THESIS

LOCAL ALGORITHMS FOR COPLACTIC SWITCHING OF YOUNG TABLEAUX

Submitted by

Kelsey M. Brown

Department of Mathematics

In partial fulfillment of the requirements

For the Degree of Master of Science

Colorado State University

Fort Collins, Colorado

Fall 2024

Master's Committee:

Advisor: Maria Gillespie

Emily King
Ewan Davies

ABSTRACT

LOCAL ALGORITHMS FOR COPLACTIC SWITCHING OF YOUNG TABLEAUX

We establish an algorithm for the local computation of the “coswitching operation" on a pair of (skew) semistandard Young tableaux, (X, T), such that that T extends X and T is Littlewood–Richardson. In this context, coswitching refers to conjugation of usual tableau switching with Jeu de Taquin rectification. If X is Littlewood–Richardson in addition to T, this algorithm aligns with the “evacuation shuffling" operation defined by Gillespie and Levinson [1] in relation to real Schubert curves. As a result, the algorithm we define has implications in real algebraic geometry.

ACKNOWLEDGEMENTS

I would like to thank my advisor, Maria Gillespie, for both introducing me to and fostering my love of algebraic combinatorics, as well as guiding me through the process of researching and writing. Additionally, I would like to thank Derek Moran, Jacob McCann, and Jake Levinson, who have all been an invaluable help in this research process. I would also like to extend my gratitude to Emily King and Ewan Davies for taking the time out of their busy schedule to serve as members of my committee. Finally, I would like to thank my mathematical twin, Kylie Schnoor, for keeping me as sane as reasonably expected and Brandon Adams for the constant, unending support that has made this project at all possible for me.

TABLE OF CONTENTS

Chapter 1

Introduction

In this thesis, we approach the problem of finding a local algorithm for coswitching tableaux (definitions in Chapter 2). Tableaux switching is an established combinatorial operation of shuffling two adjacent skew tableaux past each other. We are interested in the problem of conjugating this operation by Jeu de Taquin rectification, motivated by the problem of the intersections of Schubert varieties in $\overline{M}_{0,n}$ as defined in the subsequent subsection of Chapter 1.

We achieve the following result:

Theorem 1.0.1. *There is a local "hopping" algorithm for efficiently coswitching tableaux.*

Chapter 2 of this thesis provides the necessary background for this local algorithm, and Chapter 3 provides a detailed statement of its steps, as well as alternate versions of the algorithm statement. In Chapter 4, we discuss the extension of the algorithm to tableaux in Type B, as well as potential future applications.

1.1 Geometric motivation

For the entirety of the following section, see [2], [3], [4] for more details.

Since our algorithms are motivated by Schubert intersection problems, we include a brief discussion of a few such topics.

We begin by defining the **Grassmanian**, $\mathrm{Gr}(n,k)$, which is the set of all k-dimensional subspaces of an n-dimensional vector space, in our case, $\mathbb{C}^n$. Each element of $\mathrm{Gr}(n,k)$ is associated with a partition λ that fits inside of an ambient $k \times (n-k)$ rectangle. The set of all points with a given associated partition λ is called a **Schubert cell**.

We must also define a **flag**. In $\mathbb{C}^n$, a full flag $\mathcal{F}_\bullet$ is a chain of subspaces

$$\{0\} = F_0 \subset F_1 \subset \ldots \subset F_n = \mathbb{C}^n$$

where $\dim(F_i) = i$.

We wish to consider **Schubert varieties** defined with respect to a full flag: [4]

$$\Omega_\lambda(\mathcal{F}_\bullet) = \{V \in \mathrm{Gr}(n,k) \mid \dim(V \cap F_{n-k+i-\lambda_i}) \geq i\}.$$

These Schubert varities are the closures of the Schubert cells mentioned above.

Consider the image of the map

$$t \mapsto (1 : t : t^2 : \cdots : t^{n-1})$$

in $\mathbb{P}^{n-1}$ such that $t \in \mathbb{R}$. We call this the **rational normal curve**.

Let us consider **osculating flags**, $\mathcal{F}_t$, the family of flags such that each F_i is maximally tangent to our rational normal curve in $\mathbb{P}^1$. Then we can utilize the following powerful theorem to give us choices for our flags that will always work out nicely.

Theorem 1.1.1 (Shapiro-Shapiro Conjecture/MTV theorem [5] [6])**.** *If $t_1, \ldots, t_r \in \mathbb{R}$, $\lambda^1, \ldots, \lambda^r$ are partitions such that $\sum_i |\lambda^i| = k(n-k)$, and $\mathcal{F}_{t_1}, \ldots, \mathcal{F}_{t_r}$ are osculating flags, then*

$$\Omega_{\lambda^1}(\mathcal{F}_{t_1}) \cap \ldots \cap \Omega_{\lambda^r}(\mathcal{F}_{t_r})$$

has distinct real solutions in Gr(n,k).

