HONORS THESIS IN DETERMINANT FORMULAS FOR COUNTING LINEAR EXTENSIONS OF TREE POSETS

STEFAN GROSSER

1. Introduction

Partially ordered sets are a fundamental mathematical structure as they formalize the concept of ordering a set of objects. One important question to ask is, given a partially ordered set P, how many ways can you fill in the missing information to create a total ordering on the same set? Put another way, how many ways are there to order the elements of P respecting the order of the poset? Any one way to perform this is known as a *linear extension* of P.

Linear extensions act as a measure of complexity of a poset. The more linear extensions there are, the more information there is to fill in the partial order.

Numerous mathematical fields find linear extensions to be a useful tool; this includes representation theory on the symmetric group, enumerative combinatorics, and algorithms. Thus there is a strong motivation to be able to calculate or enumerate linear extensions of posets. However, given an arbitrary partially ordered set, it is known to be very difficult to compute the number of linear extensions. The problem is in fact $\#P$-complete (very hard) [BW91b]. However, for very specific families of posets, there are fast ways to compute this number.

Several families of posets have what is known as a *product formula* for the number of linear extensions. Two such examples are Young diagrams and rooted trees. Both of these families have what is known as the hook-length formula for the number of linear extensions. Suppose that i is an element of a poset P. Then $h(i)$ is a certain positive integer associated to i. We have the following formula for the number of linear extensions for Young diagrams and rooted trees.

$$e(P) = n! \prod_{i \in P} \frac{1}{h(i)} \tag{1}$$

where $n! = \prod_{i=1}^{n} i$ is the factorial function [Knu98, FRT+54].

An example is given in Section 2.1.1

Other families of posets like *skew Young diagrams* have no product formula for $e(P)$; however, this number still can be computed efficiently via a determinant formula. It is a point of interest on determining which families of posets have efficient formulas for counting linear extensions; more generally, we would like to know when there is a closed and computable formula for the number of linear extensions.

This thesis is based on joint work with Alejandro Morales, Jacob Matherne, and Alex Garver [GGMM].

1.1. **Methodology and Goals.** In this paper, we look for partially ordered sets which have efficiently calculated formulas for their number of linear extensions. By Atkinson in [Atk90], we know that linear extensions of tree posets (posets with acyclic Hasse diagrams) can be counted efficiently since he gave a quadratic time recursive algorithm. However, there are no known product or determinant formulas for tree posets other than zigzags, as the Jacobi–Trudi identities imply that zigzag posets have a determinant formula due to their skew shapes. This gives us the question, are there any other trees which have determinant formulas?

We will identify a larger class of tree posets that we call *calanques* obtained from zigzags by "hanging" a rooted tree to an element of the zigzag and "planting" a rooted to at most one element of the zigzag. This class contains zigzags and rooted tree posets.

In Section 3, we find an alternating formula for any finite poset based on folding edges. Specifically, a fold is taking a cover relation $x < y$ and swapping it to $y < x$. A basic relation is that

$$e(P) = e(P \backslash (x, y)) - e(P') \tag{2}$$

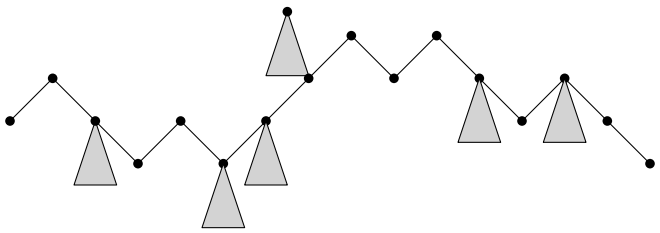


FIGURE 1. Structure of a Calanque poset

where P is a poset, $x < y$ is a covering relation, and P' is a poset retrieved from removing (x, y) and adding (y, x). Using this identity, we find a stronger alternating formula.

Our main result, proved in Section 4, is that a calanque tree poset P has a determinant formula which matches this alternating formula. This matrix, which we will designate as H_{σ_P}, will contain information on hook lengths from a completely folded P.

Theorem 1.1. *Let P be a calanque tree-poset with n elements then*

$$e(P) = n! \det(H_{\sigma_P})$$

where H_{σ_P} is a matrix whose entries are given by product formulas of the form (9).

A natural extensions once we have a formula for counting certain combinatorial objects is to find a q-analogue of the formula. In our setting, if $\mathcal{E} = (P, \omega)$ is the set of linear extensions on poset P with a labelling ω, then we are interested in counting

$$e_q(p) := \sum_{\sigma \in \mathcal{E}} q^{\text{inv}(\sigma)}$$

where $\text{inv}(\sigma)$ is the number of inversions of permutation σ. In Section 5, we give a q-analogue of our main result based on the q-analogue of the hook formula for rooted trees by Björner and Wachs [BW91a] for a certain labelling of our posets called *partitioned regular labelings*.

Theorem 1.2. *For a calanque tree poset P on n elements with a partitioned regular labeling ω*

$$\sum_{s \in \mathcal{E}(P,w)} q^{inv(s)} = [n]! \det(H^q_{\sigma_P})$$

What is interesting about these determinant formulas is that they only rely on counting linear extensions of rooted trees, which is well known through Knuth's hook length formula. This requires more elementary tools than, for example, the bijection to non-crossing paths to prove Jacobi–Trudi determinant identities.

The methods used to find determinants for calanques can be generalized to a larger class of posets that contains all posets which can be folded into a d-Complete poset while satisfying a similar condition to calanques. We explore these in Section 6.

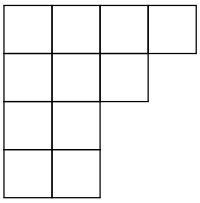
Finally, in Section 7, we give a q-analogue of Atkinson's algorithm. This is a recursive method to calculate linear extensions of tree posets. See the Appendix for an implementation.

2. Background

2.1. **Hook-Length Formula.** There is a well known hook length formula for calculating the number of linear extensions of certain poset families. We discuss these families and an algorithm to uniformly select any linear extension from a Young diagram poset.

2.1.1. *Young diagrams and Rooted Trees.* A partition is a weakly decreasing sequence of integers $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_i)$. Let $|\lambda| = \sum \lambda_l$ and let $l(\lambda) = l$. Denote the diagram of a partition λ as $[\lambda] = \{(i,j) \ : \ 1 \leq i \leq l(\lambda), 1 \leq j \leq \lambda_i\}$. We can represent this diagram visually as a *Young diagram*. Consider the partition $\lambda = (4,3,2,2)$:

(3)



A *standard Young Tableaux* is a Young diagram of shape λ where the boxes are filled in with numbers from 1 to $|\lambda|$ where the rows and columns are strictly increasing. A *semistandard* Young tableaux is where the rows are weakly increasing.

A poset can be defined from a Young diagram by placing an element in each box and having covering relations be determined by $a \geq b$ if b is to the right of or below a.

