Problem: Compute Runs/Repetitions Without Global Data Structures

Bill Smyth

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

School of Engineering & Information Technology,
Murdoch University, Perth, Australia

Algorithm Design Group, Department of Informatics
King's College London

email: smyth@mcmaster.ca

IWOCA 2014, 15–17 October 2014

Runs/Repetitions

The string

$$\begin{array}{ccccccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ \mathbf{x} = & c & g & c & c & g & c & g & c & c & g \end{array} \quad (1)$$

has runs (maximal periodicities) c^2 (twice), $cgcgc$, $(cgc)^2$, and $(cgccg)^2$. All of these are repetitions except $cgcgc$, which defines two more repetitions $(cg)^2$ and $(gc)^2$.

In general, every repetition is a substring of some run; thus computing all the runs implicitly computes all the repetitions.

Computing Runs

Runs can be computed in linear time [KK00], but only using global data structures (suffix tree, suffix array, etc.) that depend on ordering suffixes of the string.

Why??? Runs are

- sparse [PS08]: $0.25n$ expected for DNA, $0.01n$ expected for English text;
- local: generally confined to small substrings;
- unordered: unaffected by the ordering of the alphabet.

Why can't we compute runs more easily??? It would make an orders of magnitude difference in the time and space requirments.

We need to understand more about the combinatorics of squares, especially "double squares".

Double Square I

Definition

If $\mathbf{x} = \mathbf{v}^2$ has a proper prefix $\mathbf{u}^2$, $u < v < 2u$, we say that $\mathbf{x}$ is a *double square* *and write it $\mathbf{x} = DS(\mathbf{u}, \mathbf{v})$.*

Lemma (Old NPL [FSS05, FPST06])

Let $\mathbf{x} = DS(\mathbf{u}, \mathbf{v})$, where $\mathbf{u}$ has no square prefix and $\mathbf{v}$ is not a repetition. Then for all integers k and w such that $0 \leq k < v - u < w < v$ and $w \neq u$, $\mathbf{x}[k+1..k+2w]$ is not a square.

Lemma (New NPL [BFS14])

Consider a double square $DS(\mathbf{u}, \mathbf{v}) = (\mathbf{u_1}, \mathbf{u_2}, e_1, e_2)$. If $\mathbf{w}^2$ is a proper substring of $\mathbf{v}^2$, then either

(a) *$w < u$, or*

(b) *$u \leq w < v$ and the smallest generator of $\mathbf{w}$ is a conjugate (rotation) of $\mathbf{u_1}$.*

DS II

Table : Structure of $\mathbf{x}$ for subcases $S \in 1..14$: σ is the largest alphabet size consistent with u, v, k, w [FFSS12]; $\mathbf{d}$, $\mathbf{d_1}$ and $\mathbf{d_3}$ are prefixes of $\mathbf{x}$ with $d = \gcd(u, v, w)$, $d_1 = \gcd(u-w, v-u)$, $d_2 = \gcd(u, v-w)$, $d_3 = v \bmod d_2$.

Subcases S	Conditions	Breakdown of $\mathbf{x}$
$1, 2, 5, 6, 8$–10	$(\forall \mathbf{x}, \sigma = d)$	$\mathbf{x} = \mathbf{d}^{x/d}$
$3, 4, 7$	$(\forall \mathbf{x})$	$\mathbf{x} = \mathbf{d_1}^{u/d_1}\mathbf{d_1}^{v/d_1}\mathbf{d_1}^{(v-u)/d_1}$
	specified cases	$\mathbf{x} = \mathbf{d}^{x/d}$
11–14	$\sigma = d$ or $d_2 \leq 2u - v$	$\mathbf{x} = \mathbf{d}^{x/d}$
	otherwise	$\mathbf{x} = \left((\mathbf{d_3}^{d_2/d_3})^{v/d_2}\right)^2$

DS II

Lemma (S07,KS12,FFSS12,BS14)

Suppose that in $\mathbf{x} = DS(\mathbf{u}, \mathbf{v})$, $3u/2 < v < 2u$, $\mathbf{w}^2$ *occurs at* $\mathbf{x}[k+1]$, *where* $0 \leq k < v-u < w < v$, $w \neq u$. *Then for each of the 14 subcases, the corresponding structure of* $\mathbf{x}$ *is given in the above table.*

Note:

- The constraints on **u** and **v** are gone!
- In every case the assumption that $\mathbf{w}^2$ exists forces a breakdown into runs of small period, whose generator ($\mathbf{d}$, $\mathbf{d_1}$ or $\mathbf{d_3}$) is a prefix of $\mathbf{x}$; in all but a few instances (subsubcases of 3,4,7), $\mathbf{x}$ is a single repetition of small period.
- This is Structure! What do we do with it?