By the Littlewood–Richardson rule, the number of solutions is the number of chains of Littlewood-Richardon tableaux with contents λ^i that fill our $k \times (n-k)$ rectangle. [4]

Our primary goal is to study what happens to the intersection of these Schubert varieties as we vary the points on our rational normal curve. A generic intersection is given to us by the Littlewood–Richardson rule where the intersection

$$\bigcap \Omega_{\lambda^{(i)}}(\mathcal{F}_\bullet^{(i)})$$

consists of exactly the number of points in $\mathrm{Gr}(n,k)$ that correspond to the L–R coefficient $c^B_{\lambda(1),\ldots,\lambda(r)}$ where B is our ambient $k \times (n-k)$ rectangle. However, making these generic choices for flags can be rather difficult, and we might not always get distinct solutions. The Shapiro-Shapiro Conjecture means that our solution sets are always distinct points, so this gives us a natural set of solutions to study.

Much of the geometric problem reduces to the case where we have only four points, t_1, t_2, t_3, and t_4. In particular, we want to consider flags $\mathcal{F}, \mathcal{G}, \mathcal{H}$, and $\mathcal{I}$ that correspond to these four points on our rational normal curve.

To study this problem, we must recall a moduli space relevant to us for distinct $t_1, \ldots, t_n$:

$$M_{0,n}(\mathbb{R}) = \{(t_1, \ldots, t_n) \text{ distinct on } \mathbb{P}^1_{\mathbb{R}}\}.$$

Note that this moduli space is not closed since our t_i are all distinct. However, we can close up $M_{0,n}(\mathbb{R})$ using the Deligne-Mumford compactification, giving us $\overline{M_{0,n}}(\mathbb{R})$, which is the set of all genus zero stable curves with n marked points. [2] [7]

Given that an isomorphism of $\mathbb{P}^1$ is determined by where it sends three points, we can assume for a generic point of $M_{0,4}$ that the first three marked points are at $0, 1, \infty$ and the fourth varies with coordinate t. In $\overline{M_{0,4}}$, we have three additional curves, each corresponding to when t collides with one of the points at $0, 1, \infty$. Due to a construction of Speyer [2], there is a space $S = S(\alpha, \lambda, \beta, \gamma)$ with a map $\pi : S \to \overline{M_{0,4}}$ such that

$$\pi^{-1}(C_t) = \Omega_\alpha(\mathcal{F}_0) \cap \Omega_\lambda(\mathcal{F}_t) \cap \Omega_\beta(\mathcal{F}_1) \cap \Omega_\gamma(\mathcal{F}_\infty),$$

each of which are, by Shapiro-Shapiro, finite and equinumerous for $t \neq 0, 1, \infty$. Speyer proved this is a smooth covering, and the same holds for fibers above $0, 1, \infty$. In work by Gillespie and Levinson [1], the case of λ as one box was considered, and a combinatorial formula was given for the monodromy of this covering, meaning the operation on fibers of following the sheets of the

covering around $\overline{M_{0,4}}(\mathbb{R})$ [1]. In this thesis, we consider the case of a general partition λ and give a formula for the monodromy, extending the work of Gillespie and Levinson.

Chapter 2

Background

2.1 Partitions and tableaux

A **partition** $\lambda = (\lambda_1 \geq ... \geq \lambda_k)$ is a weakly decreasing sequence of nonnegative integers. The **Young diagram** of a partition λ is a left-justified collection of boxes where λ_i is the number of boxes in the i-th row. We denote the **size** of a partition as $|\lambda| = \sum \lambda_i$ and define the **length** of a partition $\ell(\lambda) = k$. We assume throughout that λ fits inside a $k \times (n-k)$ rectangle, meaning $\ell(\lambda) \leq k$ and $\lambda_i \leq n-k$ for all i.

If ρ and λ are partitions such that $\rho_i \leq \lambda_i$ for all i, we say λ **contains** ρ and write $\rho \subseteq \lambda$. If $\rho \subseteq \lambda$, we define a **skew shape** λ/ρ as the result of removing the boxes of ρ from the diagram of λ. In the case that $\rho = \varnothing$, we refer to λ as a **straight shape**.

Any way of placing positive integers in the boxes of a diagram is called a **filling**. A (skew) **semistandard Young tableau** is a filling of a skew shape such that entries are weakly increasing along the rows and strongly increasing down the columns. A semistandard Young tableau is **standard** if each entry $\{1, ..., k\}$ is used exactly once. All Young tableaux are assumed to be semistandard unless stated otherwise.

The **shape** of a tableau T refers to the skew shape associated with T and is denoted $sh(T)$. If S and T are tableaux with shapes λ/ρ and μ/λ respectively, then we say T **extends** S. A **chain of Young tableaux** is a sequence $(T_1, \ldots, T_k)$ such that each T_i extends T_{i-1}.

