AUSTRALASIAN JOURNAL OF COMBINATORICS
Volume **77(1)** (2020), Pages 51–68

Stack-sorting for words

Colin Defant

Fine Hall, 304 Washington Rd.
Princeton, NJ 08544
U.S.A.
cdefant@princeton.edu

Noah Kravitz

Grace Hopper College
Yale University, New Haven, CT 06510
U.S.A.
noah.kravitz@yale.edu

Abstract

We introduce operators hare and tortoise, which act on words as natural generalizations of West's stack-sorting map. We show that the heuristically slower algorithm tortoise can sort words arbitrarily faster than its counterpart hare. We relate the question of determining which words are sortable by hare and tortoise to more classical problems in pattern avoidance, and we derive a recurrence for the number of words with a fixed number of copies of each letter (permutations of a multiset) that are sortable by each map. In particular, we prove that the ℓ-uniform words on the alphabet $[n]$ that avoid the patterns 231 and 221 are counted by the $(\ell+1)$-Catalan number $\frac{1}{\ell n+1}\binom{(\ell+1)n}{n}$. We conclude with several open problems and conjectures.

1 Introduction

1.1 Background

Throughout this paper, the term *word* refers to a finite string of letters taken from the alphabet $\mathbb{N}$ of positive integers. Given a word $p = p_1 \cdots p_k$, we say a word $w = w_1 \cdots w_n$ *contains the pattern* p if there are indices $i_1 < \cdots < i_k$ such that $w_{i_1} \cdots w_{i_k}$ has the same relative order as p. Otherwise, we say w *avoids* p. For example, 3422155 contains the pattern 211 because the letters 322 have the same relative order in w as 211. On the other hand, 3422155 avoids the pattern 4321.

ISSN: 2202-3518

A *permutation* is a word in which no letter appears more than once; it is in the context of permutations that pattern avoidance has received the most attention. Let S_n denote the set of permutations whose letters are the elements of the set $\{1, \ldots, n\}$. The study of pattern avoidance in permutations originated in Knuth's monograph *The Art of Computer Programming* [30]. Knuth defined a sorting algorithm that makes use of a vertical *stack*, and he showed that this algorithm sorts a permutation into increasing order if and only if it avoids the pattern 231. In his 1990 Ph.D. thesis, West [39] introduced a deterministic variant of Knuth's algorithm, which we call the *stack-sorting map* and denote by s. This map operates as follows.

Place the input permutation on the right side of a vertical stack. At each point in time, if the stack is empty or the leftmost entry on the right side of the stack is smaller than the entry at the top of the stack, *push* that leftmost entry into the stack. If there is no entry on the right of the stack or if the leftmost entry on the right side of the stack is larger than the entry on the top of the stack, *pop* the top entry out of the stack and add it to the end of the growing output permutation to the left of the stack. Let $s(\pi)$ denote the output permutation that is obtained by sending π through the stack. Figure 1 illustrates this procedure for $s(4162) = 1426$.

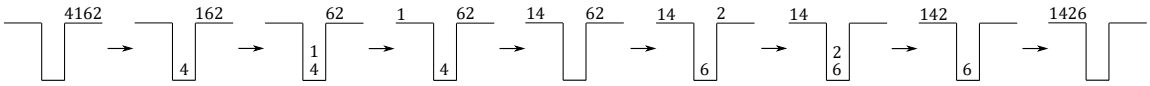

Figure 1: The stack-sorting map s sends 4162 to 1426.

If π is a permutation with largest entry n, we can write $\pi = LnR$, where L (respectively, R) is the (possibly empty) substring of π to the left (respectively, right) of the entry n. West observed that the stack-sorting map can be defined recursively by $s(\pi) = s(L)s(R)n$. It is also possible to define the map s in terms of tree traversals of decreasing binary plane trees (see [5]).

We do not attempt to give a comprehensive treatment of the extensive literature concerning the stack-sorting map s. Instead, we provide the background that is immediately relevant to our investigations and refer the interested reader to [5, 6, 18, 13] (and the references therein) for further information.

A permutation $\pi \in S_n$ is called *t-stack-sortable* if $s^t(\pi) = 123 \cdots n$, where s^t denotes the composition of s with itself t times. A 1-stack-sortable permutation is simply called *sortable*. It follows from Knuth's work that a permutation is sortable if and only if it avoids the pattern 231. According to the well-known enumeration of 231-avoiding permutations, there are C_n sortable permutations in S_n, where $C_n = \frac{1}{n+1}\binom{2n}{n}$ is the n-th Catalan number. West [39] conjectured that there are exactly

$$\frac{2}{(n+1)(2n+1)}\binom{3n}{n}$$

2-stack-sortable permutations in S_n, and Zeilberger [40] later proved this fact.

Much of the study of the stack-sorting map can be phrased in terms of preimages of permutations under s. In fact, the study of stack-sorting preimages of permutations dates back to West [39], who called $|s^{-1}(\pi)|$ the *fertility* of the permutation π and computed this fertility for some specific types of permutations. Bousquet-Mélou [7] later studied permutations with positive fertilities, which she called *sorted* permutations. In doing so, she asked for a method for computing the fertility of any given permutation. The first author achieved this in much greater generality in [17] and [18] by developing a theory of new combinatorial objects called *valid hook configurations*. These objects have since been applied to understand the stack-sorting map (see [14, 15, 17, 18, 26] and the references therein), and they have been studied as combinatorial objects in their own right in [16] and [33].

1.2 Definition of Stack-Sorting for Words

Several authors have extended the well-studied area of pattern avoidance in permutations to pattern avoidance in words [1, 2, 8, 9, 10, 27, 28, 29, 31, 32, 34]. One motivation for this line of inquiry comes from the study of sorting algorithms defined on words [1, 2]. The first order of business in this article is to extend West's stack-sorting map s so that it can operate on words. There is one point of ambiguity in how one defines this extension: should a letter be allowed to sit on top of a copy of itself in the stack? If, for instance, we send the word 221 through the stack, we want to know if the second 2 forces the first 2 to pop out of the stack. Depending on which convention is used, the output permutation could either be 122 or 212; we avoid this potential issue by considering both variations. With this background in mind, we offer the following recursive definitions of the functions hare and $\mathsf{tortoise}$ from the set of all words to itself.

Definition 1.1. First, let $\mathsf{hare}(\varepsilon) = \mathsf{tortoise}(\varepsilon) = \varepsilon$, where ε is the empty word. Now, suppose w is a nonempty word with largest letter n. If the letter n appears k times in w, then we can uniquely write $w = A_1 n A_2 n \cdots n A_{k+1}$, where the letters in the (possibly empty) words A_i are all at most $n-1$. We now define

$$\mathsf{hare}(w) = \mathsf{hare}(A_1)\,\mathsf{hare}(A_2)\cdots\mathsf{hare}(A_{k+1})nn\cdots n,$$

where there are exactly k copies of the letter n at the end of the word, and

$$\mathsf{tortoise}(w) = \mathsf{tortoise}(A_1)\,\mathsf{tortoise}(A_2)n\,\mathsf{tortoise}(A_3)n\cdots n\,\mathsf{tortoise}(A_k)n\,\mathsf{tortoise}(A_{k+1})n.$$

The map hare operates by sending a word through the stack with the convention that a letter *can* sit on top of a copy of itself in the stack. On the other hand, $\mathsf{tortoise}$ operates by sending a word through the stack with the convention that a letter *cannot* sit on top of a copy of itself. The main purpose of this article is to compare the functions hare and $\mathsf{tortoise}$ and to explore some of their properties.

1.3 Notation

We require the following notation in order to state our main results.

