Temperley–Lieb Immanants of Special Jacobi–Trudi Matrices Have Saturated Newton Polytopes

Feodor Yevtushenko

Abstract

We study whether the Schur supports of Temperley–Lieb immanants of Jacobi–Trudi matrices have saturated Newton polytopes. For a special subset of Jacobi–Trudi matrices, namely, ones that correspond to sufficiently wide skew-Schur tableaux under a bijection, we show that every Temperley–Lieb immanant has a maximal term in the canonical dominance order when expanded in the Schur symmetric polynomial basis, revealing that they have the saturated Newton polytope property. Lastly, we also offer several new conjectures and methods that may be useful in proving that general Temperley–Lieb immanants of Jacobi–Trudi matrices also have the saturated Newton polytope property.

Contents

1 Introduction

We begin with a brief introduction to several relevant concepts and properties, such as saturated Newton polytopes and Temperley-Lieb immanants. These ultimately connect our work to the broader field of enumerative combinatorics.

1.1 Saturated Newton polytope

Given a polynomial $f = \sum_{\alpha \in \mathbb{N}^n} c_\alpha x^\alpha \in \mathbb{R}[x_1, \ldots, x_n]$, the *support* of f is $\mathrm{supp}(f) = \{\alpha \in \mathbb{N}^n \mid c_\alpha \neq 0\}$, and the *Newton polytope* of f is the convex hull of $\mathrm{supp}(f)$, i.e.

$$\mathrm{Newton}(f) = \mathrm{conv}(\alpha \mid \alpha \in \mathrm{supp}(f)) \subseteq \mathbb{R}^n.$$

We say f has *saturated Newton polytope*, or f is *SNP*, if every lattice point of $\mathrm{Newton}(f)$ is in $\mathrm{supp}(f)$, i.e.

$$\mathrm{supp}(f) = \mathrm{Newton}(f) \cap \mathbb{Z}^n.$$

The SNP property was introduced by Monical–Tokcan–Yong in [MTY19] and has received considerable attention in algebraic combinatorics. For instance, the SNP property provides a sufficient condition for M-convexity and the *Lorentzian* property [Diz20]. As another example, the Newton polytope of a Schubert polynomial $\mathfrak{S}_w$ is the Schubitope, and using the Schubitope, Dizier–Yong gave an efficient algorithm to decide the vanishing of Littlewood–Richardson coefficients for Schubert calculus.

Many classical polynomials in algebraic combinatorics are known to have the SNP property. Examples include Schur polynomials, Schur P-polynomials, key polynomials, Macdonald polynomials [MTY19], Schubert polynomials [FMD18], double Schubert polynomials [CCRMM23]. See also [CS25, HND25, EY17, WZZ25].

We say a symmetric function f is *Schur positive* if it can be written as a positive sum of Schur polynomials, i.e. $f = \sum c_\lambda s_\lambda$ where $c_\lambda \geq 0$ for all λ. Its *Schur support* is the set $\{\mu \mid c_\mu \neq 0\}$. When f is Schur positive, there is an easy criterion for the SNP property. Recall the *dominance order* $\leq_D$ on the set of integer partitions

$$\mu \leq_D \lambda \text{ if } \sum_{i=1}^{k} \mu_i \leq \sum_{i=1}^{k} \lambda_i \text{ for all } k \geq 1.$$

We have the following fundamental result due to Monical–Tokcan–Yong.

Theorem 1.1 ([MTY19, Proposition 2.5]). *Suppose $f \in \mathrm{Sym}_n$ is homogeneous of degree d such that*

$$f = \sum_{\mu} c_\mu s_\mu,$$

and there exists λ such that $c_\lambda \neq 0$ and $c_\mu \neq 0$ only if $\mu \leq_D \lambda$. If $n \geq \ell(\lambda)$ and $c_\mu \geq 0$ for all μ, then f has SNP.

In other words, if f is Schur positive and the Schur support of f has a unique maximal element in the dominance order, then f has SNP.

1.2 Skew Schur polynomials

We have the following well-known corollary of Theorem 1.1:

Corollary 1.2. *Skew Schur polynomials have the SNP property.*

Proof. Recall the classical fact that every skew Schur polynomial $s_{\lambda/\mu}$ is Schur positive. Furthermore, its Schur support has a unique maximal element in the dominance order. This maximal element can be obtained by taking the content of the minimal semistandard Young tableau (SSYT) of shape λ/μ. This is the SSYT in which every box contains the smallest number possible, which is the number of boxes weakly North of it in the skew Young diagram. Hence, skew Schur polynomials have SNP. $\square$

Example 1.3. Let $\lambda = (4, 3, 3, 1)$ and $\mu = (2, 2, 1)$, we have

$$s_{\lambda/\mu} = s_{2,2,1,1} + s_{2,2,2} + s_{3,1,1,1} + 2s_{3,2,1} + s_{4,1,1}.$$

The unique maximal element in the Schur support of $s_{\lambda/\mu}$ is $(4, 1, 1)$, which is the content of the SSYT below.

$$\begin{array}{cccc} & & 1 & 1 \\ & & 2 & \\ & 1 & 3 & \\ 1 & & & \end{array}$$

1.3 Temperley–Lieb algebra

Temperley–Lieb immanants, introduced by Rhoades and Skandera in [RS05], is a subset of *dual canonical bases.* It was proved by [Hai93, RS06] that Temperley–Lieb immanants of Jacobi–Trudi matrices are Schur positive. Recently, Nguyen–Pylyavskyy [NP25] gave a combinatorial rule for expanding Temperley–Lieb immanants of Jacobi–Trudi matrices in the Schur basis in terms of Yamanouchi shuffle tableaux (see Section 2.2). These Temperley–Lieb immanants have been used by Lam–Postnikov–Pylyavskyy [LPP07] to prove several deep Schur-positivity conjectures. A more efficient formulation of Nguyen–Pylyavskyy shuffle tableaux rule was given by [NNW25] in terms of peelable tableaux.

With the new generalized Littlewood–Richardson rule for Temperley–Lieb immanants, we are now able to study their Schur support and check if they have SNP. Our main theorem is an affirmative answer to this question for a special set of Jacobi–Trudi matrices. We refer the readers to Section 2 for relevant definitions.

Theorem 1.4. *For any strict partitions μ and ν of the same length ℓ such that $\mu_\ell > \nu_1$, let $A_{\mu,\nu}$ be the corresponding Jacobi–Trudi matrix. For any Temperley–Lieb type τ, the Temperley–Lieb immanant $\mathrm{Imm}_\tau^{\mathrm{TL}}(A_{\mu,\nu})$ has saturated Newton polytope.*

Our proof generalizes the proof for skew Schur polynomials in Section 1.2. For any Temperley–Lieb type τ and shape $\mu \oslash \nu$ satisfying $\mu_\ell > \nu_1$, we show that there is a unique *minimal shuffle tableau* (to be defined in Section 7 with shape $\mu \oslash \nu$ in which every box contains the smallest number possible to achieve Temperley–Lieb type τ. The content of such a tableau is the maximal element in the Schur support of $\mathrm{Imm}_\tau^{\mathrm{TL}}(A_{\mu,\nu})$. See Section 4 for examples.

1.4 Section Overview

The subsequent section, Section 2, contains further background information about concepts such as the Temperley–Lieb algebra and shuffle tableaux. We then use these to frame our main result, Theorem 1.4, in the context of shuffle tableaux, and we use this equivalent interpretation throughout the rest of the paper. Subsequently, we present the main structure of our proof of Theorem 1.4, as well as some illustrated examples of the central steps in our proof, in Section 4. The next three sections execute this proof, with Section 5 establishing *canonical strings*, Section 6 defining *ascension paths* and integrating these with the canonical strings, and Section 7 finishing the proof through a minimality argument. The next two sections shift toward a broader exploration of generalized versions of Theorem 1.4. In particular, Section 8 offers several conjectures and methods of attack in proving this generalized version, and Section 9 summarizes our primary conjectures and hopes for future exploration of such generalized versions.

1.5 Acknowledgements

I would like to acknowledge my mentor Son Nguyen for significant guidance and review throughout the process of writing this paper. I would also like to thank the MIT PRIMES program for providing us with support and making this research possible.

2 Background

We will now describe several relevant previous results, which will situate our new results in the broader context of the Temperley–Lieb immanant and crystals.

2.1 Temperley–Lieb immanant

First, we introduce Temperley–Lieb immanant. Here, we follow the presentation in [NP25], see also [RS06, RS05]. The *Temperley–Lieb algebra* $TL_n(\xi)$ is the $\mathbb{C}[\xi]$-algebra generated by $t_1, \dots, t_{n-1}$ subject to the relations

$$\begin{aligned} t_i^2 &= \xi t_i, && \text{for } i = 1, \dots, n-1, \\ t_i t_j t_i &= t_i, && \text{if } |i-j| = 1, \\ t_i t_j &= t_j t_i, && \text{if } |i-j| > 1. \end{aligned}$$

A 321-*avoiding permutation* is a permutation in S_n that has no reduced decomposition of the form $\cdots s_i s_j s_i \cdots$ where $|i-j| = 1$. Because of the second relation, the Temperley–Lieb algebra has a natural basis $\{t_w \mid w \text{ is a 321-avoiding permutation}\}$, where $t_w := t_{i_1} \dots t_{i_k}$ for a reduced decomposition $w = s_{i_1} \dots s_{i_k}$. The map

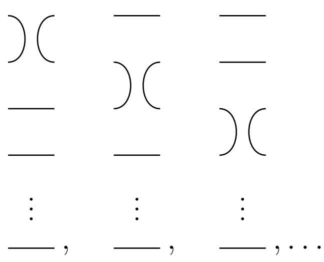
$$\theta \ : \ s_i \to t_i - 1$$

determines a homomorphism $\theta \ : \ \mathbb{C}[S_n] \to TL_n(2)$. For any permutation $v \in S_n$ and any 321-avoiding permutation $w \in S_n$, let $f_w(v)$ be the coefficient of $t_w \in TL_n(2)$ in the basis expansion of $\theta(T_v) = (t_{i_1} - 1) \dots (t_{i_k} - 1)$ for a reduced decomposition $v = s_{i_1} \dots s_{i_k}$. In [RS05], Rhoades and Skandera defined the *Temperley–Lieb immanant* of an $n \times n$ matrix $X = (x_{ij})$ by

$$\mathrm{Imm}_w^{\mathrm{TL}}(X) = \sum_{v \in S_n} f_w(v) x_{1,v(1)} \dots x_{n,v(n)}.$$

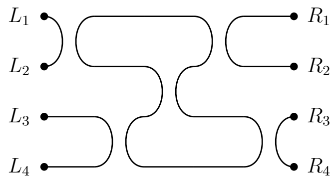
Example 2.1. Let $w = 12\dots n$ be the identity permutation, then $f_w(v) = (-1)^{\ell(v)}$ for all $v \in S_n$, so $\mathrm{Imm}^{\mathrm{TL}}_{12\dots n}(X) = \det(X)$.

Graphically, products of t_i's in $TL_n(\xi)$ can be computed graphically using *Temperley–Lieb diagrams*. The generators $t_1, t_2, t_3, \dots$ are represented by



and multiplication is represented by concatenation.

Example 2.2. The Temperley–Lieb diagram for $t_1t_3t_2t_1t_3 \in TL_4$ is



One can observe that each diagram consists of a noncrossing matching of the boundary vertices and internal loops. If the diagram of a is obtained from the diagram of b by removing k internal loops, then $b = \xi^k a$ in TL_n. Thus, we have a natural bijection between basis elements of TL_n and noncrossing matchings of $2n$ vertices.

2.2 Shuffle tableaux

Now we introduce our main object shuffle tableaux. We follow the presentation in [NP25]. Given partitions $\mu = (\mu_1 > \dots > \mu_n)$ and $\nu = (\nu_1 > \ldots > \nu_n)$, the *Jacobi–Trudi matrix* $A_{\mu,\nu}$ is the finite matrix whose i, j entry is $h_{\mu_i - \nu_j}$. By convention, we have $h_0 = 1$ and $h_m = 0$ for all $m < 0$.

Given two partitions μ and ν. We can define two skew shapes μ_R/ν_R and μ_B/ν_B where

$$\mu_R = (\mu_{2i-1} + i - k)_{i=1}^k,$$

$$\nu_R = (\nu_{2i-1} + i - k)_{i=1}^k,$$

$$\mu_B = \left(\mu_{2i} + i - \left\lfloor \frac{n+1}{2} \right\rfloor\right)_{i=1}^k,$$

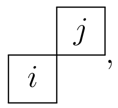
$$\nu_B = \left(\nu_{2i} + i - \left\lfloor \frac{n+1}{2} \right\rfloor\right)_{i=1}^k.$$

Then, the *shuffle diagram* D of shape $\mu \oslash \nu$ is constructed as follows.

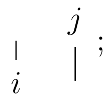
- If μ_R/ν_R has a square in position (i, j), then add into D a square in position $(2i - 1, 2j - 1)$.
- If μ_B/ν_B has a square in position (i, j), then add into D a square in position $(2i, 2j)$.

Remark 2.3. Note that in this section, rows are labeled from top to bottom, so the y-coordinate runs in the opposite direction to the "standard" Cartesian coordinate system. We maintain this convention to be consistent with [NP25] throughout this section, but we will revert to the more intuitive Cartesian system later for all our new results (see Remark 4.2 for more information).