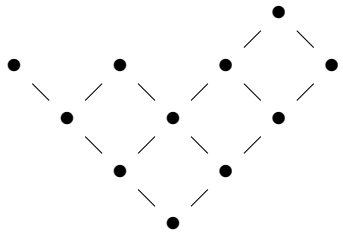
A rooted tree is a directed acyclic graph with a unique maximal element. We can consider this a poset by taking this graph as the Hasse diagram.

The hook length of cell (i, j) in a Young diagram is the number of cells to the right of or below the cell (inclusive). For example, the hook length of cell $(1, 1)$ in (3) is 7.

The hook length formula calculates the number of linear extensions of a poset of shape λ, but it was originally created to describe the number of standard Young tableaux of shape λ. A standard Young tableau of shape λ is a Young diagram of shape λ where the boxes are filled in from 1 to $|\lambda|$ such that the rows and columns are strictly increasing. The number of standard Young Tableaux of shape λ is written f^λ. Hence, we have

$$f^\lambda = n! \prod_{(i,j)\in\lambda} \frac{1}{h(i,j)}$$

Take the Young diagram above as shape λ. Then the poset of shape λ P is



By the hook length formula, we have that the number of linear extensions of P is

$$e(P) = \frac{11!}{7 \cdot 6 \cdot 3 \cdot 5 \cdot 4 \cdot 3 \cdot 2 \cdot 2} = 1320$$

For more on Young tableaux and the hook length formula, we refer to Sagan, [Sag91].

A *tree* poset is a partially ordered set that has an acyclic Hasse diagram. Although the general class tree posets does not have any know product or determinant formulas, there is still an efficient method of computing linear extensions through Atkinson's algorithm, which does this in quadratic time [Atk90].

When P is a rooted tree, we can redefine a hook $h(p)$ to be the number of elements which have p as an ancestor, including p.

$$e(P) = n! \prod_{p \in P} \frac{1}{h(p)} \tag{4}$$

2.1.2. *Hook Walks.* The hook walk algorithm allows a uniform sampling from the Young tableaux of shape λ. Proving this in fact proves the hook length formula. We present the algorithm.

Hook Walk Algorithm – Greene et al, [GNW82]

while an unlabeled cell exists **do**

Randomly select an unlabeled cell (i,j)

$C \leftarrow (i, j)$

while $h(C) > 1$ **do**

Randomly select an unlabeled cell C' in the hook of C

$C \leftarrow C'$

end while

Fill C with largest unused label and remove

end while

2.2. **Jacobi-Trudi Identity.** Similar to Young diagram of shape λ defined in Section 2.1.1, we have Young diagram of *skew shape* λ/μ. This means that we have a Young diagram of shape λ with a 'bite' taken out of it of shape μ from the upper left. More formally given partitions $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_n)$ and $\mu = (\mu_1, \mu_2, \ldots, \mu_n)$ where $\mu_i < \lambda_i$ for all i, we construct a Young diagram of shape λ/μ by removing the first μ_i boxes from row i of the Young diagram of shape λ.

Due to Feit in [Fei53], we have an identity for counting the number of standard Young tableux of skew shape, commonly know as the Jacobi-Trudi identity.

$$f^{\lambda/\mu} = n! \det \left(\frac{1}{(\lambda_i - \mu_j - i + j)!} \right)_{i,j=1}^{l(\lambda)}$$

This determinant formula also counts the number of linear extensions of posets defined on a Young diagram of skew shape.

Given that we have a determinant formula for the number of linear extensions of poset with shape λ/μ, one can ask the question, *What other class of posets have determinant formulas?*

2.3. **Inclusion Exclusion.** Inclusion Exclusion is a general principle of counting. The problem is counting the size of a union of multiple sets. Of course, if all the sets have unique elements, the answer is simply summing the individual set cardinalities. However, when they are not disjoint, we must take out a portion. For two sets, A and B, inclusion exclusion looks like this:

$$|A \cup B| = |A| + |B| - |A \cap B|$$

For three sets, A, B, and C, we have the formula

$$|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|$$

This generalizes into the following formula:

$$|A_1 \cup A_2 \cup \cdots \cup A_n| = \sum_{k=1}^{n} (-1)^{k+1} \Big(\sum_{1 \le i_1 < i_2 < \cdots < i_k \le n} |A_{i_1} \cap A_{i_2} \cap \cdots \cap A_{i_k}| \Big) \tag{5}$$

Numerous alternating formulas admit a transformation into a determinant. [Sta12] outlines a lemma to take an alternating formula (potentially provided by inclusion exclusion) and turn it into a determinant.

With this lemma, Stanley elaborated on finding an inclusion exclusion formula for descent sets of permutations, and then converting it into a determinant. We will give this lemma in the next section.

2.4. **d-Complete Posets.** Let P be a poset. P has a *diamond* if there are four elements w, x, y, z such that z covers x and y while x and y cover w. For $k \ge 3$, a *double tailed diamond poset* $d_k(1)$ can be made from a diamond by adding a $k-3$ chain to the top and the bottom of

a diamond. The *neck* elements are the $k-2$ elements about the two incomparable elements of the diamond. A d_k-interval is an interval $[w, z]$ which is isomorphic to $d_k(1)$. Additionally for $k \geq 4$, a d_k^--interval is a d_k-interval with the maximal element removed.

Definition 2.1 ([KY17]). *A poset it d-Complete if it satisfies these three properties for any* $k \geq 3$.

(1) *If* I *is a* d_k^-*-interval, then there exists an element* p *such that* p *covers the maximal elements of* I *and where* $I \cup p$ *is a* d_k-*interval.*
(2) *If* $I = [w, z]$ *is a* d_k*-interval, then* z *does not cover any elements of* P *outside* I*.*
(3) *There are no* d_k^-*-intervals which differ only in the minimal elements.*

With this in mind, we can now define hook lengths for elements in d-Complete posets.

Definition 2.2 ([KY17]). *Let* p *be an element of a d-Complete poset. Then we define its hook length* h_p *recursively.*

(1) *Suppose* p *is not in the neck of any* d_k*-interval, then* h_p *is the number of elements in* P *which are less than or equal to* p*.*
(2) *If* p *is in the neck of some* d_k*-interval, then we can take the unique element* $w \in P$ *such that* $[w, p]$ *is a* d_l*-interval, for some* $l \leq k$*. If we let* x *and* y *be the two incomparable elements in this* d_l*-interval, then* $h_p = h_x + h_y - h_w$*.*

Further detail into properties of d-Complete posets can be found in [KY17].

3. Counting Linear Extensions by Inclusion Exclusion

For certain partially ordered sets, an inclusion-exclusion formula for counting linear extensions exists. This was explored by Garver and Matherne in [GM]. We generalize this to all tree posets.

