On combinatorial models for representations of classical Lie algebras

William Louis Adamczak

University at Albany, State University of New York, wladamczak@gmail.com

The University at Albany community has made this article openly available. Please share how this access benefits you.

Follow this and additional works at: https://scholarsarchive.library.albany.edu/legacy-etàd

Part of the Physical Sciences and Mathematics Commons

Recommended Citation

This Dissertation is brought to you for free and open access by the The Graduate School at Scholars Archive. It has been accepted for inclusion in Legacy Theses & Dissertations (2009 - 2024) by an authorized administrator of Scholars Archive. Please see Terms of Use. For more information, please contact scholarsarchive@albany.edu.
On Combinatorial Models
for Representations
of Classical Lie Algebras

by

William Adamczak

A Dissertation
Submitted to the University at Albany, State University of New York
in Partial Fulfillment of
the Requirements for the Degree of
Doctor of Philosophy

College of Arts & Sciences
Department of Mathematics and Statistics
2012
Contents

0 Introduction ........................................... 1

1 Background and Notation ................................. 4
   1.1 Root Systems ...................................... 4
   1.2 The Alcove Path Model ............................. 8
   1.3 Root Systems and Weyl Groups in Types $A_{n-1}$, $B_n$, $C_n$ and $D_n$ ... 13
      1.3.1 Type $A_{n-1}$ .................................. 13
      1.3.2 Type $C_n$ .................................... 14
      1.3.3 Type $B_n$ .................................... 15
      1.3.4 Type $D_n$ .................................... 15
   1.4 Tableaux and Crystal Graphs ......................... 17
      1.4.1 Type $A_n$ ..................................... 17
      1.4.2 Type $C_n$ .................................... 19
      1.4.3 Type $B_n$ .................................... 23
      1.4.4 Type $D_n$ .................................... 24

2 Type A ................................................ 29
   2.1 Specializing the Alcove Path Model to $A_{n-1}$ ........... 29
   2.2 The Bijection Between Admissible Subsequences and Tableaux .... 34
      2.2.1 The Filling Map ................................ 34
      2.2.2 An Inverse to the Filling Map ................. 35
5 Type $D$

5.1 Specializing the Alcove Path Model to $D_n$ .................................. 93
   5.1.1 The Filling Map ......................................................... 95
   5.1.2 $a$-Configurations ..................................................... 95
   5.1.3 An Inverse to the Filling Map ....................................... 98

5.2 Splittings and Right-Admissible Subsequences ............................... 103
   5.2.1 The Pebble Game ...................................................... 104
   5.2.2 Bijection Between the Pebble Game and KN Columns ............ 104
   5.2.3 Bijection Between the Pebble Game and Right Admissible Sub-
        sequences ................................................................. 106

5.3 A Level Counting Formula for Type $D$ ...................................... 109

5.4 The Crystal Graph Structure and Root Operators .......................... 109
ABSTRACT

Lenart and Postnikov have constructed a simple combinatorial model for the characters of the irreducible representations of a complex semisimple Lie group $G$ and, more generally, for the Demazure characters, a model called the alcove path model. There are other models in this area, such as: semistandard Young tableaux and Kashiwara-Nakashima tableaux, Littelmann paths, LS-galleries, the model based on Lusztig’s parametrization of canonical bases, some models based on geometric constructions etc. The alcove path model has advantages related to its generality, simplicity, combinatorial nature, and other applications, such as Demazure modules and Schubert calculus. In particular, it leads to a far-reaching generalization of the type $A$ combinatorics of Young tableaux. In this thesis we relate the alcove path model to the semistandard Young tableaux in type $A$ and the Kashiwara-Nakashima tableaux in types $B$, $C$ and $D$. In each type we construct an explicit bijection between the objects in the two models that is compatible with the corresponding crystal structures.
Acknowledgements

The author would like to express his thanks to all who helped along the way. To Professor Lenart for his guidance along the way. To Scott, Tin, Rick, Rich, Ben, Tim, and Amanda for help in the early years in various ways. To Arthur for providing an outlet for thoughts while revising and finishing this thesis. To family and friends for their prodding to complete this.
Chapter 0

Introduction

Lenart and Postnikov have recently constructed a simple combinatorial model for the characters of the irreducible representations of a complex semisimple Lie group $G$ and, more generally, for the Demazure characters [10]. For reasons explained below, this model is called the alcove path model. This was extended to complex symmetrizable Kac-Moody algebras in [11] (that is, to infinite root systems), and its combinatorics was investigated in more detail in [8].

