THE COMPLEXITY OF UNAVOIDABLE WORD PATTERNS

PAUL SAUER

University of South Africa
e-mail address: paul.v.sauer@gmail.com

Abstract. The avoidability, or unavoidability of patterns in words over finite alphabets has been studied extensively. The word α over a finite set A is said to be unavoidable for an infinite set B^+ of nonempty words over a finite set B if, for all but finitely many elements w of B^+, there exists a semigroup morphism $\phi : A^+ \to B^+$ such that $\phi(\alpha)$ is a factor of w. We present various complexity-related properties of unavoidable words. For words that are unavoidable, we provide an upper bound to the lengths of words that avoid them. In particular, for a pattern α of length n over an alphabet of size r, we give a concrete function $N(n, r)$ such that no word of length $N(n, r)$ over the alphabet of size r avoids α.

A natural subsequent question is how many unavoidable words there are. We show that the fraction of words that are unavoidable drops exponentially fast in the length of the word. This allows us to calculate an upper bound on the number of unavoidable patterns for any given finite alphabet.

Subsequently, we investigate computational aspects of unavoidable words. In particular, we exhibit concrete algorithms for determining whether a word is unavoidable. We also prove results on the computational complexity of the problem of determining whether a given word is unavoidable.

1. Introduction

Let $\mathbb{N}$ denote the nonnegative integers. If A is a finite set, we write A^* for the set $\{a_1a_2 \dots a_n \mid a_i \in A \text{ and } n \in \mathbb{N}\}$ of *words* over A, while A^+ is the subset of all nonempty words in A^*. For $n \in \mathbb{N}$ we symbolize the set of words of length n over A by A^n. Here the *length* of a word is defined in the conventional sense: if $w \in A^*$ and $w = a_1a_2 \dots a_n$ with each $a_i \in A$, then the length $|w|$ of w is n. The set A above is sometimes called an *alphabet* and its members are called *letters*. We say that the word $v = a_1a_2 \dots a_m$ is a *factor* of the word $w = b_1b_2 \dots b_n$ if there is an i such that, for $1 \leq j \leq m$, we have $a_j = b_{i_0+j}$.

For a word w and letters $x_1, x_2, \dots, x_k$, we denote by $w^{x_1,x_2,\dots,x_k}$ the word derived from w by deleting all occurrences of each of the x_i.

We say that a word w over a finite alphabet B *reflects* a word α (or a *pattern* α, for the sake of clarity) over a finite alphabet A whenever there is a semigroup morphism $\phi : A^+ \to B^+$ such that $\phi(\alpha)$ is a factor of w. The pattern α is called *unavoidable* for a set X of words over a finite alphabet if all but finitely many $w \in X$ reflect α. The pattern

Key words and phrases: regularity, avoidability, unavoidablility.

LOGICAL METHODS
IN COMPUTER SCIENCE

DOI:10.2168/LMCS-???

α is simply called unavoidable if the preceding statement holds for every set over a finite alphabet. Otherwise α is called *avoidable*.

The study of combinatorial patterns is one of the most repeated themes in Mathematics [5], [8]. Among these studies, the unavoidability of patterns in words over finite alphabets has been explored extensively. Over the last century, this theme has resurfaced repeatedly [15], [9], [1], [16], [10], [14]. In the last decade, there has been a resurgence in the investigation of unavoidability [13]. Thue [15] proved that xxx is avoidable on the binary alphabet and xx is avoidable on the alphabet of size 3. Bean et al. [1] conducted an extensive investigation into the avoidability of patterns. One central discovery of this investigation is the notion of a letter that is *free* for a pattern.

Definition 1.1. Let A be a finite alphabet and let $w \in A^+$. A letter $x \in A$ is free for w if x occurs in w and there is no integer $n > 0$ and $a_1, a_2, \ldots, a_n, b_1, b_2, \ldots, b_n$ such that

$$\begin{gathered} xa_1 \\ b_1a_1 \\ b_1a_2 \\ b_2a_2 \\ \vdots \\ b_nx \end{gathered}$$

are all factors of w.

Free letters are connected to the phenomenon of unavoidability be the following lemma, whose proof appears in [1].

Lemma 1.2. Suppose α is a pattern with a free letter x. If α^x is unavoidable, then so α.

A surprising, complete characterization of unavoidable patterns follows from Lemma 1.2. This is commonly known as the Bean, Ehrenfeucht and McNulty (B.E.M.) Theorem.

B.E.M. Theorem 1.3. A pattern α is unavoidable if and only if it is reducible to the empty word by iteratively performing one of the following operations on the pattern:

(1) deleting every occurrence of a free letter, or
(2) replacing all occurrences of some letter x occurring in α by a different letter y, also occurring in α.

We refer to the second operation as the *identification of letters*. We can extend the definition of free letters to *free sets*.

Definition 1.4. Let A be a finite alphabet and let $w \in A^+$. A set $X \subseteq A$ is free for w if, for every pair of letters $x, y \in X$, there is no $n > 0$ and $a_1, a_2, \ldots, a_n, b_1, b_2, \ldots, b_n$ such that

$$\begin{gathered} xa_1 \\ b_1a_1 \\ b_1a_2 \\ b_2a_2 \\ \vdots \\ b_ny \end{gathered}$$

are all factors of w.

The notion of free sets allows us to reformulate the B.E.M. Theorem in a way that is sometimes more convenient for reasoning about patterns.

Theorem 1.5. A pattern is unavoidable if and only if it is reducible to the empty word by iteratively deleting free sets.

