1, \mathbf{a}_2 \in D_\alpha,$$

where $p(\mathbf{a}, D) = \sum_{\mathbf{b} \in D} p(\mathbf{a}, \mathbf{b})$.

Providing conditions C1 and C2, we can regroup the state space $\mathcal{A}^m$ with respect to the partition Γ. We write $h(D)$, where D is a class of strings and $h = f$ or g, when every element in D gives the same value under h. Define

$$\overline{\Gamma} = \{(D, i) : D \in \Gamma, 0 \leq i < g(D)\}. \tag{2}$$

Then $\overline{\Gamma}$ is a state space for $\{\overline{W}_k\}$. Note that some states in $\overline{\Gamma}$ might be transient.

In the paper, all classes in a partition of $\mathcal{A}^m$ are of the form

$$\langle \mathcal{B} a_{\ell-1} \cdots a_1 \rangle \equiv \{x_m \cdots x_{\ell+1} x_\ell a_{\ell-1} \cdots a_1 : x_\ell \in \mathcal{B} \text{ and } x_{\ell+1}, \ldots, x_m \in \mathcal{A}\}, \tag{3}$$

where $1 \le \ell \le m$, $\mathcal{B} \subset \mathcal{A}$, and $a_1, \ldots, a_{\ell-1} \in \mathcal{A}$. Write $\langle \mathcal{B} a_{\ell-1} \cdots a_1 \rangle = \langle a_\ell \cdots a_1 \rangle$ when $\mathcal{B} = \{a_\ell\}$. We call ℓ the *length* of $\langle \mathcal{B} a_{\ell-1} \cdots a_1 \rangle$.

Lemma 1. *Let* $D_\alpha = \langle \mathcal{B}_\alpha a_{\ell_\alpha - 1} \cdots a_1 \rangle$ *and* $D_\beta = \langle \mathcal{B}_\beta b_{\ell_\beta - 1} \cdots b_1 \rangle$ *with* $\ell_\alpha \ge \ell_\beta$.

(i) *Suppose* $\ell_\alpha > \ell_\beta$. *Then*

$$\begin{cases} D_\alpha \subset D_\beta & \text{if } a_{\ell_\beta - 1} \cdots a_1 = b_{\ell_\beta - 1} \cdots b_1 \text{ and } a_{\ell_\beta} \in \mathcal{B}_\beta, \\ D_\alpha \cap D_\beta = \varnothing & \text{otherwise.} \end{cases}$$

(ii) *Suppose* $\ell_\alpha = \ell_\beta$ *and* $|\mathcal{B}_\alpha| \le |\mathcal{B}_\beta|$. *Then*

$$\begin{cases} D_\alpha \subset D_\beta & \text{if } a_{\ell_\beta - 1} \cdots a_1 = b_{\ell_\beta - 1} \cdots b_1 \text{ and } \mathcal{B}_\alpha \subset \mathcal{B}_\beta, \\ D_\alpha \subsetneq D_\beta \text{ and } D_\alpha \cap D_\beta \ne \varnothing & \text{if } a_{\ell_\beta - 1} \cdots a_1 = b_{\ell_\beta - 1} \cdots b_1, \mathcal{B}_\alpha \subsetneq \mathcal{B}_\beta \\ & \quad \text{and } \mathcal{B}_\alpha \cap \mathcal{B}_\beta \ne \varnothing, \\ D_\alpha \cap D_\beta = \varnothing & \text{otherwise.} \end{cases} \tag{4}$$

In particular, if $|\mathcal{B}_\alpha| = 1$ *then the second case in (4) does not exist.*

The proof of Lemma 1 follows directly from the definition of (3).

Before introducing the construction of a partition satisfying conditions C1 and C2, we give a special example to illustrate the idea.

Example 1. Let $\mathcal{A} = \{a_1, a_2, a_3, a_4, a_5, a_6, a_7\}$, $q = 7$, $m = 5$, and $S = a_5 a_4 a_3 a_2 a_1$ (note that the values under g with respect to BM and BMH coincide in this setting). First, decompose $\mathcal{A}^5$ into the partition

$$\{\langle a_4 a_3 a_2 a_1 \rangle, \langle \mathcal{B}_4 a_3 a_2 a_1 \rangle, \langle \mathcal{B}_3 a_2 a_1 \rangle, \langle \mathcal{B}_2 a_1 \rangle, \langle \mathcal{B}_1 \rangle\},$$

where $\mathcal{B}_k = \{a : a \ne a_k\}$. Observe that

$$f(\langle a_4 a_3 a_2 a_1 \rangle) = 5,\ f(\langle \mathcal{B}_4 a_3 a_2 a_1 \rangle) = 4,\ f(\langle \mathcal{B}_3 a_2 a_1 \rangle) = 3,\ f(\langle \mathcal{B}_2 a_1 \rangle) = 2,\ f(\langle \mathcal{B}_1 \rangle) = 1,$$

and

$$g(\langle a_4 a_3 a_2 a_1 \rangle) = g(\langle \mathcal{B}_4 a_3 a_2 a_1 \rangle) = g(\langle \mathcal{B}_3 a_2 a_1 \rangle) = g(\langle \mathcal{B}_2 a_1 \rangle) = 5.$$

However, there are several possible values under g on $\langle \mathcal{B}_1 \rangle$. Thus, decompose $\langle \mathcal{B}_1 \rangle$ into

$$\langle a_2 \rangle, \langle a_3 \rangle, \langle a_4 \rangle, \langle a_5 \rangle \text{ and } \langle \{a_6, a_7\} \rangle,$$

with

$$g(\langle a_2\rangle) = 1,\ g(\langle a_3\rangle) = 2,\ g(\langle a_4\rangle) = 3,\ g(\langle a_5\rangle) = 4, \text{ and } g(\langle a_6, a_7\rangle) = 5.$$

Now, condition C1 is satisfied.

