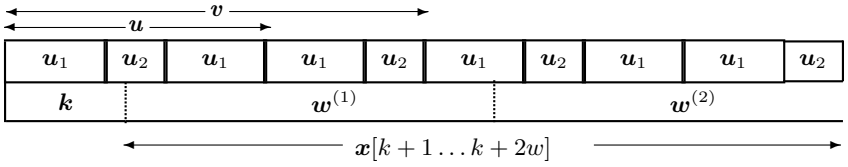
Overlapping squares in strings (01/2011)

Jenya Kopylova[1] and W. F. Smyth[2]

[1] SEQUOIA, Laboratoire d'Informatique Fondamentale de Lille, France
evguenia.kopylova@inria.fr

[2] Algorithms Research Group, Department of Computing & Software
McMaster University, Hamilton, Ontario, Canada L8S 4K1
smyth@mcmaster.ca

Question: What happens to the periodicity of a string when three squares $(\boldsymbol{u}^2, \boldsymbol{v}^2, \boldsymbol{w}^2)$ begin at neighbouring positions separated from each other by at most k?



Analysis: In all the cases we know about so far [7, 1, 2], the string breaks down locally into a repetition of small period. This document draws its focus on the 14 subcases defined in Table 1.

Table 1. 14 subcases of $3u/2 < v < 2u, 0 \le k < v - u$

Subcase S	k	$k + w$	$k + 2w$	Special Conditions
1	$0 \le k \le u_1$	$k + w \le u$	$k + 2w \le u + u_1$	$k \ge u_2$
2	$0 \le k \le u_1$	$k + w \le u$	$k + 2w \le u + u_1$	$k < u_2$
3	$0 \le k \le u_1$	$k + w \le u$	$k + 2w > u + u_1$	—
4	$0 \le k \le u_1$	$u < k + w \le u + u_1$	—	—
5	$0 \le k \le u_1$	$u + u_1 < k + w \le v$	—	—
6	$0 \le k \le u_1$	$v < k + w < 2u$	—	—
7	$u_1 < k < u_1 + u_2$	$k + w \le u + u_1$	$k + 2w \le 2u$	—
8	$u_1 < k < u_1 + u_2$	$k + w \le u + u_1$	$k + 2w > 2u$	—
9	$u_1 < k < u_1 + u_2$	$u + u_1 < k + w \le v$	—	$w < u$
10	$u_1 < k < u_1 + u_2$	$k + w \le v$	$k + 2w \le u + v$	$w > u$
11	$u_1 < k < u_1 + u_2$	$k + w \le v$	$u + v < k + 2w \le 2v - u_2$	—
12	$u_1 < k < u_1 + u_2$	$k + w \le v$	$2v - u_2 < k + 2w$	—
13	$u_1 < k < u_1 + u_2$	$v < k + w \le 2u$	—	—
14	$u_1 < k < u_1 + u_2$	$2u < k + w < 2u + u_2 - 1$	—	—

Solution: Conjectures for most of the 14 subcases were formulated with the aid of an implemented algorithm outlined in [2]:

Table 2. Generated Conjectures

Subcases S	Conditions	Breakdown of $\boldsymbol{x}/\boldsymbol{v}^2$
1,2,5,6,8–10	$(\forall \boldsymbol{x}, \sigma = d)$	$\boldsymbol{x} = \boldsymbol{d}^{(x/d)}$
3,4,7	$\sigma = d$	$\boldsymbol{x} = \boldsymbol{d}^{(x/d)}$
	$\sigma > d$	$\boldsymbol{x} = \boldsymbol{s}^{\alpha}\boldsymbol{s}[1 \ldots u_1 \bmod s]\boldsymbol{s}^{\gamma}\boldsymbol{s}[1 \ldots u_1 \bmod s]\boldsymbol{s}^{\epsilon}$
11–14	$\sigma = d$	$\boldsymbol{x} = \boldsymbol{d}^{(x/d)}$
	$\sigma > d$	?

where,

$$d = \gcd(u_1, u_2, w); s = \gcd(u - w, w - u_1)$$
$$\alpha = \lfloor u/s \rfloor; \gamma = \lfloor v/s \rfloor; \epsilon = (u_1 + u_2)/s$$

For subcases $\{1, 2, 5, 6, 8 - 10\}$, the proofs have been derived in [2]. For subcases $\{10, 11 - 14\}$ where $\sigma = d$, the proofs are outlined in [1].

The remaining subcases $\{3, 4, 7\}$ for $\sigma \geq d$ and $\{11 - 14\}$ for $\sigma > d$ remain as open problems.

Why do we want to know? To be able to use *combinatorial knowledge* about the occurrence of multiple squares at neighbouring positions to:

* provide a more precise and also computation-free analysis of the occurrence of runs in a string;
* compute repetitions (and perhaps other periodicities) directly rather than using all the heavy machinery of suffix arrays, etc.