3.1. Descent Sets.

Let $\sigma = a_1 a_2 \cdots a_n$ be a permutation of $[n]$.

Definition 3.1. *A descent set of permutation* σ *is* $D(\sigma) = \{i : a_i > a_{i+1}\}$.

For a given set $S \subset [n]$, we want to know how many permutations on n elements have S as a descent set. We denote this as $\beta_n(S)$. As seen in [Sta12], we have the following inclusion exclusion formula for $\beta_n(S)$

$$\beta_n(S) = \sum_{1 \le i_1 \le i_2 \le \cdots \le i_j \le k} (-1)^{k-j} \binom{n}{s_{i_1}, s_{i_2} - s_{i_1}, \ldots, n - s_{i_j}} \tag{6}$$

3.1.1. *Turning Descents into a Determinant.*

Lemma 3.2 ([Sta12, Example 2.2.4])**.** *Let g be any function which is defined on $[0, k+1] \times [0, k+1]$ that satisfies $g(i,i) = 1$ and $g(i,j) = 0$ if $j < i$. Then the sum*

$$D_k = \sum_{1 \le i_1 \le i_2 \le \cdots \le i_j \le k} (-1)^{k-j} g(0, i_1) g(i_1, i_2) \cdots g(i_j, k+1)$$

is equivalent to the expansion of a $(k+1) \times (k+1)$ determinant of $E_{i,j}$ where $e_{i,j} = g(i, j+1)$ and $(i,j) \in [0,k] \times [0,k]$.

This lemma can be used to show that equation (6) is $n!$ multiplied by the expansion of a the determinant.

Corollary 3.3. *$\beta_n(S)$ has the determinant formula*

$$\beta_n(S) = n! \det \left[\frac{1}{(s_{j+1} - s_i)!} \right]_{i,j=0}^{n}$$

Since descent sets form *zigzag* posets, Corollary 3.3 can also be derived from the Jacobi–Trudi identities as any zigzag is of skew shape.

Stanley also gave the following q-analogue using the same lemma.

Lemma 3.4 (Example 2.2.5, [Sta12])**.** *Let P be a zigzag.*

$$\sum_{s \in \mathcal{E}(P, \omega)} q^{inv(s)} = [n]! \det \left[\frac{1}{[s_{j+1} - s_i]!} \right]_{0}^{k}$$

where k is the size of the descent set of P.

This material is covered with greater depth in [Sta12].

3.2. **An Alternating Formula for Trees.** In [GM], Garver and Matherne found an alternating formula for what they denote as decomposable posets. We examine their formula with arbitrary trees rather than the subclass they defined.

In general, let P be a tree poset and let y cover x in P. Let $P' = P\backslash(x,y) \cup (y,x)$. Then

$$e(P) = e(P\backslash(x,y)) - e(P') \tag{7}$$

By repeated application, we can create an alternating formula of products of rooted trees, each of which are easily computable by the hook length formula. Let F be the set of edges which, if folded, would result in P' a rooted tree. If $C \subset F$, then C^{op} represents the reversed relations of F. We have that

Proposition 3.5.

$$e(P) = \sum_{C \subseteq F} (-1)^{|C|} e(P\backslash F \cup (F\backslash C)^{op})$$

as an inclusion exclusion formula for the number of linear extensions of P.

Let $\mathbb{Z}[\mathcal{P}]$ be the set of formal finite linear combinations of posets with coefficients in $\mathbb{Z}$. See an example [GM].

More precisely, this is the group algebra of $\mathcal{P}$ over $\mathbb{Z}$, where $\mathcal{P}$ is the group of finite posets with the union operation. The elements of $\mathbb{Z}[\mathcal{P}]$ are of the form $z_1P_1 + z_2P_2 + \cdots + z_kP_k$ where $z_i \in \mathbb{Z}$ and $P_i \in \mathcal{P}$.

We extend linear extensions of posets to linear extensions of elements of $\mathbb{Z}[\mathcal{P}]$ by linearity. For example, $e(\sum_i z_iP_i) = \sum_i e(z_iP_i) = \sum_i z_ie(P_i)$.

The alternating sum seen in Proposition (3.5) can be seen as an element of $\mathbb{Z}[\mathcal{P}]$

$$A_{P,F} = \sum_{C \subseteq F} (-1)^{|C|} P\backslash F \cup (F\backslash C)^{op} \tag{8}$$

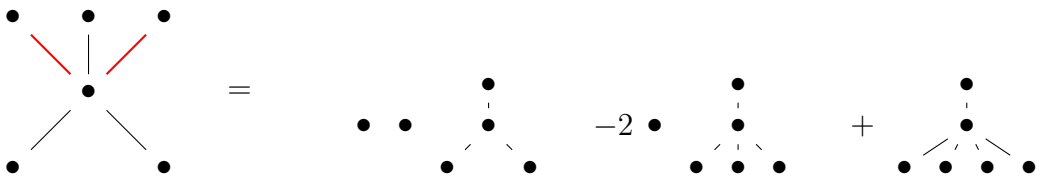
Now consider a formal matrix $(M_{i,j})_{i,j\in[n]}$ with elements of $\mathbb{Z}[\mathcal{P}]$ as entries. We define its determinant as an element of $\mathbb{Z}[\mathcal{P}]$

$$\det M = \sum_{\sigma\in S_n} M_{1,\sigma(1)}M_{2,\sigma(2)}\cdots M_{n,\sigma(n)}$$

We make the two following shorthand. In $\mathbb{Z}[\mathcal{P}]$, Let $1 = \{\}$ be the empty poset.

Example 3.6.

Let P be the following poset on the left, and let A, B and C be the posets from left to right.



where F will be selected as the red edges. This is shorthand for the following equation:

$$e(P) = e(A) - 2e(B) + e(C)$$

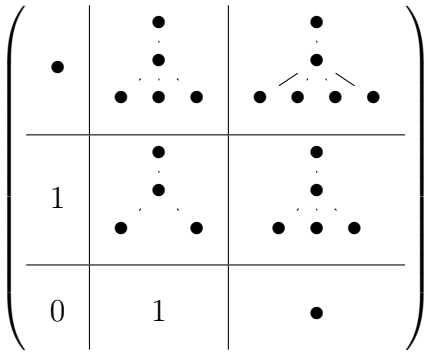
4. Determinants from Inclusion-Exclusion

We introduce an algorithm which takes in a tree poset P and outputs a matrix whose determinant expresses the number of linear extensions of P.

Let P be a tree poset. We say that P has k *folds* if there are k cover relations $x_i \lessdot y_i$ for $1 \le i \le k$, where if all of them are swapped to $y_i \lessdot x_i$ then P becomes a rooted tree M_P. We shall call this tree the *master tree of* P.

We denote as K be the set of connected components of M_P when removing all the folded edges and let the set of folded edges be E. If $n = |K|$, then we say that $\sigma \in S_n$ is an ordering of the set of K. Suppose P has k folds, then we now define a $k+1 \times k+1$ component matrix M_σ.