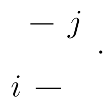
Definition 2.4 (Shuffle tableau)**.** A *shuffle tableau* of shape $\mu \oslash \nu$ is a filling of the shuffle diagram of shape $\mu \oslash \nu$ such that the entries are weakly increasing along the rows and strictly increasing along the columns. The *weight* of a shuffle tableau T, denoted $\omega(T)$, is $\prod_{i\geq 1} x_i^{c_i}$, where c_i is the number of i-entries in T. The Temperley–Lieb type of T, denoted $\psi(T)$, is obtained as follows: for each pair



if $i \leq j$, then draw two lines

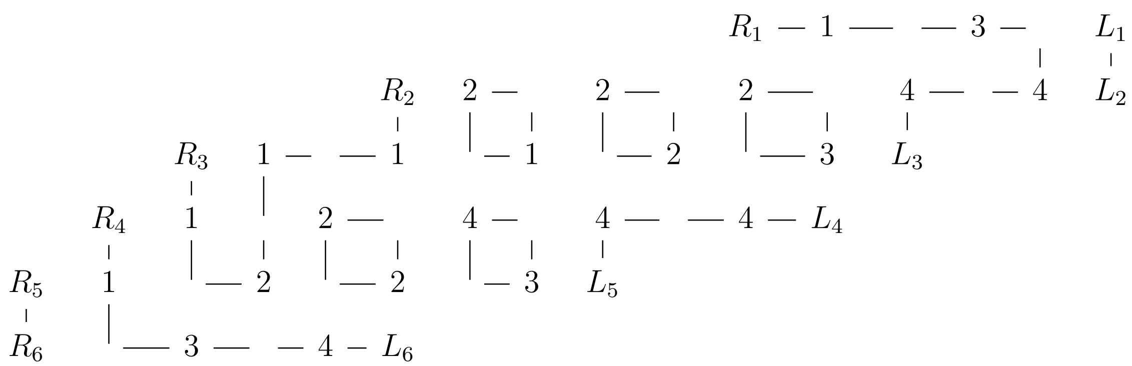


if $i > j$, then draw two lines

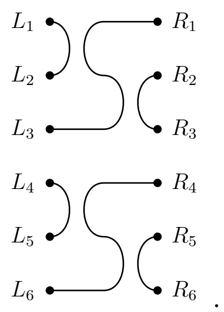


By convention, the squares North and West of the tableau have value $-\infty$, and those South and East of the tableau have value ∞. Finally, we put L_i at the end of the ith row and R_i at the beginning of the ith row.

Example 2.5. Let $\mu = (9, 9, 7, 6, 4, 2)$ and $\nu = (7, 4, 2, 1, 0, 0)$, then $\mu_R/\nu_R = (7, 6, 4)/(5, 1, 0)$ and $\mu_B/\nu_B = (7, 5, 2)/(2, 0, 0)$. A shuffle tableaux of shape $(\mu_R/\nu_R) \oslash (\mu_B/\nu_B)$ can be seen below, together with the lines drawn.



This means that the Temperley–Lieb type is



Remark 2.6. The condition in Theorem 1.4 that μ and ν are strict partitions means that the endpoints R_{i+1} (resp. L_{i+1}) is always strictly left of R_i (resp. L_i). The condition that $\mu_\ell > \nu_1$ is to make sure that L_ℓ is sufficiently right of R_1 for our purposes in the next section.

Summing over shuffle tableaux of Temperley–Lieb type τ and shape $\mu \oslash \nu$ gives $\mathrm{Imm}_\tau^{\mathrm{TL}}(A_{\mu,\nu})$.

Proposition 2.7 ([NP25, Corollary 3.10])**.** *For any basis element τ of $TL_n(2)$ and partitions μ, ν, we have*

$$\mathrm{Imm}_\tau^{\mathrm{TL}}(A_{\mu,\nu}) = \sum_{\substack{T \text{ of shape } \mu \oslash \nu \\ \psi(T)=\tau}} \omega(T).$$

Next, we recall crystal operators on shuffle tableaux and Yamanouchi shuffle tableaux.

Definition 2.8. Given a shuffle tableau T, the i-reading word w_i of T is obtained by first ignoring the columns containing both i and $i+1$, then iteratively reading the remaining i and $i+1$ from bottom to top, left to right. The *crystal operators* E_i and F_i act on $w_i(T)$ in the following (standard) way.

- View each i as a close parenthesis ")" and each $i+1$ as an open parenthesis "(".
- Match the parentheses in the usual way.
- E_i changes the leftmost unmatched $i+1$ to i (if exists), and F_i changes the rightmost unmatched i to $i+1$ (if exists).

The connected components of the graph on shuffle tableaux formed by the above crystal operators are called *Temperley–Lieb crystals.* In fact, they are type A Kashiwara crystals.

Theorem 2.9 ([NP25, Theorem 5.1])**.** *Temperley–Lieb crystals are type A Kashiwara crystals.*

A shuffle tableau T of shape $\mu \oslash \nu$ is called *Yamanouchi* if no crystal operator E_i can be applied on T. The Littlewood–Richardson rule for Temperley–Lieb immanants of $A_{\mu,\nu}$ follows from Theorem 2.9.

Theorem 2.10 ([NP25, Theorem 6.2])**.** *For any partitions μ, ν, any Temperley–Lieb type τ, and any partition λ, the coefficient of the Schur function s_λ in $\mathrm{Imm}_\tau^{\mathrm{TL}}(A_{\mu,\nu})$ is the number of Yamanouchi shuffle tableaux of shape $\mu \oslash \nu$, Temperley–Lieb type τ, and content λ.*

Remark 2.11. There is a more efficient description of Yamanouchi shuffle tableaux in terms of peelable tableaux in [NNW25]. However, we do not need this description for our purpose.

3 Interpretation and Comments

Before we proceed, we will clarify the result we are proving in connection to some broader conjectures. Recall that in the Schur function expansion of a Temperley–Lieb immanant with Temperley–Lieb type τ, we have,

Theorem 3.1 ([Hai93, RS06]). *For any τ, λ, μ, we have*

$$\mathrm{Imm}_{\tau}^{\mathrm{TL}}(A_{\mu,\nu}) = \sum_{\lambda} c_{\tau,\mu,\nu}^{\lambda} s_{\lambda}$$

where μ, ν, and λ are partitions and $c_{\tau,\mu,\nu}^{\lambda}$ are nonnegative coefficients.

Of all the partitions λ, let P be the set that have $c_{\tau,\mu,\nu}^{\lambda} > 0$. Recall the standard dominance order $\geq_D$ of partitions introduced in section 1.1. We assert the following conjecture:

Conjecture 3.2. *There exist partitions $\lambda_{min}, \lambda_{max} \in P$ such that $\lambda_{min} \leq_D \lambda \leq_D \lambda_{max}$ for all $\lambda \in P$.*

Through extensive automated code-based testing, we are highly confident that this conjecture is correct in the fully generalized case where the shape μ/ν is any shuffle tableau. Nevertheless, in this paper, we confine ourselves to the case where μ/ν is a semi-standard Young tableau, and we will solely prove the existence of λ_{max}. We also currently impose the restriction that the tableau must be sufficiently wide (see Remark 4.1).

To prove this, we will use the fact that these coefficients have an elegant combinatorial representation:

Theorem 3.3 ([NP25]). *$c_{\tau,\mu,\nu}^{\lambda}$ counts the number of Yamanouchi tableaux with shape μ/ν, Temperley–Lieb type τ, and content λ.*

We will be relying on this interpretation throughout the rest of this paper. In particular, we will prove the following:

Theorem 3.4. **??** *Fix a shuffle tableau shape μ/ν that is also the shape of a sufficiently wide semi-standard Young tableau and a Temperley–Lieb type τ. Let P be the set of contents λ of the tableau with this shape and type. We show there exists $\lambda_{max} \in P$ such that $\lambda \leq_D \lambda_{max}$ for all $\lambda \in P$.*

By Theorems 1.1, 3.1, and 3.3, proving the above would imply Theorem 1.4, the main theorem of this paper.

4 Proof Outline and Examples

In light of the previous section, here we present the outline of our proof. Recall that we are proving the desired result for all shapes $\mu \oslash \nu$ with μ and ν being strict partitions of the same length ℓ.

Remark 4.1. We also enforce $\mu_\ell > \nu_1$: geometrically, this translates to all left endpoints being strictly left of all right endpoints in the tableau (see below for the formal definition of endpoints).

1. For each shape $\mu \oslash \nu$ and Temperley–Lieb type τ, we draw *canonical strings* (Section 5). These are strings connecting the L_i's and R_i's according to the Temperley–Lieb type τ. Lemma 5.17 states that these strings are globally maximally pushed down.

2. Based on the canonical strings, for each box, we draw its *ascension path* (Section 6). Each ascension path starts from the corresponding box, ends in a cell without a box, and consists of either upsteps $(+0, +2)$ or diagonal steps $(+1, +1)$.

3. Theorem 6.10 states that for each box, the number of steps in its ascension path is the smallest number that it may contain in order to get the Temperley–Lieb type τ.
4. Finally, Theorem 7.9 states that the minimum tableau obtained by filling in each box its smallest number gives a valid shuffle tableau with Temperley–Lieb type τ.

Remark 4.2. Here and from now on in the paper, assume any coordinates (x, y) refer to the standard Cartesian coordinates - that is, $+x$ is right and $+y$ is up.

Remark 4.3. Note that in this section and all future sections, we will create and maintain a stricter definition of an "endpoint". Specifically, we have the following definition:

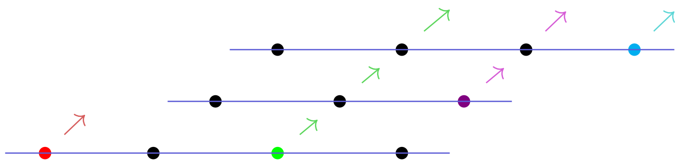
Definition 4.4. In a given row i units from the top (so at coordinate $y_i = y_0 - i + 1$), let the leftmost (smallest x-coordinate) occupied point be at (x_l, y_i) and the rightmost (largest x-coordinate) occupied point be at (x_r, y_i). Then we define the left endpoint $L_i = (x_l - 1, y_i)$ and the right endpoint $R_i = (x_r + 1, y_i)$. Note that in several prior works this convention is flipped and the L_i were on the right (and vice versa); we will instead use the far more intuitive "left-on-left, right-on-right" definition because we will heavily rely on the geometric nature of these endpoints throughout the proof.

Theorem 1.4 follows since the minimum tableau is clearly Yamanouchi: if some operator E_i can be applied, then some box can have smaller number, contradicting the minimum assumption. In addition, the content of the minimum tableau is the maximal element in the Schur support of $\mathrm{Imm}_\tau^{\mathrm{TL}}(A_{\mu,\nu})$.

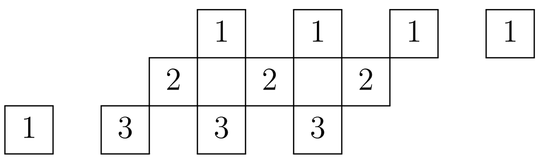
4.1 Illustrated Examples

We now present two examples of tableaux with the same shape but different Temperley–Lieb type, illustrating how our construction differs between the two cases and the effects on the tableaux, thus ultimately showcasing our entire method.

Example 4.5. Consider the shuffle diagram in the figure below, where each dot represents a box in the diagram.

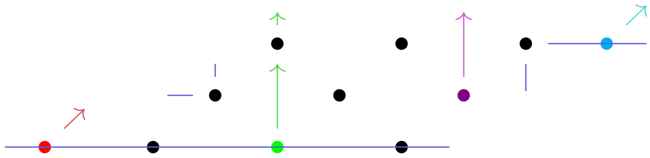


For $\tau = \{\{L_1, R_1\}, \{L_2, R_2\}, \{L_3, R_3\}\}$, the canonical strings are the blue strings in the figure. For each box, the ascension path consists of diagonal steps only. Thus, the number of steps in any box' ascension path is the number of boxes above it diagonally, including itself. Hence, the minimum tableau for this example is the following.

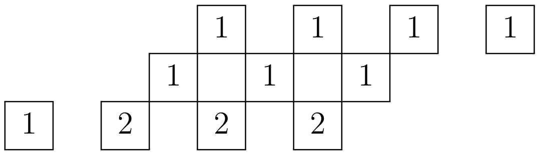


Note that under the bijection in [NP25, Remark 6.4], this minimum shuffle tableau corresponds to the minimum SSYT of shape $(5, 4, 4)/(1, 1, 0)$, which is expected since $\mathrm{Imm}_\tau^{\mathrm{TL}} = \det$ for $\tau = \{\{L_1, R_1\}, \{L_2, R_2\}, \{L_3, R_3\}\}$.

Example 4.6. Consider the same shuffle diagram as in Example 4.5 but with $\tau = \{\{L_1, L_2\}, \{R_1, R_2\}, \{L_3, R_3\}\}$, the canonical strings are as follows.



Observe that many ascension paths are different. For example, the paths corresponding to the green and violet boxes are shorter than those in Example 4.5. Thus, in this case, the minimum tableau is the following.



Remark 4.7. Once again, recall that dots in the diagrams represent boxes containing numbers in the tableau. We do this to make the diagrams in this paper more compact.

5 Canonical Strings

In this section, we define and explore *canonical strings*, which allow us to directly connect the Temperley–Lieb type to the construction of the tableau. Specifically, by greedily pushing these strings to the bottom of the grid in our construction, we can place smaller elements throughout the grid, as will be explored in Section 6.

Definition 5.1. Given a Temperley–Lieb type τ, recall that τ is a noncrossing matching of $2n$ endpoints $L_1, \dots, L_n, R_1, \dots, R_n$ with n strings. For each string, we designate one endpoint as the *starting endpoint*, and the other endpoint as the *finishing endpoint*, as follows.

- If the string connects L_i and R_j for some i, j, then the starting endpoint is L_i.
- If the string connects R_i and R_j for some $i < j$, then the starting endpoint is R_i.
- If the string connects L_i and L_j for some $i < j$, then the starting endpoint is L_j.

Definition 5.2. Given a shape $\mu \oslash \nu$ and a Temperley–Lieb type τ. A string in τ is *type* α if the starting endpoint is strictly West of the other endpoint. A string is *type* β otherwise.

The following observation is straightforward.

