AUSTRALASIAN JOURNAL OF COMBINATORICS
Volume **42** (2008), Pages 45–54

The expected number of runs in a word

Simon J. Puglisi

School of Computer Science and Information Technology
RMIT University
GPO Box 2476, Melbourne, VIC 3001
Australia
sjp@cs.rmit.edu.au

Jamie Simpson

Department of Mathematics and Statistics
Curtin University of Technology
GPO Box U1987, Perth WA 6845
Australia
simpson@maths.curtin.edu.au

Abstract

A word is a sequence of symbols taken from a (usually finite) alphabet. A run of period p in a word $\mathbf{x}$ is a factor $\mathbf{x}[m..n]$ such that $n-m \geq p$ and $\mathbf{x}[i] = \mathbf{x}[i+p]$ for all i satisfying $m \leq i < i+p \leq n$, and such that this does not hold if m is replaced by a smaller integer or n by a larger one. The number of runs in words has been a subject of interest in recent years, particularly because of connections with data compression. In this paper we investigate the expected number of runs per unit length in words of given alphabet size, and compare our results with DNA, amino acid and other sequences.

1 Introduction

A *word* is a finite or infinite sequence of symbols taken from a (usually finite) *alphabet*. Examples are sections of text, sequences of DNA codons and computer files. The study of their combinatorial properties is usually considered to have begun with the work of Thue in 1906 [15] but has accelerated in the last decade with the publication of three books ([7], [8], [9]) on the subject by M. Lothaire (who, incidentally, is not a human being, but a gang of French people). Applications exist in many areas of mathematics and computer science although computer scientists usually refer to "strings" rather than "words". The usual notation for a word of n elements is

$\mathbf{x} = \mathbf{x}[1..n]$, with $\mathbf{x}[i]$ being the ith element and $\mathbf{x}[i..j]$ the block of elements from position i to position j. The *length* of the word is its number of elements. Thus a word of length 8 on the alphabet $\{a, b, c\}$ is

$$\mathbf{x} = aabaabac$$

and $\mathbf{x}[5..7] = aba$. Some of the notation used in the area is borrowed from semigroup theory so that $\mathbf{x}[i..j]$ is a *factor* of $\mathbf{x}$, and a set of 2 or more identical adjacent factors forms a *power*. Thus the word $\mathbf{x}$ above contains the square aa which we can write as a^2, and the squares $(aab)^2$ and $(aba)^2$. A word which is not a power is *primitive*. A word $\mathbf{x}$ or factor $\mathbf{x}$ is *periodic* with period p if $\mathbf{x}[i] = \mathbf{x}[i + p]$ for all i such that $\mathbf{x}[i]$ and $\mathbf{x}[i + p]$ are in the word. In the example above $\mathbf{x}[1..2]$ and $\mathbf{x}[4..5]$ have period 1 and $\mathbf{x}[1..7]$ has period 3. Note that a power is necessarily periodic, but a periodic word is a power only if its period is a proper divisor of its length.

A *run* of period p in a word is a factor $\mathbf{x}[i..j]$ with least period p, length at least $2p$ and such that neither $\mathbf{x}[i - 1..j]$ nor $\mathbf{x}[i..j + 1]$ has period p. Runs are also called *maximal repetitions*. In the example above $\mathbf{x}[1..2]$ and $\mathbf{x}[4..5]$ are runs of period 1 and $\mathbf{x}[1..7]$ is a run of period 3. Runs have importance in data compression (see, for example, [1] and [14]) since a run $\mathbf{x}[i..j]$ with period p can be fully described by reporting only i, j, p and $\mathbf{x}[i..i + p - 1]$. In recent years there has been great interest in the maximum number of runs that can occur in a word of length n which we call $\rho(n)$. This takes larger values than one might expect, for example $\rho(32) = 26$. Here is an example of a length-32 word attaining this bound.

$$aababbabaababbababbabaababbababb$$

The runs in this word are aa (appearing 3 times), bb (5 times), $abab$ (3 times), $baba$ (2 times), $babab$ (2 times), $abaaba$ (2 times), $babbab$ (4 times), $ababbababbaba$, $ababbababb$, $aababbabaababbaba$, $ababbabaababbaba$ and

$$ababbabaababbababbabaababbababb.$$

The fact that $\rho(n) = O(n)$ was shown by Kolpakov and Kucherov [6] in 2000. However their method gave no information about the constant implied by the $O(n)$. Values for it were given in [11], [10], [12] and [2] the last and best of these being $1.6n$ by Crochemore and Ilie. In fact they claim that their result could be improved by further computation to about $1.18n$. In the other direction Franek et al [3] have shown that $\rho(n) \geq 3/(1 + \sqrt{5})n = (0.927...)n$. This theoretical study of runs has been driven (at least in part) by the desire to simplify algorithms for their computation. The only linear time algorithm for computing all the runs in a string [6] requires significant algorithmic machinery and working memory, even with recent refinements [5]. We are similarly motivated here, and hope that with an improved understanding of the average case, simpler algorithms that trade $O(n)$ worst case performance for fast expected running time may emerge.