A **word** is a string $w = w_1 w_2 \ldots$ in the symbols $\{1, 2, \ldots\}$. The **(row) reading word** of a tableau T is the word $w(T)$ obtained by reading off the entries of T from left to right starting from the bottom row. The $i, i+1$**-subword** of a word w is the result of deleting any entry in w not equal to i or $i+1$. The **weight** of T is $wt(T) = (n_1, \ldots, n_k)$ where n_i is the number of i entries in T.

Let X and T be a pair of tableaux such that T extends X, and let n be the largest entry of X. We take their union, $X \sqcup T$, to be the result of adding n to each entry of T and treating the pair as a single tableau.

The **standardization** of a word $w = w_1 \dots w_n$ is the word $std(w)$ formed by replacing the letters by $1, \dots, n$, from least to greatest, with ties broken in reading order. The standardization of a tableau is defined by standardizing its reading word.

2.2 Jeu de Taquin and equivalence relations

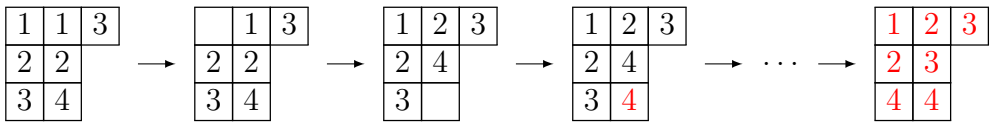
Given a skew tableau λ/μ, an **inner corner** is a box contained in the (deleted) diagram of μ such that the boxes below and to the right of it are not in μ. An **outer corner** is a box of λ such that the boxes below and to the right of it are not in λ.

If T is a skew tableau of shape λ/μ, an **inner Jeu de Taquin (JDT) slide** of T into this empty box is the result of the following procedure. First, of the two boxes to the right and below of the empty box, slide the one with the smaller entry into the empty space. Should the entries in those boxes be tied, slide the box below. This process repeats until the empty box is now an outer corner of λ. An **outer JDT slide** refers to the inverse of an inner slide.

A tableau T is **rectified** if it is of straight shape. The **rectification** of T is the tableau $\mathrm{rect}(T)$ formed by applying inner JDT slides on T in any order until we obtain a straight shape tableau. By what is frequently called the "fundamental theorem of JDT," $\mathrm{rect}(T)$ does not depend on the order in which we apply the slides.

Two skew semistandard Young tableaux S and T are **slide equivalent** ($S =_{\text{slide}} T$) if there exists a sequence of JDT slides that takes one tableau to the other.

Definition 2.2.1. Let T be a rectified semistandard tableau with largest entry n. The Schützenberger involution, or **evacuation** of T, $e(T)$, can be constructed as follows. Delete the unique entry in the upper left corner of T; let i be its value. Rectify the remaining entries of T, then fill the emptied box with $n + 1 - i$ as part of a new tableau S. Then repeat the above steps until each entry of T has been replaced, as shown in the example below.



2.3 Littlewood–Richardson tableaux

A word $w = w_1 \dots w_n$ is **ballot** if when read from the beginning to any letter, the sequence $w_1 \dots w_j$ contains at least as many i's as $i+1$'s.

A word $w = w_1 \dots w_n$, is **reverse-ballot** if when read backwards from the end to any letter, the sequence $w_n \dots w_{n-j}$ contains at least as many i's as $i+1$'s. A tableau whose reading word is reverse-ballot is called **Littlewood–Richardson** (L–R); however, we may also say the tableau itself is reverse-ballot.

For a given straight shape λ, the **highest weight** tableau with shape λ is the tableau with the i-th row filled entirely with i entries.

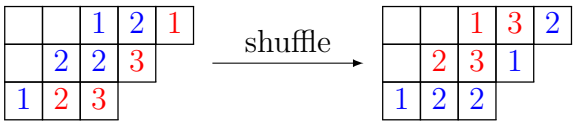
For more on the general background of tableau combinatorics, we direct readers to Fulton [8].

2.4 Switching operations on tableaux

For this section, we assume X and T are semistandard Young tableaux. We will now define several operations which switch the relative positions X and T.

Definition 2.4.1 (Tableau switching)**. Tableau switching**, or shuffling, a pair of tableau is achieved by performing successive inward JDT slides on T in the order determined by the standardization order of X from largest to smallest.

We demonstrate switching process this in the example below.



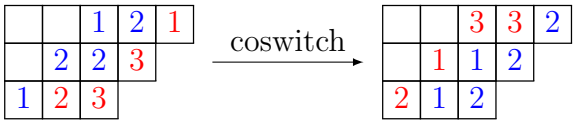
Definition 2.4.2. A function on Young tableaux is **coplactic** if it commutes with any sequence of JDT slides. We note that shuffling is *not* coplactic.

We will introduce several coplactic operations related to shuffling in this section. We are able to construct these operations by conjugating a function by rectification as described above. To extend rectification to pairs of tableau, we define $\mathrm{rect}(X, T)$ to be the two parts of $\mathrm{rect}(X \sqcup T)$.