- Let $\mathcal{W}$ denote the set of all words of finite length over the alphabet $\mathbb{N}$. This set is a monoid with concatenation as its binary operation. As such, $A_1 \cdots A_k$ denotes the concatenation of the words $A_1, \ldots, A_k$. We will often speak of a word $w = w_1 \cdots w_m$; unless otherwise stated, $w_1, \ldots, w_m$ are assumed to be the *letters* of the word w (so w has length m).

- Given a tuple $\mathbf{c} = (c_1, \ldots, c_n)$ of nonnegative integers, let $\mathcal{W}_\mathbf{c}$ be the set of all words with exactly c_i i's for each $1 \leq i \leq n$. One can think of $\mathcal{W}_\mathbf{c}$ as the set of permutations of the multiset $\{1^{c_1}, 2^{c_2}, \ldots, n^{c_n}\}$.

- Let $\mathrm{Id}_\mathbf{c}$ be the unique word in $\mathcal{W}_\mathbf{c}$ whose letters are nondecreasing from left to right. By abuse of terminology, we call $\mathrm{Id}_\mathbf{c}$ the *identity word* in $\mathcal{W}_\mathbf{c}$. We will omit the subscript when it is obvious from context.

- We call a word *normalized* if it is an element of $\mathcal{W}_\mathbf{c}$ for some vector $\mathbf{c} = (c_1, \ldots, c_n)$ in which each c_i is strictly positive. For example, 31341 is not normalized because it does not contain the letter 2.

- Let hare^k denote the map hare composed with itself k times, and define $\mathsf{tortoise}^k$ similarly. Given a word $w \in \mathcal{W}_\mathbf{c}$, let $\langle w \rangle_{\mathsf{hare}}$ be the smallest nonnegative integer k such that $\mathsf{hare}^k(w) = \mathrm{Id}_\mathbf{c}$. Similarly, let $\langle w \rangle_{\mathsf{tortoise}}$ be the smallest nonnegative integer k such that $\mathsf{tortoise}^k(w) = \mathrm{Id}_\mathbf{c}$. In particular, put $\langle \varepsilon \rangle_{\mathsf{hare}} = \langle \varepsilon \rangle_{\mathsf{tortoise}} = 0$. These values measure how "far" w is from the identity word under our generalized stack-sorting maps.

1.4 Outline of the Paper

The operators hare and $\mathsf{tortoise}$ get their names from the heuristic idea that iteratively applying the map hare to a word should produce an identity word at least as fast as iteratively applying $\mathsf{tortoise}$ does. More formally, it seems reasonable to expect that $\langle w \rangle_{\mathsf{hare}} \leq \langle w \rangle_{\mathsf{tortoise}}$ for every word w. For example, $\langle 2221 \rangle_{\mathsf{hare}} = 1 < 3 = \langle 2221 \rangle_{\mathsf{tortoise}}$. However, in some special cases, we find the fable had it right: slow and steady wins the race! That is, there exist words w for which $\langle w \rangle_{\mathsf{hare}} > \langle w \rangle_{\mathsf{tortoise}}$. In Section 2, we will construct a word η_n of length $2n+1$ such that $\langle \eta_n \rangle_{\mathsf{hare}} = 2n-2$ and $\langle \eta_n \rangle_{\mathsf{tortoise}} = n$ for each positive integer n. In the same section, we show how to rewrite these maps in terms of West's stack-sorting map s and also analyze the worst-case-scenario sorting for each map.

In Section 3, we show that a word $w \in \mathcal{W}_\mathbf{c}$ satisfies $\mathsf{hare}(w) = \mathrm{Id}_\mathbf{c}$ if and only if it avoids the pattern 231 and satisfies $\mathsf{tortoise}(w) = \mathrm{Id}_\mathbf{c}$ if and only if it avoids the patterns 231 and 221. We discuss known results concerning words that avoid the pattern 231 and present new enumerative results concerning words that avoid the patterns 231 and 221. Specifically, we provide a recurrence for $\mathcal{N}(c_1, \ldots, c_n)$, the number of words in $\mathcal{W}_{(c_1,\ldots,c_n)}$ that avoid 231 and 221. We also use generating trees to prove that

$$\mathcal{N}(\underbrace{\ell, \ell, \ldots, \ell}_{n}) = \frac{1}{\ell n + 1} \binom{(\ell+1)n}{n}.$$

In Section 4, we study what we call hare-fertility numbers and tortoise-fertility numbers. Specifically, we show that for every nonnegative integer f, there exists a word w such that $|\,\mathsf{hare}^{-1}(w)| = |\,\mathsf{tortoise}^{-1}(w)| = f$. As demonstrated in [15], this result is false if we require our words to be permutations.

In Section 5, we list several open problems and conjectures.

Remark 1.2. Given the effectiveness of valid hook configurations for understanding the stack-sorting map s, we believe that it is useful to develop an analogous theory of valid hook configurations for words. The main difficulty is figuring out how to modify the original definitions from the context of permutations; the proofs of the corresponding "fertility formulas" for hare and tortoise then follow the same basic arguments as those in [17]. We carried out these fairly dense and tedious arguments in a previous version of this paper, but we have removed them at the request of one of the referees. The reader seeking additional details about valid hook configurations for words can consult the preprint of this article on arXiv.org [20]. We also note that it could be useful to have a "decomposition lemma" similar to the one in [13].

2 The Tortoise and the Hare

We begin this section by recasting hare and tortoise explicitly in terms of the action of West's stack-sorting map s. Given a vector $\mathbf{c} = (c_1, \ldots, c_n)$ of nonnegative integers, define the maps $\phi_{\mathbf{c}}^{asc}, \phi_{\mathbf{c}}^{des} : \mathcal{W}_{\mathbf{c}} \to S_{c_1+\cdots+c_n}$ (recall that S_m is the set of permutations of $\{1, \ldots, m\}$) as follows. For each $1 \leq i \leq n$, let $p_i = c_1 + \cdots + c_{i-1}$. To obtain $\phi_{\mathbf{c}}^{asc}(w)$ from w, we replace the i's by the integers $p_i+1, p_i+2, \ldots, p_i+c_i$ in ascending order for each i. (In other words, the j-th occurrence of i becomes $p_i + j$.) To obtain $\phi_{\mathbf{c}}^{des}(w)$, we replace the i's by the integers $p_i + 1, p_i + 2, \ldots, p_i + c_i$ in descending order for each i. (In other words, the j-th occurrence of i becomes $p_i + c_i + 1 - j$.) Note that even though these maps are not surjective if any $c_i > 1$, they are always injective. We define the map $\psi_{\mathbf{c}} : S_{c_1+\cdots+c_n} \to \mathcal{W}_{\mathbf{c}}$ as follows. To obtain $\psi_{\mathbf{c}}(\pi)$ from π, we replace all of the digits $p_i + 1, p_i + 2, \ldots, p_i + c_i$ by the letter i for each $1 \leq i \leq n$. Clearly, $\psi_{\mathbf{c}} \circ \phi_{\mathbf{c}}^{asc} = \psi_{\mathbf{c}} \circ \phi_{\mathbf{c}}^{des} : \mathcal{W}_{\mathbf{c}} \to \mathcal{W}_{\mathbf{c}}$ is the identity map. Similarly, $\phi_{\mathbf{c}}^{asc} \circ \psi_{\mathbf{c}} : \mathrm{Im}(\phi_{\mathbf{c}}^{asc}) \to \mathrm{Im}(\phi_{\mathbf{c}}^{asc})$ and $\phi_{\mathbf{c}}^{des} \circ \psi_{\mathbf{c}} : \mathrm{Im}(\phi_{\mathbf{c}}^{des}) \to \mathrm{Im}(\phi_{\mathbf{c}}^{des})$ are both the identity map (restricted to the correct subset of $S_{c_1+\cdots+c_n}$). Consequently, $\psi_{\mathbf{c}}$ is a left inverse for both $\phi_{\mathbf{c}}^{asc}$ and $\phi_{\mathbf{c}}^{des}$.