This reformulation is due to Sapir [12]. See also Zimin [16]. The proof of Theorem 1.3, presented in [1], is not constructive. Therefore it gives no indication, for any given pattern, what the longest word avoiding that pattern might be. Subsequent to [1], one constructive unavoidability result was established, pertaining to the subset of patterns that represent permutations. We now discuss this result briefly.

Let $[n]$ denote the set $\{1, 2, \ldots, n\}$ and let S_n be the set of all permutations of $[n]$. We use one-line notation to express a permutation $\pi \in S_n$ – that is we write $x_1x_2 \ldots x_n$ when $\pi(i) = x_i$ for $i \in [n]$. The write $\langle \pi \rangle$ for the word $12 \ldots nx_0x_1 \ldots x_n$, where x_0 is a symbol not in $[n]$. Fouché [6] discovered the following

Theorem 1.6. For $n, r \in \mathbb{N}$ there is an $N = N(n, r) \in \mathbb{N}$ such that every $w \in [r]^N$ reflects every $\langle \pi \rangle$, where $\pi \in S_n$. Specifically, the numbers $N(n, r)$ are inductively bounded from above by

$$N(n+1, r+1) \leq 2(n+1)N(n+1, r)N(n, (2n+2)^2 r^{N(n+1,r)})$$

In the sequel, we show that a similar bound holds for all unavoidable patterns. The proof of the Main Theorem 2.3 follows Fouché's reasoning. Subsequent sections are organized as follows:

In Section 3, we investigate the density of unavoidable patterns in the space of all patterns. We establish that this density drops quite fast as the length of the pattern increases. This fact then provides a way to calculate an upper bound for the number of unavoidable patterns as function of the size of the underlying alphabet.

Section 4 is devoted to the algorithmic decision problem of whether a letter appearing in a given pattern is free. We present a concrete algorithm running in polynomial time. In Section 5, we show that there is a simple reduction from boolean formulas to patterns that maps satisfiable formulas to unavoidable patterns and unsatisfiable formulas to avoidable patterns. The final substantial part of the paper is Section 6, where we prove that the the problem of deciding whether a pattern is unavoidable is NP-complete.

2. General Bounds for Unavoidable Patterns

The main result of this section is Theorem 2.3, which provides an upper bound on the length of words that can avoid a given, unavoidable pattern. In order to establish Theorem 2.3, we first need to establish a few facts. Lemma 2.1 below gives us a method for building morphisms as the size of our alphabet increases, provided that there is a free letter in the pattern. This lemma, stated here without proof, is proved in [1].

Lemma 2.1. Let A and B be a finite alphabets and let w be a word over A. Suppose x is free for w. If there is a morphism $\phi : w^x \mapsto v$, where $v \in B^+$ is of the form $a^{i_1}X_1a^{i_2}X_2 \dots a^{i_t}X_ta^{i_{t+1}}$, each X_i being a word over $B \setminus \{a\}$, then there is a morphism $\psi : w \mapsto v$.

Since every letter in a free set is free, the following Lemma follows immediately from Theorem 1.5. This will be used in conjunction with Lemma 2.1 to build morphisms in the proof of the main result below.

Lemma 2.2. Every unavoidable pattern has a free letter.

We are now ready to prove our main result. The construction of the proof closely follows [6].

Main Theorem 2.3. For $n, r \in \mathbb{N}$ there is an $N = N(n, r) \in \mathbb{N}$ such that every $w \in [r]^N$ reflects every unavoidable pattern of length n over $[r]$. The minimal values for the numbers $N(n, r)$ are bounded from above by

$$N(n+1, r+1) \leq (n+1)N(n+1, r)N(n, (n+1)^2 r^{N(n+1,r)})$$

Proof. It is easy to see that $N(1, r) = r + 1$ and $N(1, n) = n + 1$. From here we proceed by induction to establish the stated bound. Suppose our result holds for some n and all r, as well as for $n + 1$ and some $r \geq 1$.

Let w be a word of length $(n + 1)KL$ over an alphabet A of size $r + 1$, where $K = N(n + 1, r)$ and $L = N(n, (n + 1)^2 r^{N(n+1,r)}$. We may assume that every factor of length K in w contains every letter in A, for otherwise w reflects every unavoidable pattern of length $n + 1$, by our inductive hypothesis. Consequently, the word w is of the form $a^{i_1}X_1a^{i_2}X_2 \dots a^{i_t}X_ta^{i_{t+1}}$, where each $X_i \in \{A \setminus \{a\}\}^+$ satisfies $|X_i| < K$. We may assume that $1 \leq a_{i_j} \leq n$, for otherwise the morphism $f(x) = a$ that sends every letter to a shows that every pattern of length $n + 1$ is reflected by w.

We immediately have

$$\begin{aligned}(n+1)KL = |w| &\leq (K-1)t + (t+1)(n+1) \\ &= (K+n)t + n + 1 \\ &\leq (n+1)Kt + 1\end{aligned}$$

since $K > 2$ is readily available from the definition of K. Therefore we have $t > L$ and hence w has a factor $v = a^{i_1}X_1a^{i_2}X_2 \dots a^{i_L}X_L^{i_{L+1}}$, where each $X_i \in \{A \setminus \{a\}\}^+$ satisfies $|X_i| < K$.

Define the alphabet B as the set of words of the form a^iX, with $1 \leq i \leq n$ and $X \in \{A \setminus \{a\}\}^+$ satisfies $|X_i| < K$.

$$\begin{aligned}|B| =& n(r + r^2 + \cdots + r^{K-1})\\ \leq& (n+1)^2 r^K\end{aligned}$$

since $K \geq n+1$ for every n and r.