Definition 2.4.3. Tableau coswitching (coswitch) is the coplactic extension of tableaux shuffling from the case where the inner tableau is of straight shape, giving us

$$\mathrm{coswitch}(X, T) = \mathrm{rect}^{-1} \circ \mathrm{shuffle} \circ \mathrm{rect}(X, T).$$

We give an example of this coswitch operation on the same tableau from our previous example.

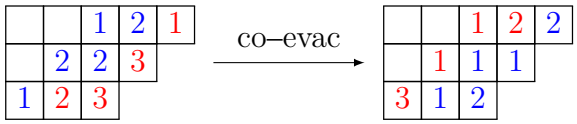


Definition 2.4.4. We define the coplactic extension of evacuation (co–evac) of a tableau T to be the result of conjugating evacuation by rectification as follows:

- **Rectify**. Compute $\mathrm{shuffle}(S, T) = (T', S)$
- **Evacuate**. Compute $e(T') = T''$
- **Unrectify**. Compute $\mathrm{shuffle}(T'', S') = (S'', T''')$.

This gives us $\text{co–evac}(T) = (T''')$ as defined above.

We give an example of this co–evac operation below.



As a note, every operation defined so far is an involution. For more details on tableaux switching operations, we direct readers to work by Benkart, Sottile, and Stroomer [9].

2.5 Coplactic operators

Crystals in type A are a set of weight raising and weight lowering coplactic operators, E_i and F_i respectively, that act on both semistandard Young tableaux and their reading words. Let w be a word. then the crystals $E_i(w)$ and $F_i(w)$ can be computed as follows. Consider the $i, i+1$-subword of w with each i and $i+1$ replaced with a "(" and ")" respectively. Starting from the end of the subword, pair off each (with the closest unpaired) to the right if it exists. If there is an unmatched (, then $E_i(w)$ changes the first unmatched (to a), otherwise $E_i(w) = \emptyset$. $F_i(w)$ will instead change the last unmatched) to a (if it exists, otherwise $F_i(w) = \emptyset$.

Example 2.5.1. Consider the word $w = 2221132122131$. Replacing each entry in the $1, 2$-subword with parentheses gives

$$((())()(()).$$

The single red (is the only unmatched symbol, so $F_1(w)$ is undefined while $E_1(w) = 1221132122131$.

Proposition 2.5.2. Let T be a Littlewood–Richardson tableau. Then F_i is defined if and only if the $i, i+1$ subword is not tied for reverse-ballot.

Proof. Crystal operators are coplactic, meaning $F_i(T)$ is defined if and only if $F_i(\mathrm{rect}(T))$ is defined. Observe that the $i, i+1$-reading word of $\mathrm{rect}(T)$ is of the form $(i+1)\ldots(i+1)i\ldots i$. We can now see that there is an unmatched i if if and only if there are more i's than $i+1$'s. □

These operators will be useful to us as we construct an algorithm that computes the coplactic extensions of the operations described above *locally*, meaning we act directly on the skew tableaux without conjugating by rectification. For information on crystal operators, we direct readers to Bump and Schilling [10].

Chapter 3

Results and proofs

3.1 Main results

We now define a *local* algorithm which computes $\mathrm{coswitch}$, meaning we act directly on the pair of skew tableaux without rectifying or unrectifying. These algorithms and their proofs can also be found in collaborative work in [11]. Our first algorithm uses a sequence of switches, which more closely resembles JDT than the later crystal algorithm.

Definition 3.1.1. (Type A hopping algorithm) Let X be a standard Young tableau such that $|X| = n$, and let T be a Littlewood–Richardson semi-standard Young tableau. Let $e(X)$ be the tableau resulting from the evacuation of X with entries labeled $y_n, \ldots, y_1$ in order of evacuation. Let d be the largest entry of T.

Then the hopping algorithm in Type A is defined as follows:

Set $i = n$ and $j = 1$.

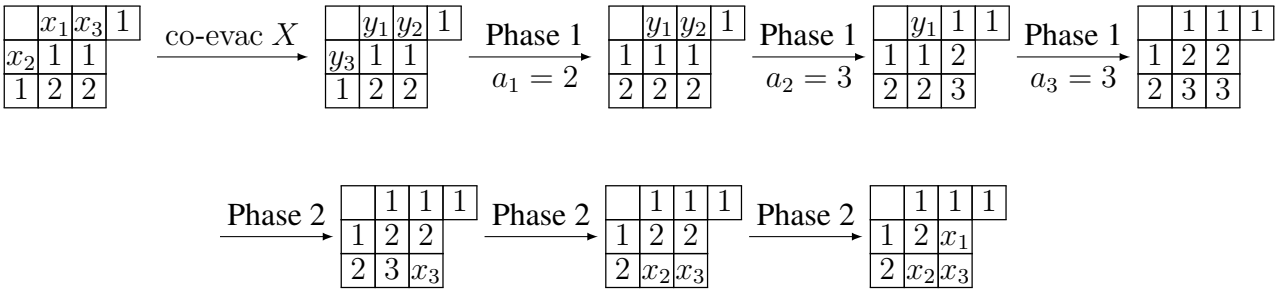
- **Phase 1a:** If y_i precedes all of the js in reading order, replace y_i by j, record $a_{n-i+1} = j$, and proceed to Phase 1b. If y_i does not precede all of the js in reading order, switch y_i with the j immediately preceding it in reading order. Increment j and repeat Phase 1a.

