UPPER BOUNDS FOR SEQUENCE SATURATION

SHIHAN KANUNGO

Abstract. In this paper, we study the saturation function $\mathrm{Sat}(n, u)$ for sequences. Saturation for sequences was introduced by Anand, Geneson, Kaustav, and Tsai (2021), who proved that $\mathrm{Sat}(n, u) = O(n)$ for two-letter sequences u and conjectured that this bound holds for all sequences. We present an algorithm that constructs a u-saturated sequence on n letters and apply it to show $\mathrm{Sat}(n, u) = O(n)$ for several families of sequences u, including all repetitions of the form $abcabc\ldots$. We further establish $\mathrm{Sat}(n, u) = O(n)$ for a broad class of sequences of the form $aa\ldots bb$. In addition, we prove that for most sequences u, there exists an infinite u-saturated sequence. For three-letter sequences of the form $abc\ldots xyz$, where a, b, c are distinct and xyz is a permutation of abc, we show—under certain structural assumptions on u—that $\mathrm{Sat}(n, u) = O(n)$. Finally, we describe a linear program that computes the exact value of $\mathrm{Sat}(n, u)$ for arbitrary n and u.

1. Introduction

The concept of pattern avoidance lies at the heart of extremal combinatorics, which investigates questions such as:

What is the largest possible structure that avoids a given substructure?

Well-known examples include Ramsey theory and the forbidden subgraph problem. Another important example is the study of **Davenport-Schinzel sequences**, which are sequences S over an alphabet of n distinct letters that satisfy the following constraints:

- For any two distinct letters a, b, the alternating subsequence $abab\ldots$ of length $s+2$ is forbidden, where s is a nonnegative integer.
- No two consecutive letters in S are the same.

A key quantity of interest is the function $\lambda_s(n)$, which denotes the length of the longest Davenport-Schinzel sequence over an n-letter alphabet. For $s \geq 3$, $\lambda_s(n)$ grows as n times a small, but nonconstant, factor. For instance,

$$\lambda_3(n) = \Theta(n\alpha(n)),$$

where $\alpha(n)$ is the inverse Ackermann function[1], known for its extremely slow growth. Originally introduced to analyze linear differential equations ([6]), Davenport-Schinzel sequences and the function $\lambda_s(n)$ have since found numerous applications in discrete geometry and geometric algorithms ([2]), as well as application to the saturation function for 0–1 matrices ([12]).

A **generalized Davenport-Schinzel sequence** extends this concept by requiring sequences to avoid a given pattern u on r distinct letters while also being r**-sparse**, meaning

Date: December 20, 2025.
2020 *Mathematics Subject Classification.* 05D99.
Key words and phrases. Sequence Saturation, Generalized Davenport-Schinzel Sequences.

[1]see [13] for a definition

every contiguous subsequence of length r has all letters distinct. Analogous to $\lambda_s(n)$, we define the **extremal function** $\mathrm{Ex}(n,u)$, which represents the longest r-sparse sequence on n letters that avoids u. The growth of $\mathrm{Ex}(n,u)$ has been studied in [13], and generalized Davenport-Schinzel sequences have many significant applications.

This paper focuses on a related function, the **saturation function** $\mathrm{Sat}(n,u)$. Saturation functions have been extensively studied in other combinatorial settings, including graphs [11, 5], posets [7], and 0-1 matrices [8]. The notion of saturation for sequences was introduced in [1], where several fundamental results were also established. Specifically, given a forbidden sequence u with r distinct letters, we say that a sequence s on a given alphabet is ***u*-saturated** if s is r-sparse, u-free, and adding any letter from the alphabet to an arbitrary position in s violates r-sparsity or induces a copy of u. Notably, it was shown that if u consists of only two distinct letters, then $\mathrm{Sat}(n,u) = O(1)$ or $\Theta(n)$, leading to the open question of whether this dichotomy extends to arbitrary sequences u. In fact, proving that $\mathrm{Sat}(n,u) = O(n)$ for all sequences u is sufficient to establish this dichotomy. Thus, we aim to address the following conjecture:

Conjecture 1 ([1])**.** *We have* $\mathrm{Sat}(n,u) = O(n)$ *for any sequence* u.

Similar dichotomies have been proven in other settings, such as for graphs [11] and for 0-1 matrices [8]. However, proving the $O(n)$ bound in the context of sequence saturation appears to be significantly more challenging.

We resolve Conjecture 1 in the following cases.

For a sequence $u = abc\dots xyz$ with 3 distinct letters a, b, c, let

$$f_0(u) = \#\{\text{ consecutive pairs of the form } ab, bc, ca \text{ in } u\},$$
$$f_1(u) = \#\{\text{ consecutive pairs of the form } ac, ba, cb \text{ in } u\},$$
$$f_2(u) = \#\{\text{ consecutive equal-letter pairs in } u\ \}.$$

Theorem 1. *Conjecture 1 holds for the following classes of sequences* u*:*

(1) $u = abcabc\cdots$,
(2) $u = aa\dots bb$ *and* u *cannot be decomposed into two subsequences* $u = u_1u_2$ *that have no letters in common,*
(3) $u = abc\dots xyz$ *is a three-letter sequence with* a, b, c *distinct, and*

$$xyz \in \{abc, bca, cab\}, \qquad f_0(u) \geq f_1(u) + 5.$$

The organization of this paper is as follows. In Section 2 we define the saturation and extremal functions for sequences. In Section 3, we introduce Algorithm 1, which produces a u-saturated sequence on n letters. Using this, we give an example of a sequence u with $\mathrm{Sat}(n,u) = O(n)$ even though $\mathrm{Ex}(n,u)$ grows faster than linear, and we additionally prove point (1) of Theorem 1. We next prove point (2) of Theorem 1 in Theorem 4. Finally, we show in Theorem 5 that for a large class of sequences u, there exists a doubly infinite sequence that is u-saturated. In Section 4, we focus on sequences u with $r = 3$ letters, and prove point (3) of Theorem 1 in Theorem 7. In Section 5, we introduce a linear program in Theorem 8 that computes the exact value of the $\mathrm{Sat}(n,u)$, similarly to the one for 0-1 matrices in [4]. In Table 8.1, we display the results of the linear program for some short sequences u and small n.

A further direction to investigate, after Conjecture 1 is resolved, would be to classify when $\mathrm{Sat}(n,u) = O(1)$ or $\Theta(n)$. The corresponding question for 0-1 matrices was investigated by

Geneson [9], who showed that almost all permutation matrices have bounded saturation function, and by Berendsohn [3], who completely resolved the classification for permutation matrices. However, the general question remains open.

