Enumeration Schemes for Words Avoiding Permutations

Lara Pudwell

November 27, 2007

Abstract

The enumeration of permutation classes has been accomplished with a variety of techniques. One wide-reaching method is that of enumeration schemes, introduced by Zeilberger and extended by Vatter. In this paper we further extend the method of enumeration schemes to words avoiding permutation patterns. The process of finding enumeration schemes is programmable and allows for the automatic enumeration of many classes of pattern-avoiding words.

1 Background

The enumeration of permutation classes has been accomplished by many beautiful techniques. One natural extension of permutation classes is pattern-avoiding words. Our concern in this paper is not attractive methods for counting individual classes, but rather developing a systematic technique for enumerating many classes of words. Four main techniques with wide success exist for the systematic enumeration of permutation classes. These are generating trees, insertion coding, substitution decomposition, and enumeration schemes. In this paper we adapt the method of enumeration schemes, first introduced for permutations by Zeilberger [5] and extended by Vatter [4] to the case of enumerating pattern-restricted words.

Definition 1. *Let $[k]^n$ denote the set of words of length n in the alphabet $\{1,\ldots,k\}$, and let $w \in [k]^n, w = w_1 \cdots w_n$. The* reduction *of w, denoted by $red(w)$, is the unique word of length n obtained by replacing the i^{th} smallest entries of w with i, for each i.*

For example, the reduction of $w = 2674423$ is $r = red(w) = 1453312$. Notice that r uses *every* letter from min(r)=1 to max(r), and furthermore $r_i \leq r_j$ iff $w_i \leq w_j$. We also say that w_1 and w_2 are *order-isomorphic* if $red(w_1) = red(w_2)$.

Definition 2. *Let $w \in [k]^n, w = w_1 \cdots w_n$ as above, and let $q \in [k]^m, q = q_1 \cdots q_m$. We say that w* contains *q if there exist $1 \leq i_1 < i_2 < \cdots < i_m \leq n$ so that $w_{i_1} \cdots w_{i_m}$ is order-isomorphic to q. Otherwise w* avoids *q.*

The study of pattern avoidance in permutations (where neither w nor q have repeated letters) has been well-studied, but less is known for the more general case of words. The groundbreaking work in this area was done by A. Burstein in his thesis [3], where he discusses words avoiding sets of permutations and uses generating function techniques to prove his results.

It is an easy exercise to fix a word w and to list all patterns of length m that w contains. However, it is a much more difficult question to fix a pattern (or set of patterns) q and enumerate all words in $[k]^n$ that avoid q for symbolic n. Our main object of study is:

Definition 3. *A frequency vector is a vector $\boldsymbol{a} = [a_1,\ldots,a_k]$ such that $k \geq 1$ and $a_i \geq 0$ for $1 \leq i \leq k$. Let $\|\boldsymbol{a}\| := \sum_{i=1}^{k} a_i$. Then, given a frequency vector $\boldsymbol{a}$ and a set of reduced words Q in $[k]^m$ for some $m > 0$, we define*

$$A_{\boldsymbol{a},Q} := \{w \in [k]^{\|\boldsymbol{a}\|} \mid w \text{ avoids } q \text{ for every } q \in Q, w \text{ has } a_i \text{ } i\text{'s for } 1 \leq i \leq k\}.$$

Notice that if $a_1 = \cdots = a_k = 1$, we reduce to the case of counting pattern-avoiding permutations. Also note that if $a_i = 0$ for some i, then we have $A_{\mathbf{a},Q} = A_{\mathbf{a}',Q}$, where $A_{\mathbf{a}'} = [a_1,\ldots,a_{i-1},a_{i+1},\ldots,a_k]$. Thus, we may assume that $a_i > 0$ for $1 \leq i \leq k$. When the set of patterns Q is clear from context, we may simply write $A_{\mathbf{a}}$.

Most of the techniques for studying pattern avoidance are dependent on what patterns are being avoided. However, one would hope for a more universal approach that enumerates pattern-avoiding strings independent of the pattern. In 1998, Zeilberger [5] introduced prefix enumeration schemes, giving a more universal framework for counting these permutation classes. Unfortunately, this method did not yet have as strong a success rate as hoped for. In 2005, Vatter [4] extended these schemes, completely

automating the enumeration of many more permutation classes. Vatter's work studies a symmetry of prefix schemes to ease notation. In 2006, Zeilberger [6] reformulated Vatter's schemes in his original notation of prefix schemes, allowing even quicker enumeration of many permutation classes. In this paper, I extend the notion of Zeilberger and Vatter's prefix schemes to enumerate words avoiding sets of permutations (i.e. pattern q has no repeated letters, but w may), and detail the success rate of this method.

2 Refinement

Assume we want to enumerate a set $A(n)$. If we cannot find a closed formula for $|A(n)|$, then ideally, we want to find a recurrence only involving n. Unfortunately, this is not always possible.

If we cannot find a direct recurrence, following Zeilberger, we introduce the notion of *refinement* as follows. Decompose $A(n)$ as $A(n) = \bigcup_{i \in I} B(n,i)$, so that the $B(n,i)$ are disjoint. Then, if we can find a recurrence for each $B(n,i)$ in terms of the other $B(n,i)$ and $A(n)$, we have a recursive formula for $A(n)$ as well. If not, then refine each $B(n,i)$ as the disjoint union $B(n,i) = \bigcup_{j \in J} C(n,i,j)$, and repeat.

In the case of words, we will use *reduced prefixes* as our refinement parameter.