For $2 \leq i \leq k+1$, we have that $m_{i,i-1} = 1$ and $m_{i,j} = 0$ where $j < i-1$. For the rest of the entries of M_σ, we have that $m_{i,j}$ is the union of connected components $\bigcup_{i \leq a \leq j} K_{\sigma(a)}$ along with any $(x, y) \in E$ such that both x and y are in any of the selected connected components. Here is an example. Consider Example 3.6. By choosing the same folds F and by ordering the connected components of the master tree from left to right, we have

$M_\sigma =$ 

Definition 4.1. *Let $f : \mathcal{P} \to \mathbb{Q}$ be a function from a rooted tree poset P to a rational number defined by*

$$f(P) = \prod_{p \in P} \frac{1}{h_p}$$

where h_p is the hook value of the element.

If M_σ contains only connected entries, then we say

$$H_\sigma = f(M_\sigma) \tag{9}$$

is the hook matrix, where $h_{i,j} = m_{i,j}$ for $j \leq i - 1$ and where $h_{i,j} = f(m_{i,j})$ otherwise.

As a prerequisite for the rest of the section, we show how to count linear extensions of unions of rooted trees.

Lemma 4.2. *Let $P = \prod_i P_i$ be a product of rooted trees P_i, where they are taken as elements of the group $\mathcal{P}$. Then*

$$e(P) = n! \prod_i f(P_i)$$

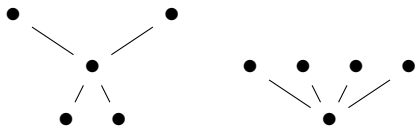
where n is the total number of elements in the rooted trees.

Proof. This follows from Knuth's heuristic argument for proving the hook length formula on rooted trees, seen in [Knu98, Theorem 5.1.4H]. Consider just one P_i. We have $n!$ total possible linear extensions. For each element of P_i, we need it to be above each of its descendents in a linear extension. The probability of this is the reciprocal of the hook length. Thus for P_i, we must multiply $n!$ by $f(P_i)$. Repeat for the other rooted trees. □

4.1. **A Determinant Formula.** In order for a poset to have an existing H_σ, we need every entry in M_σ to be a connected rooted tree. Thus we want to find the most general class of tree posets which will adhere to the domain of f.

Definition 4.3. *A calanque tree poset P is a poset obtained from a zigzag by attaching rooted trees by the root to elements in the zigzag and by attaching a leaf of a single rooted tree above the zigzag.*

Example 4.4. *Below is an example of a calanque and a tree poset that is not a calanque.*



Definition 4.5. *Let P be a tree poset and let M_P be the master tree and K be the set of connected components of M_P. We define the component graph $G_K = (V_K, E)$ where $V_K = K$ and E is the set of folds from the master tree.*

Using the component graph, we now can describe certain posets which have connected matrix entries.

Lemma 4.6. *Let P be a tree poset with a master tree M_P and component graph G_K that is a path. Then P has connected matrix entries.*

Proof. For M_σ, we order the connected components according to the path G_K. $\sigma(1)$ will be the first node in the path with degree 1, and the proceeding entry $\sigma(2)$ will be the unique

neighbor. We proceed in this fashion until the end of the path. Using this ordering σ, by definition, M_σ will have connected entries. □

In the proof of the lemma, we saw a construction for a specific σ which always gives connected entries to the component matrix M_σ. We shall call this ordering for poset P the *canonical ordering*, σ_P. The canonical ordering is always unique unless the two connected components of degree one in the component graph have $\min_{p\in K_i} \phi(p) = \min_{q\in K_j} \phi(q)$. To break this second tie, this can be resolved by placing an "x-coordinate" grading ψ and then selecting K_i as the start of the ordering if $\min_{p\in K_i} \psi(p) < \min_{q\in K_j} \psi(q)$.

Lemma 4.7. *The class of calanque tree posets has connected matrix entries*

Proof. Let $\mathcal{C}$ be a calanque. We give the following folding F for the poset. To the left of the planted rooted tree, fold all downard-left edges. To the right of the planted rooted tree, fold all upward-right edges.

With this folding, we have a path component graph, and hence connected matrix entries. □

We now can state and prove the main result.

Theorem 4.8. *Let P be a calanque tree-poset with n elements then*

$$e(P) = n! \det(H_{\sigma_p})$$

Proof. First, recall that by the previous lemma, a calanque P will have connected entries in M_{σ_P}, hence staying in the domain of f. By Lemma 4.2, it is sufficient to prove that $\det M_{\sigma_P}$ is equivalent to equation (8).

We proceed by induction on the number of folds. Suppose P has 1 fold. Then P has to be a rooted tree R with one element p additionally covering any non-root element. Note that p may be the root of another rooted tree, but without loss of generality, we assume it is a single element. Let M_P be the master tree, and let P' be the poset which removes the fold.

We have that

$$P = P' - M_P = pR - M_P = \det \begin{bmatrix} p & M_P \\ 1 & R \end{bmatrix}$$

Now suppose that every poset P with k folds has equality between $\det M_{\sigma_P}$ and its alternating formula from equation (8). A poset Q with $k+1$ folds will be the result of adding a fold to a k-fold poset P. Without loss of generality, suppose that this fold and the new connected component $\widetilde{K}$ come last in σ_Q.

Let M^P be the component matrix of P and let Q' be Q but with the edge connected to $\widetilde{K}$ folded. We see that the alternating formula from equation (7) gives us

$$Q = \widetilde{K}P - Q' \tag{10}$$

Define $\cup^+$ to be the union operation of two posets, also adding in the fold which connects them as connected components in the master tree. Then M_{σ_Q} of Q will be:

$$M_{\sigma_Q} = \begin{pmatrix} M^P_{1,1} & M^P_{1,2} & \dots & M^P_{1,k+1} & M^P_{1,k+1} \cup^+ \widetilde{K} \\ M^P_{2,1} & M^P_{2,2} & \dots & M^P_{2,k+1} & M^P_{1,k+1} \cup^+ \widetilde{K} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & M^P_{k,k+1} & M^P_{k,k+1} \cup^+ \widetilde{K} \\ 0 & 0 & \dots & 1 & \widetilde{K} \end{pmatrix}$$

Evaluating the determinant by expanding from the bottom row, we get

$$\det M_{\sigma_Q} = \widetilde{K} \det M^P - \det A \tag{11}$$

where A is M^P but adjoining $\widetilde{K}$ to the final column. By the induction hypothesis, we know that $\det M^P$ and $\det A$ both correspond to the alternating formulas of their respective posets P and Q'. Expanded, we see that equation (11) is equivalent to equation (10). □

Corollary 4.9. *Theorem 4.8 implies Corollary 3.3*

Proof. The determinant for descent sets is the same as the determinant produced by the algorithm when the rightmost element is selected as the root of the master tree. When the

rightmost vertex is selected as the root of the master tree, the corresponding descent set is equivalent to the folded edges selected. By Section 3.1, we see that the (i,j)-th entry of the determinant should be $\frac{1}{(s_{j+1}-s_i)!}$. Since the master tree is a chain, the entries of H_{σ_P} will be the exact same. □

A second proof can be provided for Theorem 4.8 using Lemma 3.2.