Further decomposition is needed for condition C2. For instance, $\langle a_2\rangle$ may move to $\langle a_2a_1\rangle$. But, $\langle a_2a_1\rangle$ overlaps with $\langle a_4a_3a_2a_1\rangle$, $\langle \mathcal{B}_4a_3a_2a_1\rangle$, and $\langle \mathcal{B}_3a_2a_1\rangle$. This make $p(\langle a_2\rangle, \cdot)$ undefined. To remedy such a case, decompose $\langle a_2\rangle$ into

$$\langle a_4a_3a_2\rangle, \langle \mathcal{B}_4a_3a_2\rangle, \text{ and } \langle \mathcal{B}_3a_2\rangle.$$

Similarly, decompose $\langle a_3\rangle$ into $\langle a_4a_3\rangle$ and $\langle \mathcal{B}_4a_3\rangle$. The final partition is given by

$$\left\{\begin{array}{c} \langle a_4a_3a_2a_1\rangle, \langle \mathcal{B}_4a_3a_2a_1\rangle, \langle \mathcal{B}_3a_2a_1\rangle, \langle \mathcal{B}_2a_1\rangle, \\ \langle a_4a_3a_2\rangle, \langle \mathcal{B}_4a_3a_2\rangle, \langle \mathcal{B}_3a_2\rangle, \\ \langle a_4a_3\rangle, \langle \mathcal{B}_4a_3\rangle, \\ \langle a_4\rangle, \langle a_5\rangle, \langle \{a_6, a_7\}\rangle \end{array}\right\}$$

and satisfies the conditions C1 and C2 for both BM and BMH algorithms.

Construct a Partition Satisfying Conditions C1 and C2

Recall that $S = s_m \cdots s_1$ and $[s_j, s_i], j \geq i$ denotes the string $s_j s_{j-1} \cdots s_i$. The string $a_\ell \cdots a_1$ is a substring of $[s_j, s_i]$ if $a_\ell \cdots a_1 = [s_{j_0}, s_{i_0}], j \geq j_0 \geq i_0 \geq i$.

Now, we construct a partition of $\mathcal{A}^m$ satisfying the conditions C1 and C2 for the BMH algorithm. First, define a family of classes of strings, Γ_h, which consists of the following three types of classes.

(a) If $\mathcal{R} = \{a : a \notin \{s_1, \ldots, s_m\}\} \neq \varnothing$ then $\langle \mathcal{R}\rangle \in \Gamma_h$. If $s_m \notin \{s_1, \ldots, s_{m-1}\}$ then $\langle s_m\rangle \in \Gamma_h$.
(b) If $[s_{m-1}, s_i]$ is not a substring of $[s_{m-2}, s_1]$ then

$$\langle s_{m-1} \cdots s_i\rangle \in \Gamma_h.$$

(c) Let

$$\mathcal{B}(j, i) = \{a : a = s_k,\ [s_{k-1}, s_{k-(j-i)-1}] = [s_j, s_i] \text{ and } k \leq m - 1\}, \tag{5}$$

for $m - 2 \geq j \geq i \geq 1$. If $\mathcal{B}(j, i) \neq \mathcal{A}$ then

$$\langle \mathcal{B}^c(j, i)s_j \cdots s_i\rangle \in \Gamma_h,$$

where $\mathcal{B}^c(j, i)$ is the complement of $\mathcal{B}(j, i)$.

From the definition of (5), $\langle \mathcal{B}^c(j, i)s_j \cdots s_i\rangle = \langle \mathcal{B}^c(\ell, k)s_\ell \cdots s_k\rangle$ when $[s_j, s_i] = [s_\ell, s_k]$. We choose one of them as a representative in Γ_h. The size of Γ_h is at most

$$2 + (m - 1) + (m - 1)(m - 2)/2 = m^2/2 - m/2 + 2.$$

Lemma 2. *The family Γ_h is a partition of $\mathcal{A}^m$.*

Proof. First, we prove the completeness. Given a string $x_m \cdots x_1$, we show that $x_m \cdots x_1 \in D$ for some $D \in \Gamma_h$. Choose the largest k such that $x_k \cdots x_1$ is a substring of $[s_{m-1}, s_1]$.

(i) If there is no such k then $x_1 = s_m$ or $x_1 \in \mathcal{R}$. Thus, $x_m \cdots x_1 \in \langle s_m \rangle$ or $\langle \mathcal{R} \rangle$.
(ii) If $x_k \cdots x_1 = [s_j, s_i]$ for some $i \le j \le m-2$ then $x_{k+1} \neq s_{j+1}$. Thus, $x_{k+1} \in \mathcal{B}^c(j,i)$ and so $x_m \cdots x_1 \in \langle \mathcal{B}^c(j,i) s_j \cdots s_i \rangle$.
(iii) Suppose $x_k \cdots x_1 \neq [s_j, s_i]$ for all $i \le j \le m-2$. Then $x_k \cdots x_1 = [s_{m-1}, s_i]$ for some $i \le m-1$ and $\langle s_{m-1} \cdots s_i \rangle \in \Gamma_h$. Thus, $x_m \cdots x_1 \in \langle s_{m-1} \cdots s_i \rangle$.

This proves the completeness of the partition.

We next prove the disjointness. Let $D_\alpha, D_\beta \in \Gamma_h$ with length ℓ_α and ℓ_β, respectively. Assume a contrario that $D_\alpha \neq D_\beta$ and $D_\alpha \cap D_\beta \neq \varnothing$. Since the classes $\langle \mathcal{R} \rangle$ and $\langle s_m \rangle$ are clearly disjoint with classes of other types, we only have to exam the following three kinds of intersections. We will apply repeatedly Lemma 1 in all the following cases.

(i) $D_\alpha = \langle s_{m-1} \cdots s_{i_1} \rangle$ and $D_\beta = \langle s_{m-1} \cdots s_{i_2} \rangle$, $i_1 < i_2$. Then $\ell_\alpha > \ell_\beta$. Thus, $D_\alpha \subset D_\beta$ and $[s_{m-1-(i_2-i_1)}, s_{i_1}] = [s_{m-1}, s_{i_2}]$, violating the condition in (b).
(ii) $D_\alpha = \langle s_{m-1} \cdots s_{i_1} \rangle$ and $D_\beta = \langle \mathcal{B}^c(j_2, i_2) s_{j_2} \cdots s_{i_2} \rangle$. If $\ell_\alpha < \ell_\beta$ then $D_\beta \subset D_\alpha$ and $[s_{m-1}, s_{i_1}] = [s_{j_2-(\ell_\beta-\ell_\alpha)+1}, s_{i_2}]$, again violating the condition in (b). If $\ell_\alpha \ge \ell_\beta$ then $D_\alpha \subset D_\beta$, $[s_{m-2-(\ell_\alpha-\ell_\beta)}, s_{i_1}] = [s_{j_2}, s_{i_2}]$ and $s_{m-1-(\ell_\alpha-\ell_\beta)} \in \mathcal{B}^c(j_2, i_2)$, a contradiction by the definition of $\mathcal{B}(j_2, i_2)$.
(iii) $D_\alpha = \langle \mathcal{B}^c(j_1, i_1) s_{j_1} \cdots s_{i_1} \rangle$ and $D_\beta = \langle \mathcal{B}^c(j_2, i_2) s_{j_2} \cdots s_{i_2} \rangle$. Assume $\ell_\alpha \ge \ell_\beta$. If $\ell_\alpha > \ell_\beta$ then $D_\alpha \subset D_\beta$, $[s_{j_1-(\ell_\alpha-\ell_\beta)}, s_{i_1}] = [s_{j_2}, s_{i_2}]$ and $s_{j_1-(\ell_\alpha-\ell_\beta)+1} \in \mathcal{B}^c(j_2, i_2)$, a contradiction by the definition of $\mathcal{B}(j_2, i_2)$. If $\ell_1 = \ell_2$ then $[s_{j_1}, s_{i_1}] = [s_{j_2}, s_{i_2}]$. Thus, $D_\alpha = D_\beta$. This is contradictory to the assumption. ∎