As an example, $\phi_{(2,2,3)}^{asc}(3313221) = 5617342$ and $\phi_{(2,2,3)}^{des}(3313221) = 7625431$. We emphasize that $\psi_{\mathbf{c}}$ depends strongly on $\mathbf{c}$. For example, $\psi_{(2,2,3)}(5617342) = 3313221$ as expected, whereas $\psi_{(2,3,2)}(5617342) = 2313221$ and $\psi_{(6,1)}(5617342) = 1112111$. The following lemma reduces the computation of hare and tortoise to computations involving s.

Proposition 2.1. *For every word $w \in \mathcal{W}_{\boldsymbol{c}}$, we have*

$$\mathsf{hare}(w) = (\psi_{\boldsymbol{c}} \circ s \circ \phi_{\boldsymbol{c}}^{des})(w) \quad \textit{and} \quad \mathsf{tortoise}(w) = (\psi_{\boldsymbol{c}} \circ s \circ \phi_{\boldsymbol{c}}^{asc})(w).$$

Moreover, for every positive integer k, we have

$$\mathsf{tortoise}^k(w) = (\psi_{\boldsymbol{c}} \circ s^k \circ \phi_{\boldsymbol{c}}^{asc})(w).$$

Proof. Fix a word $w \in \mathcal{W}_{\mathbf{c}}$. For the first statement, consider the permutation $\phi_{\mathbf{c}}^{des}(w)$. We may associate each entry of $\phi_{\mathbf{c}}^{des}(w)$ with the letter that appears in the corresponding position in w. If we keep track of the positions of individual entries and letters when we apply s to $\phi_{\mathbf{c}}^{des}(w)$ and hare to w, we see that the corresponding entries and letters enter the stack and pop out of the stack identically. Hence, when we apply $\psi_{\mathbf{c}}$ to $s(\phi_{\mathbf{c}}^{des}(w))$, each entry is taken to the correct letter value in $\mathsf{hare}(w)$. This shows that $\mathsf{hare}(w) = (\psi_{\mathbf{c}} \circ s \circ \phi_{\mathbf{c}}^{des})(w)$. The same argument shows that $\mathsf{tortoise}(w) = (\psi_{\mathbf{c}} \circ s \circ \phi_{\mathbf{c}}^{asc})(w)$.

For the second statement, it suffices to note that s maps $\mathrm{Im}(\phi_{\mathbf{c}}^{asc})$ into itself.[1] This follows from the simple observation that if $a < b$ and a appears before b in a permutation π, then a appears before b in $s(\pi)$. Indeed, $\mathrm{Im}(\phi_{\mathbf{c}}^{asc})$ is precisely the set of permutations in S_m such that the entries $p_i + 1, p_i + 2, \ldots, p_i + c_i$ appear in increasing order in π for every $i \in \{1, \ldots, n\}$. □

The maps hare and $\mathsf{tortoise}$ do in fact "sort" words in the sense that iterative applications of either map to any word will eventually reach an identity word, which is a fixed point. A natural question is how many iterations it takes to reach this fixed point. Recall that $\langle\cdot\rangle_{\mathsf{hare}}$ and $\langle\cdot\rangle_{\mathsf{tortoise}}$ measure this "distance" from the identity. In each $\mathcal{W}_{\mathbf{c}}$, this metric equals 0 for only the identity word, and it equals 1 for the nonidentity words that are completely sorted in a single pass.

Intuitively, hare should be the more efficient sorting algorithm because a later occurrence of a large letter value does not cause the previous occurrences to pop out of the stack prematurely. It is easy to show that worst-case-scenario sorting with hare is much more efficient than worst-case-scenario sorting with $\mathsf{tortoise}$. Observe that if w is a word with largest letter n, then all of the n's are at the very end of $\mathsf{hare}(w)$, whereas only one n is guaranteed to be at the end of $\mathsf{tortoise}(w)$. In fact, this "rate of progress" is the worst-case scenario for each map. In this sense, hare is faster than $\mathsf{tortoise}$.

Proposition 2.2. *Let $\boldsymbol{c} = (c_1, \ldots, c_n)$, where $c_1, \ldots, c_n$ are positive integers. For every $w \in \mathcal{W}_{\boldsymbol{c}}$, we have*

$$\langle w\rangle_{\mathsf{hare}} \leq n - 1 \quad \textit{and} \quad \langle w\rangle_{\mathsf{tortoise}} \leq c_2 + c_3 + \cdots + c_n.$$

Moreover, equality is achieved in both cases by the word $\rho \in \mathcal{W}_{\boldsymbol{c}}$ that is obtained from $\mathrm{Id}_{\boldsymbol{c}}$ by moving all of the 1's to the end of the word.

Proof. The inequalities follow directly from the observation above. Let us prove the second part of the lemma. By definition, $\rho = 2^{c_2}3^{c_3}\cdots n^{c_n}1^{c_1}$. Induction on k shows that

$$\mathsf{hare}^k(\rho) = 2^{c_2}3^{c_3}\cdots(n-k)^{c_{n-k}}1^{c_1}(n-k+1)^{c_{n-k+1}}\cdots n^{c_n}$$

[1] It is easy to see that $(s \circ \phi_{\mathbf{c}}^{des})(w) \notin \mathrm{Im}(\phi_{\mathbf{c}}^{des})$ if two letters of w with the same value are ever in the stack simultaneously during the hare-sorting process. For this reason, one cannot obtain an analogous simple formula for iterations of hare.

for each $0 \leq k \leq n-1$. Hence, $\langle \rho \rangle_{\mathsf{hare}} = n-1$. Similarly, each iterative application of tortoise to ρ moves the letter directly to the left of the 1's to the position directly to the right of the 1's (which stay together). Hence, $\langle \rho \rangle_{\mathsf{tortoise}} = c_2 + \cdots + c_n$. □

In light of the previous lemma, one might expect hare to sort all words faster than tortoise, i.e., $\langle w \rangle_{\mathsf{hare}} \leq \langle w \rangle_{\mathsf{tortoise}}$. However, this turns out not always to happen: even though hare seems to make more progress in the first few iterations, sometimes tortoise catches up and reaches the identity first! For example, we have

$$3662451 \xrightarrow{\mathsf{hare}} 3241566 \xrightarrow{\mathsf{hare}} 2314566 \xrightarrow{\mathsf{hare}} 2134566 \xrightarrow{\mathsf{hare}} 1234566$$

and

$$3662451 \xrightarrow{\mathsf{tortoise}} 3624156 \xrightarrow{\mathsf{tortoise}} 3214566 \xrightarrow{\mathsf{tortoise}} 1234566,$$

so

$$\langle 3662451 \rangle_{\mathsf{hare}} = 4 > 3 = \langle 3662451 \rangle_{\mathsf{tortoise}}.$$

The following theorem shows that tortoise can actually be arbitrarily faster than hare.

Theorem 2.3. *For any integer $n \geq 3$, the word*

$$\eta_n = 357 \cdots (2n-3)(2n)(2n)246 \cdots (2n-2)(2n-1)1$$

has length $2n+1$ and satisfies

$$\langle \eta_n \rangle_{\mathsf{hare}} = 2n-2 \quad \textit{and} \quad \langle \eta_n \rangle_{\mathsf{tortoise}} = n.$$

Proof. The proof of the theorem amounts to observing what happens to η_n under repeated applications of hare and tortoise. One could write out these calculations for general n, but we fear that doing so would only obfuscate the computations with a sea of ellipses ($\cdots$). Instead, we show the calculations for the case $n = 5$; the general case is completely analogous. For hare, note that the 1 moves only a single entry closer to the beginning of the word with each iteration after the first; for tortoise, note that the 1 moves two entries forward and that two more entries at the end become sorted with each iteration (after the second application).