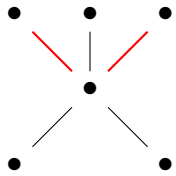
Proof. Let $\cup^+$ be defined as in the earlier proof of Theorem 4.8. If we define

$$g(i,j) = \begin{cases} 0 & j < i \\ 1 & j = i \\ f(\bigcup^+_{i \leq p < j} K_{\sigma_P(p)}) & j > i \end{cases}$$

where f is from Definition 4.1, then we follow the conditions of Lemma 3.2 and thus have the equivalence between the determinant and alternating formula. □

Example 4.10.

We shall use the poset P in Example 3.6. As a reminder P is the poset



By identifying the red edges as folds, we have the component matrix

$$M_\sigma = \left(\begin{array}{c|c|c} \bullet & \begin{array}{c}\bullet\\ \bullet\ \bullet\ \bullet\end{array} & \begin{array}{c}\bullet\\ \bullet\ \bullet\ \bullet\ \bullet\end{array} \\ \hline 1 & \begin{array}{c}\bullet\\ \bullet\ \bullet\end{array} & \begin{array}{c}\bullet\\ \bullet\ \bullet\ \bullet\end{array} \\ \hline 0 & 1 & \bullet \end{array} \right)$$

We then have

$$e(P) \;=\; 6!\det H_\sigma \;=\; 6!\det \begin{bmatrix} 1 & \frac{1}{5\cdot 4} & \frac{1}{6\cdot 5} \\ 1 & \frac{1}{4\cdot 3} & \frac{1}{5\cdot 4} \\ 0 & 1 & 1 \end{bmatrix} \;=\; 12$$

This can be verified by the observation that there are $3! \times 2 = 12$ linear extensions by ordering the bottom two elements and the top three elements.

It is a famous result that the linear extension of fence posets (zigzags with alternating inclines and declines) are the Euler zigzag numbers.

5. A q-Analogue for Calanques

We are able to generalize previous results on q-analogues for rooted trees to a q-anologue for calanques. Let $[n]$ define the polynomial $\sum_{i=0}^{n} q^i$. Then we can define

$$[n]! = [n] \cdot [n-1] \cdots [2] \cdot [1]$$

As well, define a labeled poset (P, ω) to be a bijection $\omega : P \to \langle n \rangle$, where $\langle n \rangle$ is the set of integers from 1 to n. We call this a *natural* labeling if for any $x, y \in P$ where $x < y$, we have that $\omega(x) < \omega(y)$. As well, a labeling ω is *regular* if we have the following: for all $x < z$ and $y \in P$, if $\omega(x) < \omega(y) < \omega(z)$ or $\omega(x) > \omega(y) > \omega(z)$ then $x < y$ or $y < z$. We will shortly use regular labelings for the q-analogue of Theorem 4.8

Let $\mathcal{E}(P, w)$ be the set of linear extension of P under the labeling ω. For any $\sigma \in \mathcal{E}(P, w)$, we can denote $\mathrm{inv}(\sigma)$ as the number of inversions of σ. As well, we have $\mathrm{inv}(P, \omega) = |\{(x, y) \,|\, \omega(y) < \omega(x) \text{ and } x < y\}|$ as the number of inversions of a labelling on P. A deeper analysis on natural labelings can be found in [Sta12], whereas more on regular labelings can be found in [BW89].

We have the following q-analogue of the Knuth hook-length formula of Björner and Wachs [BW89].

Lemma 5.1 ([BW89, Thm. 1.1]). *Let ω be a regular labeling on poset P. Then*

$$\sum_{\sigma\in\mathcal{E}(P,w)} q^{\mathrm{inv}(\sigma)} = q^{\mathrm{inv}(P,\omega)}\frac{[n]!}{\prod_{p\in P}[h(p)]}$$

Suppose we have a *natural* labeling where $\mathrm{inv}(P,\omega) = 0$, then this removes the power of q. This q-analogue was presented in [BW89] with the introduction of regular labelings, although it was originally found in [Sta12] with natural labelings.

There is a q-analogue for the component matrix M_σ of a poset P, as well as H_σ. We will denote these as M_σ^ω and H_σ^q respectively and are defined as follows. Let P be given a regular labeling ω_P. For each entry $p_{i,j}$ of M_σ, give it the induced labeling $\omega_{i,j}$ which keeps the same labeling from ω_P. Specifically, for $x \in p_{i,j}$, we have $\omega_{i,j}(x) = \omega_P(x)$. We then define

$$M_\sigma^\omega := (p_{i,j}, \omega_{i,j}) \tag{12}$$

where $p_{i,j}$ is the $(i,\ j)$-th entry of M_σ.

This new component matrix just adds a labeling to each entry of M_σ. Using this, we can now define H_σ^q. We have a direct q-analogue of f from Definition 4.1.

Definition 5.2. *Let $f_q : \mathcal{P}\times\Omega \to \mathcal{R}[q]$ be a function from a rooted tree poset P with labeling ω to the set of rational functions on indeterminate q defined by*

$$f_q(P) := q^{inv(P,\omega)}\prod_{p\in P}\frac{1}{[h_p]}$$

Then H_σ^q will have the following definition

$$H_\sigma^q := f_q(M_\sigma^\omega) \tag{13}$$

We now give a slight generalization of Lemma 5.1

Lemma 5.3. *Let P be a poset which is a union of several disconnected rooted trees. Let those connected components be given an ordering . If $\sigma(i) < \sigma(j)$ implies every element of K_i is given a label less than every element in K_j, then*

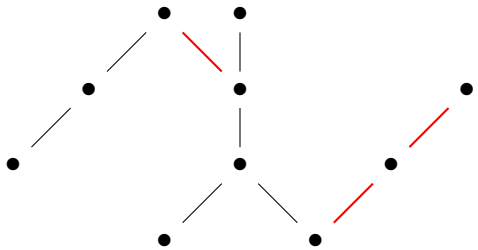
$$\sum_{\sigma \in \mathcal{E}(P,w)} q^{inv(\sigma)} = q^{inv(P,\omega)} \frac{[n]!}{\prod_{p \in P}[h_K(p)]} \tag{14}$$

where h_K denotes the hook length of element p in its corresponding rooted tree.