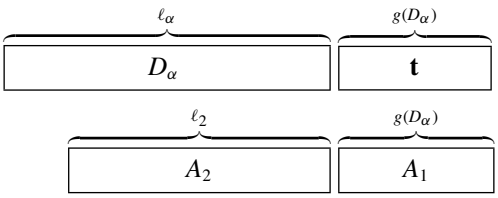
Lemma 3. *The partition Γ_h satisfies the conditions C1 and C2 with respect to the BMH algorithm.*

Proof. The rightmost character of a string determines the value of g. Every string in a class, except class $\langle \mathcal{R} \rangle$, has the same rightmost character. Clearly, $g(\langle \mathcal{R} \rangle) = m$. Thus, each class of Γ_h has only one value under g.

To show that each class of Γ_h has also only one value under f, let $D \in \Gamma_h$.

Type (a). It is trivial that $f(D) = 1$.
Type (b). If $D = \langle s_{m-1} \cdots s_1 \rangle$ then $f(D) = m$. If $D = \langle s_{m-1} \cdots s_i \rangle$ for $i > 1$ then $[s_{m-1}, s_i] \neq [s_{m-i}, s_1]$. Thus, there is at least one mismatch between $[s_{m-1}, s_i]$ and $[s_{m-i}, s_1]$, and $f(D)$ is equal to the number of comparisons till the first mismatch.
Type (c). Suppose $D = \langle \mathcal{B}^c(j,i) s_j \cdots s_i \rangle$. If $[s_j, s_i] = [s_{j-i+1}, s_1]$ then $s_{j-i+2} \in \mathcal{B}(j,i)$ and a mismatch occurs there. Thus, $f(D) = j - i + 2$. If $[s_j, s_i] \neq [s_{j-i+1}, s_1]$ then there is at least one mismatch between $[s_j, s_i]$ and $[s_{j-i+1}, s_1]$, and $f(D)$ is equal to the number of comparisons till the first mismatch. This completes the proof of condition C1.

Let $D_\alpha, D_\beta \in \Gamma_h$ with length ℓ_α and ℓ_β, respectively. We claim that the transition probability $p(D_\alpha, D_\beta)$ is well defined. The distance of shift is $g(D_\alpha)$. Obviously, $p(D_\alpha, D_\beta)$ is well defined if $\ell_\beta \le g(D_\alpha)$.



Scheme 1.

Suppose $\ell_\beta > g(D_\alpha)$. Let $D_\beta = \langle \mathcal{B}s_{j_2} \cdots s_{i_2} \rangle$. Define

$$A_1 = s_{i_2+g(D_\alpha)-1} \cdots s_{i_2} \text{ and } A_2 = \langle \mathcal{B}s_{j_2} \cdots s_{i_2+g(D_\alpha)} \rangle, \tag{6}$$

where $j_2 = i_2 + \ell_\beta - 2$. Note that A_2 may not be in Γ_h. The length of A_2 is $\ell_\beta - g(D_\alpha)$, denoted by ℓ_2. See the alignment in Scheme 1, where **t** is a random text of length $g(D_\alpha)$. To prove the well-definedness of $p(D_\alpha, D_\beta)$, we claim that D_α is either completely inside A_2 or disjoint from A_2. That is, we assume $D_\alpha \cap A_2 \neq \varnothing$ and then show $D_\alpha \subset A_2$. We will again apply repeatedly Lemma 1 in all the following cases.

If $\ell_2 < \ell_\alpha$, then $D_\alpha \subset A_2$.

If $\ell_2 > \ell_\alpha$, then $A_2 \subset D_\alpha$. If $D_\alpha = \langle s_{m-1} \cdots s_{i_1} \rangle$ then $[s_{m-1}, s_{i_1}] = [s_{j_2-(\ell_2-\ell_\alpha)+1}, s_{i_2+g(D_\alpha)}]$, contradictory to the condition in (b). On the other hand, if $D_\alpha = \langle \mathcal{B}^c(j_1, i_1)s_{j_1} \cdots s_{i_1} \rangle$ then $[s_{j_1}, s_{i_1}] = [s_{j_2-(\ell_2-\ell_\alpha)}, s_{i_2+g(D_\alpha)}]$ and $s_{j_2-(\ell_2-\ell_\alpha)+1} \in \mathcal{B}^c(j_1, i_1)$, contradictory to the definition of $\mathcal{B}(j_1, i_1)$.

If $\ell_2 = \ell_\alpha$, we examine the following three cases.

(i) If $D_\alpha = \langle s_{m-1} \cdots s_{i_1} \rangle$ then $D_\alpha \subset A_2$.

(ii) If $D_\alpha = \langle \mathcal{B}^c(j_1, i_1)s_{j_1} \cdots s_{i_1} \rangle$ and $A_2 = \langle s_{m-1} \cdots s_{i_2+g(D_\alpha)} \rangle$ then $A_2 \subset D_\alpha$, $[s_{j_1}, s_{i_1}] = [s_{m-2}, s_{i_2+g(D_\alpha)}]$ and $s_{m-1} \in \mathcal{B}^c(j_1, i_1)$, contradictory to the definition of $\mathcal{B}(j_1, i_1)$.

