CS 860 — Winter 2019

Patterns in Words

Course Project

1 Goals

To obtain credit for CS 860, you are required to do a final project. This can be either (a) original research on a topic related to the course material (see, for example, the list of open problems on the course home page) or (b) reading papers on a topic related to the course or (c) creating some useful public domain software on a topic of the course or (d) writing or rewriting existing Wikipedia pages on the topic of the course.

For choices (a), (b), and (c) you should create a 5-15 page written report and also present your work in class in a 20-30 minute presentation. Some possible topics are listed below. You may choose a topic not listed, but please check with me first. You should not do the project on a topic that you have already studied. Doing the project on a topic which you will, in the future, do more work on (such as a thesis or essay) is fine. The last few lectures are reserved for these presentations.

By Thursday, February 7, 2019 you should hand in a 1-page sheet of paper describing which topic you’ve chosen and listing a couple of references you’ve found on the subject. I’m going to cover some aspects of these topics in class, so you should check with me once you’ve made your decision, so we don’t end up covering the same material.

Your written report should be handed in on the last day of classes, April 9 2019.

Part of the goal of this project is for you to learn something about how to do research. The first step of your exploration is to learn what has already been done on the problem. I sometimes suggest a paper or two to get started, but you should not be content with just this. Trace back the citations in the listed paper to find out about earlier work. You should use any source of information you can think of. Check your text and the other reference books on the lists. Use the library: the library’s online catalogue, references such as Computing Reviews and Math Reviews, the internet, etc. Talk to people. Also try MathSciNet at

https://mathscinet.ams.org/mathscinet/search.html

and Citeseer at

http://citeseer.ist.psu.edu/index

and DBLP at

https://dblp.uni-trier.de

and Google Scholar at

https://scholar.google.ca .
Trace forward by using the “Web of Science” available on the UW library web site; this lets you find papers that have cited the paper you are interested in. Google Scholar also has this feature.

How much information is enough? If your topic is well-studied and you only bother to look at one source (one paper or book) then you haven’t done enough. (Even if the paper claims to have the ultimate solution, you should explore that for yourself.) On the other hand, if you collect a list of more than 10 papers, then you should probably narrow the field somehow, either by restricting the scope of the topic or the type of solution, or by focusing on the most recent or the most relevant work.

The second step of your exploration is then to make sense of the available results, judging which are most useful.

If you prefer a research-oriented approach, you may concentrate your efforts on trying to solve an open problem; there is a list on the course home page. Such an attempt at original work is NOT required, but I would like to encourage it, and I will mark such attempts based on the efforts, not the results. Attempting original work does not excuse you from doing a literature search, but you should spend less time on the search (just enough to be sure you are attempting something new). Some open problems are given in the course notes, too.

For your presentation: you can choose to prepare something in powerpoint, in TeX using the “beamer” package, or just on the blackboard. A handout is often helpful for people to follow the argument. Start with a general statement of the problem and motivation. Discuss, in a historical way, what has been done on the problem before. Then launch into your main discussion. There is no need to prove every result, but a talk should generally have at least one proof or two. Concentrate more on giving the main ideas than detailed technical results, and avoid case analysis when possible. Give specific examples to help build the intuition of the class members! Conclude by mentioning any open problems you found out about.

You will be marked on depth of understanding displayed (50%), clarity (33%) and presentation details (spelling, arrangement of slides, etc.) (17%).

2 An alternative for the presentation-phobic

If you can’t stand the idea of an in-class presentation, you can gain equivalent credit by creating and/or revising some Wikipedia articles on the topic of the course. We’re looking for articles on the following topics:

- Equations in words
- Dejean’s conjecture
- Combinatorics on words – the current article is a real mess and needs some drastic rewriting
- Critical exponent of a word
- Squarefree word
• Sturmian word
• Lyndon word
• Fibonacci word
• Partial word
• Periodicity
• Rich word
• Subword complexity
• Quasiperiodicity
• Primitive word
• Kolakoski word
• Lovász local lemma
• Thue-Morse word

You can also suggest articles not on the list above.

You should choose two or three of these, and print the current version and your revision
and hand in both. You will be marked on depth (30%), clarity (25%), comprehensiveness
(20%), and writing quality (25%).

3 Project suggestions

Here are some suggestions for projects.

3.1 The Oldenburger-Kolakoski word

This mysterious word is the unique sequence of 1’s and 2’s, starting with 12, that is its own
run-length encoding. There are many articles on it. To get started, look at Johan Nilsson, “A
Space-Efficient Algorithm for Calculating the Digit Distribution in the Kolakoski Sequence”,
*J. Integer Sequences* **15** (2012), Article 12.6.7 and

3.2 Permutation-containing words

These are words, that contain, either as subsequences or subwords, all permutations of
gregegan.net/SCIENCE/Superpermutations/Superpermutations.html.
3.3 Duval’s conjecture
Duval’s conjecture relates periods of words to their longest unbordered factor, and was only recently proven. See [Harju and Nowotka 2007], [Holub 2005], and [Holub and Nowotka 2012].