In this paper we consider the expected number of runs in a word of length n, that is, the mean number of runs in all words of length n found in an alphabet of given

size. We find that this decreases with alphabet size increasing from 2. (A word on a unary alphabet contains at most one run.) The papers considering $\rho(n)$ do not investigate the dependence of the maximum number of runs on the alphabet size, though experiments suggest this also is greatest with an alphabet of size 2.

The main result of this paper is Corollary 2.5 which gives a formula for the expected number of runs per unit length in words on an alphabet of size q as the word length tends to infinity. We obtain this result in the next section. In the final section we compare our results with the average number of runs appearing in long real-world sequences, such as English text and DNA sequences.

2 Main Results

Recall that the Möbius function $\mu(n)$ is defined by $\mu(1) = 1$ and if n has prime factorisation

$$n = \prod_{i=1}^{t} p_i^{\alpha_i}$$

then $\mu(n) = 0$ if $\alpha_i > 1$ for any i and $(-1)^t$ otherwise. We write $P_q(n)$ for the number of primitive words of length n on an alphabet of size q. The following result appears in the first Lothaire book [7] as Equation (1.3.7) and is easily proved using Möbius inversion.

Lemma 2.1.

$$P_q(n) = \sum_{d|n} \mu(d) q^{n/d}.$$

Our first result is the following.

Theorem 2.2. *The number of runs of period p in the set of all words of length n on an alphabet of size q is*

$$q^{n-2p-1}((n-2p+1)q - (n-2p))P_q(p)$$

for $n \geq 2p$.

Proof. We prove this by induction on n. Let $N(n, p, q)$ be the number of runs of period p in the set of all words of length n on an alphabet of size q. If $n = 2p$ each primitive prefix of length p gives rise to exactly one run. Thus there are exactly $P_q(p)$ runs among these words. Thus $N(2p, p, q) = P_q(p)$ and the formula holds for $n = 2p$.

Now suppose the statement holds for $n = k$. Each word of length $k + 1$ will contain the set of runs that is contained in its length k prefix, together with at most one additional period p run. Each length k word is the prefix of q length $k + 1$ words, so the total number of period p runs that appear in the length k prefixes of all length $k+1$ words is $qN(k, p, q)$. The only way a word of length $k+1$ can contain one more

run of period p than does its length k prefix is if it has the form $\mathbf{u}a\mathbf{v}b\mathbf{v}b$ where a and b are distinct letters, $\mathbf{u}$ and $\mathbf{v}b$ are factors, $|\mathbf{v}b| = p$ and $\mathbf{v}b$ is primitive. It is easily seen that $\mathbf{u}$ has length $k - 2p$. There are thus q^{k-2p} choices for $\mathbf{u}$, $q - 1$ choices for a and $P_q(p)$ choices for $\mathbf{v}b$. Thus the number of extra runs equals $q^{k-2p}(q-1)P_q(p)$. So the total number of runs in the q^{k+1} words of length $k+1$ is

$$\begin{aligned} & qN(k,p,q) + q^{k-2p}(q-1)P_q(p) \\ = \; & q\{q^{k-2p-1}((k-2p+1)q - (k-2p))P_q(p)\} + q^{k-2p}(q-1)P_q(p) \\ = \; & q^{k-2p}((k-2p+2)q - (k+1-2p))P_q(p), \end{aligned}$$

as required. □

We now obtain a formula for the total number of runs in the set of all words of length n on an alphabet of size q. We obtain this by summing the formula from the last theorem over all p satisfying $p \leq n/2$.