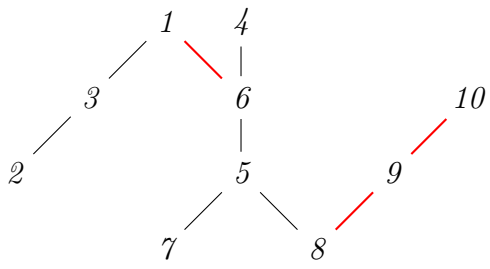
Proof. Without loss of generality, we assume P is the disjoint union of two rooted trees. We see that the hook lengths and the q-inversions of the labeling follow by Lemma 4.2 and by Lemma 5.1. We are left needing to justify $[n]!$. However, this follows by the property that any element in a later rooted tree must have a label greater than every element in a previous rooted tree. For example, if $(1, 3, 2)$ is a linear extension of K_1, and an element of label 4 is being added, the total contribution to inversions will be $[4]$ since the number of inversions entirely depends on which spot it ends up. More concisely, if l is a linear extension of K_1, an element of K_2 will give a contribution of $[|K_1| + 1]$. By merging every element of K_2 into a linear extension of K_1, we arrive at $[n]!$. □

We introduce a specific type of regular labeling that we call a *partitioned regular labeling*. Let P be a calanque and let F be folds selected to create a master tree. Consider the connected component graph G_K. We construct a labeling ω as follows. Start with a vertex of G_K which has degree 1, which we define as v_1. Use the labels 1 through $|v_1|$ (the number of elements in the connected component) to give a regular labeling for these elements of P. Following the path defined by G_K, give each connected component v_i a regular labeling using labels $1 + \sum_{j=1}^{i-1} |v_j|$ to $|v_i| + \sum_{j=1}^{i-1} |v_j|$. This labeling is a partitioned regular labeling.

Example 5.4. *Let P be the following poset*



The following labeling of P is a partitioned regular labeling.



If the labels 9 and 10 were swapped, this would no longer be a partitioned regular labeling.

Using Lemma 5.3 and a partitioned regular labeling ω, Theorem 4.8 has the following q-analogue

Theorem 5.5. *For any calanque P on n elements with a partitioned regular labeling ω*

$$\sum_{s\in\mathcal{E}(P,w)} q^{inv(s)} = [n]!\det(H^q_{\sigma_P}) \tag{15}$$

Proof. Let (P,ω) be a poset with two connected components K_1, K_2, as well as a labeling ω. Note that $q^{\mathrm{inv}(P,\omega)} = q^{\mathrm{inv}(K_1,\omega)}q^{\mathrm{inv}(K_2,\omega)}$.

This theorem follows by taking Lemma 3.2 and Lemma 5.3, as well as the note above.

Let ω be a labeling of poset P and let $\cup^+$ denote the operation seen in the proof of Theorem 4.8. Define the function

$$g(i,j) = \begin{cases} 0 & j < i-1 \\ 1 & j = i-1 \\ f_q(\bigcup^+_{i\le p\le j} K_{\sigma_P(p)}, \omega_{i,j}) & j \ge i \end{cases}$$

This follows the conditions of Lemma 3.2, giving us

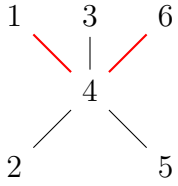
$$\sum_{s\in\mathcal{E}(P,\omega)} q^{\mathrm{inv}(s)} = \sum_{C\subseteq F}(-1)^{|C|}\sum_{t\in\mathcal{E}(P_C,\omega_C)} q^{\mathrm{inv}(t)} \tag{16}$$

$$= \sum_{C\subseteq F}(-1)^{|C|}q^{\mathrm{inv}(P_C,\omega_C)}\frac{[n]!}{\prod_{p\in P_C}[h_p]} = [n]!\det(H^q_{\sigma_P}) \tag{17}$$

where $P_C = P\backslash F \cup (F\backslash C)^{op}$ as seen in Proposition (3.5) and ω_C is the labeling ω induced on P_C.

The final equality in Equation (16) holds true since ω regular labelings are closed under folding; therefore, Lemma 5.3 can be applied. □

Example 5.6. *Let P be the same poset from Examples 3.6 and 4.10 with the following labeling ω*



where the red edges are to be folded. ω is a partitioned regular labeling for this folding. We then get the following matrices

$$M^q_{\sigma_P} = \left(\begin{array}{c|c|c} 1 & \begin{array}{ccc} & 3 & \\ & 4 & \\ 2 & 1 & 5 \end{array} & \begin{array}{cccc} & 3 & & \\ & 4 & & \\ 2 & 1 & 6 & 5 \end{array} \\ \hline 1 & \begin{array}{ccc} & 3 & \\ & 4 & \\ 2 & & 5 \end{array} & \begin{array}{ccc} & 3 & \\ & 4 & \\ 2 & 6 & 5 \end{array} \\ \hline 0 & 1 & 6 \end{array}\right)$$

$$H^q_{\sigma_P} = \begin{bmatrix} 1 & \frac{q^3}{[5][4]} & \frac{q^5}{[6][5]} \\ 1 & \frac{q^3}{[4][3]} & \frac{q^5}{[5][4]} \\ 0 & 1 & 1 \end{bmatrix}$$

Thus

$$q^6 + 3q^7 + 4q^8 + 3q^9 + q^{10} = \sum_{s \in \mathcal{E}(P,\omega)} q^{inv(s)} = [6]! \det(H^q_{\sigma_p})$$

Expanding the determinant, we see this is true.

Similarly to Corollary 4.9, we have that Lemma 3.4 is implied by Theorem 5.5.

Corollary 5.7. *Theorem 5.5 implies Lemma 3.4*

Proof. Let P be a zigzag. This follows by selecting the rightmost node of P as the root of the master tree, as well as letting ω be the labeling where $\omega(x) < \omega(y)$ if x is before y on the horizontal grading. From this, all induced labelings will have no inversions. The rest follows from the same reasoning as Corollary 4.9. □

6. Generalizing to Other Hook-Length Formulas

The proofs in previous sections have only relied on the fact that the master tree has a hook-length formula. This is of course true since any rooted tree has a hook-length formula. If all that is required is a hook-length formula to prove these results, then every result thus far can be generalized.

Instead of a master *tree*, will now just consider a *master*. In general, a master can be a d-Complete poset. We use the same notation to denote the master of poset P as M_P.

We have the same definition of the component matrix M_σ and component graph G_K, where K is the set of connected components in M_P after removing all the folds.

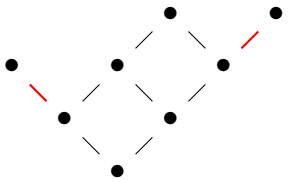
Theorem 6.1. *For every d-complete calanque P with n elements*

$$e(P) = n! \det(H_{\sigma_P})$$

where the hooks lengths in H_{σ_P} are now for d-Complete posets.

Example 6.2.

Let P be the poset below, where the red edges are selected to be folded. This is a Young diagram, which leaves our normal class of calanques as we are now hanging d-complete posets from the zigzag.