We have v is a word of length L over B. Suppose that α is any unavoidable pattern of length $n+1$ over A. Using Lemma 2.2 there is a letter $x \in A$ that is free for α. We remind ourselves that $L = N(n, (n+1)^2 r^{N(n+1,r)})$ and note, by our inductive hypothesis, that there is thus a morphism $\phi : \alpha^x \mapsto v$. Consequently, Lemma 2.1 yields that there is a morphism $\psi : \alpha \mapsto v$ and the proof is complete. □

3. Density and Counting Unavoidable Patterns

A natural subsequent question is how many unavoidable words there are. We start by showing that, for alphabets of 3 or more letters, the fraction of words that are unavoidable drops exponentially fast in the length of the word.

Lemma 3.1. Let $r > 2$ and $n > 0$. Let $p_{r,n}$ be the probability that a pattern of length n is unavoidable over [r]. We have $p_{r,n} \leq \left(\frac{r-1}{r}\right)^{n-1}$.

Proof. Let w be a word of length n over r. If $n = 1$ then w is unavoidable, so that our claim holds with $p_{r,1} = (\frac{r-1}{r})^0 = 1$. Now suppose $n > 1$. We will use the fact that xx is avoidable, established in [15]. Let $V = \{w \in [r]^n : x \in [r] \text{ and } xx \text{ is a factor of } w\}$. First we claim that every element of V is avoidable. To prove our claim, we start by noting that x is not free for any $v \in [r]^*$ that has xx as a factor. Hence any sequence of deletions of free letters applied to w results in a word that has xx as a factor. Using Theorem 1.3, our claim is proved. Let $U_{n,r}$ be the set of all unavoidable words of length n over r. By our claim above, we have $U \subseteq \bar{V} = [r]^n \setminus V$. Now we count the elements of $\bar{V}$. Let $w = w_1 w_2 \dots w_n$ be an abstract word of length n over r. For w_1 we can choose any one of the r letters in $[r]$. For each subsequent w_i, we can choose any letter from $[r]$, other than our choice of w_{i-1}. Hence $|\bar{V}| = r(r-1)^{n-1}$. It follows that $|U| \leq r(r-1)^{n-1}$ and therefore $p_{r,n} \leq \frac{r(r-1)^{n-1}}{r^n} = \left(\frac{r-1}{r}\right)^{n-1}$. □

We also know from [1] that all unavoidable patterns over $[r]$ have length less than 2^n. Combined with Lemma 3.1 above, we can now obtain an upper bound on the number of unavoidable patterns over $[r]$, where $r > 2$.

Proposition 3.2. Let $r > 2$. The number of unavoidable patters over $[r]$ is at most $r\left(\frac{(r-1)^{2^r-1}-1}{r-2}\right)$.

Proof. The number of unavoidable patterns of length n is bounded from above by

$$p_{r,n} r^n \leq \left(\frac{r-1}{r}\right)^{n-1} r^n = r(r-1)^{n-1}.$$

Since there are no unavoidable patterns of length greater than $2^n - 1$ we have the total number of unavoidable patterns is at most

$$\sum_{i=1}^{2^r-1} r(r-1)^{i-1} = r \sum_{i=0}^{2^r-2} (r-1)^i = r\left(\frac{(r-1)^{2^r-1}-1}{r-2}\right)$$

and the proof is complete. □

4. Free Letters and Computation

We now proceed to investigate the computational aspects of unavoidability, assuming a basic familiarity with algorithms and computational complexity, for which Hopcroft and Ullman [7] and [3] provide authoritative references. The computational complexity of patterns has been the subject of significant study. Rytter and Shur [11] demonstrated that the problem of finding whether a pattern is reflected in a given string is NP-complete. In the same article, they mention that the problem of determining whether a pattern is unavoidable has, at face value, properties that many other NP-complete problems have. Below, we show that their suspicions are correct. The complexity of unavoidable words in the sense of substrings, not morphisms, has also been investigated [2].

For a pattern α we construct a directed bipartite graph G_α, which we call the *graph of* α. The vertex set $V(G_\alpha)$ of G_α has two nodes ${}^0ab$ and ${}^1ab$ for each 2-factor ab of α. The pair of 2-factors $({}^0ab, {}^1cd)$ of α is an edge of G_α whenever $b = d$. Similarly, the pair $({}^1ab, {}^0cd)$ of α is an edge of G_α whenever $a = c$. The reason why we create two vertices for each 2-factor is to prevent paths of the form xa, xb, xc.

Lemma 4.1. Let α be a pattern. A letter x of α is not free if and only if there is a path in G_α from a node having x as its first component to a node having x as its second component.

Proof. If x is not free for α, then there is an $n \in \mathbb{N}$ and $a_1, a_2, \ldots, a_n, b_1, b_2, \ldots, b_n$ such that

$$\begin{array}{c} xa_1 \\ b_1a_1 \\ b_1a_2 \\ b_2a_2 \\ \vdots \\ b_nx \end{array}$$

are all factors of w. It is clear from the definition of G_α that the edges

$$\begin{gathered}({}^0xa_1, {}^1b_1a_1)\\ ({}^1b_1a_1, {}^0b_1a_2)\\ ({}^0b_1a_2, {}^1b_2a_2)\\ ({}^1b_2a_2, {}^0b_2a_3)\\ \vdots\\ ({}^0b_na_{n-1}, {}^1b_nx)\end{gathered}$$

all exist in G_α. Therefore a path from ${}^0xa_1$ to ${}^1b_nx$ exists in G_α, as desired.

Proving the converse is essentially the same as reading the construction above in reverse. □

Given Lemma 4.1, we can easily construct an efficient algorithm that decides, given a pattern α and a letter x appearing in α, whether x is free for α.