Theorem 2.3. *The number of runs in the set of all words of length n on an alphabet of size q is*

$$\sum_{p=1}^{\lfloor n/2 \rfloor} q^{n-2p-1}((n-2p+1)q - (n-2p)) \sum_{d|p} \mu(d) q^{p/d}.$$

Proof. Let the required number be $M(n,q)$. Using the notation in the proof of the last theorem we have

$$\begin{aligned} M(n,q) &= \sum_{p=1}^{\lfloor n/2 \rfloor} N(n,p,q) \\ &= \sum_{p=1}^{\lfloor n/2 \rfloor} q^{n-2p-1}((n-2p+1)q - (n-2p))P_q(p) \\ &= \sum_{p=1}^{\lfloor n/2 \rfloor} q^{n-2p-1}((n-2p+1)q - (n-2p)) \sum_{d|p} \mu(d) q^{p/d}. \end{aligned}$$

□

We now look at some values of these functions. Table 1 shows the expected number of runs in words of length n using an alphabet of size q. Since q^n words of length n can be constructed from an alphabet of size q the expected number of runs is just $M(n,q)/q^n$.

The expected number of runs decreases with the alphabet size. If we divide the figures in the table by the word lengths we get the expected number of runs per unit length. We show below that this approaches a limit depending only on the alphabet size. Some values of the expected number of runs per unit length are shown in the Table 2.

Alphabet size:	2	3	5	10
Word length: 2	0.5	0.3333	0.2	0.1
3	0.75	0.5556	0.36	0.19
5	1.4375	1.1235	0.7376	0.3871
10	3.4043	2.6318	1.7022	0.8824
20	7.4914	5.6789	3.6350	1.8733

Table 1: Expected number of runs in words of various lengths using various sized alphabets.

Alphabet size:	2	3	5	10
Word length: 2	0.2500	0.1667	0.1000	0.0500
3	0.2500	0.1852	0.1200	0.0633
5	0.2875	0.2247	0.1475	0.0774
10	0.3404	0.2632	0.1702	0.0882
20	0.3746	0.2839	0.1818	0.0937
50	0.3968	0.2965	0.1887	0.0969
100	0.4042	0.3007	0.1910	0.0980
200	0.4079	0.3028	0.1921	0.0986
Limits	0.4116	0.3049	0.1933	0.0991

Table 2: Expected number of runs per unit length in words of various lengths using various sized alphabets. The last row gives the limits obtained from Corollary 2.5.

To obtain a formula for the limits of the columns of the table we first obtain a formula for the limit of $M(n+1,q)/q^{n+1} - M(n,q)/q^n$ as n tends to infinity.

Theorem 2.4. *The limit of* $M(n+1,q)/q^{n+1} - M(n,q)/q^n$ *as* n *goes to infinity equals*

$$\frac{q-1}{q}\sum_{d=1}^{\infty}\frac{\mu(d)}{q^{2d-1}-1}.$$

Proof. We show that the limit approaches this sum as n goes through the even integers. The case for odd integers is similar. Writing $2N$ for n and substituting in the formula from Theorem 2.3 we get

$$\begin{aligned}
&M(n+1,q)/q^{n+1} - M(n,q)/q^n \\
&= \sum_{p=1}^{N} \frac{q^{2N-2p}((2N-2p+2)q-(2N+1-2p))}{q^{2N+1}} \sum_{d|p} \mu(d) q^{p/d} \\
&\quad - \sum_{p=1}^{N} \frac{q^{2N-2p-1}((2N-2p+1)q-(2N-2p))}{q^{2N}} \sum_{d|p} \mu(d) q^{p/d} \\
&= \sum_{p=1}^{N} \frac{q-1}{q^{2p+1}} \sum_{d|p} \mu(d) q^{p/d} \\
&= \frac{q-1}{q} \sum_{p=1}^{N} q^{-2p} \sum_{d|p} \mu(d) q^{p/d}.
\end{aligned}$$

We now change the order of summation, and replacing p with δd. Note that $\{(p,d) : 1 \le p \le N, d|p\} = \{(\delta d, d) : 1 \le d \le N, 1 \le \delta \le \lfloor N/d \rfloor\}$.

$$\begin{aligned}
&\frac{q-1}{q} \sum_{d=1}^{N} \mu(d) \sum_{\delta=1}^{\lfloor N/d \rfloor} q^{-(2d-1)\delta} \\
&\quad = \frac{q-1}{q} \sum_{d=1}^{N} \mu(d) \frac{1 - q^{-\lfloor N/d \rfloor (2d-1)}}{1 - q^{-(2d-1)}} q^{-(2d-1)} \\
&\quad = \frac{q-1}{q} \sum_{d=1}^{N} \frac{\mu(d)}{q^{2d-1}-1} (1 - q^{-\lfloor N/d \rfloor (2d-1)}).
\end{aligned}$$