We have $\eta_5 = 3\ 5\ 7\ 10\ 10\ 2\ 4\ 6\ 8\ 9\ 1$. Now,

$$\begin{array}{c} 3\ 5\ 7\ 10\ 10\ 2\ 4\ 6\ 8\ 9\ 1 \\ \downarrow \mathsf{hare} \\ 3\ 5\ 7\ 2\ 4\ 6\ 8\ 1\ 9\ 10\ 10 \\ \downarrow \mathsf{hare} \\ 3\ 5\ 2\ 4\ 6\ 7\ 1\ 8\ 9\ 10\ 10 \\ \downarrow \mathsf{hare} \\ 3\ 2\ 4\ 5\ 6\ 1\ 7\ 8\ 9\ 10\ 10 \end{array} \qquad \begin{array}{c} 3\ 5\ 7\ 10\ 10\ 2\ 4\ 6\ 8\ 9\ 1 \\ \downarrow \mathsf{tortoise} \\ 3\ 5\ 7\ 10\ 2\ 4\ 6\ 8\ 1\ 9\ 10 \\ \downarrow \mathsf{tortoise} \\ 3\ 5\ 7\ 2\ 4\ 6\ 1\ 8\ 9\ 10\ 10 \\ \downarrow \mathsf{tortoise} \\ 3\ 5\ 2\ 4\ 1\ 6\ 7\ 8\ 9\ 10\ 10 \end{array}$$

$$\begin{array}{c} \\ \downarrow \text{hare} \\ 2\,3\,4\,5\,1\,6\,7\,8\,9\,10\,10 \\ \downarrow \text{hare} \\ 2\,3\,4\,1\,5\,6\,7\,8\,9\,10\,10 \\ \downarrow \text{hare} \\ 2\,3\,1\,4\,5\,6\,7\,8\,9\,10\,10 \\ \downarrow \text{hare} \\ 2\,1\,3\,4\,5\,6\,7\,8\,9\,10\,10 \\ \downarrow \text{hare} \\ 1\,2\,3\,4\,5\,6\,7\,8\,9\,10\,10 \end{array} \qquad \begin{array}{c} \\ \downarrow \text{tortoise} \\ 3\,2\,1\,4\,5\,6\,7\,8\,9\,10\,10 \\ \downarrow \text{tortoise} \\ 1\,2\,3\,4\,5\,6\,7\,8\,9\,10\,10. \end{array}$$

□

Say a word w is *exceptional* if $\langle w \rangle_{\mathsf{hare}} > \langle w \rangle_{\mathsf{tortoise}}$. Let $\mathcal{E}_m$ be the set of exceptional normalized words of length m. It turns out that $\mathcal{E}_m = \emptyset$ when $m \leq 6$. We have used a computer to find that

$$\mathcal{E}_7 = \{3662451, 3664251, 6362451, 6364251\}.$$

The sets $\mathcal{E}_8$ and $\mathcal{E}_9$ have 172 and 5001 elements, respectively. Furthermore, we have checked that each element of $\mathcal{E}_8$ contains one of the words in $\mathcal{E}_7$ as a pattern. We have also found that there are 72 words w of length 9 (but no shorter words) that satisfy $\langle w \rangle_{\mathsf{hare}} = \langle w \rangle_{\mathsf{tortoise}} + 2$. These observations lead to a host of questions concerning exceptional words, many of which we list in Section 5.

3 Sortable Words

The t-stack-sortable permutations mentioned in the introduction have received a large amount of attention [5, 6, 18, 39, 40]. We define a t-hare-sortable word to be a word w such that $\mathsf{hare}^t(w)$ is an identity word. In other words, it is a word w such that $\langle w \rangle_{\mathsf{hare}} \leq t$. We define t-tortoise-sortable words similarly. Our goal in this section is to investigate the 1-hare-sortable words and 1-tortoise-sortable words. For brevity, we call these words hare-*sortable* and tortoise-*sortable*, respectively.

Recall that a permutation is sortable if and only if it avoids the pattern 231. We begin with the corresponding characterization for sortable words.

Proposition 3.1. *A word is* hare-*sortable if and only if it avoids the pattern* 231*. A word is* tortoise-*sortable if and only if it avoids the patterns* 231 *and* 221*.*

Proof. We prove the contrapositive of each statement. Let $w = w_1 w_2 \cdots w_m$. First, suppose w contains the pattern 231, i.e., there exist $1 \leq a < b < c \leq m$ such that $w_c < w_a < w_b$. Consider the action of hare on w. Because $w_a < w_b$, it is clear that

w_b will force w_a to pop out of the stack if it has not already left the stack, and this occurs before w_c even enters the stack. Hence, w_a precedes w_c in $\mathsf{hare}(w)$, which implies that $\mathsf{hare}(w) \neq \mathrm{Id}$. Second, suppose $\mathsf{hare}(w) = w'_1 w'_2 \cdots w'_m \neq \mathrm{Id}$. Then there exist $1 \leq d < e \leq m-1$ such that $w'_d > w'_e$. (We have the restriction $e \leq m-1$ because no letter is larger than w'_m.) The letter w'_d must have exited the stack before w'_e could even enter it. Let w'_f be the letter that forces w'_d to pop out of the stack. We must have $w'_f > w'_d > w'_e$. Furthermore, these letters must appear in the order w'_d, w'_f, w'_e in w, which means that these three letters form a 231 pattern in w. This establishes the first statement.

The proof of the second statement proceeds in a similar manner. The only difference is that we replace the inequalities $w_a < w_b$ and $w'_d < w'_f$ by $w_a \leq w_b$ and $w'_d \leq w'_f$. □

This proposition yields an immediate comparison between $|\mathsf{hare}^{-1}(\mathrm{Id}_{\mathbf{c}})|$ and $|\mathsf{tortoise}^{-1}(\mathrm{Id}_{\mathbf{c}})|$ for various vectors $\mathbf{c} = (c_1, \ldots, c_n)$; the result holds particular interest in light of the discussion of Section 2.

Corollary 3.2. *For any $\boldsymbol{c} = (c_1, \ldots, c_n)$, where $c_1, \ldots, c_n$ are positive integers, we have*

$$\mathsf{tortoise}^{-1}(\mathrm{Id}_{\boldsymbol{c}}) \subseteq \mathsf{hare}^{-1}(\mathrm{Id}_{\boldsymbol{c}}).$$

Moreover, equality holds exactly when $c_i = 1$ for all $i > 1$.