Definition 4. *Let $w \in [k]^n$, $w = w_1 \cdots w_n$. The* i-prefix *of w is the word obtained by reducing $w_1 \cdots w_i$, $1 \leq i \leq n$.*

For example, if $w = 152243$, the 1-prefix of w is 1, the 2-prefix of w is 12, the 3-prefix of w is 132, the 4-prefix of w is 1322, the 5-prefix of w is 14223, and the 6-prefix of w is 152243. In particular, the n-prefix of w is $red(w)$.

Now, to allow us to talk about sets, we introduce the following notation:

Definition 5. $A_{\boldsymbol{a}}\left(p_1 \cdots p_l\right) := \{w \in [k]^{\|\boldsymbol{a}\|} \mid w \textit{ has } l\textit{-prefix } p_1 \cdots p_l\}$
$A_{\boldsymbol{a}}\binom{p_1 \cdots p_l}{i_1 \cdots i_l} := \{w \in [k]^{\|\boldsymbol{a}\|} \mid w = i_1 \cdots i_l w_{l+1} \cdots w_n \textit{ has } l\textit{-prefix } p_1 \cdots p_l\}$

For example, $A_{[1,2,1]}\left(21\right) = \{3122, 3212, 3221, 2123, 2132\}$.

$$A_{[2,1,2]}\binom{121}{131} = \{13123, 13132\}.$$

$$A_{[2,1,1,1,1]}\binom{132}{154} = \{154123, 154132, 154312, 154321, 15432132, 154231\}.$$

Thus, we have

$$\begin{aligned}
A_{\mathbf{a}}(\emptyset) &= A_{\mathbf{a}}(1) \\
&= A_{\mathbf{a}}(12) \cup A_{\mathbf{a}}(11) \cup A_{\mathbf{a}}(21) \\
&= (A_{\mathbf{a}}(231) \cup A_{\mathbf{a}}(121) \cup A_{\mathbf{a}}(132) \cup A_{\mathbf{a}}(122) \cup A_{\mathbf{a}}(123)) \\
&\quad \cup (A_{\mathbf{a}}(221) \cup A_{\mathbf{a}}(111) \cup A_{\mathbf{a}}(112)) \\
&\quad \cup (A_{\mathbf{a}}(321) \cup A_{\mathbf{a}}(211) \cup A_{\mathbf{a}}(312) \cup A_{\mathbf{a}}(212) \cup A_{\mathbf{a}}(213)) \\
&= \ldots
\end{aligned}$$

Finally, for ease of notation, we make the following definition:

Definition 6. *Given a prefix p of length l, the set of* refinements *of p is the set of all prefixes of length $l+1$ whose l-prefix is p.*

For example, the set of refinements of 1 is $\{11, 12, 21\}$. The set of refinements of 11 is $\{221, 111, 112\}$. The set of refinements of 12 is $\{231, 121, 132, 122, 123\}$. This simplifies our notation to the following:

$A_{\mathbf{a}}(p) = \bigcup_{r \in \{\text{refinements of } p\}} A_{\mathbf{a}}(r)$.

Now that we have developed a way to partition $A_{\mathbf{a}}$ into disjoint subsets, we investigate methods to find recurrences between these subsets.

3 Reversibly Deletable

Following Zeilberger, we have the following:

Definition 7. *Given a forbidden pattern q, $p = p_1 \cdots p_l$ an l-prefix, and $1 \leq t \leq l$, we say that p_t is* reversibly deletable *if every instance of q in a word with prefix p involving p_t implies the presence of an instance of q without p_t.*

For example, let $q = 123$ and $p = 21$. Then $w = ij\cdots$ with $i > j$. p_1 is reversibly deletable since the only way for $p_1 = i$ to be involved in a 123 pattern is if $w = ij\cdots a\cdots b\cdots$ with $i < a < b$. But since $i > j$, we have $j < a < b$ as well so jab forms a 123 pattern without using position p_1.

Now, if p_t is reversibly deletable, we have the following recurrence, where $\hat{p_t}$ (resp. $\hat{i_t}$) indicates that the letter p_t (resp. i_t) has been deleted:

$$\left|A_{\mathbf{a}}\begin{pmatrix} p_1\cdots p_l \\ i_1\cdots i_l\end{pmatrix}\right| = \left|A_{[a_1,\ldots,a_t-1,\ldots,a_k]}\begin{pmatrix} p_1\cdots \hat{p_t}\cdots p_l \\ i_1\cdots \hat{i_t}\cdots i_l\end{pmatrix}\right|$$

That is, deleting and replacing p_t provides a bijection between the two sets.

It should be noted that in Zeilberger's original schemes for permutations, if positions t and s are both reversibly deletable, then

$$\left|A_{\mathbf{a}}\begin{pmatrix} p_1\cdots p_l \\ i_1\cdots i_l\end{pmatrix}\right| = \left|A_{[a_1,\ldots,a_t-1,\ldots,a_s-1,\ldots,a_k]}\begin{pmatrix} p_1\cdots \hat{p_t}\cdots \hat{p_s}\cdots p_l \\ i_1\cdots \hat{i_t}\cdots \hat{i_s}\cdots i_l\end{pmatrix}\right|$$

However, in the case of pattern-avoiding words, this is no longer true. Consider for example $q = 123$, $p = 11$. Both p_1 and p_2 are reversibly deletable independently, but not together. In a later section, we will revisit the question of when reversibly deletable letters in the same prefix can be deleted at the same time.