(iii) If $D_\alpha = \langle \mathcal{B}^c(j_1, i_1)s_{j_1} \cdots s_{i_1} \rangle$ and $A_2 = \langle \mathcal{B}^c(j_2, i_2)s_{j_2} \cdots s_{i_2+g(D_\alpha)} \rangle$ then $[s_{j_1}, s_{i_1}] = [s_{j_2}, s_{i_2+g(D_\alpha)}]$. By the definition of (5), $\mathcal{B}(j_2, i_2) \subset \mathcal{B}(j_2, i_2 + g(D_\alpha)) = \mathcal{B}(j_1, i_1)$. Thus, $\mathcal{B}^c(j_1, i_1) \subset \mathcal{B}^c(j_2, i_2)$ and so $D_\alpha \subset A_2$.

This completes the proof of condition C2. ∎

The partition Γ_h also satisfies the conditions C1 and C2 with respect to a modified BM algorithm that combines the good-suffix rule and the bad-character rule of the BMH algorithm. However, finer partition is needed for the BM algorithm, since there are possibly more than one value of g for some classes in Γ_h (whenever the bad-character shift is larger than the good-suffix shift). We have then to decompose those classes according to the values of g and trace back those leading to the splitting classes and further decompose them. Such a "traced-back" decomposition is continued until the length of the splitting classes is equal to 2. The detail of a formulation of the partition and the proof of condition C2 are more complex. We only give examples here.

Example 2. Assume $\mathcal{A} = \{a_1, a_2, a_3, a_4, a_5\}$, $q = 5$, $m = 6$, and $S = a_4a_3a_2a_1a_2a_1$. Then

$$\Gamma_h = \left\{ \begin{array}{c} \langle a_5\rangle, \langle a_4\rangle, \\ \langle a_3a_2a_1a_2a_1\rangle, \langle a_3a_2a_1a_2\rangle, \langle a_3a_2a_1\rangle, \langle a_3a_2\rangle, \langle a_3\rangle, \\ \langle \mathcal{B}_3a_2a_1a_2a_1\rangle, \langle \mathcal{B}_2a_1a_2a_1\rangle, \langle \mathcal{B}_3a_2a_1a_2\rangle, \langle \mathcal{B}_2a_1a_2\rangle, \langle \mathcal{B}_2a_1\rangle, \\ \langle \mathcal{B}_{1,3}a_2a_1\rangle, \langle \mathcal{B}_{1,3}a_2\rangle \end{array} \right\},$$

where $\mathcal{B}_{k_1,k_2,\ldots} = \{a : a \neq a_{k_1}, a_{k_2}, \ldots\}$. Here g is the shift function for BM algorithm. Observe that

$$g(\langle a_5\rangle) = g(\langle a_3a_2a_1a_2a_1\rangle) = g(\langle \mathcal{B}_3a_2a_1a_2a_1\rangle) = g(\langle \mathcal{B}_2a_1a_2a_1\rangle) = 6,$$
$$g(\langle a_4\rangle) = 5,\ g(\langle a_3\rangle) = 4,\ g(\langle a_3a_2a_1\rangle) = 2,$$

and

$$g(\langle a_3a_2a_1a_2\rangle) = g(\langle a_3a_2\rangle) = g(\langle \mathcal{B}_3a_2a_1a_2\rangle) = g(\langle \mathcal{B}_2a_1a_2\rangle) = g(\langle \mathcal{B}_{1,3}a_2\rangle) = 1.$$

However, the class $\langle \mathcal{B}_{1,3}a_2a_1\rangle$ gives three values under g,

$$g(\langle a_2a_2a_1\rangle) = 2,\ g(\langle a_4a_2a_1\rangle) = 3,\ g(\langle a_5a_2a_1\rangle) = 4$$

and the class $\langle \mathcal{B}_2a_1\rangle$ gives four values under g,

$$g(\langle a_1a_1\rangle) = 2,\ g(\langle a_3a_1\rangle) = 3,\ g(\langle a_4a_1\rangle) = 4,\ g(\langle a_5a_1\rangle) = 5.$$

So we decompose the class $\langle \mathcal{B}_{1,3}a_2a_1\rangle$ into

$$\langle a_2a_2a_1\rangle, \langle a_4a_2a_1\rangle, \langle a_5a_2a_1\rangle$$

and the class $\langle \mathcal{B}_2a_1\rangle$ into

$$\langle a_1a_1\rangle, \langle a_3a_1\rangle, \langle a_4a_1\rangle, \langle a_5a_1\rangle.$$

As a side effect, the decomposition of $\langle \mathcal{B}_{1,3}a_2a_1\rangle$ causes $p(\langle \mathcal{B}_{1,3}a_2\rangle, \cdot)$ not well defined. We then decompose $\langle \mathcal{B}_{1,3}a_2\rangle$ into

$$\langle a_2a_2\rangle, \langle a_4a_2\rangle, \langle a_5a_2\rangle.$$

The final resulting partition is then

$$\Gamma_{BM} = \left\{ \begin{array}{c} \langle a_5\rangle, \langle a_4\rangle, \\ \langle a_3a_2a_1a_2a_1\rangle, \langle a_3a_2a_1a_2\rangle, \langle a_3a_2a_1\rangle, \langle a_3a_2\rangle, \langle a_3\rangle, \\ \langle \mathcal{B}_3a_2a_1a_2a_1\rangle, \langle \mathcal{B}_2a_1a_2a_1\rangle, \langle \mathcal{B}_3a_2a_1a_2\rangle, \langle \mathcal{B}_2a_1a_2\rangle, \\ \langle a_1a_1\rangle, \langle a_3a_1\rangle, \langle a_4a_1\rangle, \langle a_5a_1\rangle, \\ \langle a_2a_2a_1\rangle, \langle a_4a_2a_1\rangle, \langle a_5a_2a_1\rangle, \\ \langle a_2a_2\rangle, \langle a_4a_2\rangle, \langle a_5a_2\rangle \end{array} \right\}$$

and satisfies the conditions C1 and C2 for the BM algorithm.