- **Phase 1b:** If $i = 1$, proceed to Phase 2. If $i \neq 1$, decrement i, reset $j = 1$, and repeat Phase 1a.

- **Phase 2:** Check the j value to which a_{n-i+1} is equal. Replace the first j in reading order with x_{n-i+1}. If its suffix is tied for $(j, j+1)$, increment j. If not, switch x_{n-i+1} with the nearest j after x_{n-i+1} in reading order whose suffix is tied for $(j, j+1)$, and increment j. In either case, if the new value for $j = d + 1$, increment i and repeat Phase 2 with that i value. If not, repeat Phase 2 with the original i value.

We demonstrate this algorithm in the following example.

Example 3.1.2.



Remark 3.1.3. Because they only depend on the i subword in T, we may stagger the steps of Phase 1. That is, we may perform a single step of Phase 1 on x_n through some index i_n, then perform another step on x_{n-1} through the index $i_{n-1} \leq i_n$, and so on for all entries x_j that are currently in Phase 1.

The fact that the hopping algorithm preserves the slide equivalence classes of both tableaux is not obvious. Phase 2 also implicitly relies on the remaining tableau excluding the active entry being ballot for the hops to be well defined. So, we will now show that if T starts as a Littlewood–Richardson tableau, its reading word remains reverse ballot after each step.

A **step** of the algorithm refers to a single switch performed during the algorithm.

Lemma 3.1.4. *Let X be a semistandard Young tableau and T be a Littlewood–Richardson tableau extending X. Let T', including all entries of X, be a tableau that appears after some step of the computation of* $\mathrm{hop}(X, T)$. *Then:*

1. *T' is Littlewood–Richardson when omitting every entry of X.*
2. *T' is semistandard when omitting every entry of X.*
3. *After x_j performs the i-th step of Phase 1 or Phase 2, it is an inner corner of $T'_{>i}$, the sub-tableau containing only those entries greater than i.*
4. *After running Phase 1 of* $\mathrm{hop}(X, T)$*, the resulting tableau S is Littlewood–Richardson.*

Proof. For Phase 1 the first three statements essentially follow from an inductive argument using Theorem 3.9 from [1] as the base case. Phase 2 similarly follows from Theorem 3.9, but requires the fourth statement to satisfy the initial assumptions, which we now prove.

It suffices to show that when some x_j transitions out of Phase 1 and is absorbed into T as an i entry, the resulting tableau omitting $x_1, ..., x_{j-1}$ is Littlewood–Richardson. Because ballotness is preserved by JDT, we may also simplify to the rectified setting.

Let $n \geq j$. Suppose X is a single box x_n. Notice that in the rectified setting, the marked entry x_n ends on row i when it transitions out of Phase 1 at index i [1]. Using Lemma 3.1.4, we know that T omitting x_j is Littlewood–Richardson and semistandard. So, absorbing x_j as an i entry on row i will preserve ballotness for T.

We repeat this argument for $x_{n-1}, \dots x_j$, giving us a ballot tableau omitting $x_1, \dots x_{j-1}$ and completing the proof. □

The third statement of Lemma 3.1.4 also shows that the output of $\mathrm{hop}(X, T)$ is, indeed, of the form (T', X').

Definition 3.1.5. The **row data** of $\mathrm{coswitch}(X, T)$ is $(r(x_1), \dots, r(x_n))$ where $r(x_i)$ is the row in which x_i falls after $\mathrm{coswitch}(X, T)$. The **transition data** of $\mathrm{hop}(X, T)$ is $(h(y_n), \dots, h(y_1))$ where $h(y_i)$ is the number with which y_i is replaced during Phase 1a of $\mathrm{hop}(X, T)$.

Lemma 3.1.6. *For a straight shape $X \sqcup T$, the row data, $r(X, T)$, and the transition data, $h(X, T)$, are permuations of each other.*

Proof. Let the output of $\mathrm{coswitch}(X, T)$ be (T', X'), and let the output of $\mathrm{hop}(X, T)$ be (T'', X'').

By Lemma 3.1.4, $\mathrm{coswitch}(X, T)$ and $\mathrm{hop}(X, T)$ preserve both the ballotness and weight of T. Therefore, since T' and T'' are both Littlewood–Richardson of highest weight, $T' = T''$.

Additionally, since $\mathrm{coswitch}(X, T)$ and $\mathrm{hop}(X, T)$ preserve the overall tableau shape, $\mathrm{sh}(T', X') = \mathrm{sh}(T'', X'')$. Therefore, since $T' = T''$, we get that $\mathrm{sh}(X') = \mathrm{sh}(X'')$.