Firstly, the construction of the adjacency matrix of G_α from α can be defined as follows:

```
def BUILD_G(n, alpha):
    G = [[0 for x in range(2*n-2)] for y in range(2*n-2)]
    V =  [[[0 for x in range(2)] for y in range(2)] for z in range(n-1)]
    for i in range(n-1):
        V[i][0][0] = V[i][1][0] = alpha[i]
        V[i][0][1] = V[i][1][1] = alpha[i+1]
    for i in range(n-1):
        for j in range(n-1):
           if V[i][0][0] == V[j][1][0]:
               G[i + n - 1][j] = 1
           if V[i][0][1] == V[j][1][1]:
               G[i][j + n - 1] = 1
    return (V, G)
```

We notice that the runtime is dominated by the nested for loop and therefore requires $O(n^2)$ computational steps. The subroutine as it is written is not quite optimal since multiple vertices are created if the same 2-factor is repeated. This impacts the time complexity only to a multiplicative constant and simplifies the description.

Now, given a graph $G = G_\alpha$ and a letter x, we can use a standard depth-first search algorithm to detect if x is not free. For simplicity we write a standard depth-first search subroutine.

```
def DFS(n, G, V, i, p, x, is_seen):
    is_seen[i][p] = True
    if p == 0:
        q = 1
    else:
        q = 0
    for j in range(n-1):
        if not is_seen[j][q] and
          (
          (q == 0 and G[i + n - 1][j] == 1) or
          (q == 1 and G[i][j  + n - 1]  == 1)
           ):
            if V[j][q][1] == x:
                return True
            else:
                if DFS(n, G, V, j, q, x, is_seen):
                   return True
    return False
```

We are now ready to write the subroutine determining if x is free for α.

```
def IS_FREE(alpha, x):
    if x not in alpha:
       return False
    n = len(alpha)
    V,G = BUILD_G(n, alpha)
    is_seen = [[False for i in range(n)] for j in range(n)]
    for i in range(n-1):
        is_seen = [[False for k in range(n)] for j in range(n)]
        if V[i][0][0] == x:
            if not is_seen[i][0]:
                if DFS(n, G, V, i, 0, x, is_seen):
                    return(False)
    return(True)
```

The subroutine `IS_FREE` requires $O(n^2)$ computational steps, where $n = |\alpha|$: We already know that `BUILD_G` is $O(n^2)$. In subsequent steps, `DFS` is called at most n times since every vertex is marked as seen subsequent to the invocation of `DFS`. At every invocation of `DFS`, at most n neighbors of a vertex are examined.

Let us pause for a moment to remember where we started and what we have seen along the way. Our initial definition of unavoidability sounds distinctly non-finitary: A pattern must be reflected by all but finitely many elements for every set over any finite alphabet.

Theorem 1.5 then gives us a finitary characterization of unavoidability in that we only need to look for a sequence of deletions of free letters. Most recently we have seen, in addition, that the problem of deciding whether a letter is free falls in Polynomial Time. It is hence starting to look as though the problem of determining whether a pattern is free might fall in NP: We can nondeterministically guess the sequence of deletions and verify the validity of the guess (each deletion being of a free letter) in polynomial time. We may also ask how hard this problem is, relative to other problems in NP. In the following two sections, we explore this.

5. Unavoidability and Logic

We work to establish a natural correspondence between boolean formulas and patterns. In particular, we show that given a boolean formula, we can construct a word whose unavoidability coincides with the satisfiability of of the formula. We will restrict our construction to 3-CNF boolean formulas, as the correspondence between this subset of boolean formulas and the set of all boolean formulas is well-understood (see [7]).

Let ϕ be any 3-CNF boolean formula. We construct α_ϕ, the *word of* ϕ, as follows: Suppose ϕ has n variables $x_1, x_2, \ldots, x_n$. Without loss of generality $\phi = C_1 \wedge C_2 \wedge \cdots \wedge C_m$, where each C_i is a clause of the form $(p_{i1} \vee p_{i2} \vee p_{i3})$, each p_{ij} being either a variable x_k, or its negation $\overline{x}_k$. We may also assume that any negated variables occur after any non-negated variables in each clause. We start by defining the letters in α_ϕ. These letters will fall into the following four categories:

(1) The set $X_{\alpha_\phi} = \{x_i, \overline{x}_i, : i \leq n\}$
(2) The set $Y_{\alpha_\phi} = \{a_j, b_j, c_j, d_j : j < m\}$
(3) The letter e
(4) The set $Z = \{z_i : i \leq M\}$. We choose M to be sufficiently large so that every element of this set will appear exactly once in α.

The elements of Z above are used as"separator" letters to prevent unfortunate 2-factors from occurring. We adopt the convention that we will use each letter in Z once and denote each occurrence of a letter from Z in α_ϕ by z_+. We denote the union of the sets of letters itemized above by A_α.

For each variable x_i, we create the factor

$$ex_i\overline{x}_iez_+$$

For each clause C_j in ϕ we construct a factor δ_j as the concatenation of the following factors.

