The Combinatorics of Overlapping Squares

Bill Smyth

Algorithms Research Group, Department of Computing & Software
McMaster University, Hamilton, Canada

Department of Mathematics & Statistics,
University of Western Australia, Perth, Australia

email: smyth@mcmaster.ca

Challenges in Combinatorics on Words
The Fields Institute, Toronto
24 April 2013

Abstract

I briefly review two closely-related research topics pursued over the last ten years or so:

- What is the maximum number of runs (maximal periodicities) in a string of length n?
- What are the limitations on the occurrence of overlapping squares in a string?

I suggest new strategies for dealing with these questions, as well as possible algorithmic consequences.

Outline

Repetitions & Runs

- If $\mathbf{x} = \mathbf{v}\mathbf{u}^e\mathbf{w}$, with integer $e > 1$ and $\mathbf{u}$ neither a suffix of $\mathbf{v}$ nor a prefix of $\mathbf{w}$ (e is maximum), then $\mathbf{u}^e$ is said to be a repetition in $\mathbf{x}$. The integers u and e are the period and exponent, respectively, of the repetition.
- For example, in

$$\mathbf{x} = abaababaab, \tag{1}$$

  there are repetitions a^2 (twice), $(ab)^2$ and $(ba)^2$, $(aba)^2$, and $(abaab)^2$. Each of these repetitions is a square ($e = 2$). In general, every repetition has a square prefix.
- If $\mathbf{v} = \mathbf{x}[i..j]$ has period u, where $v/u \geq 2$, and if neither $\mathbf{x}[i-1..j]$ nor $\mathbf{x}[i..j+1]$ (whenever these are defined) has period u, then $\mathbf{x}$ is said to be a maximal periodicity or run in $\mathbf{x}$ [M89] and $\mathbf{v}$ is said to have exponent $e = \lfloor v/u \rfloor$ and tail $t = v \bmod u$. When $t = 0$, the run is also a repetition.
- All of the repetitions in (1) are runs except for $(ab)^2$ and $(ba)^2$: these are prefix and suffix, respectively, of the run $\mathbf{v} = ababa$.
- In general, every repetition is a substring of some run; thus computing all the runs implicitly computes all the repetitions.

Computing Repetitions

In the early 1980s three $O(x \log x)$-time algorithms were proposed to compute all the repetitions in a given string $\mathbf{x}$:

- Crochemore [C81] describes a method of successive refinement that identifies all equal substrings of lengths $1, 2, \ldots$ until for some length ℓ every substring is unique. As remarked in [S03], his method is essentially an algorithm for suffix tree construction. Crochemore also showed that a string $\mathbf{x}$ can contain as many as $O(x \log x)$ repetitions — thus all these algorithms are optimal.
- Apostolico & Preparata [AP83] use suffix trees plus auxiliary data structures.
- Main & Lorentz [ML84] use a divide-and-conquer approach based on prior computation of the Lempel-Ziv factorization $\mathrm{LZ}_{\mathbf{x}}$.

Note: all use global data structures.

Computing LZ [ZL77]

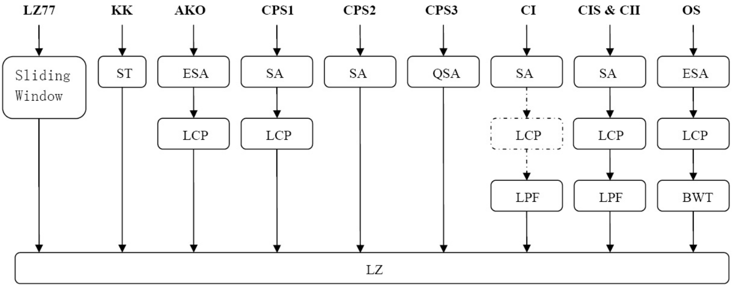


Figure: A wide variety of algorithmic approaches to the computation of the Lempel-Ziv factorization, all of them based on the computation of global data structures (from [ACIKSTY13])

Computing Runs

- In 1989 Main [M89] showed how to compute all "leftmost" runs, again from $LZ_{\mathbf{x}}$, in linear time — thus still global data structures.
- In 1999 Kolpakov & Kucherov [KK99, KK00] showed how to compute all runs from the leftmost ones, also in linear time.
- To establish linearity, they proved that the maximum number $\rho(n)$ of runs over all strings of length n satisfies

$$\rho(n) \leq k_1 n - k_2 \sqrt{n} \log_2 n \tag{2}$$

  for some universal positive constants k_1 and k_2.
- They provided computational evidence (up to $n = 60$) that $\rho(n) \leq n$ — this was their conjecture.
- Based on work by many authors over the last 10 years, it has been shown that $0.944575 < \rho(n)/n \leq 1.029$: the lower bound is combinatorial [S10], the upper largely computational [CIT11].