Remark. If we decompose all classes of type (c) into

$$\langle as_j \cdots s_i\rangle \qquad \text{for all } a \in \mathcal{B}^c(j, i),$$

then this finer partition satisfies the conditions C1 and C2 for both BM and BMH algorithms. However, the size of the partition is larger than $|\Gamma_b|$ but is still bounded above by

$$2 + (m-1) + (q-1)(m-1)(m-2)/2 = (q-1)m^2/2 - (3q-5)m/2 + q.$$

The New Transition Probability

Let Γ be a state space constructed as above for $\{W_j\}$ and $\overline{\Gamma}$ is the state space for $\{\overline{W}_k\}$ defined in (2). Let θ denote the distribution of W_1. With the same notations as in (6), the new transition probability for $\{W_j\}$ is given by

$$p(D_\alpha, D_\beta) = \begin{cases} \theta(D_\beta) & \text{if } g(D_\alpha) \geq \ell_\beta, \\ \theta(A_1) & \text{if } g(D_\alpha) < \ell_\beta \text{ and } D_\alpha \subset A_2, \\ 0 & \text{if } g(D_\alpha) < \ell_\beta \text{ and } D_\alpha \cap A_2 = \varnothing, \end{cases} \tag{7}$$

and the new transition probability for $\{\overline{W}_k\}$ is given by

$$\overline{p}((D_\alpha, i), (D_\beta, j)) = \begin{cases} 1 & \text{if } D_\alpha = D_\beta,\ 0 \leq i \leq g(D_\alpha) - 2 \text{ and } j = i + 1, \\ p(D_\alpha, D_\beta) & \text{if } i = g(D_\alpha) - 1 \text{ and } j = 0, \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

where $D_\alpha, D_\beta \in \Gamma$, $0 \leq i < g(D_\alpha)$, and $0 \leq j < g(D_\beta)$.

4. ASYMPTOTIC BEHAVIOR OF THE NUMBER OF COMPARISONS

Recall that C_n is the number of character comparisons taken by the algorithms. Observe that $C_n = \sum_{j=1}^{N_n} f(W_j)$. By applying the strong law of large number for the Markov chain $\{W_j\}$ (see Chung [7, p. 92]), there exist two constants μ_f and μ_g, specified in (14), such that

$$\frac{1}{N}\sum_{j=1}^{N} f(W_j) \to \mu_f \quad a.s. \qquad \text{and} \qquad \frac{1}{N}\sum_{j=1}^{N} g(W_j) \to \mu_g \quad a.s. \tag{9}$$

as $N \to \infty$, where *a.s.* denotes "almost surely." Clearly, $N_n \to \infty$ *a.s.* as $n \to \infty$. From (9) it follows that

$$\frac{C_n}{n} = \frac{\sum_{j=1}^{N_n} f(W_j)}{\sum_{j=1}^{N_n - 1} g(W_j) + \left(n - \sum_{j=1}^{N_n - 1} g(W_j)\right)} \to \frac{\mu_f}{\mu_g} \equiv \mu_{(S)} \quad a.s.$$

as $n \to \infty$. Thus, the limiting mean of C_n/n is $\mu_{(S)}$ by the bounded convergence theorem.

Define the weight function

$$h(\overline{W}_k) = f(W_{N_k}) 1_{\{Z_k = 0\}}.$$

Then $C_n = \sum_{k=1}^{n} h(\overline{W}_k)$. We apply the limit theorems in Chung [7, pp. 102 and 106] to the Markov chains $\{\overline{W}_k\}$ and obtain the following theorems.

Theorem 1. *The mean and the variance of C_n satisfy*

$$\mathbb{E}[C_n] = \mu_{(S)} n + o(\sqrt{n}), \tag{10}$$

$$\mathbb{V}[C_n] = B_{(S)} n + o(n), \tag{11}$$

where $B_{(S)} \geq 0$ is a constant specified in (15) below.

Theorem 2. *If* $B_{(S)} > 0$ *then*

$$\limsup_{n\to\infty} \frac{C_n - \mu_{(S)} n}{\sqrt{2B_{(S)} n \ln\ln n}} = 1 \quad a.s. \tag{12}$$

and the corresponding lim inf *is equal to* -1 *almost surely.*

Note that the constants $\mu_{(S)}$ and $B_{(S)}$ will have different values for different algorithms.

The results from the application of the limit theorems in Chung include a central limit theorem. Here, we can obtain a stronger result by applying the Berry–Esseen theorem in Bolthausen [4] to the Markov chains $\{\overline{W}_k\}$. The conditions of the theorem in Bolthausen, the third moment of a return time and the first moment of an entrance time, hold clearly if we choose $([s_{m-1}, s_1], 0)$ as a fixed state to define the entrance time and the return time.

Theorem 3. *If* $B_{(S)} > 0$ *then*

$$\sup_{-\infty<x<\infty} \left| \mathbb{P}\left(\frac{C_n - \mu_{(S)} n}{\sqrt{B_{(S)} n}} < x \right) - \Phi(x) \right| = O(n^{-1/2}), \tag{13}$$

where $\Phi(x)$ *is the standard normal distribution.*

Explicit Determination of the Quantities $\mu_{(S)}$ and $B_{(S)}$

Recall that Ω is the class of recurrent states. Let $\mathbf{P}$ be the transition matrix for $\{W_j\}$ and π be the invariant probability measure with respect to $\mathbf{P}$ (i.e., $\pi\mathbf{P} = \pi$). Then

$$\mu_f = \sum_{D\in\Omega} f(D)\pi(D) \qquad \text{and} \qquad \mu_g = \sum_{D\in\Omega} g(D)\pi(D). \tag{14}$$

Let $\overline{\Omega}$ be the class of recurrent states and $\overline{\mathbf{P}}$ be the transition matrix for $\{\overline{W}_k\}$. Let $\overline{\pi}$ be the invariant probability measure with respect to $\overline{\mathbf{P}}$. It follows from (8) that $\overline{\pi}(D, 0) = \overline{\pi}(D, 1) = \cdots = \overline{\pi}(D, g(D) - 1)$ and

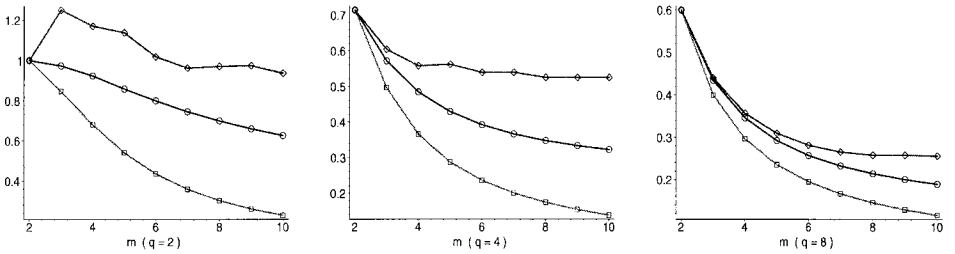
$$\overline{\pi}(D_\beta, 0) = \sum_{D_\alpha\in\Omega} \overline{\pi}(D_\alpha, g(D_\alpha) - 1)\overline{p}((D_\alpha, g(D_\alpha) - 1), (D_\beta, 0)) = \sum_{D_\alpha\in\Omega} \overline{\pi}(D_\alpha, 0)p(D_\alpha, D_\beta).$$