2. Notation and Definitions

For a sequence s, let s^t denote the concatenation of t copies of s. Let $[n]$ denote the set $\{1, \ldots, n\}$.

Throughout this paper we consider a finite sequence u that contains r distinct letters. We say sequences u and v are **isomorphic** if v can be obtained from u by a one-to-one renaming of letters. We say a sequence s **avoids** u if s does not contain a subsequence isomorphic to u. We refer to any subsequence of s isomorphic to u as a **copy** of u in s. We say that s is r**-sparse** if any r consecutive letters in s are pairwise distinct. We say that a sequence s with letters in an alphabet A is u**-saturated** if s is r-sparse, s avoids u, and inserting any letter in A between two consecutive letters in s either causes s to not be r-sparse or causes s to contain a copy of u. Finally, we say that s is u**-semisaturated** if s is r-sparse and inserting a letter either violates r-sparsity or induces a copy of u. The notions of saturation and semisaturation are the same, except for the fact that saturated sequences have to avoid u.

We define the **extremal function** $\mathrm{Ex}(n, u)$ to be the maximum length of a sequence s on n letters such that s avoids u and is r-sparse. We define the **saturation function** $\mathrm{Sat}(n, u)$ to be the minimum length of a u-saturated sequence s on n letters. Trivially, we have $\mathrm{Sat}(n, u) \leq \mathrm{Ex}(n, u)$ because every u-saturated sequence is r-sparse and avoids u.

A **0-1 matrix** is a matrix whose entries are 0 or 1. By standard convention, we represent 0-1 matrices by replacing the ones with bullets $\bullet$ and the zeros with empty spaces. For example, we write

$$\begin{pmatrix} \bullet & \bullet \\ \bullet & \end{pmatrix} \quad \text{for} \quad \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}.$$

We say that a 0-1 matrix A **contains** another 0-1 matrix B if A has a submatrix with the same dimensions as B that is entrywise greater than B.

3. General Saturation of Sequences

We are interested in showing that $\mathrm{Sat}(n, u) = O(n)$ for any sequence u. This has been shown for 2-letter sequences u in [1], but the cases for 3-letter sequences and higher remain open.

3.1. Algorithm.

In Algorithm 1 (see below), we describe a method to construct a u-saturated sequence on n letters. We say "x can be properly inserted into s" as a shorthand to mean "x can be inserted into s without violating r-sparsity and without inducing a copy of u". It is evident from the definition of the algorithm that it will always produce a u-saturated sequence. To demonstrate that $\mathrm{Sat}(n, u) = O(n)$, we must construct a u-saturated sequence on n letters for every n, where the length of the sequence grows at most linearly with n. In this subsection, we use Algorithm 1 to accomplish this task for a variety of sequences u.

To represent long sequences, we use the following method. Given a sequence $s = s_1 \ldots s_k$, we plot the points (i, s_i) for $i = 1, \ldots, n$. For instance, the sequence $1, 2, \ldots, n$ forms a line with a slope of 1, while the sequence $1, 1, \ldots, 1$ results in a horizontal line.

Algorithm 1 Constructing a u-Saturated Sequence

Input: Alphabet $A = \{1, \dots, n\}$, forbidden sequence u
Output: u-saturated sequence
Initialize the sequence: $s \leftarrow 1, 2, \dots, r-1$ ▷ Initial sequence avoids u
while it is possible to extend the sequence **do**
  for each letter $x \in A$ **do**
    if x can be *properly inserted* into s **then**
      Insert x appropriately into s to form s' ▷ Smallest x, leftmost position
      Update $s \leftarrow s'$ ▷ New sequence
      break ▷ Exit loop after the first valid insertion
    end if
  end for
end while
Return s ▷ Final sequence is u-saturated

Trivially, we know that $\text{Sat}(n, u) \leq \text{Ex}(n, u)$, so if $\text{Ex}(n, u) = O(n)$, Conjecture 1 immediately follows. A **nonlinear sequence** is defined as one for which $\text{Ex}(n, u)$ grows faster than linearly. Therefore, we only need to focus on nonlinear sequences.

In [13], the extremal function for sequences, particularly 3-letter sequences, was studied. It was shown that the sequence $u = abcacbc$ is nonlinear and "minimally nonlinear," meaning that no subsequence of u is nonlinear. Thus, $u = abcacbc$ is the first sequence that we will consider. If we run Algorithm 1 on $u = abcacbc$, we notice that a pattern appears. In Figure 1, we display the output of the algorithm using the representation described earlier.

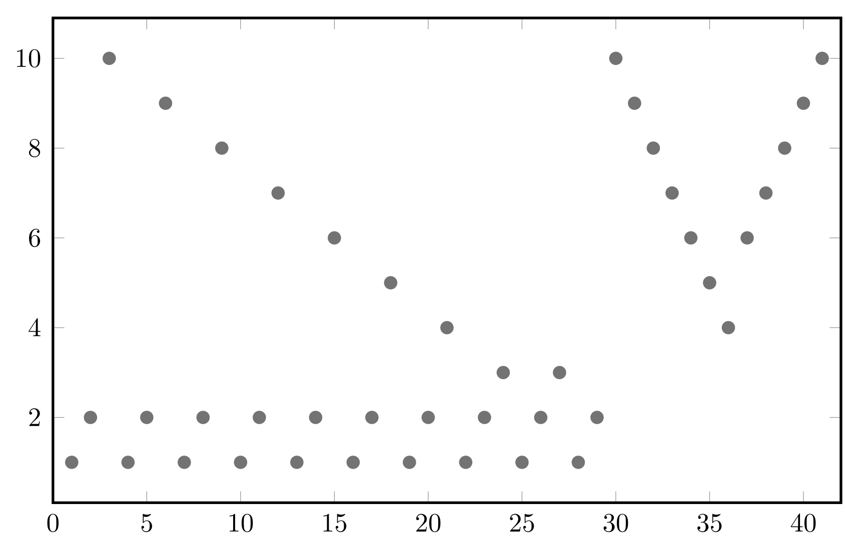


Figure 1. Algorithm 1 on $u = abcacbc$.

This example is the $n = 10$ case. The pattern (which is easily seen from the figure) corresponds to the sequence

$$1, 2, n, 1, 2, n-1, \ldots, 1, 2, 3, 1, 2, 3, 1, 2, n, n-1, \ldots, 6, 5, 4, 6, 7, \ldots, n.$$

From here it is straightforward (albeit quite tedious) to verify that this is a u-saturated sequence, for any n. Since the length of this sequence grows linearly with n, it follows that

Proposition 2. *We have* $\mathrm{Sat}(n, abcacbc) = O(n)$.