Proof. Fix some $\mathbf{c} = (c_1, \ldots, c_n)$. Since any word $w \in \mathsf{tortoise}^{-1}(\mathrm{Id}_{\mathbf{c}})$ avoids the patterns 231 and 221, it is also in $\mathsf{hare}^{-1}(\mathrm{Id}_{\mathbf{c}})$. This establishes the desired containment. Now, suppose $c_i = 1$ for all $i > 1$. Then it is impossible for any word $w \in \mathcal{W}_{\mathbf{c}}$ to contain the pattern 221, so the conditions for $w \in \mathcal{W}_{\mathbf{c}}$ being in $\mathsf{hare}^{-1}(\mathrm{Id}_{\mathbf{c}})$ and $\mathsf{tortoise}^{-1}(\mathrm{Id}_{\mathbf{c}})$ are equivalent. We can conclude that $\mathsf{hare}^{-1}(\mathrm{Id}_{\mathbf{c}}) = \mathsf{tortoise}^{-1}(\mathrm{Id}_{\mathbf{c}})$ in this case. Finally, suppose that $c_i \geq 2$ for some $i > 1$. Consider the word $w \in \mathcal{W}_{\mathbf{c}}$ that is obtained from $\mathrm{Id}_{\mathbf{c}}$ by moving all of the $i-1$'s to the right of the i's. Since w contains the pattern 221 but not the pattern 231, it is in $\mathsf{hare}^{-1}(\mathrm{Id}_{\mathbf{c}})$ but not in $\mathsf{tortoise}^{-1}(\mathrm{Id}_{\mathbf{c}})$. Hence, $\mathsf{hare}^{-1}(\mathrm{Id}_{\mathbf{c}})$ strictly contains $\mathsf{tortoise}^{-1}(\mathrm{Id}_{\mathbf{c}})$ in this case. □

We devote the remainder of this section to enumerating the hare-sortable and $\mathsf{tortoise}$-sortable words. According to Proposition 3.1, this is equivalent to the more classical problem of enumerating the words that avoid 231 and the words that avoid both 231 and 221.

Let us focus first on hare. In its most general form, our problem is to find a formula depending on $c_1, \ldots, c_n$ for the number of hare-sortable words in $\mathcal{W}_{(c_1,\ldots,c_n)}$. An explicit formula seems unattainable in this level of generality, but we can at least obtain a recurrence. In fact, this has already been done. Because of Proposition 3.1, the following theorem is equivalent to Lemma 3 in [2].

Theorem 3.3 ([2])**.** *For nonnegative integers $c_1, \ldots c_n$, let*

$$\mathcal{M}(c_1, \ldots, c_n) = |\mathsf{hare}^{-1}(\mathrm{Id}_{(c_1,\ldots,c_n)})|$$

denote the number of hare*-sortable words in* $\mathcal{W}_{(c_1,\ldots,c_n)}$*. We have* $\mathcal{M}(c_1) = 1$ *for all choices of* c_1*. For* $n \geq 2$*, we have*

$$\mathcal{M}(c_1,\ldots,c_n) = \begin{cases} \mathcal{M}(c_1+c_2,c_3,\ldots,c_n) + \sum_{r=1}^{c_1} \mathcal{M}(r,c_2-1,c_3,\ldots,c_n) & \text{if } c_2 \geq 1 \\ \mathcal{M}(c_1,c_3,\ldots,c_n) & \text{if } c_2 = 0. \end{cases}$$

The authors of [2] used Theorem 3.3 to find an explicit formula for the generating function of $\mathcal{M}(c_1,\ldots,c_n)$. Specifically, given variables $x_1, x_2, \ldots$, let $y_i = x_i(1-x_i)$. Let $A(z_1,\ldots,z_m) = \prod_{1\leq i<j\leq m}(z_i - z_j)$. The following theorem is Theorem 3 in [2].

Theorem 3.4 ([2])**.** *In the above notation, we have*

$$\sum_{a_1,\ldots,a_n\geq 0} \mathcal{M}(a_1,\ldots,a_n)x_1^{a_1}\cdots x_n^{a_n} = -\frac{\sum_{i=1}^{n}(-1)^i x_i y_i^{n-2} A(y_1,\ldots,y_{i-1},y_{i+1}\ldots,y_n)}{A(y_1,\ldots,y_n)}.$$

As a corollary of Theorem 3.4, the authors of [2] proved the surprising fact that $\mathcal{M}(c_1,\ldots,c_n)$ is a symmetric function of the arguments $c_1,\ldots,c_n$. That is, for any permutation $\sigma_1\cdots\sigma_n \in S_n$,

$$\mathcal{M}(c_1,\ldots,c_n) = \mathcal{M}(c_{\sigma_1},\ldots,c_{\sigma_n}).$$

We now turn our attention to deriving a recurrence relation for the tortoise-fertility of $\mathrm{Id}_{(c_1,\ldots,c_n)}$. Let $\mathcal{N}(c_1,\ldots,c_n) = |\,\mathsf{tortoise}^{-1}(\mathrm{Id}_{\mathbf{c}})|$ denote the number of tortoise-sortable words in $\mathcal{W}_{(c_1,\ldots,c_n)}$. Equivalently, $\mathcal{N}(c_1,\ldots,c_n)$ is the number of words in $\mathcal{W}_{(c_1,\ldots,c_n)}$ that avoid the patterns 231 and 221. The following theorem reveals $\mathcal{N}(c_1,\ldots,c_n)$ not to depend on the value of c_n.

Theorem 3.5. *For any (strictly) positive integer* c_1*, we have* $\mathcal{N}(c_1) = 1$*. Moreover, for* $n \geq 2$ *and any positive integers* $c_1,\ldots,c_n$*, we have*

$$\begin{aligned}\mathcal{N}(c_1,\ldots,c_n) = 2\mathcal{N}(c_1,\ldots,c_{n-1}) &+ \sum_{i=1}^{n-2} \mathcal{N}(c_1,\ldots,c_i)\mathcal{N}(c_{i+1},\ldots,c_{n-1}) \\ &+ \sum_{i=1}^{n-1}\sum_{k=1}^{c_i-1} \mathcal{N}(c_1,\ldots,c_{i-1},k)\mathcal{N}(c_i-k,c_{i+1},\ldots,c_{n-1}).\end{aligned}$$

Proof. The $n = 1$ case is easy: $\mathcal{W}_{(c_1)}$ consists of only the identity word, which is clearly tortoise-sortable, so $\mathcal{N}(c_1) = 1$.

Now, consider $n \geq 2$. Consider a tortoise-sortable word $w \in \mathcal{W}_{(c_1,\ldots,c_n)}$. Since w avoids the pattern 221, all but one of the n's must be at the very end of w, i.e., $w = AnBnn\cdots n$ (where there are $c_n - 1$ n's appearing at the end) for some (possibly empty) words A and B that do not contain the letter n. We can now compute

$$\mathsf{tortoise}(AnBnn\cdots n) = \mathsf{tortoise}(A)\,\mathsf{tortoise}(B)nnn\cdots n,$$

and this sorted word is the identity exactly when both A and B are tortoise-sortable and no letter of A is larger than a letter of B. Note that $AB \in \mathcal{W}_{(c_1,\ldots,c_{n-1})}$.

If A is empty, then $B \in \mathcal{W}_{(c_1,\ldots,c_{n-1})}$, and by definition there are $\mathcal{N}(c_1,\ldots,c_{n-1})$ possible choices for B. Similarly, if B is empty, then there are $\mathcal{N}(c_1,\ldots,c_{n-1})$ possible choices for A. This pair of possibilities gives the first term in the recurrence relation.

Now, suppose both A and B are nonempty and there is no letter value that appears in both A and B. Then there exists some $1 \leq i \leq n-2$ such that $A \in \mathcal{W}_{(c_1,\ldots,c_i)}$ and $B \in \mathcal{W}_{(0,\ldots,0,c_{i+1},\ldots,c_{n-1})}$ (with i 0's). In this case, there are $\mathcal{N}(c_1,\ldots,c_i)\mathcal{N}(c_{i+1},\ldots,c_{n-1})$ such pairs of sortable words (A,B). Summing over i gives the second term in the recurrence relation.