Note that since we are working in the straight shape case, $h(x_i)$ is simply the row on which x_i ends up since Phase 2 of $\mathrm{hop}(X, T)$ will just slide x_i to the end of the $h(x_i)$'th row.

Finally, note that $\mathrm{wt}(r(X,T))$ and $\mathrm{wt}(h(X,T))$ record the number of entries of X' and X'' respectively on each row. Then since $\mathrm{sh}(X') = \mathrm{sh}(X'')$, we get that $\mathrm{wt}(r(X,T)) = \mathrm{wt}(h(X,T))$. Thus, $r(X,T)$ and $h(X,T)$ must be permutations of each other. □

Theorem 3.1.7. *The output of* $\mathrm{coswitch}(X,T)$ *and the output of* $\mathrm{hop}(X,T)$ *agree.*

Proof. We will work in the case where X is straight shape. We proceed via induction on $|X|$.

Our base case is $|X| = 1$, so $X = x_1$. Then we have row data $(r(x_1))$ and transition data $(h(y_1))$, so by Lemma 3.1.6, the values $\mathrm{coswitch}(x_1,T)$ and $\mathrm{hop}(x_1,T)$ must agree.

Assume the two agree for $|X| = n-1$ and consider the case where $|X| = n$.

We perform one step of $\mathrm{coswitch}(X,T)$ which will move x_n past T, leaving us with a chain of tableaux (X', T', x_n) where $|X'| = n-1$.

If we instead were to evacuate X, leaving us with the chain (Y,T), y_1 must be the entry in the innermost corner. We remove y_1 and rectify into the empty space, which leaves us with (Y', T''), where $|Y'| = n-1$. Note that T' is the same as T'' since the content of T' and T'' have not changed and the shape of X' matches the shape of Y'.

Since evacuation is an involution and $e(X) = Y$, we know that $e(Y) = X$. Thus, this step of removing y_1 from Y and rectifying to get Y', which matches with the first step of computing $e(Y)$, must evacuate the same outer corner as the outer corner as the location of x_n in X. Evacuating Y' from this point is just the continuation of evacuating Y, so $e(Y')$ must match all of X except for x_n, but this is simply X'. Thus, $e(Y') = X'$, and since evacuation is an involution, $e(X') = Y'$.

By our inductive hypothesis, we know that $\mathrm{coswitch}(X', T')$ agrees with $\mathrm{hop}(X', T'')$, and we have row data $(r(x_1), \ldots, r(x_{n-1}))$ and transition data $(h(y_n), \ldots, h(y_2))$ which, by Lemma 3.1.6, are permutations of each other.

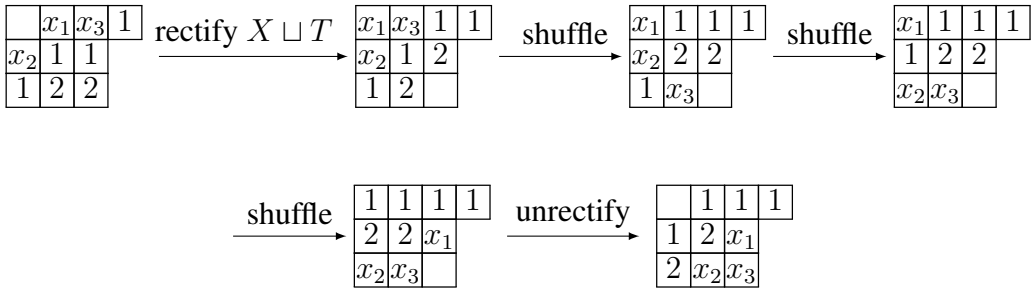
But we began by coswitching x_n past T, which really gives us row data $(r(x_1), \ldots, r(x_n))$.

Because our row data and transition data must be permutations of each other, we conclude that $r(x_n) = h(y_1)$. Since the final step of the hopping algorithm replaces y_1 by x_n, coswitching and the hopping algorithm agree for $|X| = n$.

Notice that all the steps of our algorithm involve Pieri switches and replacement of x_i entries by numbers. We know the Pieri switches to be coplactic from Gillespie–Levinson [1], and the replacements preserve both the semi-standard and ballot requirements of T at any point in our algorithm. This means that these replacements are also coplactic, so each step of our algorithm commutes with JDT slides. Thus, the output of $\mathrm{coswitch}(X, T)$ and the output of $\mathrm{hop}(X, T)$ still agree when $X \sqcup T$ is skew shape. □

We demonstrate this result in the following example by computing coswitch of the same pair of tableau from Example 3.1.2.

Example 3.1.8.



We now describe an alternate method for computing local coswitch using the crystal operators described in Section 2.5. Note that because we use crystal operators and relabeling step which preserve the standardization of the tableau, the crystal algorithm is clearly coplactic. This version has the advantage that Phase 2 is easier to state than in the hopping algorithm.