We can do this for different sequences. For every case we tried, we got a repeating pattern like the one above. Some of these had minor variations that depended on the value of n (mod d) for some d; but the $O(n)$ bound still follows. Some more examples are shown in Figure 2.

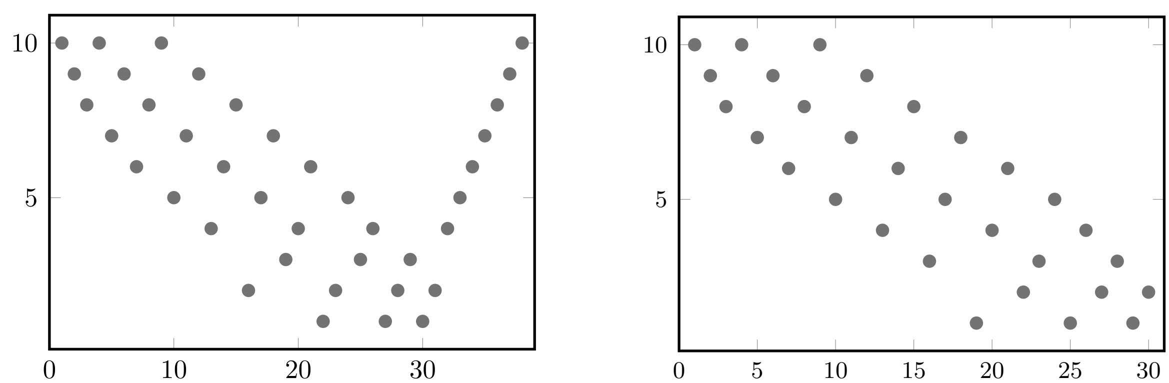


Figure 2. Algorithm 1 on $u = abbacac$ (left) and $abcacba$ (right).

The fact that we always get a pattern prompts us to posit the following.

Conjecture 2. *Let $s(n, u)$ be the sequence of produced by Algorithm 1. Then the length of $s(n, u)$ grows at most linearly in n, for any u.*

There is one important point to note about this algorithm: since we prefer to insert smaller letters, in the process of generating $s(n, u)$, right before we add the first "n", the sequence must be exactly $s(n-1, u)$. Thus, $s(n-1, u)$ is a subsequence of $s(n, u)$. This means that Conjecture 2 is equivalent to saying that if we run the algorithm starting from $s(n-1, u)$, we will add a bounded number of new letters.

We now use this algorithm to prove the conjecture when u is of the form $abcabc\ldots$. Such sequences are the three-letter analog of the Davenport-Schinzel sequences. In the case when $u = (abc)^t$, for some integer t, the algorithm outputs a sequence of the form shown in Figure 3. This example is when $t = 4, n = 10$. In general, the numbers $3, \ldots, n$ will each appear exactly $t-1$ times. The sequence obtained is thus

$$(12n)^{t-1} \ldots (124)^{t-1}(123)^{t-1}12.$$

It is straightforward to check that the resulting sequence is $(abc)^t$-saturated. Note that this is a special case of Lemma 3.4 in [1].

Similarly, when $u = (abc)^t a$ we get the pattern in Figure 4. Here $t = 4, n = 10$, and each number from 3 to $n-2$ appears exactly $3(t-1)+1$ times. This sequence can be easily generalized to arbitrary values of t. Finally, for $u = (abc)^t ab$, we get the pattern in Figure 5,

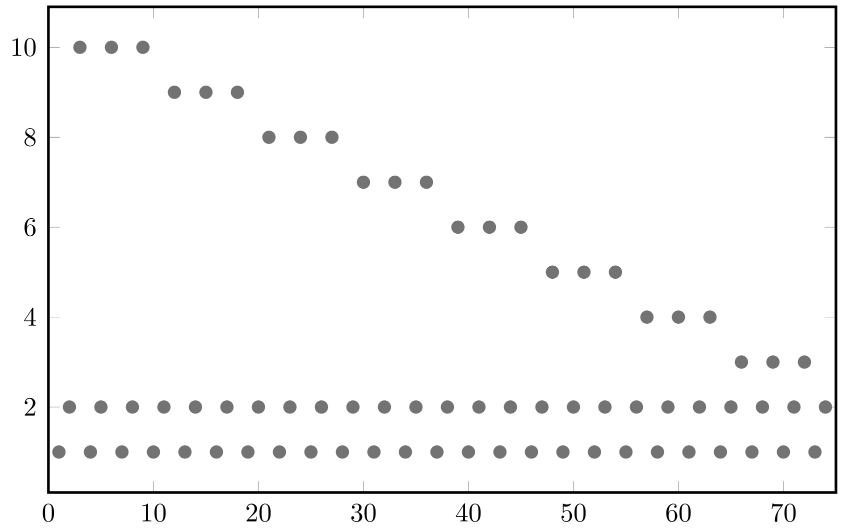


Figure 3. Algorithm 1 on $u = (abc)^t$, $t = 4$.

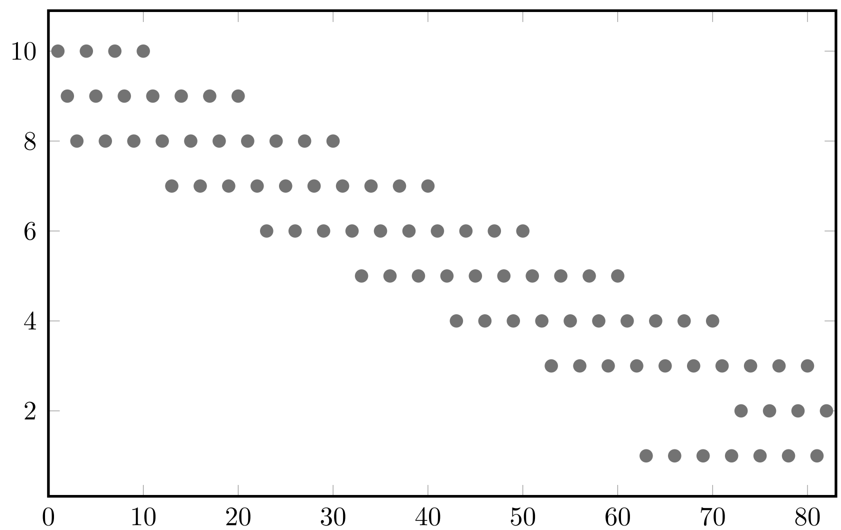


Figure 4. Algorithm 1 on $u = (abc)^t a$, $t = 4$.

where $t = 3$, $n = 10$, and each number from 3 to $n-2$ appears exactly $3(t-1)+2$ times. As before, this sequence can be easily generalized to arbitrary values of t.