Lemma 5.3. *All strings connecting L_i and L_j for some i, j are type α. All type α strings have starting endpoints L_i for some i. All strings connecting R_i and R_j for some i, j are type β.*

We can thus process all strings with a left starting endpoint in increasing order of their index. For the right endpoints, we instead interpret the order as a "nesting", where we go from the inside to the outside. Specifically, view the string connecting R_a, R_b as corresponding to the ordered pair (a, b).

Definition 5.4. Given two type β strings with endpoint indices (i, ℓ) and (j, k), we say (j, k) is nested inside (i, ℓ) if $i < j < k < \ell$. A *nested ordering* of type β strings is an ordering such that if (j, k) is nested inside (i, ℓ), (j, k) is strictly before (i, ℓ).

Now we are ready to construct the *canonical strings.*

Definition 5.5 (Canonical strings)**.** Given a shuffle diagram $\mu \oslash \nu$ and a Temperley–Lieb type τ, the canonical strings are constructed as follows.

1. First, go through all type β strings that have both endpoints as R_i in any nested order. For a string connecting R_a and R_b with $a < b$:
   - (a) Begin at R_a and go up one unit.
   - (b) If R_b is one unit right of the current cell, go there and terminate.
   - (c) Otherwise, go 2 units right if the corresponding square is free, else go 2 units up. Go back to step (b).
2. Then, go through all strings that have an endpoint of the form L_i, and process these in increasing order of i. (Skip a string if it has already been processed.) For a given string, if it has two left endpoints L_i and L_j with $i < j$,
   - (a) Begin at the starting endpoint L_i.
   - (b) Keep descending 2 units at a time until such a step is no longer legal. Then take one step right.
   - (c) If the finishing endpoint L_j is immediately above the current position, go there and terminate.
   - (d) Go 2 units right if the corresponding square is free, else go 2 units up. Go back to step (c).

   Else, it must have endpoints L_i and R_j. If it is type α,
   - (a) Begin at the starting endpoint L_i.
   - (b) Keep descending 2 units at a time until such a step is no longer legal. Then take one step right.
   - (c) If the square one unit to the right is the finishing endpoint, then go there and terminate.
   - (d) Go 2 units right if the corresponding square is free, else go 2 units up. Go back to step (c).

   Else, it must be type β, in which case,
   - (a) Begin at the starting endpoint L_i.
   - (b) If the square one unit down is the finishing endpoint, then go there and terminate.
   - (c) Go 2 units down if the corresponding square is free, else go 2 units left. Go back to step (b).

Remark 5.6. Note that this construction, and the rough central ideas of the subsequent logic, will work for *any* strict permutations μ and ν, regardless of whether $\mu_\ell > \nu_1$. Nevertheless, we enforce this condition going forward (throughout the rest of the paper) to simplify the casework. This manifests as there being no type β strings to process in the second step of the above algorithm. Some discussion on this generalized case can be found in Section 9.

Lemma 5.7. *The canonical strings are well-defined.*

Proof. First, we can see that the order in which we process strings with both right endpoints does not matter, so long as the order is nested. This is because if such a string connects R_a and R_b with $a < b$, then as per the construction, it only accesses rows with $y \in [a, b]$. Since strings cannot intersect, for a given pair of strings, such y-intervals will either be nested or disjoint.

Next, notice that strings never go in the $-x$ direction, and can only go in the $-y$ direction at the very start. This means that strings can always go 2 units up as claimed, as it is impossible for a string to be above the current string given this processing order. This also means that the termination steps described above are always possible, as the aforementioned geometry of the strings prevents them from "blocking" other strings from doing this.

Lastly, notice that in the above process, the steps for strings with an L_i endpoint involve an even number of steps down followed by a step right. Since $\mu_\ell > \nu_1$, the L_i that gets processed (first) will always be strictly to the left of the other endpoint, so following the initial step, it is impossible for the current string position to be right of the other endpoint. We will also briefly justify the legality of this step on the basis of parity. Firstly, assume for the sake of contradiction that descending $2k + 1$ squares from the start point would not intersect any already-placed strings (proceeding in the order of the aforementioned algorithm), but descending $2k + 2$ squares would. Let the start point L_i be (x_0, y_0). By trivial parity, $(x_0, y_0 - 2k - 1)$ corresponds to a numbered cell, and by construction, there must be a segment from an already-existing string of the form $(x_0, y_0 - 2k - 2) \to (x_0 + 1, y_0 - 2k - 2)$. (Specifically, by the construction order, the only already-placed possible segments with this x-value are in the up or right directions.) Refer to the construction of χ in Lemma 6.6: this new string would thus have χ-value $(x_0 + 1) \mod 2$ (by using this segment). However, the original string in question would have the same χ-value (by using the segment $(x_0, y_0 - 2k) \to (x_0, y_0 - 2k - 1)$), thus failing the adjacency condition (which states that they must have opposite χ values).

Thus, strings can only descend $2k$ squares initially for some k. If the follow-up move $(x_0, y_0 - 2k) \to (x_0 + 1, y_0 - 2k)$ is impossible by virtue of hitting a string, then there must be another string that takes a step of the form $(x_0 + 1, y_0 - 2k - 1) \to (x_0 + 1, y_0 - 2k)$, which contradicts χ parity by the same logic. Consequently, we can deduce that the rightward step is possible in every such scenario. $\square$

5.1 Enclosed Sets and Minimality

Heuristically, our construction is tailored to push strings down as much as possible, so that they "enclose" the smallest possible area. In this section, we rigorously define and prove this property. As will be shown in future sections, this makes the tableau elements as small as possible.

Definition 5.8. Let the set of tableau cells *enclosed* by a string $\mathbf{s}_{rr}$ with both right endpoints be the set of cells that lie inside or on the figure enclosed by the string and the right boundary of the tableau.

Lemma 5.9. *The enclosed set in the canonical construction for any such* $\mathbf{s}_{rr}$ *is a subset of any other possible enclosed set for the same* $\mathbf{s}_{rr}$*.*

Proof. Strong induct over all the $\mathbf{s}_{rr}$ in any nested ordering. Note that if string s_1 is nested inside s_2, replacing s_1 with a string with the same endpoints and enclosing a subset of what it initially enclosed will not hinder the placement of s_2. This is because the set of points available for s_2 is the set not enclosed by any nested string, and changing those sets to subsets of themselves can only add possible points for s_2 to pass through. Thus, by the strong inductive hypothesis, we can assume that all strings nested inside $\mathbf{s}_{rr}$ are the canonical strings.

If $\mathbf{s}_{rr}$ connects R_a and R_b for $a < b$, then trivially the canonical string only encloses certain points (x, y) with $a \leq y \leq b$. Next, we can see that the set of lattice points enclosed by $\mathbf{s}_{rr}$ is precisely

$$S_{rr} = \{x \geq mx(a + 2i) - 1 \text{ for } y \in \{a + 2i, a + 2i + 1\}\},$$

over $0 \leq i \leq \frac{b-a-1}{2}$, where $mx(y)$ is the minimum x coordinate enclosed by all internally nested strings for this given y value. (Note that the case $b - a = 1$ is already trivial, so we do not need to worry about this case.) We set $mx(a)$ as the x-coordinate of R_a. This form immediately follows from our construction, where we greedily push the string right. Note that by strong induction, we can see that right/right strings solely enclose groups of points of the form $x \geq x_0$ for a given y-value. Furthermore, $\mathbf{s}_{rr}$ has exactly $\frac{b-a-1}{2}$ right/right strings nested inside it, since if R_x connected to a left endpoint for some $a < x < b$, then the corresponding string would intersect $\mathbf{s}_{rr}$, contradiction. This means that $mx(a + 2i)$ has a corresponding string for any i, making the above well-defined.

We now show that all other strings connecting R_a and R_b must have S_{rr} as a subset. Trace any such string, starting at R_a and ending at R_b. For any $y = a+2i$, the maximum x reached by the string is clearly at most $mx(a+2i)-1$, as any higher y would intersect the internally nested strings. Thus, the string must enclose $x \geq mx(a+2i)-1$ for $y = a+2i$. Next, for any given i, consider all transitions of the form $(x, a + 2i) \to (x, a + 2i + 1)$ for a given i, and of these, consider the one with the largest x (call it x_0). By construction, all points with $y = a + 2i + 1$ and $x \geq x_0$ must be enclosed by the string. But $x_0 \leq mx(a+2i)-1$ as we established, since $(x_0, a+2i)$ must be on the path, meaning that the string encloses $x \geq mx(a + 2i) - 1$ for $y = a + 2i + 1$. Thus, all cells in S_{rr} are enclosed by the string, as desired. □

Lemma 5.10. *Starting from an arbitrary (not necessarily canonical) legal string configuration, you can always modify it to make a new string configuration so that when the string is traced starting at the leftmost endpoint and ending at the rightmost endpoint, the following are true:*

1. *The string never takes a step left.*
2. *After the string takes a step up, it never takes a step down again later.*

Proof. Trace each string from the leftmost endpoint to the rightmost endpoint of the string. Perform the following algorithm:

1. If the string ever takes a step left of the form $(x, y) \to (x-1, y)$, then consider the last step after this step that takes the form $(x-1, y+k) \to (x, y+k)$. Alternatively, if no such step exists, then there must have been a prior step of the form $(x - 1, y + k) \to (x, y + k)$, in which case consider the earliest such step. By parity (see Lemma 6.6 for more details), k must be odd. Then, delete all the intermediate steps between $(x + y)$ and $(x, y + k)$ and replace them with k purely vertical connections between the two points.

2. Now, say that the string takes a step up of the form $(x, y) \to (x, y + 1)$ and later takes a step $(x + k, y + 1) \to (x + k, y)$. As before, consider the last such step, then by parity, delete all intermediate steps and replace them with connections of the form $(x, y) \to (x + 1, y) \to \cdots \to (x + k, y)$.

Note that while different may strings may intersect throughout the course of the algorithm, they will clearly not intersect after the algorithm has terminated. This is because such an intersection would require the final strings to go left then right or up then down to intersect the newly formed segments, which trivially does not occur. Strings do not intersect themselves for the same reason. □

Definition 5.11. Similarly to $\mathbf{s}_{rr}$, we can define a set of tableau cells *enclosed* by a string $\mathbf{s}_{lr}$ (with one left and one right endpoint) or $\mathbf{s}_{ll}$ (with both left endpoints). Of the two endpoints of the string, let P_1 be the one with lower x-coordinate and P_2 be the one with higher x-coordinate. Consider the sequence $L_1, L_2, L_3, \ldots, L_\ell, R_\ell, R_{\ell-1}, \ldots R_1$, where we traverse the endpoints in a clockwise fashion. Trivially, P_1 occurs before P_2 in this sequence because of Remark 4.1. Now, consider the region with border:

- the border of the tableau over $L_1, L_2, \ldots, P_1$,
- the string $\mathbf{s}$,
- the border of the tableau over $P_2, \ldots, R_{\ell-1}, R_\ell$,
- the bottom border of the tableau.

Let the set of cells enclosed by $\mathbf{s}$ be the set of cells (x, y) lying on the boundary of or inside this region with the additional constraint that $x \in [x_{P_1}, x_{P_2}]$.

Remark 5.12. Note that in defining the string as the boundary, we draw a vertex at the center of each square with a number and connect these vertices as per the string to form the boundary of the region. This central vertex is what we tie to the (x, y) coordinate system.

Definition 5.13. Let $mx_{[x_l,x_r]}(y)$ denote the minimum (integer) x coordinate enclosed by any currently placed string(s) for a given y value, restricted to the interval $[x_l, x_r]$. If no such value exists, then $mx_{[x_l,x_r]} = x_r + 1$. If a given y is below the bottom row of the grid, then $mx_{[x_l,x_r]} = x_l$, as we can easily interpret "enclosing" as instead extending infinitely far down.

This generalizes our definition and usage of $mx(y)$ when proving Lemma 5.9.

Lemma 5.14. *The enclosed set in the canonical construction for any such* $\mathbf{s}_{lr}$ *or* $\mathbf{s}_{ll}$ *is a subset of any other possible enclosed set for the same* $\mathbf{s}$.

Proof. We apply similar logic as that for $\mathbf{s}_{rr}$. We will now strong induct on the number of such strings already placed on the grid to prove the desired statement, proceeding in the order specified by Definition 5.5. We will also add the additional constraints that over all strings $\mathbf{s}'$ not yet processed, if their corresponding x-interval is $[x_{P'_1}, x_{P'_2}]$, then $mx_{[x_{P'_1},x_{P'_2}]}(y)$ is a monotonically nondecreasing function of y, and all cells with $x \geq mx_{[x_{P'_1},x_{P'_2}]}(y)$ are enclosed, to our inductive assertion.

Note that the latter statement is trivially true when no $\mathbf{s}_{lr}$ or $\mathbf{s}_{ll}$ strings have been placed yet, as the definition(s) of $\mathbf{s}_{rr}$ make the remaining right boundary monotonic and Lemma 5.9 makes the enclosed region of all such strings a subset of the enclosed region for any other case. This establishes the base case of our induction.

We will now prove the inductive step. First, recall that by Lemma 5.10, we can assume that the string in the new construction does not take any leftward steps, so we constrain ourselves to $x \in [x_{P_1}, x_{P_2}]$. Notice that the initial descent of $\mathbf{s}$ is maximal by construction, only limited by the set of cells bounded by already placed strings in the canonical construction, and by the inductive hypothesis, this set is a subset of the set enclosed by any other construction. This means that, since the starting point is the same, the set of cells enclosed by $\mathbf{s}$ with $x = x_{P_1}$ is a subset of the cells enclosed by any other construction. Now, say $\mathbf{s}$ descends to a point (x_{P_1}, y_0), and the termination point is $P_2 = (x_{P_2}, y_{P_2})$. We can also clearly see that all cells with $y \leq y_0$ are enclosed by each $x \in [x_{P_1} + 1, x_{P_2}]$ by the monotonicity condition in the inductive step.