4 Gap Vectors

Thus far, knowing only a prefix and a forbidden pattern are enough to determine reversibly deletable positions. However, there are instances when this is not the case. Consider for example $q = 123$ and $p = 12$. With our current definition neither position of the prefix is reversibly deletable. However, observe that if $w \in [k]^n, w = ij\cdots$ with $i < j < k$, then k eventually appears in the word w. Thus, $w = ij\cdots k\cdots$ and ijk is a 123 pattern. Since every word with prefix 12 where the letter playing the role of 2 is less than k has a 123 pattern, we know that in any word with prefix 12, the second letter is necessarily k. This is the largest letter in the alphabet, so it cannot be involved in a 123 pattern, so $p_2 = k$ is trivially reversibly deletable.

To help determine the reversibly deletable positions in these more sophisticated cases, we introduce the following:

Definition 8. *Given a pattern q, a prefix $p = p_1\cdots p_l$, and letters $i_1\cdots i_l$ comprising the prefix p, let $s_1 \leq \cdots \leq s_l$ such that $\{s_1,\ldots,s_l\} = \{i_1,\ldots,i_l\}$. We say that $\mathbf{g} =< g_1,\ldots,g_{l+1} >$ is a gap vector for $[q,p,[k]^n]$ if there are no words $w \in [k]^n$ avoiding q, with prefix p and with $s_1 - 1 \geq g_1$, $s_j - s_{j-1} \geq g_j$ $(2 \leq j \leq l)$, and $k - s_l \geq g_{l+1}$.*

For example, in the case of $q = 123$ and $p = 12$ above, $\mathbf{g} =< 0,1,1 >$ is a gap vector since the set of all words in $[k]^n$ where $w = i_1i_2\cdots$, and $i_1 - 1 \geq 0$, $i_2 - i_1 \geq 1$, and $k - i_2 \geq 1$, (i.e. all words that begin with an increasing pair where $i_2 < k$) is empty. (Otherwise, $w = i_1i_2\cdots k\cdots$ contains a 123 pattern, namely i_1i_2k.)

This definition may at first seem awkward in that the the entries at the beginning and end of a vector have a slightly different meaning from the interior entries. An interior 0 denotes a repeated letter in the prefix, while an interior 1 denotes two necessarily adjacent letters. In the convention of Zeilberger and Vatter, we would have used -1 and 0 respectively instead. This change gives one advantage of notation. If prefix $p = p_1\cdots p_l$ has gap vector $< g_1,\ldots,g_{l+1} >$, then prefix $p_1\cdots\hat{p_t}\cdots p_l$ has gap vector $< g_1,\ldots,g_t + g_{t+1},\ldots,g_{l+1} >$. With the notation of 0s and -1s, further adjustment would need to be made.

Notice that as in Vatter's schemes, if $\mathbf{g}$ and $\mathbf{h}$ are vectors of length l, $g_i < h_i$ $(1 \leq i \leq l)$, and $\mathbf{g}$ is a gap vector for some $[q,p,[k]^n]$, then so is $\mathbf{h}$. In other words, the set of gap vectors for a given pattern, alphabet, and prefix form an upper ideal in the poset of vectors in $\mathbb{N}^l$, so we can find a finite basis of gap vectors for $[q,p,[k]^n]$ by choosing the minimal elements of this ideal.

5 Enumeration Schemes for Words

We define an *abstract enumeration scheme* S to be a set of triples $[p,G,R]$ where p is a reduced prefix of length l, G is a (possibly empty) set of vectors of length $l+1$ and R is a subset of $\{1,\ldots,l\}$. If d is the maximum length of a prefix p in S, we say that S is a scheme of depth d.

Such an enumeration scheme is said to be a *concrete enumeration scheme* if for all triples in S, either R is non-empty or all refinements of p are also in S. Once we have such an enumeration scheme, it can be considered as an encoding of a system of recurrences. The simplest example of such a scheme is

$$S = \{\ [\emptyset, \{\}, \{\}],\ [1, \{\}, \{1\}]\ \}$$

For prefix $\emptyset$, all refinements, i.e. $\{1\}$, belong to S. For prefix 1, $R \neq \emptyset$.

In fact, this is the scheme for counting all words in $[k]^n$. First note that it is equivalent to count all words or to count all words beginning with a 1 pattern. For words beginning with a 1 pattern, the 1 is trivially reversibly deletable (there is no forbidden pattern to avoid). This gives the following recurrence:

$$|A_{\mathbf{a}}(\emptyset)| = |A_{\mathbf{a}}(1)| = \sum_{i=1}^{k} \left| A_{\mathbf{a}}\binom{1}{i} \right| = \sum_{i=1}^{k} \left| A_{[a_1,\ldots,a_i-1,\ldots,a_k]}(\emptyset) \right|, \quad \left|A_{[a_1]}\right| = 1.$$

As expected, this gives the unique solution $|A_{\mathbf{a}}(\emptyset)| = \binom{\|\mathbf{a}\|}{a_1,\ldots,a_k}$.

We have now developed all the necessary tools to completely automatically find concrete enumeration schemes to count classes of pattern-avoiding words in the following way:

1. Initialize $S := \{[\emptyset, \{\}, \{\}]\}$.
2. Let $P = \{$refinements of all prefixes in S with no reversibly deletable elements$\}$
3. For each prefix in P, find its set G_p of gap vectors.
4. For each pair $[p, G_p]$, find the set R_p of all reversibly deletable elements, and let and $S2 = \cup_{p\in P}\{[p, G_p, R_p]\}$.
5. If $R_p \neq \{\}$ for all triples in $S2$, then return $S \cup S2$. Otherwise let $S = S \cup S2$, and return to step 2.