Proving that these constructions indeed give u-saturated sequences is straightforward but tedious. In summary, we have:

Proposition 3. *If u is of the form $abcabc\ldots$,* $\mathrm{Sat}(u) = O(n)$.

This prove point (1) of Theorem 1. If the length of u is at least 6, this result also follows from Theorem 7 below. Interestingly, the constructions above are quite similar (or the same) as the construction given in Theorem 7.

3.2. **Sequences of the form $aa\ldots bb$.** In this subsection, we prove a result that resolves Conjecture 1 for a large class of sequences.

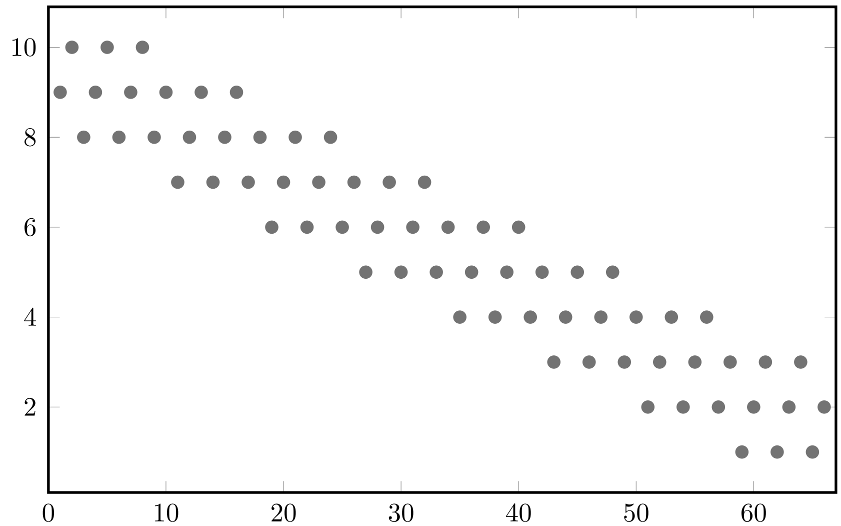


Figure 5. Algorithm 1 on $u = (abc)^t ab$, $t = 3$.

Consider a sequence u on r distinct letters. We define $s_r(u)$ to be the longest sequence of the form $a_1 a_2 \dots a_r a_1 a_2 \dots$ that avoids u. It is straightforward to verify that $s_r(u)$ is u-saturated. Let $s_r^m(u)$ be a copy of $s_r(u)$ on the letters $m+1, m+2, \dots, m+r$. The sequence $s_r(u)$ was first defined in [1] for two-letter sequences, and was a key component in the proof of the $O(n)$ bound for two-letter sequences u.

We say that u is **irreducible** if u cannot be decomposed into subsequences $u = u_1 u_2$ such that u_1 and u_2 have no letters in common.

Theorem 4. *If u is irreducible and of the form $aa \dots bb$, then* $\mathrm{Sat}(n, u) = O(n)$.

Proof. Let r be the length of u. If $r = 2$, the result follows from [1], so assume $r \geq 3$. The idea is to create a sequence out of blocks that are isomorphic to $s_r(u)$ (which is known to be u-saturated). In the case where n is not evenly divisible by r, we have to slightly modify the lengths of some of the blocks, but the general idea is the same.

Case 1: $n = rm$ for some integer m.
Define the sequence s_n as follows:

$$s_n = s_r^0(u)\, s_r^r(u) \dots s_r^{r(m-1)}(u),$$

We claim that s_n is u-saturated.

Avoidance: Partition $[n]$ into *blocks* of size r: the k-th block is $\{rk+1, \dots, rk+r\}$. Suppose that s_n contains a copy of u. By irreducibility of u, the letters in this copy all come from the same block. But within each block, the subsequence is $s_r(u)$, which avoids u. Hence, s_n avoids u.

Saturation: Suppose we insert the letter a into s_n. Then a belongs to some block $\{rk+1, \dots, rk+r\}$. If we insert a inside the corresponding $s_r^{rk}(u)$, it is easy to see that we break r-sparsity. Now suppose we insert a before or after $s_r^{rk}(u)$. Without loss of generality, we insert a to the left of $s_r^0(u)$. Then by the same argument as below in Case 2, we induce a copy of u in $a, s_r^0(u)$.

Case 2: $n = rm + j$, where $1 \leq j < r$.
We want to construct a u-saturated sequence of length n. Let $s_r(u) = (a_1 \dots a_r)^t a_1 \dots a_j$,

and define:

$$s'_r(u) = (a_1 \dots a_{r+1})^t a_1 \dots a_j.$$

We claim $s'_r(u)$ is also u-saturated and behaves like $s_r(u)$ in the saturation argument above. In particular, it is u saturated, and if we add any letter on the left or the right of $s'_r(u)$ (even if it violates sparsity) then we get a copy of u.

Avoidance: This is clear, since $s_r(u)$ avoids u.

Saturation: Note that the occurrences of the letter a_i in $s'_r(u)$ are evenly spaced with exactly r letters between them. Since $r \geq 3$, this means we cannot insert an a_i between two already-existing a_i in $s'_r(u)$ without violating r-sparsity. So it has to be inserted before the very first or after the very last a_i. Note that this is trivially true also when we are adding a_i on the left or the right of $s'_r(u)$. We claim that as long as this condition is satisfied, we have a copy of u.

We can take a subsequence s of $s'_r(u)$ containing a_i that is isomorphic to $s_r(u)$ (we just delete all occurrences of one of the letters a_k, $k > j$). Then a_i is inserted either before the first a_i in s or after the last. Without loss of generality, it is the first case, and s uses the letters $a_1, \dots, a_r$. Then, let u' be u with the first letter deleted. Since the first two letters of u are equal, it is clear that $s_r(u')$ is $s_r(u)$ with the first r letters deleted. Letting $s = a_1 \dots a_{i-1} s'$, it follows that s' is isomorphic to a sequence of the form $b_1 \dots b_r b_1 \dots b_r \dots$ that is longer than $s_r(u')$, hence contains a copy of u'. Furthermore, this copy can start with the first letter of s', which is a_i. By adding the a_i which we inserted before this a_i, we get a copy of u. This proves that $s'_r(u)$ is u-saturated, and also if we add a letter on the left or the right of $s'_r(u)$, we get a copy of u.

Now, to reach length $n = rm + j$, replace j of the blocks $s_r^{rk}(u)$ in s_{rm} with $s'_r(u)$. The resulting sequence has length n and remains u-saturated. Since we only use $O(m) = O(n)$ blocks of constant size, the total length grows linearly in n. □

This proves point (2) of Theorem 1.