The resulting number of linear extensions will be

$$e(P) = 8! \det \begin{bmatrix} 1 & \frac{1}{5\cdot4\cdot3\cdot3\cdot2} & \frac{1}{6\cdot4\cdot4\cdot3\cdot2} \\ 1 & \frac{1}{4\cdot3\cdot3\cdot2\cdot2} & \frac{1}{5\cdot4\cdot3\cdot2\cdot2} \\ 0 & 1 & 1 \end{bmatrix} = 8! \cdot \frac{1}{576} = 70$$

For verification, we see that this matches the hook length formula for counting Young tableaux on the reflected poset P' of shape $\lambda = (4, 3, 1)$.

$$f^\lambda = \frac{8!}{6 \cdot 4 \cdot 4 \cdot 3 \cdot 2} = 70$$

7. A q-analogue of Atkinson's recursive algorithm for linear extensions of tree posets

Let P be a poset with n elements. Given a in P, we define the *spectrum of* a to be the sequence of non-negative integers $(\alpha_1, \ldots, \alpha_n)$ where α_i is the number of linear extensions of P with a occurring at location i. The following result of Atkinson computes the spectrum of elements in certain posets built from two posets P and Q with respect to the spectra of elements in both P and Q.

Proposition 7.1 (Atkinson [Atk90])**.** *Let P and Q be posets with p and q elements, $(\alpha_1, \ldots, \alpha_p)$ and $(\beta_1, \ldots, \beta_q)$ be the spectrum of elements a and b in P and Q respectively. Consider the partial order on $R = P \cup Q$ with the cover relations of P and Q and $a < b$. Then the rth*

element of the spectrum of γ in R equals

$$\gamma_r = \sum_{i=\max(1,r-v)}^{\min(u,r)} \alpha_i \binom{r-1}{i-1} \binom{u+v-r}{u-i} \sum_{j=r-i+1}^{v} \beta_j.$$

Corollary 7.2 (Atkinson [Atk90])**.** *Let P be a tree poset with n elements then the number of linear extensions $e(P)$ can be computed in $O(n^2)$ operations.*

Given a in P, the q-spectrum of a is the sequence $(\alpha_1(q), \ldots, \alpha_n(q))$ of polynomials, where

$$\alpha_i(q) := \sum_{w \in \mathcal{L}(P), w_i = a} q^{\mathrm{inv}(w)},$$

then

$$e_q(P) := \sum_{w \in \mathcal{L}(P)} q^{\mathrm{inv}(w)} = \sum_{i=1}^{n} \alpha_i(q).$$

Proposition 7.3. *Let P and Q be posets with elements $[p]$ and $\{p+1, \ldots, p+q\}$, $(\alpha_1(q), \ldots, \alpha_p(q))$ and $(\beta_1(q), \ldots, \beta_q(q))$ be the spectrum of elements a and b in P and Q respectively. Consider the partial order on $R = P \cup Q$ with the cover relations of P and Q and $a < b$. Then the rth element of the q-spectrum γ of a in R equals*

$$\gamma_r(q) = \sum_{i=\max(1,r-v)}^{\min(u,r)} \alpha_i(q) q^{(u-i+1)(r-i)} \begin{bmatrix} r-1 \\ i-1 \end{bmatrix}_q \begin{bmatrix} u+v-r \\ u-i \end{bmatrix}_q \sum_{j=r-i+1}^{v} \beta_j(q).$$

Proof. A linear extension z of R with a in position r is obtained by combining a linear extension x of P with a in position i for $\max(1, r-v) \leq i \leq \min(u, r)$ and a liner extension y of Q with b in some position j with $r - i + 1 \leq j \leq v$ as follows: (i) choose $i-1$ positions S_1 out of the $r-1$ positions before a in z to insert in order the entries of x before a, (ii) choose $u-i$ positions S_2 out of the $u+v-r$ positions after a in z to insert in order the entries of x after a, (iii) insert a in position r of z, and (iv) insert y in order in the remaining positions of z. Denote this construction by $z := \phi(x, y, S_1, S_2)$.

Next, we calculate the inversions of z in terms of the inversions of x and y. Each inversion of x and y is an inversion of z. Since the labels of P are smaller than those of Q, there will be inversions from the elements of P and Q before (after) a in z. The number of these inversions

is $\text{inv}(S_1) + \text{inv}(S_2)$ where the inversions of a set $S \subset [n]$ correspond to the inversions of the binary word of length n corresponding to the positions of set S. Finally, there will be additional inversions one for each pair of one of the $r - i$ elements of Q before r in z with one of the $u - i - 1$ elements of P after r in z. Thus in total we have

$$\text{inv}(z) \;=\; \text{inv}(x) + \text{inv}(y) + \text{inv}(S_1) + \text{inv}(S_2) + (u - i + 1)(r - i).$$

We now consider the contribution to the $\gamma_r(q)$ of all the linear extensions z obtained from fixed linear extensions x and y.

$$\begin{aligned}
&\sum_{S_1 \in \binom{[r-1]}{i-1}} \sum_{S_2 \in \binom{[u+v-r]}{u-i}} q^{\text{inv}(\phi(x,y,S_1,S_2))} = \\
&= q^{\text{inv}(x)+\text{inv}(y)+(u-i+1)(r-i)} \Big(\sum_{S_1 \in \binom{[r-1]}{i-1}} q^{\text{inv}(S_1)} \Big) \Big(\sum_{S_2 \in \binom{[u+v-r]}{u-i}} q^{\text{inv}(S_2)} \Big) \\
&= q^{\text{inv}(x)+\text{inv}(y)+(u-i+1)(r-i)} \begin{bmatrix} r-1 \\ i-1 \end{bmatrix}_q \begin{bmatrix} u+v-r \\ u-i \end{bmatrix}_q,
\end{aligned}$$

where we used a well-known expansion of q-binomial coefficients [Sta12, Prop. 1.7.1].

□

We now provide an algorithm to give a labeling which runs with the q-Atkinson algorithm. Let $\mathcal{Q}$ be a tree poset with elements n elements with potential labels $[n]$. Let f be a function, $f : \mathcal{P}(Q) \to \mathcal{P}([n])$.

Lemma 7.4. *The following algorithm gives a labeling which admits a q-analogue of Atkinson. The final labeling is provided by the map f.*

$f : \mathcal{Q} \to [n]$

Let q be a list

$q \leftarrow \mathcal{Q}$

while len(q) > 0 **do**

Let *elements* be the first item in q

Select random edge e from elements

Let L be the lower elements

Let U be the upper elements

labels $\leftarrow f$(elements)

$f : L \to \{l_1 \dots l_{|L|}\}$

$f : U \to \{l_{|L|+1} \dots l_{|L|+|U|}\}$

Add L and U to q

end while

Proof. Given a random edge e which splits the current poset into two components, L and U, the possible labels of L are all lower than the possible labels of U. This is the exact property desired. Therefore, the edge ordering defined by the algorithm can be used by q-Atkinson. □

Note that this algorithm does not give every possible labeling. One way to see this is that the three element poset

can be given any labeling, and q-Atkinson can still be run. Yet the labeling algorithm is defined solely by the edge ordering, making a uniform distribution over $(n-1)!$ labelings. In this case, the algorithm uniformly generates 2 labelings instead of 6.