Let x, y and z be variables in ϕ. If C_j is of the form $x \vee y \vee z$ we add the following factors to α:

$$a_j x z_+$$
$$b_j x z_+$$
$$b_j y z_+$$
$$c_j y z_+$$
$$c_j z z_+$$
$$d_j z z_+$$
$$d_j a_j z_+$$

$$a_j b_j z_+$$
$$a_j c_j z_+$$
$$a_j d_j z_+$$

$$a_j e z_+$$

If C_j is of the form $x \vee y \vee \overline{z}$ we add the following factors to α:

$$a_j x z_+$$
$$b_j x z_+$$
$$b_j y z_+$$
$$c_j y z_+$$
$$c_j d_j z_+$$
$$\overline{z} d_j z_+$$
$$\overline{z} a_j z_+$$

$$a_j b_j z_+$$
$$a_j c_j z_+$$
$$d_j a_j z_+$$

$$a_j e z_+$$

If C_j is of the form $x \vee \overline{y} \vee \overline{z}$ we add the following factors to α:

$$a_j x z_+$$
$$b_j x z_+$$
$$b_j c_j z_+$$
$$\overline{y} c_j z_+$$
$$\overline{y} d_j z_+$$
$$\overline{z} d_j z_+$$
$$\overline{z} a_j z_+$$

$$a_j b_j z_+$$
$$c_j a_j z_+$$
$$d_j a_j z_+$$

$$a_j e z_+$$

If C_j is of the form $\overline{x} \vee \overline{y} \vee \overline{z}$ we add the following factors to α:

$$a_j b_j z_+$$
$$\overline{x} b_j z_+$$
$$\overline{x} c_j z_+$$
$$\overline{y} c_j z_+$$
$$\overline{y} d_j z_+$$
$$\overline{z} d_j z_+$$
$$\overline{z} a_j z_+$$

$$b_j a_j z_+$$
$$c_j a_j z_+$$
$$d_j a_j z_+$$

$$e a_j z_+$$

We define the word α_ϕ of ϕ as the culmination of the above construction and proceed to prove some properties of α_ϕ.

Lemma 5.1. Let $\phi = C_1 \wedge C_2 \wedge \cdots \wedge C_m$ be a 3-CNF boolean formula. Let $B \subset A_\alpha \setminus \{a_j, b_j, c_j, d_j\}$ be such that, if p_i is a literal in C_j, then the letter p_i is not in B. No letter in $\{a_j, b_j, c_j, d_j\}$ is free for α_ϕ^B.

Proof. If C_j is of the form $x \vee y \vee z$ then the path

$$a_jx$$
$$b_jx$$
$$b_jy$$
$$c_jy$$
$$c_jz$$
$$d_jz$$
$$d_ja_j$$

shows a_j is not free. Similarly the path

$$b_jx$$
$$a_jx$$
$$a_jb_j$$

yields that b_j is not free, while

$$c_jy$$
$$b_jy$$
$$b_jx$$
$$a_jx$$
$$a_jc_j$$

and

$$d_jz$$
$$c_jz$$
$$c_jy$$
$$b_jy$$
$$b_jx$$
$$a_jx$$
$$a_jd_j$$

demonstrate that c_j and d_j are not free. The arguments for the remaining three cases where C_j contains negated variables are substantially similar. □

The following lemma is easily established by inspecting α_ϕ.

Lemma 5.2. Let $\phi = C_1 \wedge C_2 \wedge \cdots \wedge C_m$ be a 3-CNF boolean formula and let α_ϕ be the word of ϕ. For $i \leq m$ the letters b_j, c_j and d_j are free for $\alpha_\phi^{a_j}$

Lemma 5.3. Let $\phi = C_1 \wedge C_2 \wedge \cdots \wedge C_m$ be a 3-CNF boolean formula. Let $B = \{y_1, y_2, \ldots, y_k\} \subseteq A_\alpha \backslash \{e\}$. Suppose $y_1, y_2, \ldots, y_k$ is a free deletion sequence for α_ϕ. Suppose furthermore that B is such that, if p_i is a literal in C_j, then the letter p_i is not in B. Then e is not free for α_ϕ^B.

Proof. Suppose C_j and B are as in the statement of the Lemma. Since $y_1, y2, \ldots, y_k$ is a free deletion sequence, Lemma 5.1 gives us that none of the letters a_j, b_j, c_j and d_j are in B.

If x_i is the first literal in C_j, then the path

$$ex_i$$
$$a_j x_i$$
$$a_j e$$

ensures that e is not free.

On the other hand, if $\overline{x_i}$ is the first literal in C_j, then we know (using our assumption that negated variables always appear after non-negated variables in a clause) that C_j is of the form $\overline{x_i} \vee \overline{y} \vee \overline{z}$, where y and z are variables of ϕ and the path

$$ea_j$$
$$\overline{z}a_j$$
$$\overline{z}e$$

shows that e is not free. □

Lemma 5.4. Let $B \subseteq A_\alpha \setminus \{e\}$. If there is an i such that both x_i and $\overline{x}_i$ are in B, then α_ϕ^B is avoidable.

Proof. If x_i and $\overline{x}_i$ are both in B, then $ex_i\overline{x}_i e^B = ee$ is a factor of α_ϕ^B. □

Lemma 5.5. Let w be a word of the form $z_+ z_+ \ldots z_+$. Every letter in w is free.

Proof. Each of the letters z_+ appears at most once in w. □

Lemma 5.6. Let $\phi = C_1 \wedge C_2 \wedge \cdots \wedge C_m$ be a 3-CNF boolean formula in n variables and let $\alpha = \alpha_\phi$ be the word of ϕ. Fix $k < n$. Let $S_k = \{p_1, p_2, \ldots, p_k\}$, where for each i, either $p_i = x_i$ or $p_i = \overline{x}_i$. Both x_{k+1} and $\overline{x}_{k+1}$ are free for $\alpha_\phi^{S_k}$.

Proof. We proceed by induction on k. For $k = 0$, we have $S_k = \emptyset$. Hence, the only 2-factor (excluding those containing z_+) that contains x_1 as the first letter is $x_1\overline{x}_1$ and the only 2-factor that contains $\overline{x}_1$ as the second letter is $x_1\overline{x}_1$. So the only path starting at a 2-factor having x_1 as the first first letter is the one-cycle from $x_1\overline{x}_1$ to itself. Our base case has thus been established.