The above equation is $\overline{\pi}(\cdot, 0)\mathbf{P} = \overline{\pi}(\cdot, 0)$. Thus, $\overline{\pi}(\cdot, 0) = \lambda\pi(\cdot)$ for some constant λ. We have $\lambda = 1/\mu_g$ since

$$1 = \sum_{D\in\Omega, 0\le i<g(D)} \overline{\pi}(D, i) = \lambda \sum_{D\in\Omega} g(D)\pi(D) = \lambda\mu_g.$$

The constant $B_{(S)}$ can be obtained by applying the formulation of the second moment in Chung to the centered function $h_c(\cdot) = h(\cdot) - \mu_{(S)}$ (see Chung [7, pp. 88 and 95–99]),

$$B_{(S)} = \sum_{\overline{D}_\alpha\in\overline{\Omega}} h_c^2(\overline{D}_\alpha)\overline{\pi}(\overline{D}_\alpha) + 2\sum_{\overline{D}_\alpha\in\overline{\Omega}} h_c(\overline{D}_\alpha)\overline{\pi}(\overline{D}_\alpha) \sum_{\overline{D}_\beta\neq\overline{D}_0} h_c(\overline{D}_\beta)\overline{\pi}(\overline{D}_\beta)(m_{\alpha,0} + m_{0,\beta} - m_{\alpha,\beta}), \tag{15}$$



m	$q=2$			$q=4$			$q=8$		
2	1.00000	1.00000	1.00000	0.71428	0.71428	0.71428	0.60000	0.60000	0.60000
3	1.25000	0.97269	0.84615	0.60416	0.57150	0.49640	0.43973	0.43422	0.39984
4	1.16964	0.92250	0.67883	0.55781	0.48451	0.36608	0.35674	0.34575	0.29645
5	1.13750	0.85767	0.53997	0.56171	0.42900	0.28706	0.30939	0.29234	0.23533
6	1.01800	0.79928	0.43479	0.53898	0.39202	0.23580	0.28129	0.25711	0.19512
7	0.96205	0.74457	0.35759	0.53923	0.36632	0.20016	0.26522	0.23237	0.16666
8	0.96986	0.69863	0.30113	0.52512	0.34765	0.17396	0.25785	0.21418	0.14545
9	0.97377	0.65919	0.25930	0.52513	0.33353	0.15386	0.25785	0.20037	0.12903
10	0.93591	0.62553	0.22765	0.52513	0.32246	0.13793	0.25618	0.18960	0.11594

Fig. 1. The maximum, average, and minimum values of $\mu_{(S)}$ for the BM algorithm.

where $\overline{D}_0$ is a fixed state (the value of $B_{(S)}$ is independent of the choice of $\overline{D}_0$) and $m_{\alpha,\beta}$ is the expected time of the first entrance of state $\overline{D}_\beta$ starting from state $\overline{D}_\alpha$. The matrix $\mathbf{M} \equiv \{m_{\alpha,\beta}\}$ can be obtained from $\overline{\mathbf{P}}$ by

$$\mathbf{M} = (\mathbf{I} - \mathbf{Z} + \mathbf{E}\mathbf{Z}_{dg})\mathbf{D},$$

where $\mathbf{I}$ is the identity matrix, $\mathbf{Z}$ is the fundamental matrix for $\overline{\mathbf{P}}$, $\mathbf{E}$ is a matrix with all entries 1, $\mathbf{Z}_{dg}$ results from $\mathbf{Z}$ by setting all off-diagonal entries equal to 0, and $\mathbf{D}$ is diagonal matrix with ith entry $1/\overline{\pi}(i)$ (see Kemeny and Snell [10, p. 79]).

5. EVALUATION OF THE CONSTANTS OF THE LIMITING MEAN AND VARIANCE

In this section, we assume that each t_i is uniformly distributed in $\mathcal{A} = \{1, \ldots, q\}$. The "average" is taken for all patterns with the same length.

First, we compute the constant $\mu_{(S)}$ for the BM algorithm. In Fig. 1, we have the maximum, average, and minimum values of $\mu_{(S)}$ for $m = 2$ to 10 and $q = 2, 4, 8$.

Patterns reaching the minimum constant are of the form $S = 11\cdots1$; patterns reaching the maximum constant are shown in Table 1.

The range of $\mu_{(S)}$ is narrowing down as q increases. In addition, we compute in Fig. 2 the standard deviation of $\mu_{(S)}$, by considering S a random pattern with length m,

$$\sigma = \sqrt{\frac{1}{q^m}\sum_S (\mu_{(S)} - \mu)^2}.$$

TABLE 1. The Patterns that Reach the Maximum Constant for the BM Algorithm

m	$q = 2$	$q = 4$	$q = 8$
2	11	11	11
3	121	121	121
4	2121	1321	1321
5	21221	14321	14321
6	221221	114321	154321
7	1221121	2114321	1654321
8	21221121	12114321	17654321
9	121122121	212114321	187654321
10	1211221211	3212114321	1187654321

Here q and m are fixed, the sum is taken over all patterns with length m, and μ is the average of $\mu_{(S)}$.

We compare our theoretical result with a recent experimental result [12] (see Fig. 3). In their experiments, the text consists of 150,000 characters built randomly and 50 patterns randomly generated for each length are searched.

The experimental result is very close to our theoretical result when $m \leq 10$ and $q = 8$. The consistency can be explained by the fact that the values of σ and $B_{(S)}$ are relatively small (see the graphs in Figs. 2 and 4).

An Estimate for Long Patterns

Note that our approach becomes less efficient when computing the exact constant for longer patterns since the size of state space is close to $m^2/2$. Nevertheless, a good estimate of the constant for most long patterns can be obtained by the procedure below.

Let

$$m_0 = \min\{k : G(k) = G(k+1) = \cdots = G(m)\}, \tag{16}$$

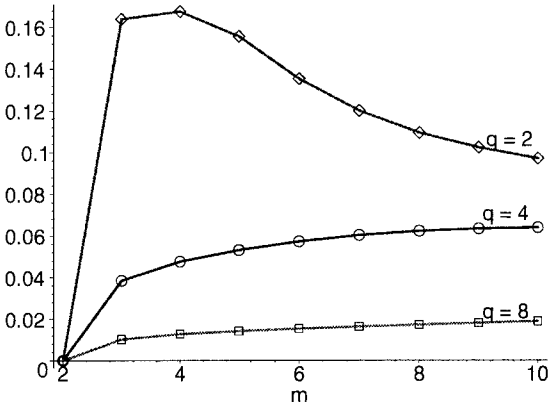


Fig. 2. The standard deviation of the linearity constant for the BM algorithm.