8. Final Remarks

Calanques can be seen as a generalization of normal zigzags. What prevents a further generalization to all trees is the requirement to tackle disconnected entries in the component matrix.

As we have seen in this paper, this determinant formula is only useful for posets where the component matrix entries all have easily computable linear extensions through hook lengths. This means that the most general result is for posets whose master is d-Complete. The question remains, are there determinant formulas out there for other tree posets?

The Jacobi–Trudi identities are more powerful in the sense that in order to compute the matrices, we don't need to count any linear extensions of smaller posets; we just need to recall information on the skew shape. Can this determinant formula be translated into such language?

For a specific poset, there might be multiple ways to select k folds, or even ways to select different numbers of folds. As long as each of these master trees are valid, then their determinant formulas all have the same value. This shows an intriguing equivalence between determinants which do not appear similar on a surface level. Future work might be found in investigating these determinant equalities.

References

[Atk90] M D Atkinson, *On computing the number of linear extensions of a tree*, Order **7** (1990), 23–25.

[BW89] Anders Björner and Michelle L Wachs, *q-hook length formulas for forests*, Journal of Combinatorial Theory, Series A **52** (1989), no. 2, 165–187.

[BW91a] Anders Björner and Michelle L Wachs, *Permutation statistics and linear extensions of posets*, Journal of Combinatorial Theory, Series A **58** (1991), no. 1, 85–114.

[BW91b] Graham Brightwell and Peter Winkler, *Counting linear extensions*, Order **8** (1991), no. 3, 225–242.

[Fei53] W. Feit, *The degree formula for the skew-representations of the symmetric group*, Proceedings of the American Mathematical Society **4** (1953), no. 5, 740.

[FRT+54] J Sutherland Frame, G de B Robinson, Robert M Thrall, et al., *The hook graphs of the symmetric group*, Canad. J. Math **6** (1954), no. 316, C324.

[GGMM] Alexander Garver, Stefan Grosser, Jacob Matherne, and Alejandro Morales, *Determinant formulas for counting linear extensions of tree posets*, In Preparation.

[GM] Alexander Garver and Jacob Matherne, *Type $\mathbb{A}_n$ exceptional sequences.*

[GNW82] Curtis Greene, Albert Nijenhuis, and Herbert S. Wilf, *A probabilistic proof of a formula for the number of young tableaux of a given shape*, Young Tableaux in Combinatorics, Invariant Theory, and Algebra (1982), 17–22.

[Knu98] D. E. Knuth, *The art of computer programming volume 3: Sorting and searching*, Addison-Wesley, 1998.

[KY17] Jang Soo Kim and Meesue Yoo, *Hook length property of d-complete posets via q-integrals*, arXiv:1708.09109 (2017).

[Sag91] Bruce Eli. Sagan, *The symmetric group: representations, combinatorial algorithms, and symmetric functions*, Wadsworth Brooks/Cole Advanced Books Software, 1991.

[Sta12] Richard P. Stanley, *Enumerative combinatorics*, 2 ed., Cambridge University Press, 2012.

9. Appendix

9.1. **Atkinson Implementation.** We provide an implementation of Atkinson's algorithm below.

```
from sage.combinat.posets.posets import FinitePoset, Poset
import random
import numpy as np


def atkinson(fp):
binom_coeff = {} # "[a, b]" ----> a choose b
n = len(fp._elements)


#Generate Pascal Triangle up to row n


binom_coeff["[0, 0]"] = 1
binom_coeff["[0, 1]"] = 0
binom_coeff["[1, 0]"] = 1
for i in range(1, n+1):
for j in range(0, i+1):
key = str([i,j])
if j == 0:
binom_coeff[key] = 1
elif j > i:
binom_coeff[key] = 0
elif i == j:
binom_coeff[key] = 1
```

```
elif j == 1:
binom_coeff[key] = i
else:
binom_coeff[key] = binom_coeff[str([i-1, j-1])] + binom_coeff[str([i-1, j])]



# Recursively decompose tree and calculate alpha spectrums

def decompose(pos, alpha):

if pos.cardinality() == 1:
return [1]

num_elements = pos.cardinality()

alpha_lower = True

num_edges = len(pos.cover_relations())

covers = pos.upper_covers(alpha)

if len(covers) == 0:
alpha_lower = False
covers = pos.lower_covers(alpha)

# beta = random.choice(covers)
beta = covers[0]
```

```
edges = pos.cover_relations()

if alpha_lower:
edges.remove([alpha, beta])

else:
edges.remove([beta,alpha])



# Separate posets

new_pos = Poset([pos._elements, edges])

sub_pos = new_pos.connected_components()

# Recurse and recombine

# Determine which connected component contains alpha
if alpha in sub_pos[0]._elements:
P = sub_pos[0]
Q = sub_pos[1]

else:
P = sub_pos[1]
Q = sub_pos[0]

u = P.cardinality()
v = Q.cardinality()
```

```
P_spec = decompose(P, alpha)
Q_spec = decompose(Q, beta)

# Calculate alpha spec

alpha_spec = []

sums_Q_spec = [Q_spec[0]]

for i in range(1, len(Q_spec)):
sums_Q_spec.append(sums_Q_spec[i-1] + Q_spec[i])

# If alpha is not maximal, calculate formula 1
if alpha_lower:
for rank in range(0, num_elements):
r = rank + 1
rank_r_val = 0
for ind in range(max(1, r - v) - 1, min(u, r)):

i = ind + 1

k = binom_coeff[str([r-1, i-1])] * binom_coeff[str([u+v-r, u-i])]

if r - i + 1 <= v:
rank_r_val += P_spec[ind] * k * (sum(Q_spec[r-i:v]))
```

```
alpha_spec.append(rank_r_val)
# Otherwise, calculate formula 2, where beta < alpha
else:
for rank in range(0, num_elements):
r = rank + 1
rank_r_val = 0
for ind in range(max(1, r - v) - 1, min(u, r)):
i = ind + 1
k = binom_coeff[str([r-1, i-1])] * binom_coeff[str([u+v-r, u-i])]
if r - i >= 1:
rank_r_val += P_spec[ind] * k * (sums_Q_spec[r-i - 1])

alpha_spec.append(rank_r_val)

return alpha_spec




# l = decompose(fp, random.choice(fp._elements))
l = decompose(fp, fp._elements[0])

return sum(l)
```