DS III: The Magical L-Root

It took only a page [BIINTT14] for six Japanese mathematicians to show that $\rho(n) \leq n-1$, a problem that dozens of smart people had been working on for 15 years.

Definition

Consider the two orderings of $\Sigma = \{c, g\}$*:*

- *F (Forward):* $c < g$
- *B (Backward):* $g < c$

and the associated lexicographic (dictionary) orderings F and B of strings $\mathbf{x}$ *on* Σ*. Then a primitive string* $\mathbf{x}$ *on* Σ *is a Lyndon word* L_F *(respectively,* L_B*) if it is the (unique) least in F-order (respectively, B-order) over all rotations* $R_j(\mathbf{x})$, $1 \leq j \leq n-1$.

For example, $\mathbf{x} = ccg$ is L_F, $\mathbf{y} = gcc$ is L_B, $\mathbf{z} = cgc$ is not a Lyndon word.

DS III: F-Root & B-Root

Definition

The F-root (respectively, B-root) of a run in $\mathbf{x}$ *is the position in* $\mathbf{x}$ *of the Lyndon word* L_F *(respectively,* L_B*) that is conjugate to the (primitive) generator of the run and leftmost in the run, except not the run's first position.*

$$\begin{array}{ccccccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ \mathbf{x} = & c & g & c & \color{red}{c} & \color{red}{g} & \color{red}{c} & \color{red}{g} & \color{red}{c} & c & g \\ & & & & & B & F & & & & \end{array} \quad (2)$$

The F-root of run $cgcgc$ in $\mathbf{x} = (cgccg)^2$ is position 6, the B-root is position 5.

DS III: L-Root

Definition

Suppose that a sentinel letter $\$ > g > c$ *is appended to* $\mathbf{x}$. *Then the L-root of a run in* $\mathbf{x}$ *is the F-root if the run is followed by* c, *the B-root otherwise.*

$$
\begin{array}{rccccccccccc}
 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 \\
\mathbf{x} = & c & g & c & c & g & c & g & c & c & g & \$ \\
 & & & & & B & F & & & \uparrow & & \\
 & & & & & & L & & & & &
\end{array} \qquad (3)
$$

Lemma

The L-roots of the runs in $\mathbf{x}$ *are distinct!*

Corollary

$\rho(n) \leq n-1$.

DS III: L-Root Example

The runs in $\mathbf{x} = (cgccg)^2$ are

- cc (twice, period 1)
- $cgcgc$ (period 2)
- $(cgc)^2$ (period 3)
- $(cgccg)^2$ (period 5)

$$\begin{array}{rccccccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 \\ \mathbf{x} = & c & g & c & c & g & c & g & c & c & g & \$ \\ \text{periods} = & & 3 & & 1 & 5 & 2 & & & 1 & & \end{array} \tag{4}$$

Hey presto!

Now What?

These very recent results give us a great deal of structural information about the behavious of squares (repeptitions) in strings: now we need to work out how to use it ...

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.

Haoyue Bai, Frantisek Franek & W. F. Smyth, **The New Periodicity Lemma Revisited**, *Discrete Applied Math.*, submitted for publication (2014).

Hideo Bannai, Tomohiro I, Shunsuke Inenaga, Yuto Nakashima, Masayuki Takeda & Kazuya Tsuruta, **The "runs" theorem**, `http://arxiv.org/abs/1406.0263` (2014).

Widmer Bland & W. F. Smyth, **Three Overlapping Squares: The General Case Characterixed & Applications**, *Theoret. Comput. Sci.* , submitted for publication (2014).

Gang Chen, Simon J. Puglisi & W. F. Smyth, **Fast & practical algorithms for computing all the runs in a string**, *Proc. 18th Annual Symp. Combinatorial Pattern Matching*, B. Ma & K. Zhang (eds.), Springer Lecture Notes in Computer Science, LNCS 4580, Springer-Verlag (2007) 307–315.

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, **Maximal repetitions in strings**, *J. Comput. Sys. Sci.* (2008) 796–807.

Maxime Crochemore, Lucian Ilie & Liviu Tinta, **The "runs" conjecture**, *Theoret. Comput. Sci. 412* (2011) 2931–2941.

Maxime Crochemore and Wojciech Rytter, **Squares, cubes, and time-space efficient strings searching**, *Algorithmica 13* (1995) 405–425.

Antoine Deza, Frantisek Franek & Adrien Thierry, **How many double squares can a string contain?**, `http://arxiv.org/abs/1310.3429` (2014).