3.4 De Bruijn words
A de Bruijn word is a (circular) string of length $2^n$ that contains all words of length $n$ as factors. There are dozens of papers on properties of such words, enumerating them, and generating the efficiently. We will cover some things in class. Here are a few papers to get you started. [Fredricksen 1982] [Moreno 2005] [Iványi 1988]

3.5 The curling number conjecture
Starting with a sequence of integers like 2121, append the largest integer power over all suffixes. So 2121 becomes 21212. Continue this process: 212122231. The curling number conjecture is that you eventually reach 1. See, for example, Chaffin, Linderman, Sloane, and Wilks, On Curling Numbers of Integer Sequences Journal of Integer Sequences, Vol. 16 (2013), Article 13.4.3.

3.6 Minimal density in power-free sequences
Given a squarefree sequence, what is the minimum density of a letter? Of course, the answer depends on the alphabet size $k$. The only interesting case is $k = 3$. Khalyavin proved that the answer is 883/3215; see [Khalyavin 2007]. Also see [Kolpakov and Kucherov 1997].

3.7 Higher dimensions
Can we avoid squares in higher dimensions?
Yes – see [Carpi 1988]. Other people have worked on the problem without realizing that Carpi had already solved it; see, for example, [Currie and Simpson 2002].

3.8 Test sets for squarefreeness, overlapfreeness, etc.
In this area we are trying to find a finite set of strings $S$ such that if a morphism $h$ is squarefree on elements of $S$, then it is squarefree everywhere.
For some recent results, look at [Richomme and Wlazinski 2004, Richomme and Wlazinski 2007].

3.9 Critical exponents
In this area we are given an infinite word represented in some fashion (typically, as the fixed point of a morphism) and we want to know what the exponent of the largest power occurring in it is.
The Ph. D. thesis of Dalia Krieger has many results in this area.
3.10 Correlations of strings
In class, lecture 2, we discussed autocorrelations. However, there is a more general notion of correlations of two strings, and you can use this to prove the result we mentioned (but did not prove) about which of two patterns is likely to occur first. See, for example, Guibas and Odlyzko, String overlaps, pattern matching, and nontransitive games, *J. Combin. Theory* **30** (1981), 183–208.

3.11 Gray codes
A Gray code for a set of length-$n$ words is an enumeration of a set in a particular order that minimizes the total number of bit flips required. Study this for a variety of different sets. For example, see M. Squire, Gray codes for $A$-free strings, *Elect. J. Combinatorics* **3** (1996), #R17.

3.12 Patterns in permutations
Avoiding certain patterns in permutations is now an active subject, focusing particularly on counting permutations avoiding certain patterns. An entire issue of the *Electronic Journal of Combinatorics* was devoted to this problem; see [http://www.combinatorics.org/Volume_9/v9i2toc.html](http://www.combinatorics.org/Volume_9/v9i2toc.html).

One of the first papers in the areas was [Simon and Schmidt 1985]. For algorithmic considerations, see Bose, Buss, and Lubiw [Bose, Buss, and Lubiw 1998].


3.13 $k$-binomial equivalence
Two words are $k$-binomially equivalent whenever the same subsequences of length $\leq k$ occur with the same multiplicities. Study this concept. You can start, for example, with a recent preprint of Lejeune, Leroy, and Rigo here: [https://arxiv.org/abs/1812.07330](https://arxiv.org/abs/1812.07330).

3.14 Two-dimensional repetitions
Study repetitions in two-dimensional arrays. For example, there is a recent paper by Amir et al., Two-dimensional maximal repetitions, in *26th Annual European Symposium on Algorithms*, 2018 LIPICS, Article No. 2, pp. 2:1–2:14.

3.15 Formal language theory and repetitions
In this area we are interested in questions like: is the set of words avoiding squares a regular or context-free language? Section 4.5 of my book, *A Second Course in Formal Languages and Automata Theory*, has some basic results in this area and is a good place to start. For a recent result, see [Rampersad 2007].
3.16 Applications to biology


3.17 Applications to cryptography

See the papers [Rivest 2005] and [Kortelainen 2011]. Also the followup papers by Andreeva et al. and Bouillaguet et al.

3.18 Applications to music

Some modern composers, such as Paris-based composer Tom Johnson, French composer Marcel Frémiot, and Danish composer Per Nørgård, use sequences avoiding repetitions to compose music. See, for example,

http://www.editions75.com/Books/TheVoiceOfNewMusic.PDF
http://www.pernoergaard.dk/eng/indhold.html
[Au, Drexler-Lemire, and Shallit 2017]

3.19 Partial words

A partial word is a word with a “don’t care” symbol that matches every other symbol of the alphabet. One can explore the notion of avoidable pattern in this setting, where a certain number of don’t care symbols are allowed. See the papers by Blanchet-Sadri and co-authors, for example, [Blanchet-Sadri, Choi, and Mercas 2011].

3.20 Parallel algorithms

We’re going to discuss some efficient algorithms to detect repetitions in words in this course, but not parallel algorithms. Look into these. You can start with [Apostolico 1992].