Unsatisfactory Situation

Moreover, the expected number of runs in a string of length n is small (Puglisi & Simpson [PS08]):

- $0.41n$ for alphabet size $\sigma = 2$;
- $0.25n$ for DNA ($\Sigma = \{A, C, G, T\}$);
- $0.04n$ for protein ($\sigma = 20$);
- $0.01n$ for English-language text.

Runs (hence repetitions) in most strings are sparse!

We have to use global data structures to compute something that is not only local in the string, but that generally occurs sparsely — obviously we need to understand better what is going on.

Combinatorial Insight?

If $\rho(n)/n$ is limited to be near one, it means that on average there is about one run starting at each position. So ... if TWO runs start at some position, then there must be some other position, probably nearby, at which NO runs start.

Runs always start with squares — what do we know about squares that begin at about the same position? What combinatorial insight do we have into the restrictions that might be imposed upon occurrences of overlapping squares? Until recently, very little:

From 1906 to 1995!

Lemma (Crochemore & Rytter [CR95])

Suppose $\mathbf{u}$ *is not a repetition, and suppose* $\mathbf{v} \neq \mathbf{u}^j$ *for any* $j \geq 1$. *If* $\mathbf{u}^2$ *is a prefix of* $\mathbf{v}^2$, *in turn a proper prefix of* $\mathbf{w}^2$, *then* $w \geq u+v$.

The Fibonacci string demonstrates that this result is best possible (squares ending at positions 6, 10, $16 = 6+10$, $26 = 10+16$):

$$\begin{array}{rcccccccccccccccccccccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 & 19 & 20 & 21 & 22 & 23 & 24 & 25 & 26 \\ \mathbf{x} = & a & b & a & a & b & \underline{a} & b & a & a & \underline{b} & a & a & b & a & b & \underline{a} & a & b & a & b & a & a & b & a & a & \underline{b} \end{array}$$

The Three Squares Lemma is a result of great insight: it tells us that if three squares occur at the same position, then one of them has to be "large". But we need to know much more: what if the three squares just overlap, just occur in the same neighbourhood? What then???

New Ideas (since 2005)

We paraphrase the accumulated results of

[FSS05, PST05, S05, FPST06, S07, KS12, FFSS12]:

The bulk of the research considers two squares $\mathbf{u}^2$ and $\mathbf{v}^2$, $u < v < 2u$, so that $\mathbf{u}$, but not $\mathbf{u}^2$, is a prefix of $\mathbf{v}$. There are two cases, whose analysis is quite different, but whose results are qualitatively the same, a breakdown of the string into runs of small period:

(C1) $v \leq 3u/2$;

(C2) $v > 3u/2$.

The details are complicated, but the main results are as follows:

$u < v \leq 3u/2$: $\mathbf{w}$ not required

Theorem (C1)

If $\mathbf{x} = \mathbf{v}^2$ *with prefix* $\mathbf{u}^2$, $u < v \leq 3u/2$, *then*

$$\mathbf{x} = \mathbf{u_1}^m \mathbf{u_2} \mathbf{u_1}^{m+1} \mathbf{u_2} \mathbf{u_1},$$

where $u_1 = v - u \leq u/2$, $u_2 = u \bmod u_1 \geq 0$, $m = \lfloor u/u_1 \rfloor \geq 2$, *and* $\mathbf{u_2}$ *is a proper prefix of* $\mathbf{u_1}$. *Moreover,* $\mathbf{x}$ *contains no runs of period* $\geq u_1$ *other than specific identifiable ones described in [KS12].*

For example, the prefix $\mathbf{f}[1..10] = \mathbf{v}^2 = (abaab)^2$ of the Fibonacci string $\mathbf{f}$ given above has proper prefix $\mathbf{u}^2 = (aba)^2$; hence $u = 3$ and $v = 5$, we find $3u/2 < v < 2u$, and so $\mathbf{u_1} = a$, $\mathbf{u_2} = b$, the shortest possible C1. Also the prefix $\mathbf{f}[1..16] = \mathbf{v}^2 = (abaababa)^2$ has proper prefix $\mathbf{u}^2 = (abaab)^2$, so that now $u = 5$, $v = 8$, again satisfying $3u/2 < v < 2u$, and $\mathbf{u_1} = ab$, $\mathbf{u_2} = b$.