There are other models in this area, such as: semistandard Young tableaux [2, 18, 22] and Kashiwara-Nakashima tableaux [19], Littelmann paths [15, 16, 17, 4], LS-galleries [3], the model in [1] based on Lusztig’s parametrization of canonical bases, some models based on geometric constructions etc. The alcove path model has advantages related to its generality, simplicity, combinatorial nature, and other applications, such as Demazure modules (which form a filtration of the irreducible modules) and Schubert calculus. In particular, it leads to a far-reaching generalization of the type $A$ combinatorics of Young tableaux.

The alcove path model is based on enumerating certain saturated chains in the Bruhat order on the corresponding Weyl group $W$. This enumeration is determined by an alcove path, which is a sequence of adjacent alcoves for the affine Weyl group.
of the Langlands dual group $G^\vee$. Alcove paths correspond to decompositions of elements in the affine Weyl group. Based on this model, one can extend to arbitrary root systems a considerable part of the very rich combinatorics of semistandard Young tableaux, which are classical combinatorial objects in the representation theory of $SL_n$. More precisely, we have:

1. cancellation free character formulas, including Demazure character formulas [10];

2. a Littlewood-Richardson rule for decomposing tensor products of irreducible representations and a branching rule [11];

3. a combinatorial description of the crystal graphs corresponding to the irreducible representations [11];

4. a combinatorial realization of certain fundamental involution on the canonical basis, which exhibits the crystals as self-dual posets, corresponds to the action of the longest Weyl group element on an irreducible representation, and generalizes Schützenberger’s involution on tableaux [8];

5. a generalization to arbitrary root systems of Schützenberger’s sliding algorithm (also known as jeu de taquin), which has many applications to the representation theory of the Lie algebra of type $A$ [8].

The goal of this paper is to relate the alcove path model to the semistandard Young tableaux in type $A$ and the Kashiwara-Nakashima tableaux in types $B$, $C$ and $D$. In each type we construct an explicit bijection between the objects in the two models that is compatible with the corresponding crystal structures. Having a nice translation between the general model given by the alcove path model and the explicit model given by tableaux will allow us to simplify formulas expressed in terms of the alcove path model in the classical types. Examples of formulas simplified by use of this translation include Schwer’s Formula for Hall-Littlewood polynomial [21] and
Ram-Yip formulas for Macdonald polynomials[20]. The first results in this direction may be found in [12], [13], and [14], all recent papers of Lenart. Future applications will include an efficient construction, based on the alcove model, of a basis for an irreducible representation as well as applications with Demazure characters.
Chapter 1

Background and Notation

1.1 Root Systems

In this section and the following, we recall root systems and the alcove path model in the representation theory of semisimple Lie algebras, closely following [10, 11].

Let $G$ be a connected, simply connected, simple complex Lie group. Fix a Borel subgroup $B$ and a maximal torus $T$ such that $G \supset B \supset T$. As usual, we denote by $B^-$ be the opposite Borel subgroup, while $N$ and $N^-$ are the unipotent radicals of $B$ and $B^-$, respectively. Let $\mathfrak{g}$, $\mathfrak{h}$, $\mathfrak{n}$, and $\mathfrak{n}^-$ be the complex Lie algebras of $G$, $T$, $N$, and $N^-$, respectively. Let $r$ be the rank of the Cartan subalgebra $\mathfrak{h}$. Let $\Phi \subset \mathfrak{h}^*$ be the corresponding irreducible root system, and let $\mathfrak{h}^* \subset \mathfrak{h}^*$ be the real span of the roots. Let $\Phi^+ \subset \Phi$ be the set of positive roots corresponding to our choice of $B$. Then $\Phi$ is the disjoint union of $\Phi^+$ and $\Phi^- = -\Phi^+$. We write $\alpha > 0$ (respectively, $\alpha < 0$) for $\alpha \in \Phi^+$ (respectively, $\alpha \in \Phi^-$), and we define sgn($\alpha$) to be 1 (respectively, $-1$).