It is clear that steps 1, 2, and 5 can be done completely automatically. In the following sections, we will prove that steps 3 and 4 can be done completely rigorously and automatically as well.

It should be noted that as in the case of permutations, the operations of complement and reversal are involutions on the set of words in $[k]^n$ with some useful properties. Namely, if p is a forbidden pattern, p^c is its complement (formed by replacing $i \to k+1-i$), and p^r is its reversal, in the notation of section 1, we have:

$$A_{[a_1,\ldots,a_k],\{p\}} = A_{[a_k,\ldots,a_1],\{p^c\}}$$

$$A_{[a_1,\ldots,a_k],\{p\}} = A_{[a_1,\ldots,a_k],\{p^r\}}$$

Let $Av(p)$ denote the set of all words avoiding p. If, we can find a scheme for $Av(p)$, then we have a system of recurrences for counting $Av(p)$, $Av(p^c)$, $Av(p^r)$, and $Av(p^{rc})$. If we are unable to automatically find a scheme for $Av(Q)$ directly, we may use these natural symmetries, or Wilf equivalences, on patterns to find an equivalent scheme.

6 Finding Gap Vectors Automatically and Rigorously

Recall that for a fixed set of patterns Q and a prefix p of length l, $\mathbf{g}$ is a gap vector if there are no words avoiding Q with prefix p and spacing given by $\mathbf{g}$. Thus, to study gap vectors we consider the following sets:

$A(Q, p, \mathbf{g}) := \{w \in [1 + \|\mathbf{g}\|]^n \ \ | \ n \geq l, \ \ w$ avoids Q, w has prefix p, $\{s_1, \ldots, s_l\} = \{w_1, \ldots, w_l\}$ with $s_1 \leq \cdots \leq s_l$, and $s_1 = g_1 + 1$, $s_j = s_{j-1} + g_j \ (j > 1)\}$

where $\|\mathbf{g}\| = \sum_i g_i$, the norm of $\mathbf{g}$. Thus, $A(Q, p, \mathbf{g})$ is the set of all Q-avoiding words with alphabet size $k = 1 + \|\mathbf{g}\|$ whose first l elements form a p pattern composed of the letters $g_1 + 1, g_2 + g_1 + 1, \ldots, g_l + \cdots + g_1 + 1$.

Not all pairs $(p, \mathbf{g})$ result in a non-empty set $A(Q, p, \mathbf{g})$. If the set $A(Q, p, \mathbf{g})$ is empty, then $\mathbf{g}$ is a gap vector.

Denote $G(p) := \{\mathbf{g} \in \mathbb{N}^{l+1} | A(Q, p, \mathbf{g}) \neq \emptyset$ for all $n \geq l\}$. The set $\mathbb{N}^{l+1} \setminus G(p)$ is the same as the set of all gap vectors that was introduced previously. We observed before that the set of gap vectors is an upper ideal in $\mathbb{N}^{l+1}$. Since $\mathbb{N}^{l+1}$ is partially

well-ordered, we may define the set of gap vectors in terms of a basis by specifying the minimal elements not in $G(p)$. We are guaranteed, by the poset structure of $\mathbb{N}^{l+1}$ under product order, that this basis is finite.

Now that we are concerned with determining a *finite* set of vectors, two questions remain: (1) How can we determine all gap vectors of a particular norm?, and (2) What is the maximum norm of a gap vector in the basis?

First, following Zeilberger, we may find all gap vectors of a specific norm $k = \|\mathbf{g}\|$ in the following way. Intuitively, a gap vector $\mathbf{g}$ specifies the relative spacing of the initial entries of a word beginning with prefix p. Consider prefix $p = p_1 \dots p_l$, sorted and reduced to be $s = s_1 \cdots s_l$ and potential gap vector $\mathbf{g} =< g_1, \dots, g_{l+1} >$. This means that there are g_1 entries smaller than s_1, $max\{0, g_{i+1} - 1\}$ entries between s_i and s_{i+1} (for $2 \leq i \leq l-1$), and finally g_{l+1} entries larger than s_l.

Let $\frac{1}{g_1+1}, \frac{2}{g_1+1}, \dots, \frac{g_1}{g_1+1}$ be the g_1 elements smaller than s_1.

Let $s_i + \frac{1}{g_{i+1}+1}, s_i + \frac{2}{g_{i+1}+1}, \dots, s_i + \frac{max\{0,g_{i+1}\}}{g_{i+1}+1}$ be the elements between s_i and s_{i+1}, $(2 \leq i \leq l-1)$.

Let $s_l + \frac{1}{g_{l+1}+1}, \dots, s_l + \frac{g_{l+1}}{g_{l+1}+1}$ be the g_{l+1} elements larger than s_l.

Extending the definition of reduction to fractional elements, we may consider all words of length $l + \|\mathbf{g}\|$ which begin with s and end with some permutation of the set of fractional letters above. There are $\leq (g_1 + \cdots + g_{l+1})!$ such possibilities. If *each and every* one of these words contains an element of Q, then we know that $\mathbf{g}$ is a gap vector for prefix p since the set of words beginning with p, avoiding Q, and obeying the gap conditions imposed by $\mathbf{g}$ is the empty set.

Now, we have a rigorous way to find all gap vectors of a specific norm, but the question remains: what is the maximum norm of elements in the (finite) basis of gap vectors guaranteed above?

First, it should help to remember how gap vectors are used. The notion of gap vector was introduced to help determine when a particular letter of a word prefix is reversibly deletable. We revisit this concept more rigorously.