$3u/2 < v < 2u$

Theorem (C2)

Suppose $\mathbf{u}^2$ *and* $\mathbf{v}^2$, $3u/2 < v < 2u$, *occur at the same position* i *in* $\mathbf{x}$. *Then* $\mathbf{v} = \mathbf{u_1 u_2 u_1 u_1 u_2}$, *where* $u_1 = 2u - v,\ u_2 = 2v - 3v$. *If moreover a third square* $\mathbf{w}^2$ *occurs at position* $i+k$, *where* $v-u < w < v$, $w \neq u$, $0 \leq k < v-u$, *then* $\mathbf{x}[i..i+2v-1]$ *breaks down into runs of small period according to 14 well-defined subcases [KS12, FFSS12].*

I confess that it is an exaggeration to call this a "theorem" – two of the 14 subcases have been only partly proved [FPST06, FFSS12]. Nevertheless there is convincing evidence from extensive computer simulations [KS12] that the incomplete cases do satisfy the stated constraint.

Two Subcases

We show Subcases 5 & 13: for both it is true [KS12] that $\mathbf{v} = \mathbf{d}^{v/d}$, with $\mathbf{d}$ a prefix of $\mathbf{v}$ of length $d = \gcd(u, v, w)$.

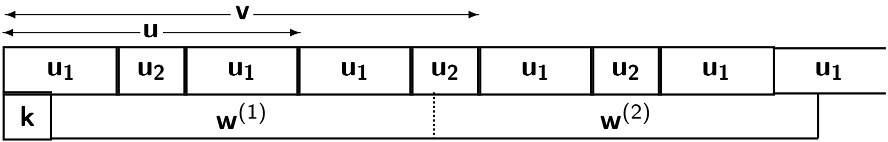


Figure: Subcase 5: $0 \leq k \leq u_1,\ u+u_1 < k+w \leq v$

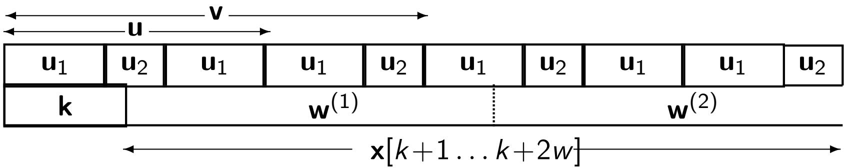


Figure: Subcase 13: $u_1 < k < u_1+u_2,\ v < k+w \leq 2u$

Along the Way ...

In connection with (C2), a new and useful lemma[1] emerged: what happens when both $\mathbf{x}$ and some rotation (cyclic shift) of $\mathbf{x}$ have the same period?

Lemma

Suppose both $\mathbf{x}$ *and* $R_v(\mathbf{x})$, $0 < v < x$, *have period* u, *where* $\ell = x \bmod u > 0$ *and* $e = \lfloor x/u \rfloor$. *Let* $\mathbf{x}_v$ *denote* $R_v(\mathbf{x})$, *and let* $d = \gcd(u, \ell)$. *Then*

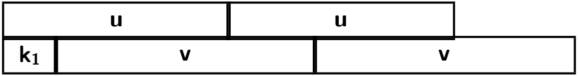
- (a) *if* $e = 1$ *and* $v \geq \ell$, $\mathbf{x}_{v-\ell}[1..2\ell]$ *is a square of period* ℓ;
- (b) *if* $e = 1$ *and* $v \leq \ell$, $\mathbf{x}[1..v+\ell]$ *has period* ℓ;
- (c) *if* $e > 1$ *and* $v < u$, $\mathbf{x}[1..v+\ell]$ *has period* ℓ; *if moreover* $v+d \geq u$, *then* $\mathbf{x}$ *is a repetition of period* d;
- (d) *if* $e > 1$ *and* $u \leq v \leq x-u$, $\mathbf{x}[1..u+\ell]$, *hence* $\mathbf{x}$, *is a repetition of period* d;
- (e) *if* $e > 1$ *and* $x-u < v$, *where* $v' = v-(x-u)$, $\mathbf{x}[v'+1..u+\ell]$ *has period* ℓ; *if moreover* $v' \leq d$, *then* $\mathbf{x}$ *is a repetition of period* d.

[1] Credit to 23 PhD students in Informatics at the University of Warsaw, who verified the result up to $x = 4000$!

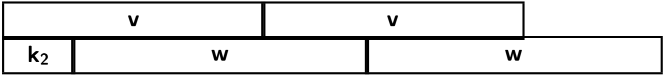
The General Case

Clearly, quite apart from the two missing subcases, there is much more work to be done:

- What can be said when $w > v$ (as in the case of the Three Squares Lemma), but with $k > 0$?
- What if $\mathbf{u}^2$ and $\mathbf{v}^2$ are not coincident?
- What if $\mathbf{w}^2$ occurs to the left of $\mathbf{u_2}$ — or somewhere in between $\mathbf{u}^2$ amd $\mathbf{v}^2$?
- In other words, we need a general case that puts together



and



In fact, a start has recently been made in this direction [S13], but an analysis of the combinaotorial possibilities requires consideration of many more subcases.