We now shift our focus to proving that the points enclosed by the canonical $\mathbf{s}$ with $x \in [x_{P_1} + 1, x_{P_2}]$ must also be enclosed by $\mathbf{s}$ in any other construction. Now, however, notice that restricting our view to $x \in [x_{P_1}, x_{P_2}]$ (and imposing a hard cutoff for horizontal motion at $x = x_{P_2}$) gives us a scenario where we can apply the exact same logic as used in Lemma 5.9. Specifically, that lemma's exact logic shows that any such string must enclose

$$S = \left\{ x \geq mx_{[x_{P_1}, \infty]}(y_0) - 1 \text{ for } y \in \{y_0 + 2i, y_0 + 2i + 1\} \right\},$$

over $y \in [y_0, y_{P_2}]$ and corresponding $i \in \mathbb{Z}$, and it also must trivially enclose

$$S = \{x \geq x_{P_2} \text{ for } y \in [y_0, y_{R_2}]\},$$

since the string must pass through and thus include *some* x for each $y \in [y_0, y_{R_2}]$. Combining these yields that the string must ultimately enclose

$$S = \left\{ x \geq mx_{[x_{P_1}, x_{P_2}]}(y_0) - 1 \text{ for } y \in \{y_0 + 2i, y_0 + 2i + 1\} \right\},$$

but this, combined with the $x = x_{P_1}$ and $y \leq y_0$ cases from earlier, is precisely the set of cells enclosed by $\mathbf{s}$, as can be seen from the construction. (For the specifics of this logic, refer to the proof of Lemma 5.9.)

We will now demonstrate the desired $[x_{P_1'}, x_{P_2'}]$ condition. Indeed, notice that by virtue of the processing order, $x_{P_1} < x_{P_1'}$. If $x_{P_2} < x_{P_1'}$, then the values of $mx_{[x_{P_1'}, x_{P_2'}]}(y)$ are clearly not affected since $\mathbf{s}$ does not access any points with x-coordinates in this interval. Otherwise, notice that $\mathbf{s}'$ is "above" (encloses a strict superset of) $\mathbf{s}$. Since the path of $\mathbf{s}$ only has up/right steps after the initial downward step, the newly enclosed cells that it contributes within the x-interval $[x_{P_1'}, x_{P_2'}]$ continue to make $mx_{[x_{P_1'}, x_{P_2'}]}(y)$ monotonically nonincreasing, as it either adds cells to decrease the value (with a monotonic boundary) for a given y or does not access that y. Additionally, since we only consider $\mathbf{s}'$ instances that enclose $\mathbf{s}$, we can also see that all $x \geq mx_{[x_{P_1'}, x_{P_2'}]}(y)$ in the scope $[x_{P_1'}, x_{P_2'}]$ are filled.

We have proven our entire inductive hypothesis, so we are done by induction. □

Lemma 5.15. *For a given Temperley-Lieb type τ, if any tableau with shape μ/ν has Temperley-Lieb type τ, then the canonical string construction will successfully produce a valid tableau.*

Proof. First, perform the algorithm of Lemma 5.10 on the string construction. Now, consider processing the strings in the following order:

1. All $\mathbf{s}_{rr}$ in any nested order.

2. All $\mathbf{s}_{lr}$ and $\mathbf{s}_{ll}$ in ascending order of i where the (smaller index) left endpoint is L_i.

At each step, we compare the set of all possible remaining non-enclosed points in the new construction to the canonical construction, after which we place the string down and move on to the next string in the order. Trivially, each string in the canonical construction only encloses points with x-coordinates between or equal to the corresponding endpoints of the strings. Additionally, for these x-values, the canonical construction's string will only enclose a subset of the y-values that any other construction will enclose, meaning that the set of remaining cells for the rest of the strings to pass through in any other construction is a subset of the cells for the canonical construction.

We can now induct on the number of strings already processed (base case trivial). Additionally, if we can draw any string connecting the two endpoints, then we can also draw the canonical string by virtue of the geometry imposed by Lemmas 5.7 and 5.10. Thus, by induction, if we can draw any string configuration, we can draw the canonical string configuration too. ☐

Definition 5.16. We now define what it means for a string to be maximally pushed downward. Specifically, trace the string from the endpoint of lower x-coordinate to the one of higher x-coordinate, and consider all the steps of the form $(x, y) \to (x + 1, y)$ for a given fixed x. Consider the step of this form with y minimal; define this as $y_0(x)$. A string with endpoints P_1 and P_2 (where $x_{P_1} < x_{P_2}$) is *maximally pushed down* if, for all $x \in [x_{P_1}, x_{P_2} - 1]$, $y_0(x)$ is minimal over such strings.

Lemma 5.17. *Canonical strings are maximally pushed down.*

Proof. As before, perform the algorithm of Lemma 5.10 on the string construction, then process the strings in the following order:

1. All $\mathbf{s}_{rr}$ in any nested order.
2. All $\mathbf{s}_{lr}$ and $\mathbf{s}_{ll}$ in ascending order of i where the (smaller index) left endpoint is L_i.

Say that we have already placed some number of strings (possibly 0). Consider the next string $\mathbf{s}$ in the order.

First, note that by Lemmas 5.9 and 5.14, the set of available points for $\mathbf{s}$ is a superset of the set of available points for any other configuration. We claim that for each $x \in [x_{P_1}, x_{P_2} - 1]$, $y = y_0(x)$ is the smallest value such that $(x, y) \to (x + 1, y)$ is a legal step inside the tableau that is contained in the set of available points this string. Indeed, this follows from how we construct the strings in the canonical construction. (Note that y mod 2 is fixed by Lemma 6.6.) Thus, if y were any lower in a different configuration, then the segment $(x, y) \to (x + 1, y)$ would either be out of the grid or would intersect an already-placed string, contradiction. ☐

6 Ascension Paths

Next, we define the idea of an *ascension path* based on the string construction(s) from the previous section, which ultimately allows us to place lower bounds on the values of tableau elements. Specifically, we find a sequence of points starting at a given point and ending above the grid. By proving that the difference between consecutive points in this sequence is at least 1, we can add all these differences to bound our elements.

For each box, we will now construct its ascension path. Recall the following two rules:

Lemma 6.1. *Consider two squares with elements a and b.*

- *If a square containing a is 2 units above a square containing b, then $b \geq a + 1$.*
- *If squares containing a and b are in the configuration*

$$\begin{array}{cc} - & a \\ b & - \end{array}$$

then $b \geq a+1$. Specifically, if a crossing (defined later in Definition 6.2) starts at b and ends at a, then $b \geq a+1$.

Given a fixed Temperley–Lieb type τ and a square P in the shuffle diagram, we will locate a coordinate Q, which may not necessarily be in the shuffle diagram, such that for any shuffle tableau, we will be able to take a series of steps, which is a combination of $(0,2)$ steps and $(1,1)$ steps, to get from P to Q. We call such series a *path*. Furthermore, the number of steps in every path from P to Q is the same, as will shortly be shown in Theorem 6.5. Along any path, after each step, the entry contained in the box decreases by at least 1. Hence, such path gives a lower bound for the entry in P.

6.1 Canonical Ascension Paths

Now, we give an algorithm to find Q based on the canonical strings and their layout.

1. Start at P.
2. If on a point that is not in the shuffle diagram, mark this as Q and terminate.
3. If we are at x in the following configuration and $(**) - y$ is on a canonical string *or* $x - (*)$ is on a canonical string

$$\begin{array}{ccc} (**) & — & y \\ x & — & (*) \end{array}$$

   then go to y. Go back to step 2.
4. Else, go up 2 units. Go back to step 2.

Definition 6.2. From now on, we refer to moves where $x - (*)$ is on a canonical string as *type 1 crossings*, and moves where $(**) - y$ is on a canonical string but not $x - (*)$ as *type 2 crossings*.

Definition 6.3. We refer to any "execution" of either step 3 or step 4 as a *jump*.

Trivially, each jump increases the sum $x+y$ of the current coordinates (x,y) by 2.

Definition 6.4. For a given box P, let f_P be the number of jumps performed by the above algorithm. In particular, if P is not in the shuffle diagram, set $f_P = 0$.

Theorem 6.5 (Coordinate Step Formula)**.** *If the corresponding Q is (x_Q, y_Q), then $f_P = (x_Q + y_Q - x_P - y_P)/2$.*

Proof. The final coordinate sum is $x_Q + y_Q$ and the initial sum is $x_P + y_P$. Since each jump corresponds to an increase of 2 in this sum, the number of jumps is just $(x_Q + y_Q - x_P - y_P)/2$, as desired. □

In the next section, we will show that filling the tableau with all the minimum P values corresponding to the above bound (the f_P values) generates a valid tableau that has the same path structure as the canonical strings. For now, though, we will begin by proving that the f_P values give lower bounds on the corresponding tableau elements.

We will now prove some assertions about the Q-finding construction.

Lemma 6.6. *We can assign a parity χ to each string that is locally preserved along the string such that "adjacent" strings have opposite parities.*

Proof. Notice that, overall, each string has a left endpoint and a right endpoint. Trace the string in order from the left endpoint to the right endpoint. At any given position, consider unit squares in the following configuration, where A and B are tableau cells that contain numbers and are diagonally adjacent:

$$\begin{tikzcd} {} & B \arrow[l] \arrow[d] \\ A \arrow[u] \arrow[r] & {} \end{tikzcd}$$

Definition 6.7. Call all such (A, B) a *diagonal pair*.

A string can run either parallel or antiparallel to one of the arrows shown in the figure. Let the coordinates of A be (x_A, y_A). If the string runs parallel, then let $\chi = (x_A + 1) \mod 2$, and if it runs antiparallel, then let $\chi = x_A \mod 2$. We can now see why χ is conserved. Each arrow between adjacent points lies on exactly one such square (with the above geometry). Adjacent segments along the string correspond to diagonally adjacent corresponding squares. These squares have opposite values of both $x_A \mod 2$ and the parallel/antiparallel value, meaning that χ is conserved.

Definition 6.8. Strings are defined to be *adjacent* if, on the original Temperley-Lieb diagram, it is possible to draw a segment connecting them without going "outside" the diagram or crossing any other string.

Notice that if the left endpoint of the string is at $l_a = (x_{l_a}, y_{l_a})$, then χ will just be $x_{l_a} \mod 2$, independent of what the string is. Additionally, $(x_{l_a} - a) \mod 2$ is also trivially a constant. Since adjacent paths' left endpoints are separated by "blocks" of left/left paths, which have an even number of left endpoints in total (each path has 2), adjacent paths will have opposite parities for their a-values. This means that adjacent paths will have opposite χ values by the above. (Note that it is also straightforward to demonstrate adjacency between non-nested strings with endpoints on the opposite sides of the tableau via parity.) $\square$

We can now see that the path for crossings in the canonical construction is as follows:

1. First, depending on where the initial point is, there may or may not be a type-2 crossing (as the very first crossing).

2. After that, all crossings will be type-1.

We can see that undergoing a crossing flips the parity of the x-coordinate of the current point (on the path). Additionally, analogously adjacent strings will have flipped χ values. Since all horizontal steps are rightward in the canonical construction, after performing a crossing, the parity of the x-coordinate is now "aligned" to the parity of the next string, allowing the next crossing to be type-1. This parity flipping guarantees that each path will undergo at most one type-2 crossing in the canonical construction.

6.2 Generalized Ascension Paths

Next, we will show that a valid path between P and Q exists for **any** numbering in the grid, which will ultimately allow us to demonstrate elementwise minimality. To do this, we will construct a new path (for a given arrangement of numbers in the grid) that has one up/right step for every up/right step taken from P to Q in the canonical construction, which in turn corresponds to crossing strings. If the canonical construction does a type-2 crossing, then call it s_0. Let the canonical construction perform type-1 crossings on strings $s_1, s_2, s_3, \ldots, s_k$ in that order. Specifically, we have the following:

1. Start at P.
2. If at Q, terminate.
3. Else, if s_0 exists and are currently able to perform a type-2 crossing on s_0, and have not done a type-2 crossing on s_0 yet, then do the type-2 crossing. Go back to step 2.
4. Else, if s_0 either doesn't exist or you have done a type-2 crossing on it, define the index $m \in [0, k]$ such that you have performed type-1 crossings on s_i iff $1 \leq i \leq m$. If $m < k$ and currently able to perform a type-1 crossing on s_{m+1}, then do the type-1 crossing. Go back to step 2.
5. Go 2 steps up. Go back to step 2.

Note that in the canonical construction, if s_0 exists, then s_0 and s_1 are the same string because by virtue of the geometry of the canonical strings, after doing the type-2 crossing on a given string, one will vertically follow the string until it goes right, after which one will thus perform a type-1 crossing on the same string.

Lemma 6.9. *At any given position in the algorithm, the path will be able to do the crossing specified above after sufficiently many vertical steps, if such a crossing exists.*

Proof. We will prove this by inducting on the number of crossings α already performed. In particular, we claim that if the canonical path for P performs at least α crossings, then for any other tableau, the above algorithm will perform at least α crossings, with the sequence of the first α crossings identical to the first α in the canonical path. Additionally, we also prove that each crossing s_i ($i \geq 1$) that increases the x-coordinate from x_0 to $x_0 + 1$ corresponds to the segment $(x_0, y) \to (x_0 + 1, y)$ of minimal y on s_i (when s_i is traced left to right). Note that this can also be interpreted as leaving the region enclosed by s_i for the first time (see Remark 5.12 for some nuances.) Both conditions are trivially vacuously true for $\alpha = 0$.

For the inductive step, assume this is true for a given α, so we will induct $\alpha \to \alpha + 1$. Recall Lemma 5.17, which states that given a string $\mathbf{s}$ and x-value, the canonical string has a step $(x, y) \to (x + 1, y)$ with y minimal over all such configurations and steps (if such a step exists). Additionally, by χ-based parity, $y \mod 2$ is fixed over all such configurations.