Now suppose the lemma holds for some k. Again, the only 2-factor containing x_k as the first letter is $x_k\overline{x}_k$ and the only 2-factor that contains $\overline{x}_k$ as the second letter is $x_k\overline{x}_k$. The lemma immediately follows. □

Lemma 5.7. Let ϕ be a 3-CNF boolean formula and let $\alpha = \alpha_\phi$ be the word of ϕ. If α is unavoidable, then there is a free deletion sequence where the letters z_+ are deleted after all other letters are deleted.

Proof. By Lemma 5.5, it suffices to note that deleting any letter z_+ cannot make any letter free that is not already free. □

Lemma 5.8. Let $\phi = C_1 \wedge C_2 \wedge \cdots \wedge C_m$ be a 3-CNF boolean formula and let $\alpha = \alpha_\phi$ be the word of ϕ. If α is unavoidable, then there is a free deletion sequence where every free set that is deleted contains exactly one letter.

Proof. Suppose α is as in the statement of the lemma. There is a partition of A_α into sets $B_1, B_2, \ldots, B_k$ such that, for every $i < k$, we have that B_{i+1} is a free set for $\alpha^{B_1, B_2, \ldots, B_i}$. Assume for contradiction that there is some t such that $|B_t| > 1$ and every deletion sequence of the individual letters in B_t results in no letter $y \in A_\alpha \setminus D \setminus X_{al}$ being free for α^D, where $D = B_1 \cup B_2 \cup \cdots \cup B_{t-1} \cup E$ and $E \subset B_t$ is the set of letters in B_t that are already deleted. Let x be the last letter in E that was deleted and let $y \in B_t \setminus E$ be not free for α^D.
Case 1. $x = x_i$ for some i.

Subcase 1.1. $y = x_j$ for some j. We have that the deletion of x_i resulted in a path from a 2-factor having x_j as its first component to a 2-factor that has x_j as its second component. We observe that the only only 2-factors that can possibly be newly created by the deletion of x_i are among the following forms:

(1) $a_k z_+$, $b_k z_+$, $c_k z_+$ or $d_k z_+$. Since each letter z_+ appears only once in α we can conclude that these factors are not in our path.
(2) $e\overline{x}_i$. We conclude that there is a path from a 2-factor having x_j as its first letter to $e\overline{x}_i$. But this means there is a path in $\alpha^{D \setminus \{x_i\}}$ from a 2-factor having x_j as its first letter to ex_i. Thus x_i and x_j cannot be in the same free set, a contradiction.
(3) ee. Similarly to the previous item, we conclude that x_i and x_j cannot be in the same free set as it implies a path from $x_i e$ to a 2-factor having x_j as its second component before the deletion of x_i.

Subcase 1.2. $y = a_j$, or $y = b_j$, or $y = c_j$, or $y = d_j$ for some j. We follow the same reasoning as Subcase 1.1 and arrive at the same conclusion, showing y not free implies either a path from a 2-factor having y as its first letter to a two factor having x as its second component, or vice versa.

Subcase 1.3. $y = e$. Our reasoning is substantially similar to the previous two subcases.
Case 2. $x = \overline{x}_j$ for some j. This is symmetric to Case 1.
Case 3. $y = a_j$, $y = b_j$, $y = c_j$, or $y = d_j$ for some j. The deletion of x results only in new 2-factors containing one or more of the z_+ letters, so a new path from a 2-factor having y as its first letter to a 2-factor having y as its second letter could not have been created by virtue of deleting x.
Case 4. $x = e$. We know $y \neq x_j$ for any j since this would imply the existence of the 2-factor ex_j, negating the assumption that x_j and e are in the same free set. Similarly $y \neq \overline{x}_j$ by virtue of the 2-factor $\overline{x}_j e$ and $y \neq a_j$ because of either $a_j e$ or ea_j would have been a 2-factor before the deletion of e, so we are left with the possibilities of $y = b_j$, $y = c_j$ or $y = d_j$.

Suppose $y = b_j$. From Lemma 5.2 we have that $a_j \notin D$. If C_j consists of three negated variables, then we know that $\overline{z} \in D$, where $\overline{z}$ is the last literal in C_j, for otherwise the path

$$ea_j$$
$$\overline{z}a_j$$
$$\overline{z}e$$

would contradict the assumption that e is free. But then there is no path from a 2-factor having b_j as its first component to a 2-factor having b_j as its second component, contradicting that b_j is not free. On the other hand, if C_j contains a non-negated variable, we arrive at a similar contradiction using the path

$$ez$$
$$a_jz$$
$$a_je$$

where z is the first literal in C_j. The arguments for $y = c_j$ and $y = d_j$ substantially identical. The cases are exhausted. This concludes the proof. □

Lemma 5.9. Let $\phi = C_1 \wedge C_2 \wedge \cdots \wedge C_m$ be a 3-CNF boolean formula in n variables and let $\alpha = \alpha_\phi$ be the word of ϕ. If α is unavoidable, then there is a deletion sequence of free letters that starts by deleting either x_i or $\overline{x}_i$, for $i \leq n$.

Proof. Suppose α is unavoidable. By Lemma 5.8 there is a deletion sequence of free letters reducing α to the empty word. We may assume by Lemma 5.7 that all the letters z_+ appear at the end of the deletion sequence. We know from Lemma 5.6 that it is possible to delete x_i or $\overline{x}_i$ as the ith letter in a deletion sequence of free letters. We need to establish that we can alter any deletion sequence of free letters to one where the first n deletions are as described by Lemma 5.6.