We claim that as N goes to infinity the sum of the terms involving $q^{-\lfloor N/d \rfloor (2d-1)}$ goes to zero. The absolute value of this sum is

$$\sum_{d=1}^{N} \frac{|\mu(d)|}{q^{2d-1}-1} q^{-\lfloor N/d \rfloor (2d-1)}.$$

Since $\lfloor N/d \rfloor (2d-1) \ge N$ the sum is at most

$$\sum_{d=1}^{N} \frac{|\mu(d)|}{q^{2d-1}-1} q^{-N} < q^{-N} \sum_{d=1}^{\infty} \frac{1}{q^{2d-1}-1} < 2q^{-N}$$

which clearly approaches 0. Thus the infinite sum approaches

$$\frac{q-1}{q} \sum_{d=1}^{\infty} \frac{\mu(d)}{q^{2d-1}-1}.$$

as required. □

Corollary 2.5. *The expected number of runs per unit length in a word on an alphabet of size q tends to*

$$\frac{q-1}{q}\sum_{d=1}^{\infty}\frac{\mu(d)}{q^{2d-1}-1}$$

as the word length approaches infinity.

Proof. For fixed q and using earlier notation, the expected number of runs per unit length in a word of length n is $M(n,q)/nq^n$. We write L for

$$\frac{q-1}{q}\sum_{n=1}^{\infty}\frac{\mu(d)}{q^{2d-1}-1}$$

and let ϵ be a positive number. We will define a number N below and show that for $n > N$ we have

$$\left|\frac{M(n,q)}{nq^n} - L\right| < \epsilon.$$

Note that

$$\frac{M(n,q)}{nq^n} = \frac{1}{n}\sum_{m=1}^{n-1}\left\{\frac{M(m+1,q)}{q^{m+1}} - \frac{M(m,q)}{q^m}\right\}.$$

Since $M(1,q) = 0$ we do not need to allow for the beginning of the sum. By Theorem 2.4 there exists N_1 such that $m > N_1$ implies that

$$\left|\frac{M(m+1,q)}{q^{m+1}} - \frac{M(m,q)}{q^m} - L\right| < \epsilon/2. \tag{2.1}$$

Let

$$B = \left|\sum_{m=1}^{N_1}\left\{\frac{M(m+1,q)}{q^{m+1}} - \frac{M(m,q)}{q^m}\right\}\right|$$

and choose $N > N_1$ so that

$$\left|\frac{B - (N_1+1)L}{N}\right| < \epsilon/2.$$

Thus, for $n > N$,

$$
\begin{aligned}
& \left| \frac{M(n,q)}{nq^n} - L \right| \\
= & \left| \frac{1}{n} \sum_{m=1}^{n-1} \left\{ \frac{M(m+1,q)}{q^{m+1}} - \frac{M(m,q)}{q^m} \right\} - L \right| \\
= & \left| \frac{1}{n} \left\{ \sum_{m=1}^{N_1} \left\{ \frac{M(m+1,q)}{q^{m+1}} - \frac{M(m,q)}{q^m} \right\} \right.\right. \\
& \qquad \left.\left. + \sum_{m=N_1+1}^{n-1} \left\{ \frac{M(m+1,q)}{q^{m+1}} - \frac{M(m,q)}{q^m} \right\} \right\} - L \right| \\
= & \left| \frac{1}{n} \left\{ B + \sum_{m=N_1+1}^{n-1} \left\{ \frac{M(m+1,q)}{q^{m+1}} - \frac{M(m,q)}{q^m} \right\} \right\} - L \right| .
\end{aligned}
$$

Thus, using (2.1),

$$
\begin{aligned}
\frac{B}{n} + \frac{n-1-N_1}{n} \left(L - \frac{\epsilon}{2} \right) - L & < \frac{M(n,q)}{nq^n} - L \\
& < \frac{B}{n} + \frac{n-1-N_1}{n} \left(L + \frac{\epsilon}{2} \right) - L \\
\Rightarrow \frac{B-(N_1+1)L}{n} - \frac{(n-1-N_1)\epsilon}{2n} & < \frac{M(n,q)}{nq^n} - L \\
& < \frac{B-(N_1+1)L}{n} + \frac{(n-1-N_1)\epsilon}{2n} \\
\Rightarrow -\epsilon < \frac{M(n,q)}{nq^n} - L & < \epsilon
\end{aligned}
$$

as required. □

Some values of the limit obtained in this corollary are shown in Table 2.

3 Experimental Results

We compare the number of runs per unit length in various data files with the expected numbers from Corollary 2.5. The discrepancies give some indication of how the data files differ from random sequences on the same alphabets. The data files were broken into chunks of 100,000 symbols and the runs were counted for each chunk and averaged. This was necessary to reduce processing time, particularly on the large genomic data. Human Genome actually refers to a large substring of the whole Human Genome, specifically chromosomes 18, 19 and 22 concatenated together. The Protein string was a 150Mb subset of the Genbank non-redundant protein set [4].