Firstly, if $\alpha = 0$, then this means that the new configuration will have the same crossing available (because of χ) after some number of $(+0, +2)$ steps (namely, at least as many as in the canonical construction), given that the starting point is the same. Additionally, if s_0 exists (and thus $s_0 = s_1$), keep taking vertical steps after the initial crossing until such a step would leave the aforementioned enclosed region. This next step will thus be aligned for a type-1 crossing over s_1 by χ parity, and we know it exists because Lemma

5.17 guarantees the existence of this step at some y, and had it been earlier, the path would have (at least temporarily) left the enclosed region.

From now on, assume that one has already performed a crossing on some s_i for $i \geq 1$. Say that the most recent crossing was from $(x_0 - 1, y_0 - 1)$ to (x_0, y_0) in the new construction. According to Lemma 5.17 and parity, s_{i+1} must have some segment of the form $(x_0, y_0 + 2k) \to (x_0 + 1, y_0 + 2k)$ for some integer k; of all such segments, take the one with minimal k. Now, notice that the region enclosed by s_{i+1} is a strict superset of the region enclosed by s_i. Assume for the sake of contradiction that $k < 0$, meaning that $(x_0, y_0 + 2k + \varepsilon)$ would not be enclosed by s_{i+1} for any $\varepsilon > 0$. But by the inductive hypothesis on s_i, $(x_0, y_0 - \varepsilon)$ is in the enclosed region of s_i for any $\varepsilon > 0$. Combining these yields a contradiction if $k < 0$, so $k \geq 0$. Thus, the path goes up from (x_0, y_0) to $(x_0, y_0 + 2k)$ and then crosses s_{i+1} in a type-1 crossing to $(x_0 + 1, y_0 + 2k + 1)$, as desired. Since this has the lowest y out of all such transitions for s_{i+1}, the strong inductive hypothesis is satisfied, as desired. $\square$

Theorem 6.10. *In any valid numbering in the grid for the given shape μ/ν and Temperley–Lieb type τ, the number in cell P is at least the value of f_P in the corresponding canonical string construction.*

Proof. In the canonical construction, if the canonical ascension path starts at some point $P = (x_P, y_P)$ and ends at $Q = (x_Q, y_Q)$. By the coordinate step formula, $f_P = \frac{x_Q + y_Q - x_P - y_P}{2}$, and since each crossing increases the x-coordinate by 1, the canonical ascension path performs $x_Q - x_P$ crossings. Now, consider any other valid numbering that preserves μ/ν and τ. By Lemma 6.9, there exists a path starting at P consisting of $(+0, +2)$ jumps and the same number of crossings with this new string configuration as for the canonical path for the canonical construction. Consider the ending point of the last crossing in this new path, call it Q'. Note that $x_Q = x_{Q'}$ since the paths have the same change in x-coordinate, $2 \mid y_Q - y_{Q'}$ by parity, and $y_Q - y_{Q'} \geq 0$ since the endpoints are monotonic. (Specifically, if $y_Q < y_{Q'}$, then the starting point of the final crossing in the new path would be above the grid, contradiction.) Thus we can take $\frac{y_Q - y_{Q'}}{2}$ vertical $(+0, +2)$ jumps from Q to Q'. Just as for the canonical construction, each jump increases the sum of coordinates $x + y$ by 2, so the overall number of steps from P to Q is just $\frac{x_Q + y_Q - x_P - y_P}{2} = f_P$.

Now, consider the sequence of elements visited jumps in the new tableau from P to Q. Trivially, the second-to-last element is in the grid and thus has value ≥ 1. Now, letting s_i denote the value of the i-th element in the above sequence (with $s_1 = P$ and $s_{f_P+1} = Q$), then $s_{f_P} \geq 1$, and by Lemma 6.1, $s_i - s_{i+1} \geq 1$ for all $1 \leq i < f_P$. Summing such terms, we have $s_1 = s_{f_P} + \sum_{i=1}^{f_P - 1}(s_i - s_{i+1}) \geq f_P$ as desired. $\square$

7 Minimal Shuffle Tableau

To complete the proof for Theorem 1.4, we now construct the minimal shuffle tableau, demonstrate its minimality, and prove that it is a valid shuffle tableau.

For a given shape μ/ν and Temperley–Lieb type τ, perform the canonical string construction and calculate f_P for every cell in the tableau.

Definition 7.1. Let the tableau T formed by placing f_P in every cell P be called the *canonical tableau.*

Theorem 7.2. *The canonical tableau is minimal in every element and thus achieves the maximal partition in the dominance order of the tableau content over all Yamanouchi tableaux of shape μ/ν and type τ.*

Proof. By Theorem 6.10, the number in cell P can be no less than f_P, meaning that the tableau is minimal in every element. This, in turn, implies that it is Yamanouchi, as otherwise a crystal operator E_i could decrease some $i+1 \to i$, in turn violating minimality. We now demonstrate that T is thus trivially maximal in the overall dominance order $\leq_D$ of partitions.

Let $n_i(T_0)$ denote the number of occurrences of i in an arbitrary Yamanouchi tableau T_0. If $n_{i+1}(T_0) > n_i(T_0)$ for some i, then this means that in considering the crystal operator E_i, there would be at least one unmatched $i+1$ that E_i could decrease to i while keeping all other elements unchanged. This would contradict minimality under E_i, the definition of Yamanouchi tableaux, so $n_{i+1}(T_0) \leq n_i(T_0)$. This, in turn, implies that the partition of the content of any Yamanouchi tableau is just $(n_1(T_0), n_2(T_0), n_3(T_0), \ldots)$. Let λ be the content of the canonical tableau T and λ_0 be the content of any other Yamanouchi tableau T_0 with the same shape and Temperley–Lieb type.

Proving T is maximal over T_0 is by definition equivalent to proving:

$$\lambda \geq_D \lambda_0 \leftrightarrow \sum_{i=1}^{k} \lambda_i \geq \sum_{i=1}^{k} (\lambda_0)_i \leftrightarrow \sum_{i=1}^{k} n_i(T) \geq \sum_{i=1}^{k} n_i(T_0),$$

which translates to T having at least as many elements x with $x \leq k$ than T_0. This is trivially satisfied by virtue of elementwise minimality, as for a given x, $x \leq k$ in T_0 only if $x \leq k$ in T. Thus, T is maximal over the partition dominance order as expected. □

Remark 7.3. Note that having *minimal* elements implies being *maximal* in the partition ordering, and vice versa. Again, this is due to smaller elements v contributing to the earlier elements λ_v of the partition.

We will now show that the corresponding tableau is indeed a legal tableau with the correct Temperley–Lieb type. To do this, we first verify that all row and column monotonicity conditions are satisfied. Next, we verify that every individual segment of a canonical string is successfully generated, meaning that the strings overall construct a tableau with Temperley–Lieb type τ.

Remark 7.4. As before, we prove this under the assumption $\mu_\ell > \nu_1$, but we are confident that this logic can be extended to the slightly more general case where μ and ν are any valid strict partitions without too many major changes.

7.1 Row and Column Monotonicity

We will now prove the necessary row and column monotonicity, thus demonstrating that the tableau is valid, by virtue of proving that the desired condition is valid for locally adjacent pairs. To do this, we will perform casework on the local crossing geometry in conjunction with the coordinate step formula.

Lemma 7.5. *The nonstrict row monotonicity condition is satisfied.*

Proof. Consider two points $P_1 = (x, y)$ and $P_2 = (x+2, y)$; we claim $f_{P_1} \leq f_{P_2}$. Let the maximum possible y-value of any cell in the tableau be y_{max} (in other words, this is the topmost y-coordinate). We will prove the above assertion through strong induction on $y_{max} - y$. We can take $y_{max} - y < 0 \longrightarrow f_P = 0$ as the base case(s). Furthermore, notice that if we assign all f-values to the left of the grid as 0, then nonstrict row monotonicity continues to hold. Now, for the inductive step: assume all rows with y-coordinates strictly greater than the current one satisfy the nonstrict row monotonicity condition. Consider the first steps in the Q-finding algorithms for points P_1 and P_2. If both steps are the same (either both being a 2-cell jump upward or both being a crossing), then we are trivially done by induction on $y' = y+2$ or $y' = y+1$, respectively. Else, exactly one of the two paths (from P_1 and P_2) starts with a crossing. We have two cases based on whether the crossing is on the P_1 path or the P_2 path.

First, consider the case where the first crossing is on the P_2 path, so the P_2 path proceeds from $P_3 = (x+3, y+1)$ after the first step. We split into subcases:

- If neither the path starting at P_1 nor the path starting at P_3 has any crossings, then the path at P_1 ends at (x, h_1) and the path at P_3 ends at $(x+3, h_3)$. Trivially, by parity and the monotonicity of the top/right boundary of the tableau, $h_3 \geq h_1 - 1$. By the coordinate step formula, we thus have,

$$f_{P_2} - f_{P_1} = \frac{(x+3) + h_3 - x - y}{2} - \frac{(x+1) + h_1 - x - y}{2} = \frac{h_3 - h_1 + 2}{2} \geq 0,$$

as desired.

- Else, one of the two paths has a subsequent crossing. Over all such crossings, consider the one with smallest y-coordinate of the bottom-left cell in the crossing. For now, assume that this is in the path corresponding to P_1; we will address the P_2 case in the next entry. By assumption, the crossing in the P_1 path goes from $(x, y+2k)$ to $R_1 = (x+1, y+2k+1)$ for some integer $k \geq 1$. Additionally, since there is no lower crossing on the $P_2 \to P_3$ path, that path takes k vertical steps from $P_3 = (x+3, y+1)$ to $R_3 = (x+3, y+2k+1)$. Thus, by the coordinate step formula, $f_{P_2} - f_{P_1}$ equals

$$\begin{aligned} &\left(\frac{(x+3) + (y+2k+1) - (x+2) - y}{2} + f_{R_3}\right) \\ &- \left(\frac{(x+1) + (y+2k+1) - x - y}{2} + f_{R_1}\right) \\ &= f_{R_3} - f_{R_1} \geq 0 \end{aligned}$$

by the inductive hypothesis on $y' = y + 2k + 1$, as desired.

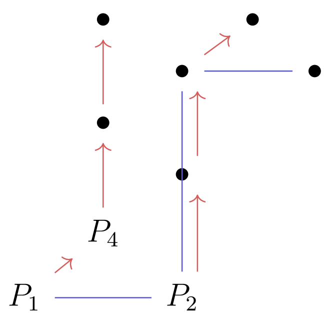
- Lastly, take the subcase where the first such crossing is on the P_2 path. By construction, the path takes $k \geq 0$ vertical steps from $(x+3, y+1)$ to $(x+3, y+2k+1)$ and then takes a crossing to $R_3' = (x+4, y+2k+2)$. If the path starting at P_1 does not reach a y-coordinate greater than or equal to $y+2k+2$, then trivially $f_{P_1} \leq \frac{x+(y+2k+1)-x-y}{2} < k+1$ and $f_{P_2} \geq \frac{(x+4)+(y+2k+2)-(x+2)-y}{2} \geq k+2 > f_{P_1}$, as desired. Else, it does, meaning that by construction, it reaches $R_1 = (x, y+2k+2)$ with $k+1$ vertical steps. By the coordinate step formula, $f_{P_2} - f_{P_1}$ equals

$$\left(\frac{(x+4) + (y+2k+2) - (x+2) - y}{2} + f_{R_3'}\right) - \left(\frac{x + (y+2k+2) - x - y}{2} + f_{R_1}\right)$$
$$= f_{R_3'} - f_{R_1} + 1 \geq 0$$

by the inductive hypothesis on $y' = y+2k+2$, as desired. Notice that this argument still works even if R_3 and/or R_1 do not correspond to tableau cells.

Next, consider the case where the first crossing is on the P_1 path, so its first step is to $P_4 = (x+1, y+1)$. We now take cases based on the local string geometry. Notice that strings cannot end on cells on or "inside" the tableau.

- If the initial crossing is type-1, then there is a string passing through $P_1 = (x, y)$ and also thus $(x+1, y)$. The string trivially cannot go through $(x+1, y+1)$ since this would violate diagonal pair geometry. Additionally, the string cannot go through $(x+1, y-1)$, since our construction for the string-placing algorithm does not create "corners" with this geometry. As a result, the string must go through $(x+2, y)$. Now, the string cannot go through $(x+2, y-1)$ because this would violate the aforementioned corner geometry. If it goes through $(x+3, y)$, then it would cause a crossing in the path starting at P_2, contradicting the assumption of this section. Thus, the string proceeds vertically from $(x+2, y)$. Here is a diagram of this scenario:



  Notice that from here, after several vertical steps, the string can either terminate, go left, or go right. By the nature of our string-placing algorithm, the string cannot have any segments corresponding to an upward step immediately followed by a leftward step. Thus, we have two cases, which we will examine separately.