Finally, consider the case where there is some value $1 \leq i \leq n-1$ that appears in both A and B. Then there exists $1 \leq k \leq c_i - 1$ such that A contains k i's and B contains $c_i - k$ i's. Hence, we have $A \in \mathcal{W}_{(c_1,\ldots,c_{i-1},k)}$ and $B \in \mathcal{W}_{(0,\ldots,0,c_i-k,c_{i+1},\ldots,c_{n-1})}$ (with $i-1$ 0's). As above, there are $\mathcal{N}(c_1,\ldots,c_{i-1},k)\mathcal{N}(c_i-k,c_{i+1},\ldots,c_{n-1})$ such pairs of sortable words (A,B). Summing over i and k gives the third term in the recurrence relation. This exhausts all possibilities. □

Some formulas for $\mathcal{N}(c_1,\ldots,c_n)$ and $\mathcal{M}(c_1,\ldots,c_n)$ for small values of n are as follows:

$$\mathcal{N}(c_1) = 1, \quad \mathcal{N}(c_1,c_2) = c_1 + 1,$$

$$\mathcal{N}(c_1,c_2,c_3) = \frac{1}{2}c_1^2 + c_1c_2 + \frac{3}{2}c_1 + c_2 + 1 = \frac{1}{2}(c_1+1)(c_1+2c_2+2);$$

$$\mathcal{M}(c_1) = 1, \quad \mathcal{M}(c_1,c_2) = \binom{c_1+c_2}{c_1},$$

$$\mathcal{M}(c_1,c_2,c_3) = 2^{c_1+c_2+c_3} - \sum_{i=1}^{3}\sum_{r=0}^{c_i-1}\binom{c_1+c_2+c_3}{r}.$$

Using these formulas, one can show that $\mathcal{N}(c_1,\ldots,c_n)$ grows as c_i^{n-i} in each i (e.g., $\mathcal{N}(c_1,c_2,c_3)$ grows quadratically in c_1 and linearly in c_2). We also remark that this type of argument yields a similar but more complicated recurrence relation for $\mathcal{M}(c_1,\ldots,c_n)$; we do not pursue this line of inquiry here.

Although the general formula in Theorem 3.5 looks complicated, it simplifies in some special cases. In particular, we investigate the ℓ-uniform (normalized) words. These are words in which each letter value that appears in the word appears exactly ℓ times, i.e., $\mathbf{c} = (\ell,\ell,\ldots,\ell)$.

To count these words, we make use of *generating trees*, an enumerative tool that was introduced in [11] and studied extensively afterward [3, 36, 37, 38]. In particular, generating trees have been used to study pattern avoidance in permutations. To describe a generating tree of a class of combinatorial objects, we first specify a scheme by which each object of size n can be uniquely generated from an object of size $n-1$. We then label each object with the number of objects it generates. The generating tree consists of an "axiom" that specifies the labels of the object(s) of size 1 along with a "rule" that describes the labels of the objects generated by each object with a given label. For example, in the generating tree

$$\text{Axiom: } (2) \qquad \text{Rule: } (1) \rightsquigarrow (2), \quad (2) \rightsquigarrow (1)(2),$$

the axiom (2) tells us that we begin with a single object of size 1 that has label 2. The rule $(1) \rightsquigarrow (2),\ (2) \rightsquigarrow (1)(2)$ tells us that each object of size $n-1$ with label 1 generates a single object of size n with label 2, whereas each object of size $n-1$ with label 2 generates one object of size n with label 1 and one object of size n with label 2. This example generating tree describes objects counted by the Fibonacci numbers.

Theorem 3.6. *The number of ℓ-uniform words on the alphabet $[n]$ that avoid the patterns* 231 *and* 221 *is*

$$\mathcal{N}(\underbrace{\ell, \ell, \ldots, \ell}_{n}) = \frac{1}{\ell n + 1}\binom{(\ell+1)n}{n}.$$

Proof. The authors of [3] show (their Example 9) that $(\ell+1)$-ary trees, which are counted by the $(\ell+1)$-Catalan numbers $\frac{1}{\ell n+1}\binom{(\ell+1)n}{n}$, can be described via the generating tree

$$\text{Axiom: } (\ell+1) \qquad \text{Rule: } (m) \rightsquigarrow (\ell+1)(\ell+2)\cdots(\ell+m) \quad \text{for every } m \in \mathbb{N}. \tag{3.1}$$

Fix some positive integer ℓ, and let $\mathcal{P}_\ell(231, 221)$ denote the set of all normalized ℓ-uniform words that avoid the patterns 231 and 221; we will show that these words can be described using the generating tree in (3.1).

Let us say that a word $w' \in \mathcal{P}_\ell(231, 221)$ over the alphabet $[n]$ is generated from a word $w \in \mathcal{P}_\ell(231, 221)$ over the alphabet $[n-1]$ if we can obtain w' by inserting ℓ copies of the letter n into spaces between the letters in w. For example, when $\ell = n = 3$, the word 121122 generates the words

$$312112233, \quad 132112233, \quad 121132233, \quad 121123233, \quad 121122333. \tag{3.2}$$

Because w' avoids 221, the last $\ell - 1$ letters of w' all have value n. Therefore, w' is determined by specifying w along with the position j of the first appearance of the letter n in w'. In the above example, the possible positions j where we could have placed the first appearance of the letter 3 were $1, 2, 5, 6, 7$. In general, we can place the first appearance of n into position j if and only if $1 \le j \le \ell(n-1)+1$ and there do not exist α, β such that $1 \le \alpha < j \le \beta \le \ell(n-1)$ and $w_\alpha > w_\beta$. Indeed, this follows from the requirement that the new word w' avoids 231. We label the word w with the number of such positions j, or, equivalently, the number of words that w generates.

Suppose we are given the word $w \in \mathcal{P}_\ell(231, 221)$ over the alphabet $[n-1]$. Let m be the label of w. Let $j_1 < \cdots < j_m$ be the positions where we can place the letter n so that, after appending an additional $\ell - 1$ copies of n to the end of the word, we obtain a word $w' \in \mathcal{P}_\ell(231, 221)$ over the alphabet $[n]$ that is generated by w. If we place the letter n in the j_r^{th} position between letters of w and then append an additional $\ell-1$ copies of n to the end, we obtain a word w' with label $\ell + r$. Indeed, the words generated by w' can be formed by inserting the letter $n+1$ into one of the positions $j_1, \ldots, j_r, \ell(n-1)+2, \ldots, \ell n+1$ between letters in w' and appending $\ell - 1$

copies of $n+1$ to the end. Therefore, w (which has label m) generates words with labels $\ell+1, \ell+2, \ldots, \ell+m$. For example, the word 121122 has label 5 and generates the words in (3.2), which have labels $4, 5, 6, 7, 8$, respectively. This is precisely the rule in the generating tree in (3.1). Of course, the only word in $\mathcal{P}_\ell(231, 221)$ over the alphabet [1] is $11\cdots1$ (of length ℓ). This word has label $\ell+1$, which yields the axiom of the generating tree in (3.1). □

4 Fertility Numbers

Recall that West defined the fertility of a permutation π to be $|s^{-1}(\pi)|$. In [15], the first author defined a *fertility number* to be a nonnegative integer f such that there exists a permutation with fertility f. Among other things, he showed that $3, 7, 11, 15, 19$, and 23 are not fertility numbers, and he has conjectured that infinitely many positive integers are not fertility numbers. By analogy, we define a hare-*fertility number* to be a nonnegative integer f such that there exists a word w with $|\,\mathsf{hare}^{-1}(w)| = f$. We define tortoise-*fertility numbers* similarly. It turns out that hare-fertility and tortoise-fertility numbers are much less mysterious than ordinary fertility numbers.