Kangmin Fan, Simon J. Puglisi, W. F. Smyth & Andrew Turpin, **A new periodicity lemma**, *SIAM J. Discrete Math. 20–3* (2006) 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, Springer-Verlag (2005) 257–265.

Martin Farach, **Optimal suffix tree construction with large alphabets**, *Proc. 38th IEEE Symp. Found. Computer Science*, IEEE Computer Society (1997) 137–143.

Aviezri S. Fraenkel & Jamie Simpson, **How many squares can a string contain?**, *J. Combinatorial Theory, Series A82–1* (1998) 112–120.

Frantisek Franek, Robert C. G. Fuller, Jamie Simpson & W. F. Smyth, **More results on overlapping squares**, *J. Discrete Algorithms 17* (2012) 2–8.

Frantisek Franek, R. J. Simpson & W. F. Smyth, **The maximum number of runs in a string**, *Proc. 14th Australasian Workshop on Combinatorial Algs.*, Mirka Miller & Kunsoo Park (eds.) (2003) 26–35.

Frantisek Franek & Q. Yang, **An asymptotic lower bound for the maximal number of runs in a string**, *Internat. J. Foundations of Computer Science1* (2008) 195–203.

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.), Springer 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.

Lucian Ilie, **A note on the number of squares in a word**, *Theoret. Comput. Sci. 380–3* (2007) 373–376.

Juha Kärkkäinen, Giovanni Manzini & Simon J. Puglisi, **Permuted longest-common-prefix array**, *Proc. 20th Annual Symp. Combinatorial Pattern Matching*, Gregory Kucherov & Esko Ukkonen (eds.), Springer Lecture Notes in Computer Science, LNCS 5577, Springer Verlag (2009) 181–192.

Toru Kasai, Gunho Lee, Hiroki Akimura, Setsuo Arikawa & Kunsoo Park, **Linear-time longest-common-prefix computation in suffix arrays and its applications**, *Proc. 12th Annual Symp. Combinatorial Pattern Matching*, Amihood Amir & Gad M. Landau (eds.), Springer Lecture Notes in Computer Science, LNCS 2089, Springer-Verlag (2001) 181–192.

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.
- Abraham Lempel & Jacob Ziv, **On the complexity of finite sequences**, *IEEE Trans. Information Theory 22* (1976) 75–81.
- 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.
- Udi Manber & Gene W. Myers, **Suffix array: a new method for on-line string searches**, *Proc. First Annual ACM-SIAM Symp. Discrete Algs.* (1990) 319-327.
- Udi Manber & Gene W. Myers, **Suffix array: a new method for on-line string searches**, *SIAM J. Computing* 22–5 (1993) 935–948.
- G. Manzini, **Two space saving tricks for linear time LCP computation**, *Proc. 9th Scandinavian Workshop on Algorithm Theory*, T. Hagerup & J. Katajainen (eds.), Springer Lecture Notes in Computer Science, LNCS 3111, Springer-Verlag (2004) 372–383.
- Wataru Matsubara, Kazuhiko Kusano, Akira Ishino, Hideo Bannai & Ayumi Shinohara, **New lower bounds for the maximum number of runs in a string**, *PSC* (2008) 140–145.

Edward M. McCreight, **A space-economical suffix tree construction algorithm**, *J. Assoc. Comput. Mach. 32–2* (1976) 262–272.

Yuta Mori, **libdivsufsort**: `http://code.google.com/p/libdivsufsort/`

Ge Nong, Sen Zhang & Wai Hong Chan, **Linear time suffix array construction using D-critical substrings**, *Proc. 20th Annual Symp. Combinatorial Pattern Matching*, Gregory Kucherov & Esko Ukkonen (eds.), Springer Lecture Notes in Computer Science, LNCS 5577, Springer-Verlag (2009) 54–67.

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.

Simon J. Puglisi, W. F. Smyth & Andrew Turpin, **A taxonomy of suffix array construction algorithms**, *ACM Computing Surveys 39–2* (2007) Article 4, 1–31.

Simon J. Puglisi & Andrew Turpin, **Space-time tradeoffs for longest-common-prefix array computation**, *Proc. 19th Internat. Symp. Algs. & Computation*, S.-H. Hong, H. Nagamochi & T. Fukunaga (eds.) (2008) 124–135.

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.), Springer Lecture Notes in Computer Science, 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.

W. F. Smyth, **Repetitive perhaps, but certainly not boring**, *Theoret. Comput. Sci. 249–2* (2000) 343–355.

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.

Esko Ukkonen, **On-line construction of suffix trees**, *Algorithmica 14* (1995) 249–260.

Peter Weiner, **Linear pattern matching algorithms**, *Proc. 14th Annual IEEE Symp. Switching & Automata Theory* (1973) 1–11.

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