3.21 Connections with logic

Look into some of the connections of repetition-free words with various problems in logic. For example, you can look at [Stolboushkin and Taitslin 1983], [Urzyczyn 1983], [Stolboushkin 1983], and [Kfoury 1985].
3.22 The runs conjecture

A run is a maximal subword that is of exponent at least 2. A longstanding conjecture was that a word of length $n$ contains at most $n$ runs. This was finally proven by [Bannai, I, Inenaga, Nakashima, Takeda, and Tsuruta 2017].

3.23 Anti-powers in words

A $k$th anti-power in a word is a block of $k$ consecutive distinct words of some length $t$. Study this concept. Start with Fici et al., Anti-powers in infinite words, J. Combin. Theory Ser. A 157 (2018), 109–119.

They define $N(k, r)$ to be the smallest integer $N$ such that every binary string of length $\geq N$ has either a $k$-power or an $r$-antipower. (An $r$-antipower is $r$ consecutive blocks of the same length that are distinct.) What is the asymptotic growth rate of $N(k, r)$?

Numerical evidence suggests $N(k, 3) = 2k$ for $k \geq 7$ and $N(k, 4) = 4k$ for $k \geq 11$. And $N(k, 5) = 6k + 4$ for $k \geq 10$. But apparently no proofs of these yet! Furthermore, I conjecture that for each $r \geq 2$ a relation of the form $N(k, r) = (2r - 4)k + O(1)$ holds, where the constant in the $O(1)$ term can depend on $r$.

3.24 Efficient generation of strings with forbidden patterns

Given a string $w$ over a $k$-letter alphabet, we’d like to efficiently generate all the length-$n$ strings avoiding $w$. Read, for example, Ruskey and Sawada, Generating necklaces and strings with forbidden substrings, in COCOON 2000, Lect. Notes in Comput. Sci. 1858 (2000), 330–339.

3.25 Uniform words

A word $w$ over an alphabet $\Sigma$ is said to be uniform if for every pair of words $u$ and $v$ of the same length, the number of occurrences of $w$ in $u$ and $v$ differ by at most 1. Study this concept. See, for example, Carpi and de Luca, Uniform words, Advances in Applied Math. 32 (2004), 485–522. There are also more recent papers by Kamae, who apparently did not know of the Carpi-de Luca result.

3.26 Prefix-normal words

Study prefix-normal words: these are words where no factor has more 1’s than the prefix of the same length. See, for example, P. Burcsi et al., On prefix normal words and prefix normal forms, Theoret. Comput. Sci. 659 (2017), 1–13.

3.27 Computing the longest unbordered factor

Study algorithms for computing the longest unbordered factor of a word. See, for example, P. Gawrychowski et al., Computing the longest unbordered substring, SPIRE 2015, Lect.
3.28 Shelton and Soni’s work on squarefree words

Shelton and Soni, in a series of three papers, proved the following beautiful claim: there is a fixed constant $c$ such that if a ternary squarefree word $w$ has a right-extension $wx$ that is squarefree and $|x| \geq c|w|^{3/2}$, then $w$ is the prefix of an infinite squarefree word. Read the papers and determine the value of $c$. Shelton and Soni did their work at a time when computers were much slower than today, so it should be possible to verify their calculations relatively quickly with a good program. Can you prove a similar result for binary cubefree words?

See [Shelton 1981a, Shelton 1981b, Shelton and Soni 1982].

4 Suggestions for software projects

• Implement the algorithm of Currie and Rampersad for determining whether a morphism generates an abelian $n$’th power-free word. See [Currie and Rampersad 2012]. There is also a recent improvement, due to Rao and Rosenfeld [Rao and Rosenfeld 2016]

• Implement the linear-time algorithm for testing whether a word is rich, due to [Groult, Prieur, and Richomme 2010]

References

[Apostolico 1992]

[Au, Drexler-Lemire, and Shallit 2017]

[Bannai, I, Inenaga, Nakashima, Takeda, and Tsuruta 2017]

[Blanchet-Sadri, Choi, and Mercas 2011]

[Bose, Buss, and Lubiw 1998]
[Carpi 1988]

[Currie and Simpson 2002]

[Currie and Rampersad 2012]

[Fredricksen 1982]

[Groult, Prieur, and Richomme 2010]

[Harju and Nowotka 2007]

[Holub 2005]

[Holub and Nowotka 2012]

[Iványi 1988]

[Kfoury 1985]

[Khalyavin 2007]

[Kolpakov and Kucherov 1997]
R. Kolpakov and G. Kucherov. Minimal letter frequency in \( n \)-th power-free binary words.

[Kortelainen 2011]

[Moreno 2005]

[Rampersad 2007]

[Rao and Rosenfeld 2016]

[Richomme and Wlazinski 2004]

[Richomme and Wlazinski 2007]

[Rivest 2005]

[Shelton 1981a]

[Shelton 1981b]

[Shelton and Soni 1982]
[Simon and Schmidt 1985]

[Stolboushkin 1983]

[Stolboushkin and Taitslin 1983]

[Urzyczyn 1983]