Let $\Phi^+ \subset \Phi$ be the set of positive roots corresponding to our choice of $B$. Then $\Phi$ is the disjoint union of $\Phi^+$ and $\Phi^- = -\Phi^+$. We write $\alpha > 0$ (respectively, $\alpha < 0$) for $\alpha \in \Phi^+$ (respectively, $\alpha \in \Phi^-$), and we define sgn($\alpha$) to be 1 (respectively, $-1$). We also use the notation $|\alpha| := \text{sgn}(\alpha)\alpha$. Let $\alpha_1, \ldots, \alpha_r \in \Phi^+$ be the corresponding simple roots, which form a basis of $\mathfrak{h}_\mathbb{R}$. Let $\langle \cdot, \cdot \rangle$ denote the nondegenerate scalar product on $\mathfrak{h}_\mathbb{R}$ induced by the Killing form. Given a root $\alpha$, the corresponding coroot is $\alpha^\vee := 2\alpha/\langle \alpha, \alpha \rangle$. The collection of coroots $\Phi^\vee := \{\alpha^\vee | \alpha \in \Phi\}$ forms the dual root
The Weyl group \( W \subset \text{Aut}(\mathfrak{h}^*_\mathbb{R}) \) of the Lie group \( G \) is generated by the reflections \( s_\alpha : \mathfrak{h}^*_\mathbb{R} \to \mathfrak{h}^*_\mathbb{R} \), for \( \alpha \in \Phi \), given by

\[
s_\alpha : \lambda \mapsto \lambda - \langle \lambda, \alpha^\vee \rangle \alpha.
\]

In fact, the Weyl group \( W \) is generated by the simple reflections \( s_1, \ldots, s_r \) corresponding to the simple roots \( s_i := s_{\alpha_i} \), subject to the Coxeter relations:

\[
(s_i)^2 = 1 \quad \text{and} \quad (s_is_j)^{m_{ij}} = 1 \quad \text{for any } i, j \in \{1, \ldots, r\},
\]

where \( m_{ij} \) is half of the order of the dihedral subgroup generated by \( s_i \) and \( s_j \). An expression of a Weyl group element \( w \) as a product of generators \( w = s_{i_1} \cdots s_{i_l} \) which has minimal length is called a reduced decomposition for \( w \); its length \( \ell(w) = l \) is called the length of \( w \). The Weyl group contains a unique longest element \( w_\circ \) with maximal length \( \ell(w_\circ) = \#\Phi^+ \). For \( u, w \in W \), we say that \( u \) covers \( w \), and write \( u \triangleright w \), if \( w = us_\beta \), for some \( \beta \in \Phi^+ \), and \( \ell(u) = \ell(w) + 1 \). The transitive closure “\( \triangleright \)” of the relation “\( \triangleright \)” is called the Bruhat order on \( W \).

The Hasse diagram of the Bruhat order for any type will be the graph whose vertices are the elements of the Weyl group in that type and whose labels are given by \( w \overset{(i,j)}{\rightarrow} v \) provided that \( w = v(i,j) \) and \( \ell(v) = \ell(w) + 1 \), i.e. \( v \) covers \( w \).

The weight lattice \( \Lambda \) is given by

\[
\Lambda := \{ \lambda \in \mathfrak{h}^*_\mathbb{R} \mid \langle \lambda, \alpha^\vee \rangle \in \mathbb{Z} \text{ for any } \alpha \in \Phi \}. \tag{1.1.1}
\]

The weight lattice \( \Lambda \) is generated by the fundamental weights \( \omega_1, \ldots, \omega_r \), which are defined as the elements of the dual basis to the basis of simple coroots, i.e., \( \langle \omega_i, \alpha_j^\vee \rangle = \delta_{ij} \).
The set $\Lambda^+$ of dominant weights is given by

$$\Lambda^+ := \{ \lambda \in \Lambda \mid \langle \lambda, \alpha^\vee \rangle \geq 0 \text{ for any } \alpha \in \Phi^+ \}.$$ 

Let $\rho := \omega_1 + \cdots + \omega_r = \frac{1}{2} \sum_{\beta \in \Phi^+} \beta$. The height of a coroot $\alpha^\vee \in \Phi^\vee$ is $\langle \rho, \alpha^\vee \rangle = c_1 + \cdots + c_r$ if $\alpha^\vee = c_1 \alpha_1^\vee + \cdots + c_r \alpha_r^\vee$. Since we assumed that $\Phi$ is irreducible, there is a unique highest coroot $\theta^\vee \in \Phi^\vee$ that has maximal height. (In other words, $\theta^\vee$ is the highest root of the dual root system $\Phi^\vee$. It should not be confused with the coroot of the highest root of $\Phi$.) We will also use the Coxeter number, that can be defined as $h := \langle \rho, \theta^\vee \rangle + 1$.