For any $r \in [l]$, the set $A(Q, p, \mathbf{g})$ embeds naturally (remove the entry (p_r) and reduce) into $A(Q, d_r(p), d_r(\mathbf{g}))$ where $d_r(p)$ is obtained by deleting the rth entry of p and reducing. $d_r(\mathbf{g})$ is obtained by sorting p, and finding the index i corresponding to p_r, then letting $d_r(\mathbf{g}) =< g_1, \dots, g_{i-1}, g_i + g_{i+1}, g_{i+2}, \dots, g_{l+1} >$.

Sometimes this embedding of $A(Q, p, \mathbf{g})$ into $A(Q, d_r(p), d_r(\mathbf{g}))$ is a bijection. If this is true for all gap vectors $\mathbf{g}$ that obey $G(p)$, that is, this embedding is a bijection whenever the set $A(Q, p, \mathbf{g})$ is non-empty, then we say that p_r is *reversibly deletable* for p with respect to Q. Notice that this equivalent to the notion of reversibly deletable introduced previously.

Adapting notation from Vatter, we have the following proposition, which puts a bound on the number of gap conditions to check before declaring an element to be reversibly deletable.

Proposition 1. *The entry p_r of the prefix p is reversibly deletable if and only if*

$$|A(Q, p, \boldsymbol{g})| = |A(Q, d_r(p), d_r(\boldsymbol{g}))|$$

for all $\boldsymbol{g} \in G(p)$ with $\|\boldsymbol{g}\| \leq \|Q\|_\infty + l - 2$, where $\|\boldsymbol{g}\|$ denotes the sum of the entries of $\boldsymbol{g}$, $\|Q\|_\infty$ denotes the maximum length of a pattern in Q, and l is the length of p.

Proof. If p_r is reversibly deletable then the claim follows by definition. To prove the converse, suppose that p_r is not reversibly deletable. We trivially have that $|A(Q, d_r(p), d_r(\mathbf{g}))| \geq |A(Q, p, \mathbf{g})|$, and since p_r is not reversibly deletable, we now have $|A(Q, d_r(p), d_r(\mathbf{g}))| > |A(Q, p, \mathbf{g})|$ for some $\mathbf{g} \in G(p)$. Pick $\mathbf{g} \in G(p)$ and $p^* \in A(Q, d_r(p), d_r(\mathbf{g}))$ so that p^* cannot be obtained from a word in $A(Q, p, \mathbf{g})$ by removing p_r and reducing.

Now, form the Q-containing word p' by incrementing every entry of p^* that is at least p_r by 1 and inserting p_r into position r. p' is the word that would have mapped to p^*, except that p' contains a pattern $\rho \in Q$, and thus is in $A(\emptyset, p, \mathbf{g}) \setminus A(Q, p, \mathbf{g})$.

Now, pick a specific occurrence of $\rho \in Q$ that is contained in p'. Since $p^* = red(p' - p(r))$ avoids Q, this occurence of ρ must include the entry p_r. Let p'' be the reduction of the subsequence of p' formed by all entries that are either in the chosen occurence of ρ or in prefix p (or both). p'' is now a word of length $\leq \|Q\|_\infty + l - 1$. Since all gap vectors $\mathbf{g}$ have $\|g\| = k - 1$ where k is the size of the alphabet, we have that p'' lies in $A(\emptyset, p, \mathbf{h})$ for some $\mathbf{h}$ with $\|\mathbf{h}\| \leq \|Q\|_\infty + l - 2$. On the other hand, $red(p'' - p(r))$ avoids Q, so that $|A(Q, d_r(p), \mathbf{h})| > |A(Q, d_r(p), \mathbf{h})|$, as desired. □

Although not as sharp as the original bound of $\|\mathbf{g}\| \leq \|Q\|_\infty - 1$ given by Vatter for pattern-avoiding *permutations*, this still gives a bound on the depth of gap vectors that only increases linearly with the depth of the enumeration scheme. Now we have found a completely rigorous way to compute a basis for all gap vectors corresponding to a given prefix p. Finally, we turn our attention to the notion of reversibly deletable elements.

7 Finding Reversibly Deletable Elements Rigorously

Intuitively, to show that p_r is reversibly deletable, we must show that every conceivable forbidden pattern involving p_r implies the presence of another forbidden pattern not involving p_r. For example, in the case of $Q = \{1234\}$ and $p = 123$, for position p_3 we first compute that $\mathbf{g} =< 0, 1, 1, 1 >$ is a basis for $G(p)$ with respect to p, thus $p_3 = k$, the largest letter in the word. Since the third (and largest) letter cannot be in a 1234-pattern, p_3 is trivially reversibly deletable.

As a more instructive example, consider $p = 4213$ and $Q = \{43215\}$ and check if p_3 is reversibly deletable. To check, note that the only ways that $p_3 = 1$ can participate in a 43215 pattern is if we have (1) **421abc**, where $c > 4 > 1 > a > b$, (2) 4**21abc** where $c > 2 > 1 > a > b$, or (3) **421ab** where $b > 4 > 2 > 1 > a$.

Consider the first case. If this happens, then we have 2 letters smaller than "1", and one letter larger than "4", i.e. our word has the gap condition $< 2, 0, 0, 0, 1 >$. In this case, form the word $4213abc$ and delete the 1, to obtain $423abc$. In this case $43abc$ forms a 43215 pattern, so p_3 is ok.