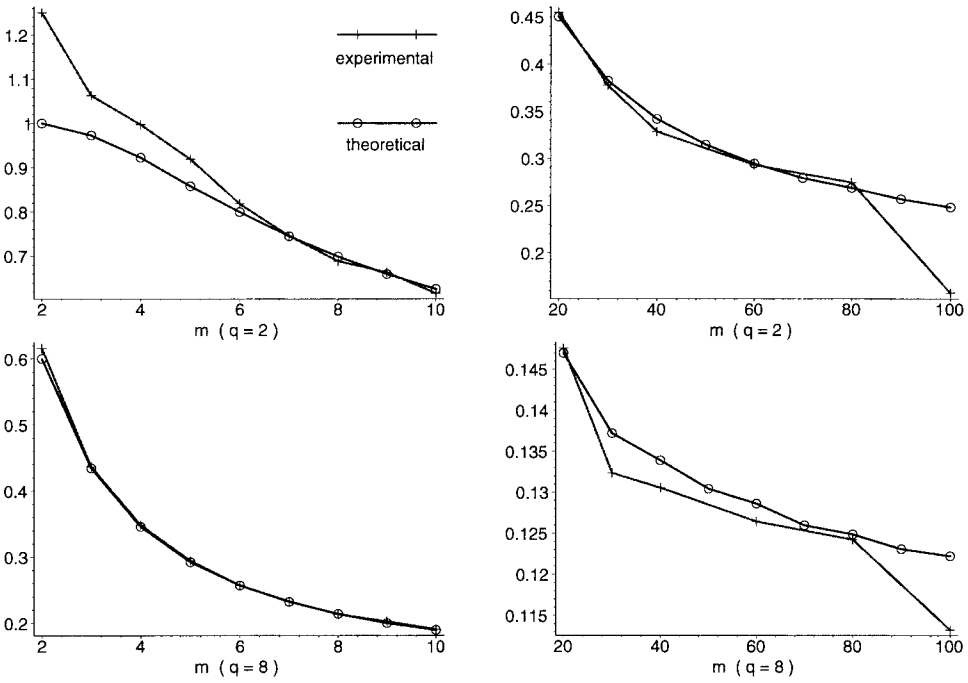


Fig. 3. The experimental C_n/n vs. the average of the theoretical constant $\mu_{(S)}$ for the BM algorithm in the cases $q = 2$ and 8.

where G is the function of good-suffix shift. Then m_0 is usually small compared to m for long patterns. For instance, if $q = 2$, $m = 20$, and

$$S = 1\,2\,1\,2\,2\,1\,2\,2\,2\,2\,1\,2\,1\,2\,2\,2\,2\,1\,2\,1,$$

then $G(1) = 1$, $G(2) = 19$, $G(3) = 2$, $G(4) = G(5) = \cdots = G(20) = 17$, and so $m_0 = 4$.

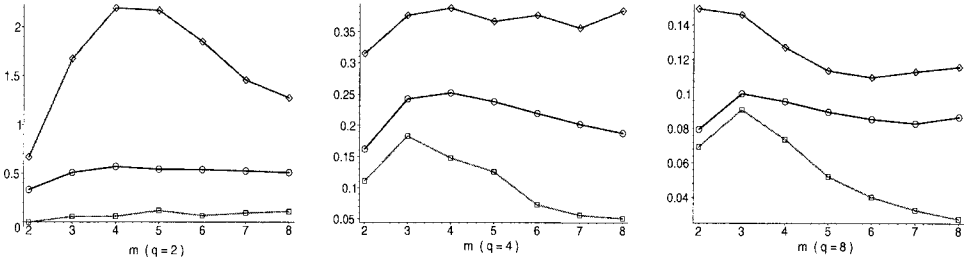
For the BM algorithm, we take $m_0 - 1 \le K \le m - 1$; for the BMH algorithm, we take $1 \le K \le m - 1$. A partition Γ_0 can be defined similarly to Γ_h as follows.

(a) If $\mathcal{R} = \{a : a \notin \{s_1, \ldots, s_m\}\} \neq \varnothing$ then $\langle \mathcal{R} \rangle \in \Gamma_0$. If $s_k \notin \{s_1, \ldots, s_{k-1}\}$ for $k > K$ then $\langle s_k \rangle \in \Gamma_0$.
(b) If $[s_K, s_i]$ is not a substring of $[s_{K-1}, s_1]$ then $\langle s_K \cdots s_i \rangle \in \Gamma_0$.
(c) If $\mathcal{B}(j, i) \neq \mathcal{A}$ then $\langle \mathcal{B}^c(j, i) s_j \cdots s_i \rangle \in \Gamma_0$, where

$$\mathcal{B}(j, i) = \{a : a = s_k,\ [s_{k-1}, s_{k-1-(j-i)}] = [s_j, s_i] \text{ and } k \le K\}$$

and $K - 1 \ge j \ge i \ge 1$.

The partition Γ_0, which is of size bounded above by $(K^2 + K)/2 + q$, is good enough for the BMH algorithm (but finer partition is needed for the BM algorithm as mentioned in Section 3). The condition in (16) guarantees that there is only one value under the function g of the BM algorithm on $\langle s_K \cdots s_1 \rangle \equiv D_1$. Indeed, the partition satisfies conditions C1 and C2, except for f on D_1. Nevertheless, we can compute the invariant probability



m	$q=2$			$q=4$			$q=8$		
2	0.66666	0.33333	0.00000	0.31486	0.16180	0.11078	0.14933	0.07933	0.06933
3	1.67501	0.50879	0.05664	0.37581	0.24226	0.18282	0.14585	0.10016	0.09078
4	2.19151	0.56904	0.05887	0.38728	0.25189	0.14766	0.12702	0.09562	0.07358
5	2.16630	0.54183	0.11848	0.36609	0.23777	0.12555	0.11337	0.08947	0.05199
6	1.84905	0.53762	0.06577	0.37610	0.21902	0.07276	0.10955	0.08529	0.03996
7	1.45500	0.52482	0.09254	0.35569	0.20176	0.05588	0.11285	0.08282	0.03243
8	1.27292	0.50784	0.10871	0.38313	0.18743	0.05080	0.11564	0.08658	0.02726

Fig. 4. The maximum, average, and minimum values of $B_{(S)}$ for the BM algorithm.