Theorem 4.1. *For every nonnegative integer f, there exists a word w such that $|\,\mathsf{hare}^{-1}(w)| = |\,\mathsf{tortoise}^{-1}(w)| = f$.*

Proof. It is clear that the word 21 has fertility 0 under both hare and tortoise and that the word 1 has fertility 1 under each map. In [15], it is shown that the permutation

$$\xi_m = m(m-1)\cdots321(m+1)(m+2)\cdots(2m)$$

has fertility $2m$ for every integer $m \geq 1$. Since hare and tortoise both restrict to the map s on the set of permutations, this tells us that

$$|\,\mathsf{hare}^{-1}(\xi_m)| = |\,\mathsf{tortoise}^{-1}(\xi_m)| = |s^{-1}(\xi_m)| = 2m.$$

For each integer $m \geq 1$, let

$$\xi'_m = m(m-1)\cdots3211(m+1)(m+2)\cdots(2m)$$

be the word obtained from ξ_m by inserting an additional 1 directly next to the 1 in ξ_m. We now claim that

$$|\,\mathsf{hare}^{-1}(\xi'_m)| = |\,\mathsf{tortoise}^{-1}(\xi'_m)| = 2m+1,$$

which will complete the proof. We will exhibit a proof of this claim only for hare; the proof for tortoise is completely analogous.

Let X_m be the set of words in $\mathsf{hare}^{-1}(\xi'_m)$ in which the two occurrences of the letter 1 appear consecutively. Given a word in X_m, one can obtain a permutation in $s^{-1}(\xi_m)$ by deleting one of the 1's; note that this map is in fact a bijection from X_m to $s^{-1}(\xi_m)$.

(For example, with $m = 3$, the word $3624115 \in X_3$ is mapped to the permutation $362415 \in s^{-1}(\xi_3)$.) So $|X_m| = |s^{-1}(\xi_m)| = 2m$, and it remains to show that there is exactly one word in $\mathsf{hare}^{-1}(\xi'_m)$ in which the two occurrences of 1 do not appear consecutively.

Suppose $v \in \mathsf{hare}^{-1}(\xi'_m)$ does not have two consecutive 1's. For each $i \in \{2, \ldots, m\}$, let $f(i)$ be the leftmost letter in v that is larger than i and appears to the right of i. (Such a $f(i)$ exists because otherwise in $\mathsf{hare}(v)$, the letter i would be followed by only larger letters.) Note that $f(i)$ is the letter that forces i to leave the stack when we apply hare to v. Let $f(1)$ be the letter that appears immediately to the right of the first 1 in v. For every $i \in \{2, \ldots, m\}$, the letter $f(i)$ must appear in v to the left of every letter that is smaller than i, since otherwise such a smaller letter would exit the stack before i when we apply hare. So we must have

$$v = m\, f(m)\, (m-1)\, f(m-1) \cdots 2\, f(2)\, 1\, f(1)\, 1.$$

In particular, the sequence $f(1), \ldots, f(m)$ is some rearrangement of $m+1, \ldots, 2m$. We claim that the $f(i)$'s are increasing. Indeed, if we had $f(i) > f(j)$ for some $1 \leq i < j \leq m$, then $f(j)$ would exit the stack before the second 1 could enter the stack; this would force $f(j)$ to appear to the left of the second 1 in ξ'_m, which is impossible. Thus, $f(1) < f(2) < \cdots < f(m)$, and we conclude that $f(i) = m + i$ for all $i \in \{1, \ldots, m\}$. Then

$$v = m(2m)(m-1)(2m-1) \cdots 2(m+2)1(m+1)1$$

is the last word in $\mathsf{hare}^{-1}(\xi'_m)$, and this completes the proof. □

5 Concluding Remarks and Further Directions

The introduction of the maps hare and $\mathsf{tortoise}$ leads to a variety of interesting problems, which we list in this section.

Theorem 2.3 tells us that for each positive integer k, there is a word η_{k+2} of length $2k+5$ with the property that $\langle \eta_{k+2} \rangle_{\mathsf{hare}} - \langle \eta_{k+2} \rangle_{\mathsf{tortoise}} = k$. More precisely, $\langle \eta_{k+2} \rangle_{\mathsf{hare}} = 2k+2$ and $\langle \eta_{k+2} \rangle_{\mathsf{tortoise}} = k+2$. We suspect that η_{k+2} is minimal among such words in the sense of the following conjectures.

Conjecture 5.1. *If w is a word of length m, then*

$$\langle w \rangle_{\mathsf{hare}} \leq \langle w \rangle_{\mathsf{tortoise}} + \frac{m-5}{2}.$$

Conjecture 5.2. *For every word w, we have*

$$\langle w \rangle_{\mathsf{hare}} \leq 2\langle w \rangle_{\mathsf{tortoise}} - 2.$$

After Theorem 2.3, we defined an exceptional word to be a word w such that $\langle w \rangle_{\mathsf{hare}} > \langle w \rangle_{\mathsf{tortoise}}$. We also let $\mathcal{E}_m$ denote the set of exceptional normalized words

of length m. We have calculated that $|\mathcal{E}_m| = 0$ for $m \leq 6$, $|\mathcal{E}_7| = 4$, $|\mathcal{E}_8| = 172$, and $|\mathcal{E}_9| = 5001$. Let $\mathcal{NW}_m$ denote the set of normalized words of length m. We are interested in the ratios $|\mathcal{E}_m|/|\mathcal{NW}_m|$. These values for $m = 7, 8, 9$ are (approximately) 0.000085, 0.000315, 0.000706. This leads us to the following question.

Question 5.3. *Determine the asymptotics of the sequence of ratios*

$$\frac{|\mathcal{E}_m|}{|\mathcal{NW}_m|}.$$

Each element of $\mathcal{E}_8$ contains one of the words in $\mathcal{E}_7$ as a pattern. This suggests that it could be possible to find conditions based on pattern avoidance that are necessary and/or sufficient for a word to be exceptional.

We saw in Section 4 that every nonnegative integer is a hare-fertility number and a tortoise-fertility number. In other words, if we define maps $\mathcal{F}_{\mathsf{hare}}, \mathcal{F}_{\mathsf{tortoise}} : \mathcal{W} \to \mathbb{N} \cup \{0\}$ by $\mathcal{F}_{\mathsf{hare}}(w) = |\,\mathsf{hare}^{-1}(w)|$ and $\mathcal{F}_{\mathsf{tortoise}}(w) = |\,\mathsf{tortoise}^{-1}(w)|$, then

$$\mathcal{F}_{\mathsf{hare}}(\mathcal{W}) = \mathcal{F}_{\mathsf{tortoise}}(\mathcal{W}) = \mathbb{N} \cup \{0\}.$$

Let $\mathcal{P}$ denote the set of all permutations. The first author has conjectured [15] that there are infinitely many positive integers that are not in the set $\mathcal{F}_{\mathsf{hare}}(\mathcal{P}) = \mathcal{F}_{\mathsf{tortoise}}(\mathcal{P})$ (where these sets are identical because hare, tortoise, and s all agree on permutations). It would be interesting to see if this phenomenon persists when we restrict attention to certain natural sets of words. For example, we have the following question. Recall that a 2-uniform word is a word in which each letter that appears actually appears exactly twice.