Putting It All Together

- With deeper combinatorial insight, perhaps we can classify the possible periodic structures at each position in a string,
- so that a computer program can do a left-to-right scan to compute all the repetitions using an order of magnitude less time and space than present algorithms;
- and thus deal with terabytes tomorrow the way we process gigabytes today:
- an advance in software based on combinatorics.

????

- Anisa Al-Hafeedh, Maxime Crochemore, Lucian Ilie, Evguenia Kopylova, W. F. Smyth, German Tischler & Munina Yusufu, **A comparison of index-based Lempel-Ziv LZ77 factorization algorithms**, *ACM Computing Surveys* (2012) to appear.
- Alberto Apostolico & Franco P. Preparata, **Optimal off-line detection of repetitions in a string**, *Theoret. Comput. Sci. 22* (1983) 297–315.
- Maxime Crochemore, **An optimal algorithm for computing all the repetitions in a word**, *Inform. Process. Lett. 12–5* (1981) 244–248.
- Maxime Crochemore, Lucian Ilie & Liviu Tinta, **The "runs" conjecture**, *TCS 412–27* (2012) 2931–2941.
- Maxime Crochemore and Wojciech Rytter, **Squares, cubes, and time-space efficient strings searching**, *Algorithmica 13* (1995) 405–425.
- Kangmin Fan, Simon J. Puglisi, W. F. Smyth & Andrew Turpin, **A new periodicity lemma**, *SIAM J. Discrete Math. 20–3* (2005) 656–668.
- Kangmin Fan, R. J. Simpson & W. F. Smyth, **A new periodicity lemma** (preliminary version), *Proc. 16th Annual Symp. Combinatorial Pattern Matching*, Springer Lecture Notes in Computer Science LNCS 3537 (2005) 257–265.
- Frantisek Franek, Robert C. G. Fuller, Jamie Simpson & W. F. Smyth, **More results on overlapping squares**, *J. Discrete Algorithms* (2012) to appear.

Mathieu Giraud, **Not so many runs in strings**, *Proc. 2nd Internat. Conf. on Language & Automata Theory & Applications*, Carlos Martín-Vide, Friedrich Otto & Henning Fernau (eds.), Lecture Notes in Computer Science, LNCS 5196, Springer-Verlag (2008) 232–239.

Mathieu Giraud, **Asymptotic behavior of the numbers of runs and microruns**, *Inform. & Computation* 207–11 (2009) 1221–1228.

Roman Kolpakov & Gregory Kucherov, **Finding maximal repetitions in a word in linear time**, *Proc. 40th Annual IEEE Symp. Found. Computer Science* (1999) 596–604.

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

Evguenia Kopylova & W. F. Smyth, **The three squares lemma revisited**, *J. Discrete Algorithms 11* (2012) 3–14.

Michael G. Main, **Detecting leftmost maximal periodicities**, *Discrete Applied Maths. 25* (1989) 145–153.

Michael G. Main & Richard J. Lorentz, **An $O(n \log n)$ algorithm for finding all repetitions in a string**, *J. Algorithms 5* (1984) 422–432.

Simon J. Puglisi & R. J. Simpson, **The expected number of runs in a word**, *Australasian J. Combinatorics 42* (2008) 45–54.

Simon J. Puglisi, R. J. Simpson & W. F. Smyth, **How many runs can a string contain?**, *Theoret. Comput. Sci. 401* (2008) 165–171.

Simon J. Puglisi, W. F. Smyth & Andrew Turpin, **Some restrictions on periodicity in strings**, *Proc. 16th Australasian Workshop on Combinatorial Algs.* (2005) 263–268.

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.

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

Jamie Simpson, **Modified Padovan words and the maximum number of runs in a word**, *Australasian J. Combinatorics* 46 (2010) 129–145.

Bill Smyth, *Computing Patterns in Strings*, Pearson Addison-Wesley (2003) 423 pp.

W. F. Smyth, **Computing periodicities in strings — a new approach**, *Proc. 16th Australasian Workshop on Combinatorial Algs.* (2005) 481–488.

W. F. Smyth, **Three overlapping squares: the general case characterized**, *Theoret. Comput. Sci.* , submitted for publication (2013).

Jacob Ziv & Abraham Lempel, **A universal algorithm for sequential data compression**, *IEEE Trans. Information Theory 23* (1977) 337–343.