It suffices to show that we can always invert the deletion order whenever an $x \in X_\alpha$ is deleted immediately after some letter $y \notin X_\alpha$ and $\overline{x}$ is deleted after x, where $\overline{\overline{x}} = x$.
Case 1. $y = a_j$. We start by noting that x is already free before the deletion of a_j since no new 2-factor that does not contain a letter z_+ is created by deleting any letter not in X_α or in Z. Suppose x is a non-negated variable, i.e. $x = x_i$ for some $i \leq n$. Suppose for contradiction that inverting the deletion order of x_i and a_j results in a_j not being free. We notice that the only new 2-factor (excluding ones with z_+ letters) created by the deletion of x_i is $e\overline{x}_i$, so after the deletion of x_i there is a path from a 2-factor having a_j as its first letter to $e\overline{x}_i$ and a path from $e\overline{x}_i$ to a 2-factor having a_j as its second letter. We now notice that the only 2-factor having $\overline{x}_i$ as its second letter is $e\overline{x}_i$, so the immediate predecessor to $e\overline{x}_i$ in our malignant path has e as its first letter. But this means that the immediate successor to $e\overline{x}_i$ in the path has $\overline{x}_i$ as its second letter. But again the only 2-factor having $\overline{x}_i$ as its second letter is $e\overline{x}_i$, so the path cannot proceed to any 2-factor not already in the path. Hence there is already a 2-factor having a_j as its second letter at some earlier point in the path, contradicting our assumption that a_j was free before x_i was deleted. Supposing, on the other hand, that $x = \overline{x}_i$ leads to the same contradiction through symmetric reasoning, where we end up in a dead end at the 2-factor xe.
Case 2. $y \in \{b_j, c_j, d_j\}$. The argument is essentially the same as Case 1.
Case 3. $y = e$. Suppose again, for contradiction, that e is not free as the result of deleting x_i. Again the only new 2-factor created is $e\overline{x}_i$, so there is a path from $e\overline{x}_i$ to a 2-factor having e as its second letter. But since the only 2-factor having $\overline{x}_i$ as its second letter is $e\overline{x}_i$, we find ourselves back at the contradiction described in Case 1. For $x = \overline{x}_i$ the argument is, once again, symmetric.

The cases are exhausted and the proof is complete. □

Proposition 5.10. If ϕ is a 3-CNF boolean formula and $\alpha = \alpha_\phi$ is the word of ϕ, then ϕ is satisfiable if and only of α is unavoidable.

Proof. Suppose ϕ with variables $x_1, \ldots, x_n$ and clauses $C_1, \ldots, C_m$ is satisfiable. Let $x_1 = e_1, x_2 = e_2, \ldots, x_n = e_n$, with each $e_i \in \{0, 1\}$, be a satisfying assignment for ϕ. We show that α_ϕ will reduce to the empty set by deleting all its letters in the following stages:

(1) For $i \leq n$, delete x_i if $e_i = 1$, otherwise delete $\overline{x}_i$.
(2) Next, for $j \leq m$, delete a_j, b_j, c_j and the d_j.
(3) Delete the letter e.
(4) Delete the remaining x_i and $\overline{x}_i$.
(5) Delete the remaining characters z_+ in any order.

Furthermore, every letter that is deleted will be free at the stage when the deletion happens.

Lemma 5.6 guarantees that every deletion in Stage (1) above is of a free letter. Since ϕ is satisfiable, every clause $C_j = (p_1 \vee p_2 \vee p_3)$ has at least one literal that is set to 1. If $p_1 = x_i = 1$, then x_i is deleted in Stage (1). Consequently a_j is free after Stage (1) and can be deleted in Stage (2). The deletion of a_j, in turn, causes b_j, c_j and d_j to become free. The remaining cases among $p_k = x_i = 1$ and $p_k = \overline{x}_i = 0$ lead to a_j, b_j, c_j and d_j being deleted in a similar fashion. We can therefore successfully complete the deletions in Stage (2).

After the completion of Stage (2) the only 2-factors (once again ignoring the z_+) containing e, are of the form ep_i and p_ie, where for each i we have either $p_i = x_i$ or $p_i = \overline{x}_i$. Furthermore, for each i the same 2-factors are the only ones containing p_i. Therefore e is free and consequently Stage (3) can be completed.

After the completion of Stage (3), there are no 2-factors left that do not contain one of the z_+. Since every letter z_+ is unique, we can safely complete Stage (4). Now all that remains is letters of the form z_+ and hence, using Lemma 5.5, we can delete the remaining letters. It follows, by Theorem 1.5, that α_ϕ is unavoidable, as desired.

Now suppose ϕ is unsatisfiable. For contradiction, suppose α_ϕ is unavoidable. Using Lemma 5.9, we may assume that the first n deletions are $p_1, p_2, \ldots, p_n$ with, for every i, either $p_i = x_i$ or $p_i = \overline{x}_i$. Define the following assignment on ϕ: If $p_i = x_i$, then set the variable x_i to 1, otherwise set x_i to 0. Since ϕ is not satisfiable, we know that there is some clause C_j that is not satisfied by our chosen assignment. But this means that none of the p_i in the first n deletions appear in C_j and consequently none of the letters a_j, b_j, c_j and d_j are free after the first n deletions, by Lemma 5.1. In addition, by Lemma 5.3, we have that e is not free. In order to free any of these letters, we have to delete at least one letter x_i or $\overline{x}_i$ which has, thus far not been deleted. But this means, for some i, both x_i and $\overline{x}_i$ have been deleted. Using Lemma 5.4, we have a contradiction. □

6. Unavoidability and Computational Complexity

We define the *Word Unavoidability Problem* as follows: Given a pattern α over a finite alphabet, determine if α is unavoidable. We refer to the set of unavoidable patterns as WU.