Data type	Alphabet size	Average Runs	Expected Number
Human genome (DNA)	4	0.255	0.2374
Protein	24	0.054	0.0416
King James Bible	63	0.004	0.0159
Alice In Wonderland	74	0.011	0.0135
Wall Street Journal	94	0.010	0.0106
Source Code	90	0.041	0.0111
Executable Program	256	0.008	0.0040
Fibonacci	2	0.764	0.4116
Run Maximal Word	2	0.927	0.4116

Table 3: The number of runs per unit length in words of various types of data. The fourth column gives the expected number of runs per unit length from Corollary 2.5

Run Maximal refers to the words from [3], which are *conjectured* to be run maximal, and Fibonacci is a nearly 15Mb instance of the ubiquitous Fibonacci word (see, for example, [13]). Source Code is implementations of various fundamental algorithms and data structures such as stacks, queues and sorting routings written in the C programming language. The Executable Program is the compiled source code.

Many of the runs in the Alice in Wonderland and Wall Street Journal texts come from the formatting of the text, rather than the text itself. For instance a significant number of runs in those files are a result of multiple new lines and other consecutive whitespace characters like tab and space. The King James Bible on the the other hand is more continuous and so perhaps is more indicative of the number of runs usually occurring in the English language.

In future work we hope to apply insight gained by these results in developing new algorithms for computing runs that have good average case performance, but that may be more straightforward than those currently available. We have shown the expected number of runs to be relatively high — almost half the conjectured maximum for binary strings [3] — perhaps indicating algorithms that run in expected $O(n)$ time will not be substantially simpler than their worstcase counterparts. Nonetheless, we think this is a direction worthy of further pursuit.

References

[1] Alberto Apostolico and Stefano Lonardi, Off-line compression by greedy textual substitution, *Proc. IEEE* 88(11) (2000), 1733–1744.

[2] Maxime Crochemore and Lucian Ilie, *Maximal repetitions in strings*, submitted, 2007

[3] Frantisek Franek, Jamie Simpson, and W.F. Smyth, The maximum number of runs in a string, In *Proc. 14th Australas. Workshop Combin. Algorithms* (eds. M. Miller and K. Park), Seoul (2003), 36–45.

[4] D.A. Benson, I. Karsch-Mizrachi, D.J. Lipman, J. Ostell, and D.L. Wheeler, Genbank, *Nucleic Acids Research* 33 (2005), D34–D38.

[5] G. Chen, S.J. Puglisi and W.F. Smyth, Fast and practical algorithms for computing all the runs in a string, In *CPM 2007* (eds. B. Ma and K. Zhang), Lec. Notes Comp. Sci. 4580, Springer-Verlag, Berlin (2007), 307–315.

[6] Roman Kolpakov and Gregory Kucherov, On maximal repetitions in words, *J. Discrete Algorithms* 1(1) (2000), 159–186.

[7] M. Lothaire, *Combinatorics on Words*, Encyclopedia of Mathematics and its Applications 17, Cambridge, 1997.

[8] M. Lothaire, *Algebraic Combinatorics on words*, Encyclopedia of Mathematics and its Applications 90, Cambridge, 2002.

[9] M. Lothaire, *Applied Combinatorics on Words* Encyclopedia of Mathematics and its Applications 105, Cambridge, 2005.

[10] Simon J. Puglisi, Jamie Simpson and Bill Smyth, *How many runs can a string contain?*, submitted, 2006.

[11] Wojciech Rytter, The number of runs in a string: Improved analysis of the linear upper bound, In *STACS 2006* (eds. B. Durand and W. Thomas), Lec. Notes Comp. Sci. 3884, 184–195. Springer-Verlag, Berlin, 2006.

[12] Wojciech Rytter, The number of runs in a string, *Information and Computation* 205, no. 9, Sept. 2007, 1459–1469.

[13] B. Smyth, *Computing Patterns in Strings*, Pearson Addison-Wesley (2003), 423 pp.

[14] A. Turpin and W.F. Smyth, An approach to phrase selection for offline data compression, In *Proc. 25th Australian Comp. Sci. Conf.* (ed. M.J. Oudshoorn), Melbourne (2002), 267–273.

[15] Axel Thue, Über unendliche zeichenreihen, *Norske Vid. Selsk. Skr. I. Mat. Nat. Kl. Christiana* 7 (1906), 1–22.

(Received 5 July 2007; revised 1 Dec 2007)