Now, consider the case where $21abc$ is our 43215 pattern. Again, we have 2 letters smaller than "1", but our final letter is bigger than "2", so we may have any of the following gap vectors: $< 2, 0, 1, 0, 0 >$ (that is, $2 < c < 3$), $< 2, 0, 0, 1, 0 >$ (that is, $3 < c < 4$), or $< 2, 0, 0, 0, 1 >$ (that is, $c > 4$). Again, we must test all 3 cases, to check for implied instances of 43215. For gap $< 2, 0, 1, 0, 0 >$, we have $4213abc$ with $b < a < 1 < 2 < c < 3 < 4$, so $423abc$ reduces to 635214. There is no implied 43215 pattern, so p_3 is not reversibly deletable.

These two examples give the general idea for how to test if a position p_r is reversibly deletable:

1. List all possible bad patterns involving p_r.
2. For each possible bad pattern involving p_r, list all gap spacings the pattern may have with respect to the prefix p.
3. If each gap spacing of the bad pattern implies a different instance of a bad pattern, then p_r is reversibly deletable. Otherwise, it is not.

Furthermore, if there are non-trivial gap vectors, we may rule out many of the above cases in our computation because the gap vectors imply that the set of all such words with no bad pattern is empty.

Now that we have shown how to completely automatically determine the set of all gap vectors, and the set of reversibly deletable entries of a given prefix, we revisit the notion of independence of reversibly deletable elements. We showed earlier that if both p_r and p_s are reversibly deletable entries of prefix p, we cannot necessarily delete both p_r and p_s. We now show an important case where elements *may* be deleted simultaneously.

Proposition 2. *Let S be a concrete enumeration scheme, let p be a prefix in S and let p_r, p_s be reversibly deletable elements of p. If neither $d_r(p)$ nor $d_s(p)$ is a member of S, and p_s is reversibly deletable for some i-prefix of $d_r(p)$ in S, then p_r and p_s may be deleted simultanously.*

We will denote this embedding (deleting p_r and p_s simultanously) as $d_{r,s}$.

Proof. Suppose that p, r, s, S are as above. Since $d_r(p)$ is not in S, there must be some i-prefix p^* of $d_r(p)$ in S with a reversibly deletable element p^*_j. Since p^*_j is reversibly deletable for p^*, it is also reversibly deletable for $d_r(p)$ (which begins with p^*), therefore this position is also reversibly deletable. □

In short, this proposition shows that while we may not always be able to delete more than one prefix entry at a time, when it is necessary to obtain a prefix in scheme S, it can always be done.

We have now shown how to completely rigorously find all components of a concrete enumeration scheme for pattern-avoiding words.

8 The Maple Package `mVATTER`

The above algorithm has been programmed in the Maple package `mVATTER`, available from the author's website. The main functions are `SchemeF`, `MiklosA`, `MiklosTot`, and `SipurF`.

`SchemeF` inputs a set of patterns and a maximum depth scheme to search for, and outputs a concrete enumeration scheme for words avoiding q of the specified maximum depth. `SchemeF` also makes use of the natural symmetries of pattern avoiding

words: reversal and complement. If it cannot find a scheme for a set of patterns, it tries to find a scheme for a symmetry-equivalent pattern set and returns that scheme instead.

`MiklosA` inputs a scheme, a prefix, and a frequency vector and returns the number of words obeying the scheme and the vector, having that prefix. To count *all* words with a specific frequency vector avoiding a specific set of patterns, try `MiklosA(SchemeF(Patterns, SchemeDepth), [], [frequency vector])`.

`MiklosTot` inputs a scheme, and positive integers k and n, and outputs the total number of words in $[k]^n$ obeying the scheme.

`SipurF` inputs a list $[L]$, a maximum scheme depth, an integer r, and a list of length r. It outputs all information about schemes for words avoiding one word of each length in L. For example, `SipurF([3],2,4,[10,8,6,6])` outputs all information about words avoiding permutation patterns of length 3. It will output the first 10 terms in the sequence of the number of permutations (1 copy of each letter) avoiding a pattern, the first 8 terms in the sequence of the number of words with exactly 2 copies of each letter, and the first 6 terms in the sequences with exactly 3 or 4 copies of each letter.

`SipurF` has been run on $[L]$ for various lists of the form $[3^a, 4^b]$, and the output is available from the author's website.

9 A Collection of Failures

Although this notion of enumeration schemes for words is successful for many sets of patterns, there is more to be done.

There are many cases where enumeration schemes of Vatter and Zeilberger fail. Unfortunately, the new schemes for words avoiding permutation patterns will necessarily fail whenever Zeilberger and Vatter's schemes fail for the same patterns. Namely, the chain of prefixes with no reversibly deletable elements from the permutation class enumeration scheme will still have no reversibly deletable elements for words, since there are even more possibilities for a bad pattern to occur.

Further, this paradigm only succeeds for avoiding permutations (i.e., the patterns to be avoided have precisely one copy of each letter). The key observation is that gap vectors keep track of spacing, but they do not keep track of frequency. More precisely:

Proposition 3. *If $Q = \{\rho\}$ where ρ has a repeated letter, then there is no finite enumeration scheme for words avoiding Q.*

Proof. To show that there is no finite enumeration scheme, we must exhibit a chain of prefixes which have no reversibly deletable elements with respect to Q. Consider the structure of ρ. We have $\rho = q_1 l q_2 l q_3$, where l is the first repeated letter of ρ. Thus, $q_1 l q_2$ is a permutation.