It is easy to see that the class of sequences in this theorem contains infinitely many nonlinear sequences.

3.3. **Infinite Sequences.** We can also consider u-saturated *infinite* sequences. In other words, let s be an infinite sequence on some alphabet A (which can be infinite). We say s is u-saturated if s is r-sparse (defined in the same way as in the finite-length case), s does not contain a copy of u as a subsequence, and inserting a new letter between two letters of s either violates r-sparsity or induces a copy of u.

In the following theorem, we prove that for many sequences u, there exists a doubly infinite sequence s on an infinite alphabet A such that s is u-saturated.

Say that u is **strongly irreducible** if for any two letters α, β in u, it is not the case that all the α's occur before all the β's or vice versa. Note that this implies the notion of irreducibility introduced earlier.

Theorem 5. *Let u be a strongly irreducible sequence on r letters that contains each of its letters more than once. Then there exists a doubly infinite sequence on the alphabet $\{a_1, \dots, a_{r-1}\} \cup \mathbb{Z}$ that is u-saturated.*

Proof. The idea is similar to the proof of Theorem 4. Define $t_r(k)$ to be the infinite sequence over the alphabet $\{a_1, \dots, a_{r-1}, x\}$ given by

$$\dots (a_1 \dots a_{r-1})(a_1 \dots a_{r-1})(a_1 \dots a_{r-1} x)^k (a_1 \dots a_{r-1})(a_1 \dots a_{r-1}) \dots.$$

Since u contains each of its letters at least twice, the sequence $t_r(1)$ avoids u. Moreover, for sufficiently large k, it is clear that $t_r(k)$ contains a copy of u. Let p be the maximum integer such that $t_r(p)$ avoids u. Note that $p \geq 1$.

For $n \in \mathbb{Z}$, define

$$t[n] = (a_1 \dots a_{r-1} n)^p.$$

We then set

$$s = \dots t[-1]t[0]t[1]t[2] \dots = \bigcup_{n=-\infty}^{\infty} t[n].$$

We claim that s is u-saturated. It is clearly r-sparse.

Avoidance. Suppose s contains a copy of u. Since u is strongly irreducible, such a copy can involve at most one letter from $\mathbb{Z}$. Thus it must lie on the alphabet $\{a_1, \dots, a_{r-1}, n\}$ for some $n \in \mathbb{Z}$. But the subsequence of s on these letters is isomorphic to $t_r(p)$, which by definition avoids u. Hence s avoids u.

Saturation. We cannot insert any a_i without breaking r-sparsity. If we insert an integer n without violating r-sparsity, then s acquires a subsequence isomorphic to $t_r(p+1)$ over the alphabet $\{a_1, \dots, a_{r-1}, n\}$. Since p was chosen maximally, $t_r(p+1)$ contains a copy of u. It follows that s is u-saturated. □

Remark 1. Consider the finite analogue of the sequence s, namely

$$t[1]t[2] \dots t[n].$$

The same argument shows that inserting a letter between $t[k]$ and $t[n-k]$ necessarily violates either r-sparsity or u-avoidance, where k is a constant depending only on u. Thus this construction yields a sequence that is "almost u-saturated." Moreover, if we run Algorithm 1 starting from this sequence, the contiguous block

$$t[k]t[k+1] \dots t[n-k]$$

is preserved. It is natural to ask whether this additional structure can be exploited to prove that the algorithm terminates in $O(n)$ steps, thereby resolving Conjecture 1 (and a version of Conjecture 2) for the class of sequences considered in Theorem 5.

4. Saturation for Sequences on 3 letters

In [1], it was shown that every sequence on two letters has a linear saturation function. The proof provided an explicit construction of a u-saturated sequence on n letters for arbitrary n, relying heavily on the sequence $s_2(u)$. This construction extends naturally to an arbitrary sequence u on r letters. The same argument shows that the resulting sequence is **u-semisaturated**, i.e., saturated except that it need not avoid u. However, when u involves more than two letters, the sequence obtained from this construction may fail to avoid u.

In this section, we restrict attention to the case where u has three letters, and prove that under certain conditions the construction of [1] indeed yields a u-saturated sequence. Consequently, we obtain $\mathrm{Sat}(n, u) = O(n)$ in this setting.

Throughout, let u be a fixed three-letter sequence of length greater than or equal to six of the form

$$u = abc \dots xyz,$$

where a, b, c and x, y, z are distinct letters, and x, y, z is a permutation of $\{a, b, c\}$. Define a family of sequences $s[n]$ on n letters recursively as follows. Set $s[3] = s_3(u)$. Given $s[n-1]$, let

x, y denote its last two letters, and let z be a new letter not appearing in $s[n-1]$. Construct $s[n]$ by appending to $s[n-1]$ a copy of $s_3(u)$ on the letters x, y, z, omitting its initial two letters x, y. In particular, the terminal segment of $s[n]$ is a full copy of $s_3(u)$ on x, y, z.

This definition of $s[n]$ is directly analogous to the construction of the u-saturated sequence in Theorem 3.10 of [1].

Proposition 6. *The sequence $s[n]$ is u-semisaturated.*

Proof. The proof is analogous to the last part of the proof of Theorem 3.10 in [1]. □

Thus, it remains to determine when $s[n]$ avoids u.

We begin by analyzing the length of $s_3 := s_3(u)$. By definition, $s_3 = 123123\ldots$. If s_3' denotes s_3 with one new letter appended at the end (in a way that preserves 3-sparsity), then s_3' contains a copy of u. Moreover, this copy necessarily uses both the first and last letters of s_3'. Hence $a = 1$ and $(b, c) = (2, 3)$ or $(3, 2)$ in this copy.

To compute $|s_3'|$, imagine first writing the copy of u, then inserting letters from $\{1, 2, 3\}$ between consecutive letters so that the resulting sequence has the form $123123\ldots$. The number of insertions depends on the consecutive pair:

- 0 insertions for $12, 23, 31$,
- 1 insertion for $13, 21, 32$,
- 2 insertions for $11, 22, 33$.

Define

$$f_0(u) = \#\{\text{ consecutive pairs of the form } ab, bc, ca \text{ in } u\},$$
$$f_1(u) = \#\{\text{ consecutive pairs of the form } ac, ba, cb \text{ in } u\},$$
$$f_2(u) = \#\{\text{ consecutive equal-letter pairs in } u\,\}.$$

Clearly,

$$f_0(u) + f_1(u) + f_2(u) = |u| - 1.$$

If the copy of u in s_3' uses the letters $1, 2, 3$ in this order, then

$$|s_3'| = |u| + f_1(u) + 2f_2(u),$$

while if it uses $1, 3, 2$, then

$$|s_3'| = |u| + f_0(u) + 2f_2(u).$$

Therefore,

$$|s_3| = |u| - 1 + f_2(u) + \min\{f_0(u), f_1(u)\}. \tag{4.1}$$

Theorem 7. *Let $u = abc\ldots xyz$ be a three-letter sequence with a, b, c distinct. Suppose*

$$xyz \in \{abc, bca, cab\}, \qquad f_0(u) \geq f_1(u) + 5.$$

Then $s[n]$ avoids u, and hence $\mathrm{Sat}(n, u) = O(n)$.