π as long as the transition probability is well defined. An estimate of the constant is given by

$$\left|\mu_{(S)} - \frac{(K+1)\pi(D_1) + \sum_{D\neq D_1} f(D)\pi(D)}{\sum_{D\in\Gamma_0} g(D)\pi(D)}\right| \leq \frac{(m-K-1)\pi(D_1)}{\mu_g} = \frac{(m-K-1)}{q^K}$$

since $K+1 \leq f \leq m$ on D_1 and $\pi(D_1) = \mu_g/q^K$. We can choose a suitable K to control the order of the errors.

We compute the constant for each pattern individually. So, the number of patterns cannot be too large. In the right-hand part of Fig. 3, we use random samples of 2000 patterns for the estimation of the average of $\mu_{(S)}$.

Finally, we compute the value of $B_{(S)}$ for the BM and the BMH algorithm for $m = 2$ to 8 in the cases $q = 2, 4, 8$. The results are presented in Figs. 4 and 5.

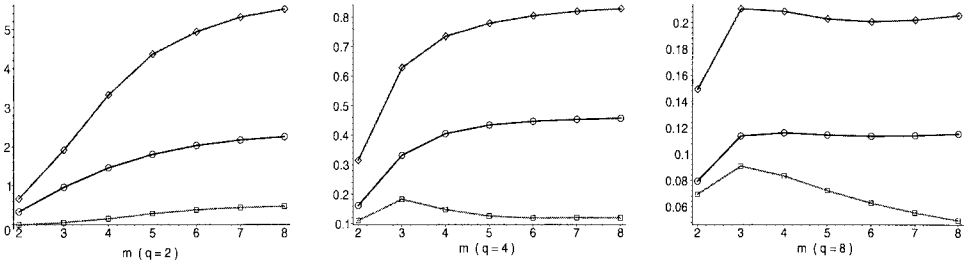
Note that $B_{(S)} = 0$ in the special case that $q = 2$ and $S = 1\ 2$.

6. CONCLUSION

In the paper we investigated the original BM and the BMH algorithms. Our approach may be readily modified for other variants of the BM algorithm. The extension from iid to Markov also seems straightforward. However, the transition matrix remains complex and thus a simplification of $\mu_{(S)}$ and $B_{(S)}$ seems very difficult.

We conclude by mentioning some interesting extensions for the BM algorithm.

- **Random pattern.** Evaluate the constant μ and B under the assumption that the pattern is random with a fixed length. Characterize the asymptotic behavior as the pattern length $m \to \infty$ (but comparatively small with n).



m	$q=2$			$q=4$			$q=8$		
2	0.66666	0.33333	0.00000	0.31486	0.16180	0.11078	0.14933	0.07933	0.06933
3	1.91601	0.96191	0.05664	0.62871	0.33162	0.18282	0.21041	0.11377	0.09078
4	3.32617	1.45803	0.15429	0.73453	0.40518	0.14766	0.20841	0.11620	0.08338
5	4.36665	1.80198	0.28515	0.77859	0.43410	0.12555	0.20264	0.11446	0.07221
6	4.93547	2.02716	0.37890	0.80364	0.44675	0.11890	0.20065	0.11367	0.06277
7	5.31093	2.16853	0.44018	0.81898	0.45336	0.12064	0.20190	0.11409	0.05533
8	5.51516	2.25432	0.47454	0.82757	0.45728	0.11975	0.20535	0.11534	0.04943

Fig. 5. The maximum, average, and minimum values of $B_{(S)}$ for the BMH algorithm.

- **Maximum and minimum.** Investigate the asymptotic behavior of the maximum and minimum values of $\mu_{(S)}$ and the combinatorial properties of patterns reaching these values.
- **Large deviation.** Derive estimates for the probabilities of large deviations via the constructed Markov chain.

ACKNOWLEDGMENTS

We thank the referees for helpful suggestions and comments.

REFERENCES

[1] R. A. Baeza-Yates, G. H. Gonnet, and M. Régnier, Analysis of Boyer–Moore type string searching algorithms, Proc 1st ACM-SIAM Symposium on Discrete Algorithms, San Francisco 1990, pp. 328–343.

[2] R. A. Baeza-Yates and M. Régnier, Average running time of the Boyer–Moore–Horspool algorithm, Theoret Comput Sci 92 (1992), 19–31.

[3] G. Barth, An analytical comparison of two string matching algorithms, Inform Process Lett, 18 (1984), 249–256.

[4] E. Bolthausen, The Berry–Esseen theorem for functionals of discrete Markov chains, Z. Wahrscheinlichkeitstheorie Verw. Gebiete 54 (1980), 59–73.

[5] R. S. Boyer and J. S. Moore, A first string searching algorithm, Commun ACM 20 (1977), 762–772.

[6] C. Charras and T. Lecroq, Handbook of Exact String-Matching Algorithms, King's College London Publications, 2004.

[7] K. L. Chung, Markov Chains with Stationary Transition Probabilities, second edn., Springer-Verlag, New York, 1967.

[8] R. Cole, Tight bounds on the complexity of the Boyer–Moore string matching algorithm, SIAM J Comput 23 (1994), 1075–1091.

[9] P. Flajolet, W. Szpankowski, and B. Vallée, Hidden word statistics, Preprint. Available on http://algo.inria.fr/flajolet/Publications/FlSzVa02.pdf.

[10] J. G. Kemeny, and J. L. Snell, Finite Markov Chains, Van Nostrand, 1960; Springer-Verlag, 1976.

[11] H. M. Mahmoud, R. T. Smythe and M. Régnier, Analysis of Boyer–Moore–Horspool string-matching heuristic, Random Struct Algor 10 (1997), 169–186.

[12] P. D. Michailidis and K. G. Margaritis, On-line string matching algorithms: Survey and experimental results, Int J Comp Math 76 (2001), 411–434.

[13] M. Régnier, Knuth–Morris–Pratt algorithm: An analysis, Mathematical Foundations of Computer Science 1989, Lecture Notes in Computer Science, Vol. 379, Springer-Verlag, Berlin, 1989, pp. 431–444.

[14] R.T. Smythe, The Boyer–Moore–Horspool heuristic with Markovian input, Random Struct Algor 18 (2001), 153–163.