Definition 3.1.9. (Type A crystal algorithm) Let X be a standard Young tableau such that $|X| = n$, and let T be a Littlewood–Richardson semi-standard Young tableau.

Then the crystal algorithm in Type A is defined as follows:

Evacuate X. Replace the entries of $e(X)$ in reading order by $-(n-1), \ldots, 0$. Set $j = 0$ and $i = n$.

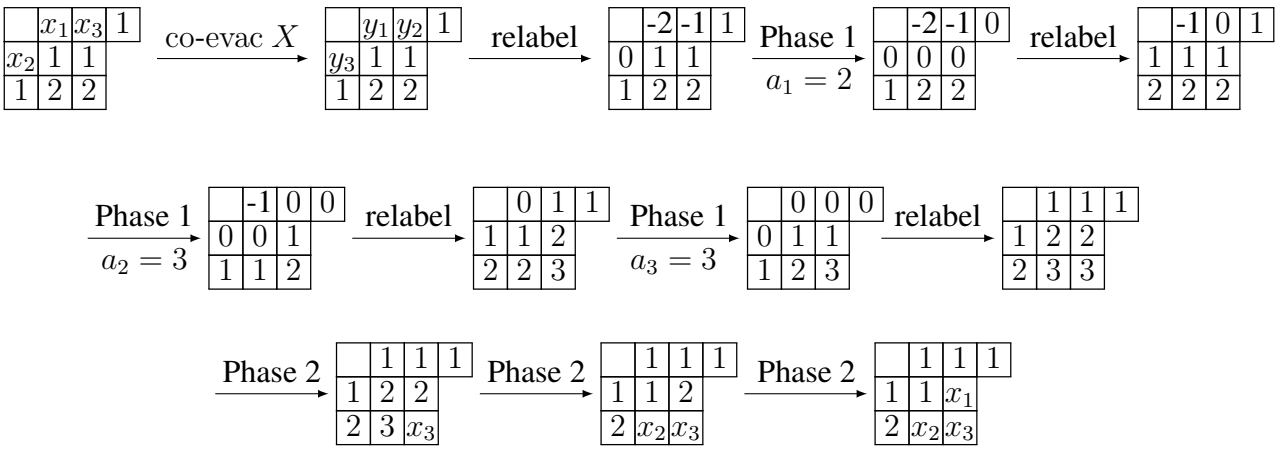
- **Phase 1a:** If a $j+1$ precedes the singular j in reading order, apply E_j a number of times equal to one less than the number of $(j+1)$'s. Increment j by one and repeat Phase 1a.

Otherwise, record $a_{n-i+1} = j+1$, increment every entry less than $j+1$ by one, and proceed to Phase 1b.

- **Phase 1b:** If $i = 1$, proceed to Phase 2. Otherwise, decrement i by one, reset j to be zero, and return to Phase 1a.
- **Phase 2:** If $i = n$, we are done. Otherwise, apply $F_{a_{n-i+1}}, \ldots, F_n$. Replace the new $n+1$ entry by x_{n-i+1}. Increment i by one and repeat Phase 2.

We demonstrate this algorithm in the following example.

Example 3.1.10.



Theorem 3.1.11. *The output of* $\mathrm{crystal}(X,T)$ *and the output of* $\mathrm{hop}(X,T)$ *agree.*

Proof. In $\mathrm{crystal}(X,T)$, at each application of E_j, we isolate the $j+1$ entry that precedes the unique j and lower all other $(j+1)$'s to j's. This makes the $j+1$ that preceded j our new unique number. This has the same effects as directly switching y_i with the j preceding it in reading order in $\mathrm{hop}(X,T)$. The only difference is that in $\mathrm{crystal}(X,T)$, our entries have been lowered, but we correct this by incrementing at the end of Phase 1a, so Phase 1a from both $\mathrm{crystal}(X,T)$ and $\mathrm{hop}(X,T)$ agree. Phase 1b is the same for both algorithms, so we need only consider whether their Phase 2's agree.

In Phase 2 of $\mathrm{crystal}(X,T)$, we start by applying F_{a_i}. This raises the first j entry whose $(j, j+1)$ suffix is tied, as that will be the last unmatched j. In $\mathrm{hop}(X,T)$, we replaced the first j in

reading order by x_{n-i+1}. Swapping x_{n-i+1} with this same j entry with a tied suffix puts a j back where x_{n-i+1} was and isolates the same entry that $\mathrm{crystal}(X,T)$ raised to be a $j+1$. We repeat this process in both $\mathrm{crystal}(X,T)$ and $\mathrm{hop}(X,T)$ until the same final entry is x_{n-i+1}, so Phase 2 agrees for both algorithms. □

Remark 3.1.12. We may compute Phase 1 and Phase 2 using either of the descriptions above. In general, the hopping description of Phase 1 is the most efficient and practical, while the crystal description of Phase 2 is better.

Definition 3.1.13. (Type A mixed algorithm) Let X be a standard Young tableau such that $|X| = n$, and let T be a Littlewood–Richardson semi-standard Young tableau. Let $e(X)$ be the tableau resulting from the evacuation of X with entries labeled $y_1, \ldots, y_n$ in order of evacuation.