For further background information and examples, please see:

`http://www.cas.mcmaster.ca/~bill/cv.shtml`
and
`http://www.cas.mcmaster.ca/~bj`

and the accompanying document on ***"The Maximum Number of Runs in a String"*** *by Bill Smyth.*

Problem
The Maximum Number of Runs in a String (2008)

Bill Smyth[1,2]

[1] Algorithms Research Group, Department of Computing & Software
McMaster University, Hamilton, Ontario, Canada L8S 4K1
smyth@mcmaster.ca
www.cas.mcmaster.ca/cas/research/algorithms.htm

[2] Digital Ecosystems & Business Intelligence Institute
and Department of Computing, Curtin University, GPO Box U1987
Perth WA 6845, Australia
smyth@computing.edu.au

Given a nonempty string $\boldsymbol{u}$ and an integer $e \geq 2$, we call $\boldsymbol{u}^e$ a ***repetition***; if $\boldsymbol{u}$ itself is not a repetition, then $\boldsymbol{u}^e$ is a ***proper repetition***. Given a string $\boldsymbol{x}$, a ***repetition in*** $\boldsymbol{x}$ is a substring

$$\boldsymbol{x}[i..i+e|\boldsymbol{u}|-1] = \boldsymbol{u}^e,$$

where $\boldsymbol{u}^e$ is a proper repetition and neither $\boldsymbol{x}\big[i+e|\boldsymbol{u}|..i+(e+1)|\boldsymbol{u}|-1\big]$ nor $\boldsymbol{x}[i-|\boldsymbol{u}|..i-1]$ equals $\boldsymbol{u}$. We say the repetition has ***period*** $|\boldsymbol{u}|$ and ***exponent*** e; it can be specified by the integer triple $(i, |\boldsymbol{u}|, e)$. It is well known [4] that the maximum number of repetitions in a string $\boldsymbol{x} = \boldsymbol{x}[1..n]$ is $\Theta(n \log n)$, and that the number of repetitions in $\boldsymbol{x}$ can be computed in $\Theta(n \log n)$ time [4, 3, 13].

A string $\boldsymbol{u}$ is a ***run*** iff it is periodic of (minimum) period $p \leq |\boldsymbol{u}|/2$. Thus $\boldsymbol{x} = abaabaabaabaab = (aba)^4ab$ is a run of period $|aba| = 3$. A substring $\boldsymbol{u} = \boldsymbol{x}[i..j]$ of $\boldsymbol{x}$ is a ***run*** or ***maximal periodicity in*** $\boldsymbol{x}$ iff it is a run of period p and neither $\boldsymbol{x}[i-1..j]$ nor $\boldsymbol{x}[i..j+1]$ is a run of period p. The run $\boldsymbol{u}$ has ***exponent*** $e = \lfloor |\boldsymbol{u}|/p \rfloor$ and possibly empty ***tail*** $\boldsymbol{t} = \boldsymbol{x}[i+ep..j]$ (proper prefix of $\boldsymbol{x}[i..i+p-1]$). Thus

$$\begin{array}{c} \scriptstyle 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9\ 10\ 11\ 12\ 13\ 14 \\ \boldsymbol{x} = b\ a\ a\ a\ b\ a\ a\ b\ a\ a\ \ b\ \ a\ \ b\ \ a \end{array}$$

contains a run $\boldsymbol{x}[3..12]$ of period $p = 3$ and exponent $e = 3$ with tail $\boldsymbol{t} = a$ of length $t = |\boldsymbol{t}| = 1$. It can be specified by a 4-tuple $(i, p, e, t) = (3, 3, 3, 1)$. and it includes the repetitions $(aab)^3$, $(aba)^3$ and $(baa)^2$ of period $p = 3$. In general it is easy to see that for $e = 2$ a run ***encodes*** $t+1$ repetitions; for $e > 2$, p repetitions. Clearly, computing all the runs in $\boldsymbol{x}$ specifies all the repetitions in $\boldsymbol{x}$. The idea of a run was introduced in [12].

Let $r_{\boldsymbol{x}}$ denote the number of runs that actually occur in a given string $\boldsymbol{x}$, and let $\rho(n)$ denote the maximum number of runs that can possibly occur in any string $\boldsymbol{x}$ of given length n. A string $\boldsymbol{x} = \boldsymbol{x}[1..n]$ such that $r_{\boldsymbol{x}} = \rho(n)$ is said to be ***run-maximal***.