Proof. We use a similar argument to that of Theorem 3.10 in [1], but the 3-letter case is much more complicated (which is also why we need to impose many more conditions). The proof proceeds by induction and a careful case analysis of where a hypothetical copy of u could appear. The parameters $f_0(u)$ and $f_1(u)$ control how much space such a copy would need to take.

We proceed by induction on n. The base case holds since $s[3] = s_3$ is u-saturated, and therefore avoids u. Assume inductively that $s[n-1]$ avoids u.

Suppose for contradiction that $s[n]$ contains a copy of u. We distinguish two cases.

Case 1: One of the letters is z. It is clear from the conditions that u is strongly irreducible, so the other two letters must be x and y, as all other letters appear strictly before the first z. The subsequence of $s[3]$ consisting of x, y is of the form

$$Pxyzxyz\dots,$$

where the part after P is isomorphic to s_3 and P contains only x, y. Because the first three letters of u are distinct, any copy of u in this subsequence can use at most two letters before the first z, and these must be distinct. Thus it suffices to consider the cases $P = y$:

$$xyzxyz\dots \quad \text{and} \quad yxzxyz\dots,$$

with both sequences of length $|s_3|$. The first is isomorphic to s_3, hence avoids u. In the second, if a copy of u exists, it must use the initial yx; otherwise, the same copy could be embedded in the first sequence, a contradiction. Thus the copy is on the alphabet $\{y, x, z\}$, and its length is at least

$$|u| + f_0(u) + 2f_2(u) - 2,$$

where we subtract 2 because no insertions are needed between the initial yx and the following xz. Since $f_0(u) \geq f_1(u) + 3$, this contradicts (4.1).

Case 2: None of the letters is z. Let the letters be α, β, γ in the order of their first appearance in $s[3]$. The first γ appears when extending $s[k-1]$ to $s[k]$ for some $k < n$. Since u is strongly irreducible, some α must appear after some γ, which is only possible if α is among the last two letters of $s[k-1]$. Similarly, β must also be among the last two letters. Without loss of generality, let β be the last letter and α the second-last (if $k - 1 = 0$, then α, β, γ are simply the first three letters of $s[n]$, and the same argument applies). Thus there is a copy of u in

$$P\alpha\beta\gamma\alpha\beta\gamma\dots Q,$$

where P contains only α, β, the middle block has length $|s_3|$, and Q contains only the last two letters of the middle block. We now consider three subcases, depending on the ending of the middle block.

Case 2.1: The middle block ends $\alpha\beta\gamma$. We must check the following sequences of length $|s_3|$:

$$\alpha\beta\gamma\dots\alpha\beta\gamma, \quad \beta\alpha\gamma\dots\alpha\beta\gamma, \quad \alpha\beta\gamma\dots\alpha\gamma\beta, \quad \beta\alpha\gamma\dots\alpha\gamma\beta.$$

The first two follow from Case 1. For the latter two, let $\widetilde{u}$ be the reversal of u. Since $xyz \in \{abc, bca, cab\}$, $f_i(\widetilde{u}) = f_i(u)$, so $\widetilde{u}$ satisfies the theorem's hypotheses (note that we must rename the letters a, b, c when computing $f_i(\widetilde{u})$). The reversal of the third sequence is isomorphic to the second sequence of Case 1 for $\widetilde{u}$, hence avoids $\widetilde{u}$, so the third sequence avoids u. For the fourth, any copy of u must use both the initial $\beta\alpha$ and the terminal $\gamma\beta$; otherwise it embeds in one of the previous sequences. Thus its length is at least

$$|u| + f_0(u) + 2f_2(u) - 4.$$

Since this is $\leq |s_3|$ yet $f_0(u) \geq f_1(u) + 5$, we again contradict (4.1).

Case 2.2: The middle block ends $\beta\gamma\alpha$. Here the relevant sequences are

$$\alpha\beta\gamma\dots\alpha\beta\gamma\alpha, \quad \beta\alpha\gamma\dots\alpha\beta\gamma\alpha, \quad \alpha\beta\gamma\dots\alpha\beta\alpha\gamma, \quad \beta\alpha\gamma\dots\alpha\beta\alpha\gamma,$$

each of length $|s_3|$. The proof is exactly analogous to Case 2.1.

Case 2.3: The middle block ends $\gamma\alpha\beta$. Here the sequences are

$$\alpha\beta\gamma\ldots\alpha\beta\gamma\alpha\beta,\quad \beta\alpha\gamma\ldots\alpha\beta\gamma\alpha\beta,\quad \alpha\beta\gamma\ldots\alpha\beta\gamma\beta\alpha,\quad \beta\alpha\gamma\ldots\alpha\beta\gamma\beta\alpha,$$

each of length $|s_3|$. The argument is identical to the previous subcases.

In all cases we obtain a contradiction, so $s[n]$ avoids u. Since $|s[n]|$ grows linearly in n, it follows that $\mathrm{Sat}(n,u) = O(n)$. ☐

This proves point (3) of Theorem 1.

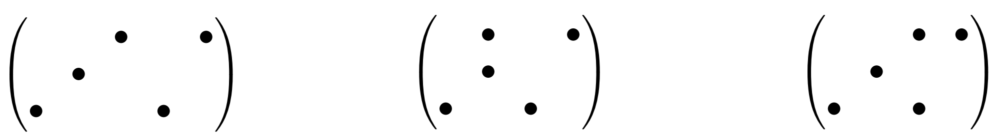
Example 1. Some examples of sequences u satisfying the conditions of Theorem 7 are

$$abcacbcabca, abcbabcabc, abcbcbacbabcabcabc.$$

Additionally, all sequences of the form $(abc)^t$, $(abc)^t a$, $(abc)^t ab$ satisfy the conditions as long as $t \geq 2$. Thus, Theorem 7 implies Theorem 3. Finally, note that for any sequence u on letters a, b, c, $(abc)u(abc)^t$ satisfies the conditions for large enough t. Thus $\mathrm{Sat}(n, (abc)u(abc)^t) = O(n)$ for large enough t.