First we consider a simple case. Suppose that $q_1 = \emptyset$. Then, consider the chain of prefixes of the form $p_i = 1 \dots i$. Consider an occurence of ρ beginning with j, $1 \leq j \leq i$. Since j is the first repeated letter in forbidden pattern ρ, there is no other letter in p_i which can take its place. Thus we have an infinite chain of prefixes with no reversibly deletable element.

Now, suppose that $|q_1| \geq 1$ and the final letter of q_1 is $> l$. For $1 \leq i \leq |q_1|$, we let $p_i = (q_1)_1 \cdots (q_1)_i$. Now, for $i > |q_1|$, let $d = i - |q_1|$, and make the following construction:

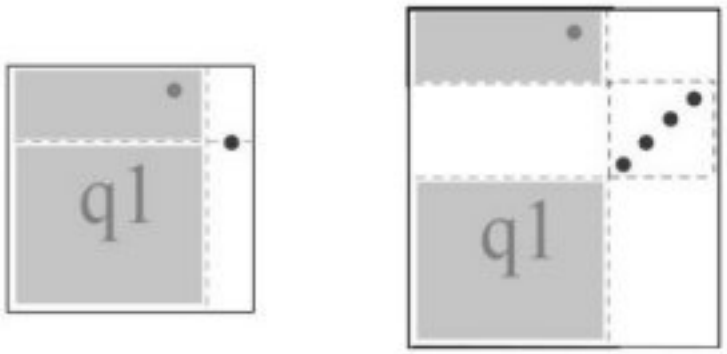
$$(q_1^*)_i = \begin{cases} (q_1)_i, & \text{if } (q_1)_i < l; \\ (q_1)_i + d & \text{if } (q_1)_i > l; \end{cases}$$

Then, $p_i = (p_i)_1 \cdots (p_i)_i$ where

$$(p_i)_j = \begin{cases} (q_1^*)_i, & \text{if } j \leq |q_1^*|; \\ l + (j - (|q_1^*| + 1)) & \text{if } j > |q_1^*|; \end{cases}$$

In essence, for large i, $p_i = q_1^* l^*$, where l has been replaced by increasing sequence l^*, and all entries of q_1 greater than l are incremented accordingly.

Viewing a prefix as a function from $\{1, \dots, i\}$ to $\{1, \dots, i\}$, the prefixes p_i of length $|q_1| + 1$ and $|q_1| + 4$ are displayed below as an example.



Now, consider the occurence of forbidden pattern ρ that uses element j of the monotone run at the end of p_i as l. Since this is the first repeated letter in the pattern, no matter how ρ occurs in the word, the role of l must be played by j. Thus there are no reversibly deletable entries of p_i.

For the remaining case: $|q_1| \geq 1$ and the final letter of q_1 is smaller than l, repeat the construction above, but with a decreasing run instead of an increasing run at the end of p_i for large i. Again, for each letter in p_i, we can pick an occurence of ρ that demonstrates that letter is not reversibly deletable.

□

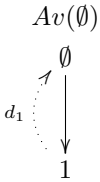
This seeming shortcoming raises the question whether there is yet another way to extend schemes. Recall that in this paper, we have modified Zeilberger's original schemes which use prefixes for refinement. On the other hand, Vatter took symmetries and refined by the patterns formed by the smallest entries of a permutation. In the study of restricted permutations, these two notions are equivalent, but for words, this is no longer true. Indeed, in Vatter's notation, if we refine by adding one letter at a time, the repeated letters in words cause $1 \to 11 \to 111 \to \cdots$ to often be an infinite chain in schemes for pattern-avoiding words. Ideally we would like to find schemes that do not depend on the alphabet size or on specifying frequency of letters. One way to circumvent this difficulty is to refine words by adding multiple letters at a time. These results will be given in a future paper.

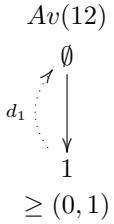
10 Examples and Successes

Despite the holes for future progress discussed in section 9, prefix enumeration schemes for words have a reasonable success rate, especially when avoiding *sets* of permutation patterns. This method is 100% successful when avoiding sets of patterns of length 3, and enjoys fairly high success when avoiding sets of 3 or more patterns simultanously. Some of the nicer results are displayed below.

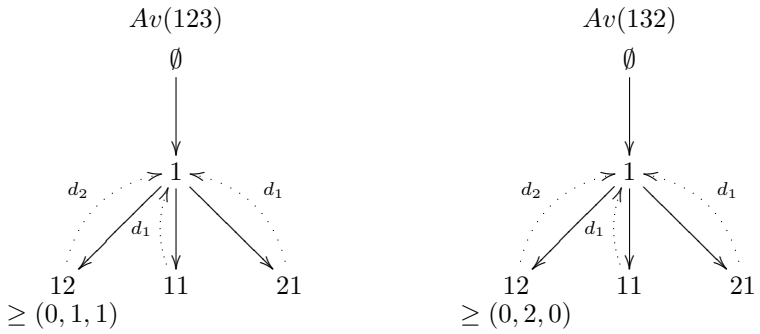
We draw an enumeration scheme as a directed graph, where the vertices are prefixes. A solid arrow goes from a prefix to any of its refinements. A dotted arrow, denoting reversibly deletable letters, goes from a prefix to one of its i-prefixes, and is labelled by the corresponding deletion map. If there are any gap vectors for a given prefix, the basis for those gap vectors is written below that prefix.

Many of the enumeration schemes for permutations carry over to enumerating words almost directly. Some simple examples include the scheme for counting all words, and the scheme for counting words avoiding the pattern 12.