In [10, 11] it was shown that there exist universal positive constants k_1 and k_2 such that

$$\rho(n)/n < k_1 - k_2 \log_2 n/\sqrt{n},$$

but the proof was nonconstructive and provided no way of estimating the magnitude of k_1 and k_2. In [10], using a brute force algorithm, a table of $\rho(n)$ was computed for $n = 5, 6, \ldots, 31$, giving also for each n an example of a run-maximal string; for every n in this range, $\rho(n)/n < 1$ and $\rho(n) \leq \rho(n-1)+2$. In [8] an infinite sequence $X = \{\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots\}$ of strings was described, with $|\boldsymbol{x_{i+1}}| > |\boldsymbol{x_i}|$ for every $i \geq 1$, such that

$$\lim_{i\to\infty} r\boldsymbol{x_i}/|\boldsymbol{x_i}| = \frac{3}{2\phi},$$

where $\phi = \frac{1+\sqrt{5}}{2}$ is the golden mean. Moreover, it was conjectured that in fact

$$\lim_{n\to\infty} \rho(n)/n = \frac{3}{2\phi}. \tag{1}$$

Recently a different and simpler construction was found [9] to yield another infinite sequence X of strings for which the ratio $r\boldsymbol{x_i}/|\boldsymbol{x_i}|$ approached the same limit; in addition, it was shown that for every $\epsilon > 0$ and for every sufficiently large $n = n(\epsilon)$, $\frac{3}{2\phi} - \epsilon$ provides an asymptotic lower bound on $\rho(n)/n$.

In 2006 considerable progress was made on the estimation of an upper bound on $\rho(n)/n$:

* $\rho(n)/n \leq 5.0$ [15];
* $\rho(n)/n \leq 3.48$ [14];
* $\rho(n)/n \leq 3.44$ [16][1];
* $\rho(n)/n \leq 1.6$ [5].

Thus the problems may be stated as follows:

Is conjecture (1) true?
In any case, characterize the function $\rho(n)/n$.

Help may be found in recent work studying the limitations imposed on the existence and length of runs in neighbourhoods of positions where two runs are known to exist [7, 17].

Additional results in 2008

In [6] new perspectives on the problem are discussed. Based on further computational work, Lucian Ilie's website

`http://www.csd.uwo.ca/faculty/ilie/`

claims $\rho(n)/n \leq 1.048n$.

References

1. R. J. Simpson, **Intersecting periodic words**, *Theoret. Comput. Sci.* 374 (2007) 58–65.

[1] Unfortunately, this bound has recently been found to be invalid, due to an error in a proof (2008)

2. Evguenia Kopylov & W. F. Smyth, **The three squares lemma revisited**, J. Discrete Algorithms (submitted for publication).
3. Alberto Apostolico & Franco P. Preparata, **Optimal off-line detection of repetitions in a string**, *Theoret. Comput. Sci. 22* (1983) 297–315.
4. Maxime Crochemore, **An optimal algorithm for computing the repetitions in a word**, *Inform. Process. Lett. 12–5* (1981) 244–250.
5. Maxime Crochemore & Lucian Ilie, **Maximal repetitions in strings**, *J. Comput. Sys. Sci.* (2007) to appear.
6. Maxime Crochemore, Lucian Ilie & Liviu Tinta, **Towards a solution to the "runs" conjecture**, *Proc.* 19^{th} *Annual Symp. Combinatorial Pattern Matching* (2008) to appear.
7. Kangmin Fan, Simon J. Puglisi, W. F. Smyth & Andrew Turpin, **A new periodicity lemma**, *SIAM J. Discrete Math. 20–3* (2006) 656–668.
8. Frantisek Franek, R. J. Simpson & W. F. Smyth, **The maximum number of runs in a string**, *Proc.* 14^{th} *Australasian Workshop on Combinatorial Algs.*, M. Miller & K. Park (eds.) (2003) 26–35.
9. Frantisek Franek & Qian Yang, **An asymptotic lower bound for the maximum-number-of-runs function**, *Proc. Prague Stringology Conference '06*, Jan Holub & Jan Žd'árek (eds.) (2006) 3–8.
10. Roman Kolpakov & Gregory Kucherov, *Maximal Repetitions in Words or How to Find all Squares in Linear Time*, Rapport LORIA 98-R-227, Laboratoire Lorrain de Recherche en Informatique et ses Applications (1998) 22 pp.
11. Roman Kolpakov & Gregory Kucherov, **On maximal repetitions in words**, *J. Discrete Algorithms 1* (2000) 159–186.
12. Michael G. Main, **Detecting leftmost maximal periodicities**, *Discrete Applied Maths. 25* (1989) 145–153.
13. Michael G. Main & Richard J. Lorentz, **An** $O(n \log n)$ **algorithm for finding all repetitions in a string**, *J. Algs. 5* (1984) 422–432.
14. Simon J. Puglisi, R. J. Simpson & W. F. Smyth, **How many runs can a string contain?**, *Theoret. Comput. Sci.* (2008) to appear.
15. Wojciech Rytter, **The number of runs in a string: improved analysis of the linear upper bound**, *Proc.* 23rd *Symp. Theoretical Aspects of Computer Science*, B. Durand & W. Thomas (eds.), LNCS 2884, Springer-Verlag (2006) 184–195.
16. Wojciech Rytter, **The number of runs in a string**, *Information & Computation 205–9* (2007) 1459–1469.
17. R. J. Simpson, **Intersecting periodic words**, *Theoret. Comput. Sci.* (2007) 58–65.