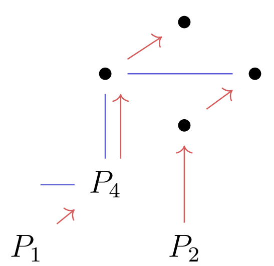
  - If the string proceeds vertically and terminates, then the path starting at P_2 will follow the string, with the string ending at $(x+2, y+2k+1)$ for some nonnegative integer k. (Recall that in a given row with leftmost cell (x_0, y_0), there is a string endpoint at $(x_0 - 1, y_0)$.) As such, the path starting at P_2 will terminate at $(x+2, y+2k+2) = (x+2, h_2)$, since $(x+2, y+2k)$ is in the grid while $(x+2, y+2k+2)$ is not (both because of the endpoint coordinates). We can now see that the path starting at P_1 must also continue vertically until it terminates. This is because it already had a crossing, so for it to have another crossing, the subsequent crossing must be type-1. This, however, would imply that there exists a string between two points of the form $(x+1, y+2a+1)$ and $(x+2, y+2a+1)$, where the crossing has bottom-left corner at $(x+1, y+2a+1)$. Since $a \leq k$ by virtue of $(x+1, y+2a+1)$ being a cell in the tableau, and strings cannot intersect, this would contradict the assumption of the string through P_1 not having a leftward step after moving vertically up from P_2. Thus, the path starting at P_1 terminates at some $(x+1, h_1)$. By parity and the monotonicity of the top/left boundary, knowing that path starting from $(x+2, y)$ terminates at $h_2 = y+2k+2$, we trivially have $h_1 \leq (y+2k+2)+1$ and that $h_1 - y$ is odd. Now, terminating at $(x+1, y+2k+3)$ would imply that $(x+1, y+2k+1)$ is in the grid by construction. But this is a string endpoint, so it must be outside the grid, implying that $h_1 \leq y+2k+1$. This gives

$$f_{P_2} - f_{P_1} = \frac{(x+2) + (y+2k+2) - (x+2) - y}{2} - \frac{(x+1) + h_1 - x - y}{2}$$
$$= \frac{2k+1-h_1}{2} \geq 0,$$

    as desired.

- If the string takes a horizontal step, then by diagonal pair parity, it must trivially be from $(x+2, y+2k)$ to $(x+3, y+2k)$ for some positive integer k (after k steps of moving 2 units vertically). Analogously, the path starting at P_2 will take k vertical steps to $(x+2, y+2k)$ followed by a crossing to $R_3 = (x+3, y+2k+1)$. Note that by the same logic as the previous section, the path starting at P_1 cannot take any crossings $(x+1, y+2a+1) \to (x+2, y+2a+2)$ for $a < k$ because this would involve a string going between $(x+1, y+2a+1)$ and $(x+2, y+2a+1)$, contradicting the above. This means that either the path from P_1 terminates at $(x+1, y+2a+1)$ for some $a \le k$, implying
$$\begin{aligned} f_{P_1} &= \frac{(x+1)+(y+2a+1)-x-y}{2} \\ &\le k+1 = \frac{(x+3)+(y+2k+1)-(x+2)-y}{2} \le f_{P_2}. \end{aligned}$$
Alternatively, the path must pass through $R_1 = (x+1, y+2k+1)$. We know $f_{R_3} \ge f_{R_1}$ by the inductive hypothesis on $y' = y+2k+1$, giving:
$$\begin{aligned} f_{P_2} - f_{P_1} &= \left(f_{R_3} + \frac{(x+3)+(y+2k+1)-(x+2)-y}{2}\right) \\ &- \left(f_{R_1} + \frac{(x+1)+(y+2k+1)-x-y}{2}\right) = f_{R_3} - f_{R_2} \ge 0, \end{aligned}$$
as desired, finishing this case.

- On the other hand, if the initial crossing is type-2, we get a slightly different string geometry, but the logic is functionally equivalent. In particular, we must have a string through $(x, y+1)$ and $(x+1, y+1)$. Furthermore, the path through P_1 initially crosses over to $P_4 = (x+1, y+1)$. Notice that trivially by construction, $(x+1, y+1)$ must be inside the grid, so the string through this point must also connect to a different point than $(x, y+1)$. Notice that, in our construction, strings never go left, meaning that the string can take several (possibly zero) steps up before either terminating or going right. We branch based on these cases:
  - If the string proceeds vertically and terminates, then say the highest position with an element in this column is at $(x+1, y+2k+1)$ for some integer $k \ge 0$. By construction, the string terminates at $(x+1, y+2k+2)$ and thus the path starting at P_1 terminates at $(x+1, y+2k+3)$. Thus, $f_{P_1} = \frac{(x+1)+(y+2k+3)-(x)-(y)}{2} = k+2$, so we want to show f_{P_2} exceeds this value. Notice that since $(x+1, y+2k+2)$ is a left-side string endpoint, all tableau columns with x-coordinate $x' > x+1$ will have a cell at some $y' \ge y+2k+2$ (if they have any cells at all). Thus, the path from $P_2 = (x+2, y)$ ends at (x_0, y_0) with $x_0 \ge x+2$ and $y_0 \ge y+2k+4$, as it ends 2 units above the topmost point in the column $x' = x_0$. This yields $f_{P_2} = \frac{x_0+y_0-(x+2)-y}{2} \ge \frac{(x+2)+(y+2k+4)-(x+2)-y}{2} = k+2 \ge f_{P_1}$ as desired.
  - Else, the string takes some steps up before going right, and we have the following configuration:



Note that the string can only take an even number of steps up, as going up an odd number of units before going right would violate diagonal pair geometry. Thus, the string goes up to $(x+1, y+2k+1)$ for some $k \geq 0$ before taking a step right to $(x+2, y+2k+1)$. Notice that since the column at $x' = x+2$ is nonempty, this cannot be an endpoint, so the string must proceed to a new point. Going either up or down would violate local pair geometry, so it takes another rightward step to $(x+3, y+2k+1)$. The string starting at P_1 crosses to $(x+1, y+1) = P_4$ before trivially being forced to take at least k vertical jumps, thus passing through $(x+1, y+2k+1)$. Now, we claim that the path starting at P_2 must first take a vertical steps, pass through $(x+2, y+2a)$, and then cross to $(x+3, y+2a+1)$ for some integer $0 \leq a \leq k$. If this were false, then the path would trivially take k vertical steps to $(x+2, y+2k)$, at which point the string in question would force a crossing to $(x+3, y+2k+1) \longrightarrow a = k$ is valid, contradiction. Thus, a exists and corresponds to the first crossing performed by this string. Thus, the path starting at P_1 passes through $R_1 = (x+1, y+2a+1)$ and the path starting at P_2 passes through $R_2 = (x+3, y+2a+1)$ for some $0 \leq a \leq k$. By the coordinate sum formula,

$$f_{P_1} - f_{R_1} = \frac{(x+1) + (y+2a+1) - (x) - (y)}{2} = a+1,$$

and

$$f_{P_2} - f_{R_2} = \frac{(x+3) + (y+2a+1) - (x+2) - (y)}{2} = a+1.$$

Jointly, these yield

$$f_{P_2} - f_{P_1} = (f_{R_2} + a + 1) - (f_{R_1} + a + 1) = f_{R_2} - f_{R_1} \geq 0$$

by the inductive hypothesis on $y' = y + 2a + 1 > y$, finishing.

We have successfully exhausted all cases, completing our proof. □

Lemma 7.6. *The strict column monotonicity condition is satisfied.*

Proof. Consider two points $P_1 = (x, y)$ and $P_2 = (x, y+2)$; we will show $f_{P_1} > f_{P_2}$. If the first step in the path finding algorithm of P_1 is a vertical step 2 units up, then trivially $f_{P_1} = f_{P_2} + 1$ since all subsequent steps are identical to the corresponding steps in P_2. Else, the first step in the path finding algorithm for P_1 is an crossing to $P_3 = (x+1, y+1)$.

Notice by parity that since the path starting at $P_1 \to P_3$ has already made a crossing, it has the correct x-coordinate parity for it to make a type-1 crossing should it ever intersect a subsequent string (since "adjacent" strings have opposite parities). Since the x-coordinate parity of P_2 is opposite that of P_3, its first crossing, if any, **must** be type-2. We will now take cases.

Firstly, if neither the path starting at P_2 nor the path starting at P_3 take any crossings, then they will have their Qs at (x, h_2) and $(x+1, h_3)$, respectively. By parity and the monotonicity of the top/left boundary of the grid, we must have $h_3 \geq h_2 - 1$. Thus, by the coordinate formula for the number of steps,

$$f_{P_1} - f_{P_2} = \frac{(x+1) + h_3 - x - y}{2} - \frac{x + h_2 - x - (y+2)}{2} = \frac{3 + h_3 - h_2}{2} \geq 1,$$

as desired.

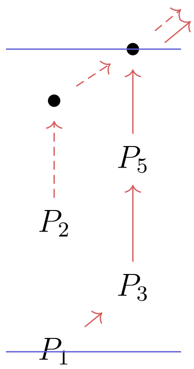
Else, at least one of these paths performs a crossing. Of all the crossings by either path, consider the crossing of smallest y coordinate (of the bottom-right, starting point in the crossing). If this is a type-2 crossing by the path starting at P_2, then the paths starting at P_1 and P_2 meet at the endpoint of this crossing and thus correspond to the same $Q = (x_Q, y_Q)$. By the coordinate step formula,

$$f_{P_1} - f_{P_2} = \frac{x_Q + y_Q - x - y}{2} - \frac{x_Q + y_Q - x - (y+2)}{2} = 1,$$

as desired. If not, then this is a type-1 crossing by the path starting at P_3 (in other words, a type-1 crossing by the path starting at P_1, not counting the very first crossing by this path). The endpoint of this crossing is thus $R_1 = (x+2, y+2+2k)$ for some nonnegative integer k. By assumption, the path starting at P_2 either passes through $R_2 = (x, y+2+2k)$ or performs several vertical steps before terminating at a strictly lower y-value. In the latter case, f_{P_1} corresponds to at least $k+2$ steps while f_{P_2} corresponds to strictly fewer than k steps, so $f_{P_1} > f_{P_2}$ as desired. In the former, by the coordinate step formula,

$$\begin{aligned} &f_{P_1} - f_{P_2} \\ =& \left(f_{R_1} + \frac{(x+2)+(y+2+2k)-x-y}{2}\right) - \left(f_{R_2} + \frac{x+(y+2+2k)-x-(y+2)}{2}\right) \\ =& \, 2 + f_{R_1} - f_{R_2} \geq 1, \end{aligned}$$

since $f_{R_1} \geq f_{R_2}$ by Lemma 5.1, finishing the proof.



□

7.2 String Generation

We will now show that this tableau correctly generates all string elements as described. Again, we will rely on geometric casework and the coordinate sum formula.

Lemma 7.7. *All horizontal string elements are correctly generated.*

Proof. We will show that every diagonal pair correctly generates a horizontal string element, if it is present. Consider the following setup:

$$(**) - P_2$$

$$P_1 - (*)$$

We will prove that P_1 and P_2 both yield the same Q value. Indeed, if $P_1 - (*)$ is a string, then the first move starting at P_1 will be a type-1 crossing to P_2. P_2 will thus not have any type-2 crossings by parity, with all subsequent steps for P_1 and P_2 being identical. On the other hand, if $(**) - P_2$ is a string but $P_1 - (*)$ is not, then the first move for P_1

will be a type-2 crossing to reach P_2, at which point all subsequent steps will be identical (by the same logic). As such, in both cases, they yield the same final Q, making the value at P_1 exactly one greater than the value at P_2 (by virtue of the one extra step). Consequently, the pair of segments will be horizontal, as desired. $\square$

Lemma 7.8. *All vertical string elements are correctly generated.*

Proof. We will split the following into two cases, based on whether a vertical string element has an L_a endpoint or a corner above it. It is clear to see that in the canonical construction, all such segments fall into one of the two categories.

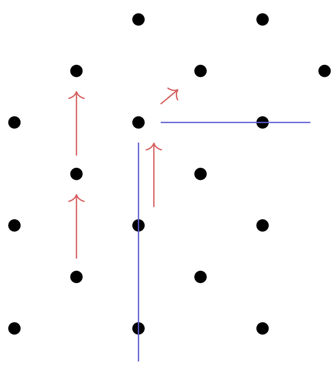
First, consider vertical string segments that have an L_a endpoint directly above them; let the diagonally adjacent pair of points that could potentially generate the string element be $P_1 = (x, y)$ and $P_2 = (x + 1, y + 1)$.

The first subcase is where L_a is above P_1, say $L_a = (x, y + 2a + 1)$ for some integer $a \geq 0$. Then the path starting at P_1 ends at $P_3 = (x, y + 2a + 2)$ (since it just follows the string) and the path starting at P_2 ends at some point $P_4 = (x_1, y_1)$. Trivially $x_1 \geq x + 1$, $y_1 \geq y + 2a + 2$ by construction and $(x_1 + y_1) - (x + y)$ is even (since each step jump is even), so $f_{P_2} = \frac{x_1+y_1-(x+1)-(y+1)}{2} \geq \lceil \frac{(y+2a+2)-(y+1)}{2} \rceil = a + 1 = f_{P_1}$, as desired.

The second subcase is where L_a is above P_2. Say $L_a = (x+1, y+2a+2)$ for some integer $a \geq 0$, so by construction, there is a string from $(x+1, y)$ directly to $(x+1, y+2a+2)$. This means the path starting at P_2 terminates at $(x+1, y+2a+3)$ and $f_{P_2} = a+1$. Additionally, the path starting at P_1 cannot make any crossings, as any such crossing would be induced by a string that intersects the L_a-containing string in an invalid way, contradiction. Thus the path at P_1 takes only vertical steps and terminates at $(x, y + 2a + 2 - 2b)$ for some $b \geq 0$, meaning $f_{P_1} = a + 1 - b \leq f_{P_2}$ as desired. We are now done will all subcases that involve an L_a endpoint above.

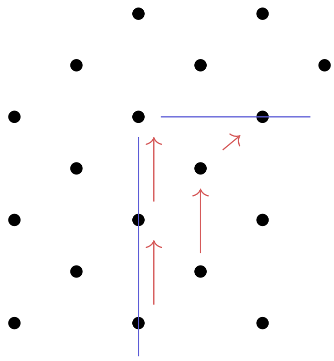
Next, consider the vertical string segments that have a corner but not an L_a endpoint above them. Again, we split into cases based on whether the generating pair is on the left side or the right side of the string (call this string the **primary string** for now).