Theorem 6.1. The Word Unavoidability Problem is NP-complete.

Proof. We note that, given a 3-CNF boolean formula ϕ, the construction of the word α_ϕ of ϕ requires a number of computational steps that is linear in the length of ϕ: For every

variable x_i, we need to add a factor $dx_i\overline{x}_i d$. For every clause we need to add a constant number of factors that are derived purely from the literals in that clause.

Proposition 5.10 therefore leaves us very little work to do. All that remains is to prove $WU \in NP$. Using Theorem 1.3 and the algorithm `IS_FREE` above, we write the following test for unavoidability:

```
IS_UNAVOIDABLE(alpha)
    A[] = the distinct letters in alpha
    B[] = the distinct letters in alpha and all pairs of letters in A
    n = |B|
    nondeterministically guess the permutation pi on [n]
        for i = 1 to n:
            if B[pi(i)] is a single letter and occurs in alpha:
               x = B[pi(i)]
               if IS_FREE(alpha, x):
                    delete every occurrence of x from alpha
               else:
                    nondeterministic guess dies
            else:
               x, y = B[pi(i)]
               if x and y are both letters in alpha:
                  replace every occurrence of y in alpha with x
        return True
     return False
```

Each branch of nondeterminism completes at most $|A_\alpha|$ deletions and $|A_\alpha|^2$ identifications of letters. Since `IS_FREE` runs in polynomial time, so does each branch of `IS_UNAVOIDABLE`. The number of branches of nondeterminism is bounded from above by the number of permutations on $|A_\alpha| + |A_\alpha|^2$. □

7. Conclusion

Many interesting questions remain regarding the complexity of unavoidable patterns [4]. The bounds established in Theorem 2.3 above are not primitive recursive. We do not know if there is a primitive recursive upper bound, nor do we know what lower bounds exist, for any significantly general subset of patterns.

8. Acknowledgements

This article has been written in partial fulfillment of the requirements for the degree Doctor of Philosophy in Operations Research at the University of South Africa. Special and sincere thanks go to Willem Fouché and Petrus Potgieter for continuous insight and guidance. Many

thanks also to Narad Rampersad and James Currie who read earlier versions of this paper and communicated problems to me.

References

[1] **D R Bean**, **A Ehrenfeucht**, **G F McNulty**, http://projecteuclid.org/euclid.pjm/1102783913 *Avoidable patterns in strings of symbols.*, Pacific Journal of Mathematics 85 (1979) 261–294

[2] **B Blakeley**, **F Blanchet-Sadri**, **J Gunter**, **N Rampersad**, *On the Complexity of Deciding Avoidability of Sets of Partial Words*, from: "Developments in Language Theory", (V Diekert, D Nowotka, editors), Springer Berlin Heidelberg, Berlin, Heidelberg (2009) 113–124

[3] **T H Cormen**, **C E Leiserson**, **R L Rivest**, *Introduction to Algorithms*, The MIT Press and McGraw-Hill Book Company (1989)

[4] **J Currie**, http://www.jstor.org/stable/2324790 *Open Problems in Pattern Avoidance*, The American Mathematical Monthly 100 (1993) 790–793

[5] **K Devlin**, https://books.google.com/books?id=HW26uSG2yVgC *Mathematics: The Science of Patterns: The Search for Order in Life, Mind and the Universe*, A Scientific American Library paperback, Henry Holt and Company (1996)

[6] **W L Fouché**, *Unavoidable regularities and factor permutations of words*, from: "Proc. Royal Society Edinburgh", volume 125A, Cambridge University Press (1995) 519–524

[7] **J Hopcroft**, **J Ullman**, *Introduction to Automata Theory, Languages, and Computation*, Addison-Wesley, Reading, Massachusetts (1979)

[8] **F R Madelaine**, http://dx.doi.org/10.2168/LMCS-5(2:13)2009 *Universal Structures and the logic of Forbidden Patterns*, Logical Methods in Computer Science Volume 5, Issue 2 (2009)

[9] **M Morse**, **G A Hedlund**, http://dx.doi.org/10.1215/S0012-7094-44-01101-4 *Unending chess, symbolic dynamics and a problem in semigroups*, Duke Math. J. 11 (1944) 1–7

[10] **P Roth**, http://dx.doi.org/10.1007/BF01178567 *Every binary pattern of length six is avoidable on the two-letter alphabet*, Acta Informatica 29 (1992) 95–107

[11] **W Rytter**, **A M Shur**, *On Searching Zimin Patterns*, CoRR abs/1409.8235 (2014)

[12] **M Sapir**, *Problems of Burnside type and the finite basis property in varieties of semi- groups.*, Izv. Akad. Nauk. SSSR. Ser. Mat. 51 (1987) 319–340

[13] **M Sapir**, **V Guba**, **M Volkov**, https://books.google.com/books?id=HYWBDQEACAAJ *Combinatorial Algebra: Syntax and Semantics*, Springer Monographs in Mathematics, Springer International Publishing (2016)

[14] **U Schmidt**, http://dx.doi.org/10.1007/BF00292112 *Long unavoidable patterns*, Acta Informatica 24 (1987) 433–445

[15] **A Thue**, https://books.google.com/books?id=-gwpGwAACAAJ *Über unendliche Zeichenreihen*, Skrifter udgivne af Videnskabsselskabet i Christiania: Mathematisk-naturvidenskabelig Klasse (1906)

[16] **A I Zimin**, http://stacks.iop.org/0025-5734/47/i=2/a=A05 *BLOCKING SETS OF TERMS*, Mathematics of the USSR-Sbornik 47 (1984) 353