Then the mixed algorithm in Type A is defined as follows:

Evacuate X. Let y_i be the ith entry of $e(X)$. Set $i = n$ and $j = 1$.

- **Phase 1a:** If y_i precedes all of the js in reading order, replace y_i by j, record $a_i = j$, and proceed to Phase 1b. If y_i does not precede all of the js in reading order, switch y_i with the j immediately preceding it in reading order. Increment j and repeat Phase 1a.

- **Phase 1b:** If $i = 1$, proceed to Phase 2. If $i \neq 1$, decrement i, reset $j = 1$, and repeat Phase 1a.

- **Phase 2:** Apply $F_{a_i}, \ldots, F_n$. Replace the new $n+1$ entry by x_{n-i+1}. Increment i by one and repeat Phase 2.

Chapter 4

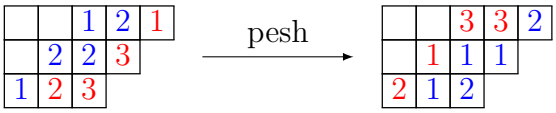
Related results and future directions

To discuss a key related result, we must first define another switching operation on tableaux.

Definition 4.0.1. We define **partial evacuation shuffling** (pesh) to be

$$\mathrm{pesh}(X,T) = \mathrm{coswitch}(\text{co–evac}(X),T).$$

We demonstrate this operation in the example below.



Corollary 4.0.2. *If we do not evacuate X in either* $\mathrm{hop}(X,T)$ *or* $\mathrm{crystal}(X,T)$*, the output we get instead is* $\mathrm{pesh}(X,T)$ *[11].*

Proof. Let $\mathrm{alg}(X,T)$ be either local algorithm applied to (X,T) without first evacuating X. Since both algorithms output $\mathrm{coswitch}(X,T)$, then

$$\mathrm{coswitch}(X,T) = \mathrm{alg}(\text{co–evac}(X),T).$$

We then use the fact that coplactic evacuation is an involution to rewrite the statement above:

$$\mathrm{coswitch}(\text{co–evac}(X),T) = \mathrm{alg}(\text{co–evac}(\text{co–evac}(X)),T) = \mathrm{alg}(X,T).$$

Notice that this is exactly the definition of pesh, meaning the output of alg must agree with that of pesh

$$\mathrm{alg}(X,T) = \mathrm{coswitch}(\text{co–evac}(X),T) = \mathrm{pesh}(X,T).$$

□

In addition to this additional local computation, we can extend these local algorithms to the same computations on tableaux in Type B without much editing necessary. [11]

Gillespie and Levinson found a connection with genomic tableaux [12] in the case of a single box tableau that allows them to compute the genus of Schubert curves. We then pose the following question for possible future work as an extension of the work done by Gillespie and Levinson:

Question 1. *Is there any connection to genomic tableaux and K-theory in this general case that generalizes the results of Gillespie–Levinson in [1]?*

Bibliography

[1] M. Gillespie and J. Levinson. Monodromy and k-theory of Schubert curves via generalized Jeu de Taquin. *Journal of Algebraic Combinatorics*, 45(1):191–243, August 2016.

[2] David E Speyer. Schubert problems with respect to osculating flags of stable rational curves. *Algebraic Geometry*, 1:14-45, 2014.

[3] M. Gillespie, J. Levinson, and K. Purbhoo. Schubert curves in the orthogonal Grassmannian. *Discrete & Computational Geometry*, 69(4):981–1039, 2023.

[4] M. Gillespie. Variations on a theme of Schubert calculus. In *Recent Trends in Algebraic Combinatorics*. Springer International Publishing, 2019.

[5] E. Mukhin, V. Tarasov, and A. Varchenko. Schubert calculus and representations of general linear group. *J. Amer. Math. Soc.*, 22(4):909–940, 2009.

[6] J. Levinson and K. Purbhoo. A topological proof of the Shapiro–Shapiro conjecture. *Invent. math.*, 226(2):521–578, 2021.

[7] R. Cavalieri. Moduli spaces of pointed rational curves, July 2016. Lecture Notes from Combinatorial Algebraic Geometry program at the Fields Institute.

[8] W. Fulton. *Young Tableaux: With Applications to Representation Theory and Geometry*. Cambridge University Press, 2003.

[9] G. Benkart, F. Sottile, and J. Stroomer. Tableau switching: Algorithms and applications. *Journal of Combinatorial Theory, Series A*, 76(1):11–43, 1996.

[10] D. Bump and A. Schilling. *Crystal Bases*. World Scientific, 2017.

[11] K. Brown and D. Moran. Local algorithms for coswitching and partial evacuation shuffling of tableaux, in preparation.

[12] O. Pechenik and A. Yong. Genomic tableaux. *Journal of Algebraic Combinatorics*, 45(3):649–685, 2017.