Question 5.4. *What can we say about $\mathcal{F}_{\mathsf{hare}}(\mathcal{P}_2)$ and $\mathcal{F}_{\mathsf{tortoise}}(\mathcal{P}_2)$, where $\mathcal{P}_2$ denotes the set of all 2-uniform words?*

Recall that a word w is t-hare-sortable (respectively, t-tortoise-sortable) if $\langle w \rangle_{\mathsf{hare}} \leq t$ (respectively, $\langle w \rangle_{\mathsf{tortoise}} \leq t$). We have not said anything about these families of words when $t \geq 2$. It would be interesting to investigate t-hare-sortable words and t-tortoise-sortable words in general. In the past, there has been a huge amount of interest in 2-stack-sortable permutations [4, 5, 6, 12, 22, 23, 25, 39, 40]. It is probably very difficult to obtain an explicit formula for the number of 2-hare-sortable words (or 2-tortoise-sortable words) in $\mathcal{W}_{\mathbf{c}}$ for arbitrary vectors $\mathbf{c}$, but deriving recurrences might be possible. Also, one might be able to prove more refined statements about specific choices of $\mathbf{c}$, such as $(1, \underbrace{2, \ldots, 2}_{n-1})$, $(\underbrace{1, \ldots, 1}_{n-1}, 2)$, and $(\underbrace{2, \ldots, 2}_{n})$.

Finally, let us mention that the authors of [14, 19, 26] have found several interesting properties of *uniquely sorted* permutations, which are permutations with fertility 1. Let us say a word w is *uniquely* hare-*sorted* if $|\,\mathsf{hare}^{-1}(w)| = 1$ and is *uniquely* tortoise-*sorted* if $|\,\mathsf{tortoise}^{-1}(w)| = 1$. We propose the investigation of uniquely hare-sorted words and uniquely tortoise-sorted words as a potential area for future research.

Acknowledgments

This research was conducted at the University of Minnesota Duluth REU and was supported by NSF/DMS grant 1650947 and NSA grant H98230-18-1-0010. The authors wish to thank Joe Gallian for running the REU program and providing encouragement. This paper also benefited from the suggestions of anonymous referees. The first author was also supported by a Fannie and John Hertz Foundation Fellowship and an NSF Graduate Research Fellowship.

References

[1] M. H. Albert, R. E. L. Aldred, M. D. Atkinson, C. Handley and D. Holton, Permutations of a multiset avoiding permutations of length 3, *European J. Combin.* **22** (2001), 1021–1031.

[2] M. D. Atkinson, S. A. Linton and L. A. Walker, Priority queues and multisets, *Electron. J. Combin.* **2** (1995), #R24.

[3] C. Banderier, M. Bousquet-Mélou, A. Denise, P. Flajolet, D. Gardy and D. Gouyou-Beauchamps, Generating functions for generating trees, *Discrete Math.* **246** (2002), 29–55.

[4] D. Bevan, R. Brignall, A. E. Price and J. Pantone, Staircases, dominoes, and the growth rate of 1324-avoiders, *Electron. Notes Discrete Math.* **61** (2017), 123–129.

[5] M. Bóna, *Combinatorics of permutations*, CRC Press, 2012.

[6] M. Bóna, A survey of stack-sorting disciplines, *Electron. J. Combin.* **9.2** (2002-2003), #A1.

[7] M. Bousquet-Mélou, Sorted and/or sortable permutations, *Discrete Math.* **225** (2000), 25–50.

[8] P. Brändén and T. Mansour, Finite automata and pattern avoidance in words, *J. Combin. Theory Ser. A* **110** (2005), 127–145.

[9] A. Burstein, "Enumeration of words with forbidden patterns", Ph.D. Thesis, University of Pennsylvania, 1998.

[10] A. Burstein and T. Mansour, Words restricted by patterns with at most 2 distinct letters, *Electron. J. Combin.* **9.2** (2002), #R3.2.

[11] F. R. K. Chung, R. L. Graham, V. E. Hoggatt Jr. and M. Kleiman, The number of Baxter permutations, *J. Combin. Theory Ser. A* **24** (1978), 382–394.

[12] R. Cori, B. Jacquard and G. Schaeffer, Description trees for some families of planar maps, *Proc. 9th FPSAC* (1997), 196-208.

[13] C. Defant, Counting 3-stack-sortable permutations, *J. Combin. Theory Ser. A* **172** (May 2020; available online).

[14] C. Defant, Catalan intervals and uniquely sorted permutations, *J. Combin. Theory Ser. A* **174** (August 2020; available online).

[15] C. Defant, Fertility numbers, *J. Comb.* (to appear).

[16] C. Defant, Motzkin paths and valid hook configurations, Preprint arXiv: 1904.10451.

[17] C. Defant, Postorder preimages, *Discrete Math. Theor. Comput. Sci.* **19** 1 (2017), #3.

[18] C. Defant, Preimages under the stack-sorting algorithm, *Graphs Combin.* **33** (2017), 103–122.

[19] C. Defant, M. Engen and J. A. Miller, Stack-sorting, set partitions, and Lassalle's sequence, *J. Combin. Theory Ser. A* (to appear).

[20] C. Defant and N. Kravitz, Stack-sorting for words, Preprint arXiv:1809.09158.

[21] E. Duchi, V. Guerrini, S. Rinaldi and G. Schaeffer, Fighting fish, *J. Phys. A.* **50.2** (2017), #024002.

[22] S. Dulucq, S. Gire and J. West, Permutations with forbidden subsequences and nonseparable planar maps, *Discrete Math.* **153.1** (1996), 85–103.

[23] W. Fang, Fighting fish and two-stack-sortable permutations, *Sém. Lothar. Combin.* **80B** (2018), Art. #7.

[24] H. W. Gould, Some generalizations of Vandermonde's convolution, *Amer. Math. Monthly* **63** (1956), 84–91.

[25] I. Goulden and J. West, Raney paths and a combinatorial relationship between rooted nonseparable planar maps and two-stack-sortable permutations, *J. Combin. Theory Ser. A* **75.2** (1996), 220–242.

[26] H. Mularczyk, Lattice paths and pattern-avoiding uniquely sorted permutations, Preprint arXiv:1908.04025.

[27] S. Heubach and T. Mansour, Avoiding patterns of length three in compositions and multiset permutations, *Adv. Appl. Math.* **36** (2006), 156–174.

[28] S. Heubach and T. Mansour, Combinatorics of compositions and words, CRC Press, 2009.

[29] S. Kitaev, "Patterns in Permutations and Words", Monographs in Theoretical Computer Science, Springer, Heidelberg, 2011.

[30] D. E. Knuth, *The Art of Computer Programming, Volume 1: Fundamental Algorithms*, Addison-Wesley, Reading, Massachusetts, 1973.

[31] T. Mansour, Restricted 132-avoiding k-ary words, Chebyshev polynomials, and continued fractions, *Adv. Appl. Math.* **36.2** (2006), 175–193.

[32] L. K. Pudwell, "Enumeration schemes for pattern-avoiding words and permutations", Ph.D. Thesis, Rutgers University, 2008.

[33] M. Sankar, Further bijections to pattern-avoiding valid hook configurations, *Adv. Appl. Math.* (to appear).

[34] C. D. Savage and H. S. Wilf, Pattern avoidance in compositions and multiset permutations, *Adv. Appl. Math.* **36** (2006), 194–201.

[35] R. Stanley, *Enumerative Combinatorics, Volume 1, Second Edition*, Cambridge University Press, Cambridge, UK, 2012.

[36] V. Vatter, Finitely labeled generating trees and restricted permutations, *J. Symbolic Comput.* **41** (2006), 559–572.

[37] J. West, Generating trees and forbidden subsequences, *Discrete Math.* **157** (1996), 363–374.

[38] J. West, Generating trees and the Catalan and Schröder numbers, *Discrete Math.* **146** (1995), 247–262.

[39] J. West, "Permutations with restricted subsequences and stack-sortable permutations", Ph.D. Thesis, MIT, 1990.

[40] D. Zeilberger, A proof of Julian West's conjecture that the number of two-stack-sortable permutations of length n is $2(3n)!/((n+1)!(2n+1)!)$, *Discrete Math.* **102** (1992), 85–93.

(Received 2 June 2019; revised 30 Dec 2019, 15 Mar 2020)