- If it is on the left, then we have the following:



  Let the generating points of the string segment be $P_1 = (x, y)$ and $P_2 = (x+1, y+1)$. By construction, there must be a corner at $(x+1, y+2k+1)$ for some integer $k \geq 0$. The path starting at P_2 must thus take k vertical steps to $(x + 1, y + 2k + 1)$ and then makes a crossing to $P_4 = (x+2, y+2k+2)$. Meanwhile, the path starting at P_1 must take $k+1$ vertical steps to $P_3 = (x, y+2k+2)$, as otherwise any crossing before this would imply a horizontal string segment that intersects the string in question, contradiction. Thus $f_{P_2} - f_{P_1} = (k + 1 + f_{P_4}) - (k + 1 + f_{P_3}) = f_{P_4} - f_{P_3} \geq 0$ by Lemma 7.5.

- Now, if it is on the right, then we have the following:



We can apply similar logic. Let the generating points of the string segment be $P_1 = (x, y)$ and $P_2 = (x+1, y+1)$. Similar to what we had before, by construction, there must be a corner at $P_3 = (x, y+2k)$ for $k \geq 1$, and the path starting at P_1 takes k vertical steps to get there, by virtue of traversing along the string. Now, we claim that the path starting at P_2 takes $k-1$ vertical steps to $(x+1, y+2k-1)$ and then makes a crossing (either type-1 or type-2) to $P_4 = (x+2, y+2k)$. The existence of the final crossing conditional on being at $(x+1, y+2k-1)$ is trivially guaranteed, so we focus on proving that there are no "premature" crossings. Indeed, examining the configuration geometry implies that there would have to be a horizontal string segment from $(x+1, y+2a+1) - (x+2, y+2a+1)$ for some $a \in [0, k-1]$; consider the maximal such a out of all such string segments.

We will now argue that the existence of this string contradicts the "maximally pushed down" nature of the primary string. In particular, while the primary string currently passes through $(x, y+2k-2) - (\mathbf{x}, \mathbf{y} + \mathbf{2k}) - (x+2, y+2k)$, we claim that we can instead push the primary string down through $(x, y+2k-2) - (\mathbf{x} + \mathbf{2}, \mathbf{y} + \mathbf{2k} - \mathbf{2}) - (x+2, y+2k)$ while keeping the rest of the path the same. For the sake of contradiction, assume this is impossible. Then, there must exist another string passing through $(x+1, y+2k-3) - (x+1, y+2k-1) - (x+3, y+2k-1)$. Call this the secondary string. Since all strings are monotonic in some direction (either up/right or up/left) in our canonical construction, the second string cannot have gone left when going up before $y' = y+2k-1$, and it cannot have gone right either at any step with $y \leq y' \leq y+2k-3$ because it would have otherwise intersected the primary string. Thus, the secondary string passes through $(x+1, y) - (x+1, y+2k-1) - (x+3, y+2k-1)$, contradicting the existence of such an a.

Thus, if a exists, then it must be possible for the primary string to get pushed down to $(x, y+2k-2) - (x+2, y+2k-2) - (x+2, y+2k)$. But this is impossible, as it contradicts the established idea of strings in the canonical construction being maximally pushed down. In turn, this implies that premature crossings do not exist and that the path from P_2 indeed passes through P_4 as claimed. Lastly,

$$f_{P_2} - f_{P_1} = (f_{P_4} + k) - (f_{P_3} + k) = f_{P_4} - f_{P_3} \geq 0$$

by the coordinate sum formula and Lemma 7.5, finishing this case.

Again, we have exhausted our cases, finishing this proof. □

Theorem 7.9. *The canonical tableau is a valid elementwise minimal shuffle tableau with shape μ/ν and Temperley–Lieb type τ.*

Proof. By construction, each cell in the canonical tableau contains a positive integer. Jointly, Lemmas 7.5 and 7.6 are sufficient show that the canonical tableau is a legal shuffle tableau, as it obeys the necessary local row and column conditions.

Additionally, Lemmas 7.7 and 7.8 show that all strings described in the canonical construction are successfully generated in the canonical tableau. Since these strings were created in a way that connects the left and right endpoints with a given Temperley–Lieb type τ, this implies that the canonical tableau has the desired Temperley–Lieb type.

Finally, Theorem 7.2 gives elementwise minimality. $\square$

We are now ready to prove Theorem 1.4.

Proof. Consider a given arbitrary shape μ/ν satisfying the desired conditions and an arbitrary Temperley–Lieb type τ. If no shuffle tableaux with shape μ/ν and type τ exist, then we are trivially done. Else, by Lemma 5.15, the canonical construction successfully returns a valid tableau, which by Theorem 7.9, is minimal in every element. This proves Theorem **??**, which, as established in Section 3, is equivalent to Theorem 1.4, finishing. $\square$

8 Further Exploration

Throughout this section and all future sections, we drop the semi-standard Young tableau restriction and focus on general shuffle tableaux. In other words, we expand Theorem 3.4 to the following conjecture:

Conjecture 8.1. *Fix a shuffle tableau shape μ/ν for any such legal tableau shape and a Temperley–Lieb type τ. Let P be the set of contents λ of the tableau with this shape and type. We show there exists $\lambda_{max} \in P$ such that $\lambda \leq \lambda_{max}$ for all $\lambda \in P$.*

While we do not have a full proof for this case, we present some initial new conjectures as well as outline techniques that could be used to address the general case. Many of the following conjectures were motivated by large-scale code-based checking of small cases, and in general, we strongly recommend large-scale automated checking as an efficient method to find new relevant conjectures. The methods shown in [NNW25] can be used to efficiently generate all Yamanouchi tableaux of a given shape and type.

Note that for a given Yamanouchi tableau T, if $n_T(i)$ is the number of occurrences of i in the tableau, then $n_T(i) \geq n_T(i+1)$. This is because if $n_T(i+1) > n_T(i)$, then applying the crystal operator E_i to T would find and decrement an unmatched $i+1$, contradicting the fact that T is Yamanouchi. Thus, the sorted content of T is just $\nu_T = (n_T(1), n_T(2), n_T(3), \dots)$. Consequently, if one were to have $\lambda \leq \lambda_{max}$, then one would expect T_{max} to have larger proportions of "smaller" entries.

In fact, for all the examples checked, the code suggests a substantially stronger version of the above statement.

Conjecture 8.2. *For all tableau $T \in S$, each entry of T_{max} is less than or equal to the corresponding entry in T.*

For a given entry $x_{max} \in T_{max}$ and a corresponding entry $x \in T$, consider the infinite tuples y_{max} and y. Let the x_{max}-th entry of y_{max} be 1 and let all other entries be 0, and define y via x accordingly. If $x_{max} \leq x$, we know that $y \leq y_{max}$ (in terms of majorization). Note that summing all the y_{max} by adding the corresponding components yields λ_{max} and summing all the y yields λ, and so $\lambda \leq \lambda_{max}$. Thus, if the strengthened conjecture is true and a tableau T_{max} with all entries minimal exists, then the corresponding λ_{max} will be maximal in the dominance order of permutations, proving the original conjecture.

Thus, it suffices to prove the stronger conjecture. Note that the nature of the stronger conjecture immediately suggests algorithmically decreasing entries of the current tableau until one reaches T_{max}.

Let $A \lesssim B$ denote A and B being two SSYTs of the same shape and TL type with all entries of A being less than or equal to the corresponding entry of B. (A and B do not have to be Yamanouchi.) Note that trivially both equalities of shape/TL type and the entry less than or equal conditions are transitive, so $\lesssim$ is transitive, in other words,

$$A \lesssim B \lesssim C \implies A \lesssim C.$$

Thus, to prove conjecture 8.2, if one fixes a given shape and TL type, it suffices to show that of all SSYTs T of that shape and type, all but one (T_{max}) have the property that there exists a tableau T' with $T' \lesssim T$ and $T' \neq T$. Note that we don't have to explicitly maintain the Yamanouchi condition, as if a given tableau T is not Yamanouchi, then there exists a crystal operator E_i so that $E_i T \lesssim T$ (trivially, $E_i T \neq T$). Since the sum of the entries of the tableau is finite, one can only apply a finite number of crystal operators (each decreasing an entry by 1) before one is forced to terminate.

In other words, if T_0 is a Yamanouchi tableau and there exists a tableau $T' \neq T_0$ with $T' \lesssim T_0$, then by repeatedly applying crystal operators until none can be found, one can a Yamanouchi tableau $T_1 \lesssim T' \longrightarrow T_1 \lesssim T_0$.

In fact, testing large datasets through brute force strongly suggests the following even stronger conjecture.

Conjecture 8.3. *Fix a given shape μ/ν and TL type τ. Then, not only does there exist an aforementioned T_{max}, but for all $T \neq T_{max}$ with this shape and type, there exists an entry of T such that decreasing this entry by 1 produces a new tableau T' with the same shape and TL type.*

By the above, Conjecture 8.3 is strong enough to imply Conjectures 8.2 and thus 8.1. Conjecture 8.3 also readily suggests a more accessible method of proof:

1. Compile a set of operations that is strong enough to find an entry to decrease for all tableau $T \neq T_{max}$.
2. Use the definitions of the operations above to argue that it is impossible to have two tableaux T_1 and T_2 of the same shape and TL type such that neither of them can be decremented by applying the operations.
3. Conclude that T_{max} exists and is unique, proving all the above conjectures.

Step 2 above may be done through either "direct" comparison or by iterative constructing T_{max} and, in doing so, arguing its uniqueness.

In creating this new conjecture, we have greatly refined the original conjecture, making it far more accessible to direct proof. The next section will outline and describe partial progress for Step 1.

8.1 Decremental Algorithms

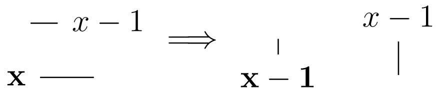
In this section, we give progress toward obtaining a complete series of algorithms that can perform the decrease-by-1 step in Conjecture 8.3. Trivially, the crystal operators E_i constitute such an algorithm. Next, let a given entry be *mutable* if decreasing it by 1 creates a new legal crystal tableau. In other words, if x is the value of the entry, then all of the following must be satisfied:

- x is at least 2.
- x is at least 2 greater than the entry immediately above it, if such an entry exists.
- x is strictly greater than the entry immediately to the left of it, if such an entry exists.

Now, plant a coordinate system on the tableau such that going right by 1 increases the x coordinate by 1 and going up by 1 increases the y coordinate by 1.

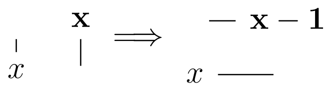
Definition 8.4. Let an entry x at (a, b) be *topright-blocked* if the number at $(a+1, b+1)$ exists and is equal to $x - 1$.

Equivalently, decrementing x to $x - 1$ changes the local Temperley–Lieb type of the graph:



Definition 8.5. Analogously, let an entry x at (a, b) be *bottomleft-blocked* if the entry at $(a - 1, b - 1)$ exists and equals x.

Again, decrementing x to $x - 1$ changes the local Temperley–Lieb type of the graph, this time on the other side:

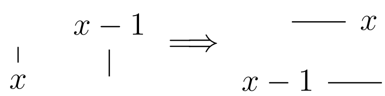


Note that any mutable entry that is neither topright-blocked nor bottomleft-blocked can trivially be decremented, so from now on, we will consider cases where every mutable entry is topright-blocked, bottomleft-blocked, or both.

For reasons motivated shortly, we will now consider cases where it is known whether the given entry x is on a cycle.

Lemma 8.6. *Consider a case where a given entry P, with initial value x, is either topright-blocked or bottomleft-blocked, but not both. If P is on a cycle either before or after the decrement (not necessarily both), then the decrement does not change the Temperley–Lieb type.*

Proof. Let the coordinates of P be (a, b). If P is topright-blocked but not bottomleft-blocked, then we only have the following change in the Temperley–Lieb diagram:



If P is known to be initially on a cycle, then there is a path ℓ in the rest of the tableau connecting (a, b) to $(a, b+1)$. Thus, whatever path the remaining two entries in the above figure were a part of, the path changes from directly connecting $(a+1, b)$ and $(a+1, b+1)$ to

$$(a + 1, b) - (a, b) - \ell - (a, b + 1) - (a + 1, b + 1),$$

thus not affecting the overall connectivity. Similarly, if x (now $x - 1$) is known to be on a cycle after the decrement, then there is a path ℓ among the rest of the tableau entries connecting (a, b) to $(a+1, b)$. Thus, while the final path involving $(a, b+1)$ and $(a+1, b+1)$ directly connects the two, the initial path is

$$(a, b+1) - (a, b) - \ell - (a+1, b) - (a+1, b+1),$$

meaning that the initial path has the same ultimate connectivity as the final path for the purposes of calculating the Temperley–Lieb type.

If P is bottomleft-blocked but not topright-blocked, then the proof is the mirror image analog of the above proof, thus covering both cases. $\square$

Lemma 8.7. *Consider a case where a given entry P, with initial value x. is both topleft-blocked and bottomright-blocked. If P is on a cycle in the Temperley–Lieb graph both before **and** after decrementing x, then the Temperley–Lieb type is unaffected.*

Proof. Let P have coordinates (a, b). We have the following local change in the Temperley–Lieb diagram, where P is in the center:

$$\begin{array}{ccc} & (A) - x-1 & \\ (B) & x \;—\; (D) & \\ \mid & \mid & \\ x & (C) & \end{array} \implies \begin{array}{ccc} & (A) & x-1 \\ & \mid & \mid \\ (B) - & x-1 & (D) \\ & x \;—\; (C) & \end{array}$$

Since P is initially on a cycle, (C) and (D) are connected by an external path ℓ_1. Since P is on a cycle at the end, (A) and (B) are connected by an external path ℓ_2. Thus, among the entries shown, the connectivity changes from a path

$$(a+1, b+1) - (A) - \ell_2 - (B) - (a-1, b-1)$$

and a cycle

$$(C) - (a, b) - (D) - \ell_1 - (C)$$

to a path with the same endpoints

$$(a+1, b+1) - (D) - \ell_1 - (C) - (a-1, b-1)$$

and a new cycle

$$(A) - (a, b) - (B) - \ell_2 - (A),$$

thus leaving the ultimate connectivity for the Temperley–Lieb type unchanged. $\square$

Remark 8.8. Notably, knowing that P is on a cycle either before or after the decrement, but not both, is *not* sufficient to conclude that the Temperley–Lieb type is unchanged. In particular, it's straightforward to find a counterexample for which this is not the case: just take the standalone diagrams

$$\begin{array}{ccc} & & —\;1 \\ & 2\;— & \\ \mid & \mid & \\ 2 & & \end{array} \implies \begin{array}{ccc} & & 1 \\ & \mid & \mid \\ & —\;1 & \\ 2\;— & & \end{array}$$

where the Temperley–Lieb type evidently changes.