5. A Linear Program for $\mathrm{Sat}(n,u)$

In this section, we describe a linear program that computes the exact value of $\mathrm{Sat}(n,u)$. This is similar to the linear program found in [4], but it is significantly more complicated. Let u be a sequence of length ℓ with r distinct letters. Call an $r \times \ell$ 0-1 matrix P a u-pattern if P has a single 1 per column, and the sequence obtained by listing the row indices of the ones from left to right is isomorphic to u. Call a $r \times (\ell - 1)$ 0-1 matrix Q a u^+-pattern if every column has exactly one 1, except for a single column c which has two ones; and the sequence obtained by listing the row indices of the ones from left to right, where the one in c with a higher row index is placed first, is isomorphic to u. Define a u^--pattern similarly, except the one in c with a lower row index is placed first. For example, if $u = abcac$ then we have the following examples:



u-pattern $\qquad$ u^+-pattern $\qquad$ u^--pattern

For a positive integer m, let $[m]$ denote the set $\{1, \ldots, m\}$.

5.1. **Integer program** $\mathrm{IP}_{u,n}(N)$**.** Let N be an integer greater than $\mathrm{Sat}(n,u)$. Define $\mathcal{P}$ to be the set of all index sets $P' \subset [n] \times [N]$ such that P' forms a copy of some u-pattern P in the all-ones matrix $\mathbf{1}_{n\times N}$. For each pair $(i,j) \in [n] \times [N]$, let

$$A_{i,j} = \{P' \in \mathcal{P} : (i,j) \in P'\}$$

denote the collection of patterns in $\mathcal{P}$ that include the entry (i,j). Note that $A_{i,j} \neq \varnothing$ for all (i,j), and each pattern $P' \in \mathcal{P}$ satisfies $|P'| = \ell$, where ℓ is the size of any u-pattern.

Define $\mathcal{P}^+$, $\mathcal{P}^-$, $A^+_{i,j}$, and $A^-_{i,j}$ analogously for the sets of modified u-patterns. Again, each $P' \in \mathcal{P}^\pm$ satisfies $|P'| = \ell$.

Define variables $x_{i,j}$ for $1 \le i \le r$ and $1 \le j \le N$ and $y_{P'}$ for each $P' \in \mathcal{P}$ or $\mathcal{P}^{\pm}$. For notational convenience, set

$$f(P') := \sum_{(i,j)\in P'} x_{i,j}.$$

The integer program $\mathrm{IP}_{u,n}(N)$ is

$$\min \sum_{i=1}^{n} \sum_{j=1}^{N} x_{i,j}$$

subject to

$$x_{i,j} \in \{0,1\} \qquad \forall (i,j) \in [n]\times[N] \tag{5.1}$$

$$y_{P'} \in \{0,1\} \qquad \forall P' \in \mathcal{P} \tag{5.2}$$

$$f(P') < \ell \qquad \forall P' \in \mathcal{P} \tag{5.3}$$

$$(\ell-1)y_{P'} - f(P') \le 0 \qquad \forall P' \in \mathcal{P}\cup\mathcal{P}^{\pm} \tag{5.4}$$

$$y_{P'} - f(P') \ge -\ell+2 \qquad \forall P' \in \mathcal{P}\cup\mathcal{P}^{\pm} \tag{5.5}$$

$$\sum_{P'\in A_{i,j}\cup A^{+}_{i,j}} y_{P'} - \sum_{t=i+1}^{n} x_{t,j} + \sum_{t=j-r+1}^{j+r-2} x_{i,t} \ge 0 \qquad \forall (i,j) \in [n]\times[N] \tag{5.6}$$

$$\sum_{P'\in A_{i,j}\cup A^{+}_{i,j}} y_{P'} - \sum_{t=1}^{i-1} x_{t,j} + \sum_{t=j-r+2}^{j+r-1} x_{i,t} \ge 0 \qquad \forall (i,j) \in [n]\times[N] \tag{5.7}$$

$$\sum_{P'\in A_{i,j}\cup A^{-}_{i,j}} y_{P'} - \sum_{t=i+1}^{n} x_{t,j} + \sum_{t=j-r+2}^{j+r-1} x_{i,t} \ge 0 \qquad \forall (i,j) \in [n]\times[N] \tag{5.8}$$

$$\sum_{P'\in A_{i,j}\cup A^{-}_{i,j}} y_{P'} - \sum_{t=1}^{i-1} x_{t,j} + \sum_{t=j-r+1}^{j+r-2} x_{i,t} \ge 0 \qquad \forall (i,j) \in [n]\times[N] \tag{5.9}$$

$$\sum_{i=1}^{n} x_{i,j} \le 1 \qquad \forall j \in [N] \tag{5.10}$$

$$\sum_{i=1}^{n} x_{i,j} \ge \sum_{i=1}^{n} x_{i,j+1} \qquad \forall j \in [N-1] \tag{5.11}$$

$$\sum_{t=j}^{j+r-1} x_{i,t} \le 1 \qquad \forall (i,j) \in [n]\times[N-r+1]. \tag{5.12}$$

$$x_{i,i} = 1 \qquad i \in [r-1] \tag{5.13}$$

We are now ready to state the following theorem.

Theorem 8. *For $N > \mathrm{Sat}(n,u)$, the optimal value of $IP_{u,n}(N)$ is* $\mathrm{Sat}(u,n)$.

Proof. The interpretations for the conditions are as follows: Equation (5.3) says that there is no copy of u, Equations (5.6), (5.7), (5.8) and (5.9) encode the saturation condition, and

Equation (5.12) encodes the sparsity condition. The other conditions make sure everything is in the right "syntax".

Suppose we have numbers $x_{i,j}, y_{P'}$ satisfying the above conditions. Let $X = [x_{ij}]$. Then by conditions (5.10) and (5.11), columns $1, \dots, k$ of X have exactly one 1, and columns $k+1, \dots, N$ are empty, for some k. We define the sequence s on n letters with length k, by s_j being the unique index such that $x_{s_j,j} = 1$. We claim that s is a u-saturated sequence. First, condition (5.12) implies s is r-sparse. Next, condition (5.3) implies there is no copy of u in s, so s avoids u. Now we need to show that s is u-saturated. Consider $y_{P'}$ for $P' \in \mathcal{P} \cup \mathcal{P}^{\pm}$. Note that $f(P') \leq \ell - 1$ for $P' \in \mathcal{P}^{\pm}$ since P' only has $\ell - 1$ nonzero columns. Conditions (5.4) and (5.5) then implies $y_{P'} = 0$ if $f(P') < \ell - 1$ and $y_{P'} = 1$ if $f(P') = \ell - 1$, i.e. P' is missing exactly one letter. Since P' has a column with two ones, it must be one of the ones in that column if $y_{P'} = 1$. Now, we claim that conditions (5.6), (5.7),(5.8),(5.9) together imply that s is u-saturated. Together, they correspond to adding a i either before or after s_j. If $j > k$, then $x_{t,j} = 0$ for all t, so the conditions evidently hold. Now consider the case when $j \leq k$, so $x_{s_j,j} = 1$. If $i = s_j$ then $x_{i,j} = x_{s_j,j} = 1$ so all of the conditions are satisfied. Now suppose $i < s_j$. Then $x_{s_j,j} = 1$ so $\sum_{t=i+1}^{n} x_{t,j} = 1$, and $\sum_{t=1}^{i-1} x_{t,j} = 0$. Thus conditions (5.7) and (5.9) are satisfied. We now claim that (5.6) corresponds to adding the letter i before s_j. Note that

$$\sum_{t=i+1}^{n} x_{t,j} = 1$$

implies that one of the following conditions is true:

- $y_{P'} \geq 1$ for some $P' \in A_{ij}$, or
- $y_{P'} \geq 1$ for some $P' \in A_{ij}^{+}$, or
- $\displaystyle\sum_{t=j-r+1}^{j+r-2} x_{i,t} \geq 1.$

The third is easily seen to be equivalent to violating the sparsity condition if i is inserted before s_j. The second implies that if we make the (i, j) spot into a 1, we get a copy of P' for some $P' \in A_{ij}^{+}$. Since column j is the only one with two nonzero entries, both (i, j) and (s_j, j) must occur in P. Then, by listing out the row-indices of this copy from left to right, putting the one for (i, j) before the one for (s_j, j), we get a copy of u. Finally, we similarly get that the first implies that an i before the s_j induces a copy of u in s.

Similarly, condition (5.8) implies that inserting an i after s_j induces a copy of u, and if $s_j > i$, we have a similar proof. Thus, s is u-saturated. Note that (5.13) implies that s is nontrivial. Furthermore, it is clear that any u-saturated sequence has length $\geq \ell - 1 \geq r - 1$, so by an appropriate renaming of the letters in s, we can make (5.13) be satisfied.

Likewise, for any u-saturated sequence s, we can define numbers $x_{i,j}$ by reversing the above process. Then conditions (5.4) and (5.5) uniquely determine the $y_{P'}$, and using the fact that s is u-saturated, we can simply reverse the above arguments to show that the remaining conditions are satisfied. Since the length of s is equal to $\sum x_{i,j}$, this completes the proof. □

The choice of the integer N above can be found by trial and error (or by finding some u-saturated sequence with length N).

u	n	$\mathrm{Sat}(n,u)$	u-saturated sequence
aba	2	2	12
aba	3	3	123
aba	4	4	1234
$abab$	2	3	121
$abab$	2	5	12321
$aabb$	2	5	12121
$aabb$	3	7	1232123
$aaab$	2	5	12121
$aaab$	3	7	1213231
$ababa$	2	4	1212
$ababa$	3	7	1212323
$ababb$	2	5	12121
$ababb$	3	7	1212313
$ababa$	2	4	1212
$ababa$	3	7	1212323
$abca$	3	3	123
$ababa$	4	4	1234
$abcb$	3	3	1231
$abcb$	4	5	12341
$abcc$	3	5	12312
$abcc$	4	6	123142
$abcba$	3	3	1231
$abcba$	4	5	12341

Table 8.1. Exact Values of $\mathrm{Sat}(n,u)$

Using the SAGE implementation of this linear program in [10], we get the results in Table 8.1. Note that when $n = r$ is the number of distinct letters of u, we see that $\mathrm{Sat}(r,u)$ is simply the length of $s_r(u)$. This is easy to prove in general, because any r-sparse sequence on $[r]$ is isomorphic to a sequence of the form $12\dots r12\dots$.

It is not feasible to use this linear program to compute $\mathrm{Sat}(n,u)$ when u is long or n is large, because that means that N will be large, implying that the number of variables blows up. We wonder if the algorithm can be improved to compute more nontrivial values of $\mathrm{Sat}(n,u)$.

Acknowledgements

I am grateful to my mentor, Jesse Geneson, for proposing this project and for his guidance during the preparation of this paper. I thank James Alitzer for collaborating on the linear program in Section 5. While working on this paper, I was part of MIT PRIMES-USA, a year-long math research program hosted by MIT. I would like to express my gratitude to the PRIMES program and its organizers for making this experience possible.

References

[1] Anand et al. "Sequence saturation". In: *Discrete Applied Mathematics* 360 (2025), pp. 382–393.

[2] Mikhail J. Atallah. "Some dynamic computational geometry problems". In: *Computers and Mathematics with Applications* 11.12 (1985), pp. 1171–1181.

[3] Benjamin Aram Berendsohn. "An exact characterization of saturation for permutation matrices". In: *Comb. Theory* 3 (2021), #17.

[4] Andrew Brahms et al. *Saturation of 0-1 matrices*. 2025. arXiv: `2503.03193 [math.CO]`.

[5] Gábor Damásdi et al. "Saturation problems in the Ramsey theory of graphs, posets and point sets". In: *European Journal of Combinatorics* 95 (2021), p. 103321.

[6] Harold Davenport and Andrzej Schinzel. "A Combinatorial Problem Connected with Differential Equations". In: *American Journal of Mathematics* 87.3 (1965), pp. 684–694.

[7] Michael Ferrara et al. "The saturation number of induced subposets of the Boolean lattice". In: *Discret. Math.* 340 (2017), pp. 2479–2487.

[8] Radoslav Fulek and Balázs Keszegh. "Saturation Problems about Forbidden 0-1 Submatrices". In: *SIAM J. Discret. Math.* 35 (2020), pp. 1964–1977.

[9] Jesse Geneson. "Almost all Permutation Matrices have Bounded Saturation Functions". In: *Electron. J. Comb.* 28 (2020), P2.16.

[10] Shihan Kanungo. Sequence Saturation Code (2025), online at `github`.

[11] László Kászonyi and Zs. Tuza. "Saturated graphs with minimal number of edges". In: *Journal of Graph Theory* 10 (1986), pp. 203–210.

[12] Balzá's Keszegh. "On linear forbidden submatrices". In: *Journal of Combinatorial Theory, Series A* 116.1 (2009), pp. 232–241.

[13] Seth Pettie. "Generalized Davenport–Schinzel sequences and their 0–1 matrix counterparts". In: *Journal of Combinatorial Theory, Series A* 118.6 (2011), pp. 1863–1895.

San José State University San José, CA 95192
Email address: `shihan.kanungo@sjsu.edu`