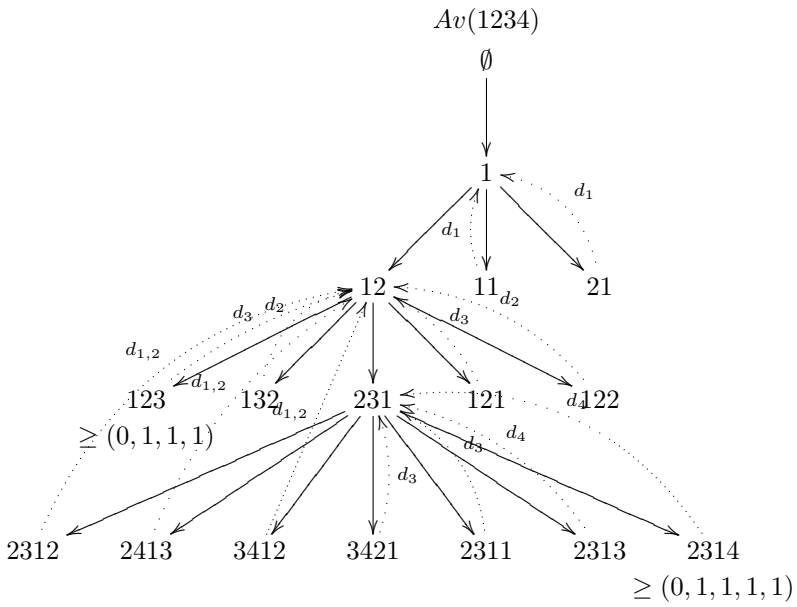




The first non-trivial examples are schemes for avoiding one pattern of length 3. These schemes are nearly identical to the permutation schemes, only with the 11 prefix now included. The symmetry of these schemes gives an alternate explanation that $Av(123)$ and $Av(132)$ are Wilf-equivalent for words as well as for permutations.



A more interesting example is:



We conclude this section with statistics comparing the success rate of Vatter and Zeilberger's schemes for permutations versus the success rate of schemes for words. As discussed above, the current success of word schemes is bounded above by the success of permutation schemes. We consider success rate to be the percentage of trivial Wilf classes of patterns which can be enumerated via schemes. Notice that there are fewer Wilf classes when enumerating permutations, since the operations of reverse, complement, and inverse all give trivial equivalences, while in the case of words, inverses no longer exist. Finally, we use the notation of the program `SipurF`, described above. For example, avoiding the list $[3, 4]$ means to avoid one pattern of length 3 and one pattern of length 4.

Pattern Lengths to Avoid	Permutation Scheme Success	Word Scheme Success
[2]	1/1 (100%)	1/1 (100%)
[2,3]	1/1 (100%)	1/1 (100%)
[2,4]	1/1 (100%)	1/1 (100%)
[3]	2/2 (100%)	2/2 (100%)
[3,3]	5/5 (100%)	6/6 (100%)
[3,3,3]	5/5 (100%)	6/6 (100%)
[3,3,3,3]	5/5 (100%)	6/6 (100%)
[3,3,3,3,3]	2/2 (100%)	2/2 (100%)
[4]	2/7 (28.6%)	2/8 (25%)
[3,4]	17/18 (94.4%)	9/24 (37.5%)
[3,3,4]	23/23 (100%)	27/31 (87.1%)
[3,3,3,4]	16/16 (100%)	20/20 (100%)
[3,3,3,3,4]	6/6 (100%)	6/6 (100%)
[3,3,3,3,3,4]	1/1 (100%)	1/1 (100%)
[4,4]	29/56 (51.8%)	?/84 (in process)
[3,4,4]	92/92 (100%)	38/146 (26%)
[3,3,4,4]	68/68 (100%)	89/103 (86.4%)
[3,3,3,4,4]	23/23 (100%)	29/29 (100%)
[3,3,3,3,4,4]	3/3 (100%)	3/3 (100%)

11 Future Work

This modification of Zeilberger and Vatter's enumeration schemes provides the beginning of a universal method for counting pattern-avoiding words. The success rate is quite good for avoiding sets of patterns, and has reasonable success for avoiding single patterns. The following questions remain open:

- Find other general techniques for enumerating classes of permutation-avoiding words not counted by prefix schemes.
- Find a method for counting words avoiding other words. This will be explored in a subsequent paper of the author.
- Find ways to simplify schemes to compute even more values in the sequence of the number of pattern-avoiding words for fixed k and q.
- Find ways to convert concrete enumeration schemes to closed forms or generating functions when possible.

References

[1] M. Albert, R. Aldred, M.D. Atkinson, C. Handley, D. Holton, *Permutations of a multiset avoiding permutations of length 3*, Europ. J. Combin. **22**, 1021-1031 (2001).

[2] P. Brändén, T. Mansour, *Finite automata and pattern avoidance in words*, Joural Combinatorial Theory Series A **110:1**, 127-145 (2005).

[3] A. Burstein, *Enumeration of Words with Forbidden Patterns*, Ph.D. Thesis, University of Pennsylvania, 1998.

[4] V. Vatter, *Enumeration Schemes for Restricted Permutations*, Combinatorics, Probability, and Computing, to appear.

[5] D. Zeilberger, *Enumeration Schemes, and More Importantly, Their Automatic Generation*, Annals of Combinatorics **2**, 185-195 (1998).

[6] D. Zeilberger, *On Vince Vatter's Brilliant Extension of Doron Zeilberger's Enumeration Schemes for Herb Wilf's Classes*, The Personal Journal of Ekhad and Zeilberger, 2006. `http://www.math.rutgers.edu/~zeilberg/pj.html`.