We now motivate the discussion on cycles. Evidently, if we have information on whether a given entry is part of a cycle (call this *cyclicity*), then we can justify a decremental step. However, being on a cycle isn't immediately a local property. Fortunately, we can still obtain some cycle information locally. For this, we need a lemma.

Lemma 8.9 ([NP25])**.** *In a tableau where entries i and j are in relative positions where i is one unit left and one unit up of j and $i \geq j$, both i and j lie on cycles (not necessarily the same cycle).*

Using this lemma to test for cyclicity immediately before and after a given decrement can be used to eliminate the majority of cases. In particular, if one is decrementing x at (a, b) to $x - 1$, then one can look at the entries at $(a - 1, b + 1)$ and $(a + 1, b - 1)$ and compare their values to x via Lemma 8.9 to potentially deduce cyclicity. Based on the values of $(a - 1, b - 1)$ and $(a + 1, b + 1)$, one can use this in conjunction with Lemmas 8.6 or 8.7 to justify the decrement.

Remark 8.10. Furthermore, note that it may be helpful to shift the given entry up or down either before or after the decrement. In particular, say you are decrementing a value x at (a, b) down to $x - 1$, however, there exists some $x' > x$ where increasing x to x' doesn't change the TL diagram at all. Then, if there is a value y at $(a + 1, b - 1)$ with $x' \geq y$, then one can conclude that x lies on a cycle, even if initially $x < y$.

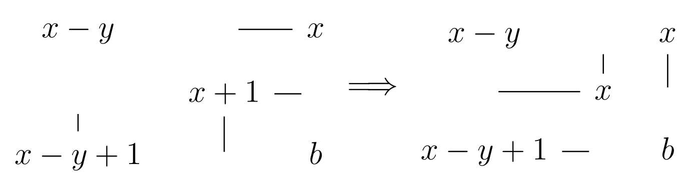
One can perform an analogous argument on the other side of (a, b) if there exists an entry at $(a - 1, b + 1)$.

Conjecture 8.11. *One can extend the definition of crystal operators so that when checking if a given element is unpaired in the crystal operator definition, after deleting all $(i, i+1)$-overlap pairs, one only checks whether there are are at lest as many is in the row immediately above the current row than $i + 1$s in the current row (as opposed to pairing with respect to all rows above the current row). If this condition fails, then the given tableau is not minimal and one can decrement an element.*

By recreating the proof in [NP25] while observing that the arguments only require local comparisons between adjacent rows, we can generalize the argument to include all generalized comparisons (in the correct direction). We also verified the validity of this generalization for small cases with code.

We will use this new crystal operator definition from now on, as it is slightly stronger by giving us more local information that we can directly use to prove cyclicity.

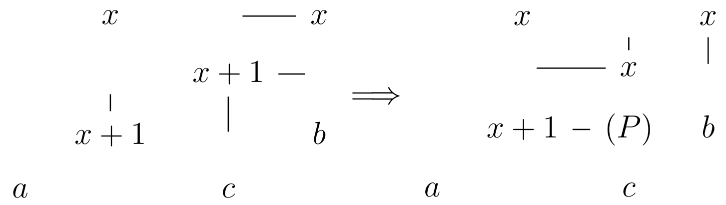
Conjecture 8.12. *In certain cases, configurations of the following form can be decremented as shown:*



for $x, y \in \mathbb{Z}_+$.

We sketch some arguments for this. According to a large data bank of brute-forced structures, configurations with $y > 0$ appear to be able to be broken down by first incrementing $x - 1 - y \longrightarrow x - 1$ and $x - y \longrightarrow x$, decrementing the central x to $x - 1$, arguing that this preserves the Temperley–Lieb type, and then reversing the decrement.

As such, we focus on the $y = 0$ case for now. Assume that all numbers considered in the below steps are nonzero integers (and are not $\pm\infty$), meaning that we can freely adjust them at will. Also assume that for the sake of contradiction, the current configuration cannot be decremented any further. We have the following diagram for such a transformation:

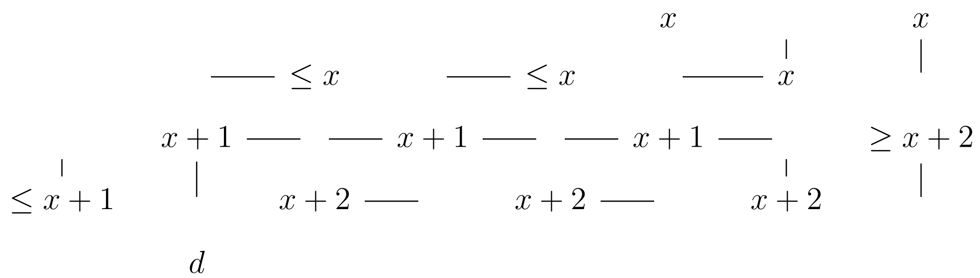


It would suffice to prove that in the final graph, $x+1 \quad - \quad P$ lies on a cycle, as by arguments similar to the ones discussed above, this would finish. (Other cases that prove the existence of other cycles also finish.) Notice that trivially $b \geq x+1$ by the row condition and $b = x+1$ would give that b and the $x+1$ immediately above-right of it are on the same cycle, which would be strong enough to finish. Thus, we can take $b \geq x+2$.

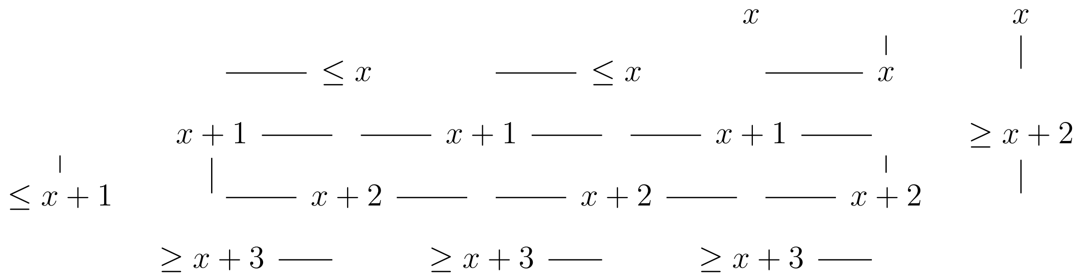
For such an operation to be legal, since the central $x+1$ is strictly above c in the initial graph, we must have $c \geq x+2$. We now have two cases. First, consider the case where $c > b$. Notice that in the original diagram, decreasing b to $b' = x+1$ preserves all row and column conditions. Also note that $c > b > x+1 = b'$, so the local geometry of the b/b' and c pair is unchanged. Furthermore, after performing this modification, we have $x+1 \geq b'$, so by Lemma 8.9 $x+1$ and b' both lie on cycles, so the Temperley–Lieb type is unchanged under performing $b \longrightarrow b'$. Since $b > b'$, this gives a strictly lower tableau, contradicting minimality.

Else, we have $c \leq b$, so the segments in the (b, c) pair are vertical. Now, notice that if $c \geq x+3$ in the initial diagram, then $(x+1, c)$ is not an overlapping pair since $c \neq x+2$. This means that by Conjecture 8.11, there must be an entry equal to $c-1$ in the row immediately above c's row. But $x+1 < c-1$ and $b > c-1$, so since rows are nondecreasing, no elements can satisfy this, contradiction. This forces $c = x+2$. Note that this argument fails if $c = \infty$ since 8.10's logic no longer applies.

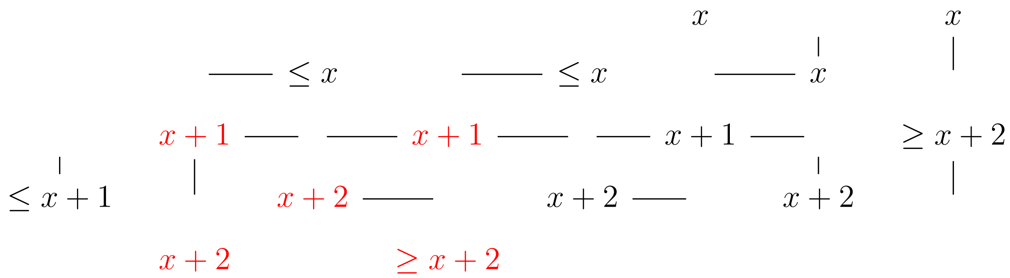
Now, we have two cases on the $(a, x+1)$ geometry. We know $a \leq c = x+2$, so either $a = x+2$ and the segments are horizontal or $a \leq x+1$ and the segments are vertical. We process the first case first. Consider the $x+2$s in the same row as a and c that are leftward or at c. Notice that the number two cells above $c = x+2$ is x, and by the row condition, all numbers in that row leftward of x must be $\leq x$. As a result, none of the aforementioned $x+2$s are $(x+1, x+2)$-overlapped, meaning that there are at least as many $x+1$s in the row immediately above c's row as there are $x+2$s leftward or at c in c's row. We have the following diagram:



We have two cases. If $d \geq x+3$, then all elements right of d must also be $\geq x+3$, thus completing the cycle and proving that the initial and final TL types are the same:



Else, we know that by the column condition, $d > x+1$, so we must have $d = x+2$. Now, we just have the same scenario as in the claim of this lemma in the 2x2 box with bottom-left corner at d. We may likely be able to set up an induction-based argument by decrementing the center element of this instead of the initial element:



This would complete the $a = x+2$ case, so now we can just assume that $a \leq x+1$ and the corresponding segments are vertical. It may likely be possible to extend this logic "down 2 rows" and repeat, further continuing with the induction-based approach.

Now, for the main conjecture of this section.

Conjecture 8.13. *All cases where an element can be decremented fall into one of the above categories:*

1. *An application of Conjecture 8.11,*
2. *An application of either Lemma 8.6 or Lemma 8.7 in conjunction with Remark 8.10,*
3. *An application of Conjecture 8.12.*

We have manually verified this conjecture with code for all skew-Schur tableaux that have size at most 8 (in other words, the cells of the tableau are a subset of an 8 by 8 grid of squares), and so we are confident that this holds in the general case.

9 Concluding Remarks

In this paper, we have proved a specific version of the Saturated Newton Polytope (SNP) assertion for semi-standard Young tableaux, namely, Theorem 1.4.

The generalized analogous version, Conjecture 8.1, remains open. Nevertheless, we are confident that some combination of the arguments presented in the first proof and the ideas offered in the previous section may be used to approach this general case. We are especially confident that dropping the extra condition $\mu_\ell > \nu_1$ will not significantly affect the nature of the proof. Specifically, one can likely generalize our string construction and subsequent logic to cover this case as well, and the structure of the ascension path logic should also remain roughly the same.

We encourage readers to explore these general cases with our new techniques.

References

[CCRMM23] Federico Castillo, Yairon Cid-Ruiz, Fatemeh Mohammadi, and Jonathan Montaño. Double Schubert polynomials do have saturated Newton polytopes. In *Forum of Mathematics, Sigma*, volume 11, page e100. Cambridge University Press, 2023.

[CS25] Jack Chen-An Chou and Linus Setiabrata. Newton polytopes of fireworks Grothendieck polynomials. *arXiv preprint arXiv:2508.09107*, 2025.

[Diz20] Avery J St Dizier. *Combinatorics of Schubert Polynomials.* Cornell University, 2020.

[EY17] Laura Escobar and Alexander Yong. Newton polytopes and symmetric Grothendieck polynomials. *Comptes Rendus. Mathématique*, 355(8):831–834, 2017.

[FMD18] Alex Fink, Karola Mészáros, and Avery St Dizier. Schubert polynomials as integer point transforms of generalized permutahedra. *Advances in Mathematics*, 332:465–475, 2018.

[Hai93] Mark Haiman. Hecke algebra characters and immanant conjectures. *Journal of the American Mathematical Society*, 6(3):569–595, 1993.

[HND25] Dang Tuan Hiep and Khai-Hoan Nguyen-Dang. Supersymmetric Schur polynomials have saturated Newton polytopes. *arXiv preprint arXiv:2507.22528*, 2025.

[LPP07] Thomas Lam, Alexander Postnikov, and Pavlo Pylyavskyy. Schur positivity and Schur log-concavity. *American journal of mathematics*, 129(6):1611–1622, 2007.

[MTY19] Cara Monical, Neriman Tokcan, and Alexander Yong. Newton polytopes in algebraic combinatorics. *Selecta Mathematica*, 25(5):66, 2019.

[NNW25] Chau Nguyen, Son Nguyen, and Dora Woodruff. Shuffle Tableaux, Littlewood–Richardson Coefficients, and Schur Log-Concavity. *arXiv preprint arXiv:2506.00349*, 2025.

[NP25] Son Nguyen and Pavlo Pylyavskyy. Temperley–Lieb crystals. *International Mathematics Research Notices*, 2025(7), 2025.

[RS05] Brendon Rhoades and Mark Skandera. Temperley-Lieb immanants. *Annals of Combinatorics*, 9(4):451–494, 2005.

[RS06] Brendon Rhoades and Mark Skandera. Kazhdan–Lusztig immanants and products of matrix minors. *Journal of Algebra*, 304(2):793–811, 2006.

[WZZ25] Bo Wang, Candice XT Zhang, and Zhong-Xue Zhang. Newton polytopes of dual k-Schur polynomials. *Advances in Applied Mathematics*, 162:102773, 2025.