Let $W_{\text{aff}}$ be the affine Weyl group for the Langlands dual group $G^\vee$. The affine Weyl group $W_{\text{aff}}$ is generated by the affine reflections $s_{\alpha,k} : h^*_R \rightarrow h^*_R$, for $\alpha \in \Phi$ and $k \in \mathbb{Z}$, that reflect the space $h^*_R$ with respect to the affine hyperplanes

$$H_{\alpha,k} := \{ \lambda \in h^*_R \mid \langle \lambda, \alpha^\vee \rangle = k \}.$$ 

(1.1.2)

Explicitly, the affine reflection $s_{\alpha,k}$ is given by

$$s_{\alpha,k} : \lambda \mapsto s_{\alpha}(\lambda) + k \alpha = \lambda - (\langle \lambda, \alpha^\vee \rangle - k) \alpha.$$

The hyperplanes $H_{\alpha,k}$ divide the real vector space $h^*_R$ into open regions, called alcoves. Each alcove $A$ is given by inequalities of the form

$$A := \{ \lambda \in h^*_R \mid m_{\alpha} < \langle \lambda, \alpha^\vee \rangle < m_{\alpha} + 1 \text{ for all } \alpha \in \Phi^+ \},$$

where $m_{\alpha} = m_{\alpha}(A), \alpha \in \Phi^+$, are some integers.

The fundamental alcove $A_{\circ}$ is given by
\[ A_\circ := \{ \lambda \in h^*_R \mid 0 < \langle \lambda, \alpha^\vee \rangle < 1 \text{ for all } \alpha \in \Phi^+ \}. \]

Let \( A_\lambda = A_{\circ - \lambda} \) be the alcove obtained by the affine translation of the fundamental alcove by a weight \( \lambda \in \Lambda \). We then define \( v_\lambda \) as the element of \( W_{aff} \) that translates \( A_\circ \) to \( A_\lambda \), i.e. \( v_\lambda(A_\circ) = A_\lambda \) [7].
1.2 The Alcove Path Model

In this section we establish the needed background for the alcove path model. This section will also closely follow the background given in [10, 11].

Definition 1.2.1. A $\lambda$-chain of roots is a sequence of positive roots $(\beta_1, \ldots, \beta_l)$ which is determined as indicated below by a reduced decomposition $v_{-\lambda} = s_{i_1} \cdots s_{i_l}$ of $v_{-\lambda}$ as a product of generators of $W_{\text{aff}}$:

$$
\beta_1 = \alpha_{i_1}, \quad \beta_2 = \bar{s}_{i_1}(\alpha_{i_2}), \quad \beta_3 = \bar{s}_{i_1}\bar{s}_{i_2}(\alpha_{i_3}), \ldots, \beta_n = \bar{s}_{i_1} \cdots \bar{s}_{i_{l-1}}(\alpha_{i_l}).
$$

When the context allows, we will abbreviate “$\lambda$-chain of roots” to “$\lambda$-chain”. The $\lambda$-chain of reflections associated with the above $\lambda$-chain of roots is the sequence $(\hat{r}_1, \ldots, \hat{r}_l)$ of affine reflections in $W_{\text{aff}}$ given by

$$
\hat{r}_1 = s_{i_1}, \quad \hat{r}_2 = s_{i_1}s_{i_2}s_{i_1}, \quad \hat{r}_3 = s_{i_1}s_{i_2}s_{i_3}s_{i_2}s_{i_1}, \ldots, \hat{r}_l = s_{i_1} \cdots s_i \cdots s_{i_1}.
$$

We now present an equivalent definition of a $\lambda$-chain of roots.

We say that two alcoves are adjacent if they share a common wall. Given a pair of adjacent alcoves $A$ and $B$ with common wall $H_{\beta,k}$ with $\beta \in \Phi$ pointing in the direction from $A$ to $B$ we write $A \xrightarrow{\beta} B$.

Definition 1.2.2. An alcove path is a sequence of alcoves

$$
A_0 \xrightarrow{\beta_1} A_1 \xrightarrow{\beta_2} \cdots \xrightarrow{\beta_l} A_l
$$

such that $A_{i-1}$ and $A_i$ are adjacent, for $i = 1, \ldots, l$. We say that an alcove path is reduced if it has minimal length among all alcove paths from $A_0$ to $A_l$. If $A_0 = A^\circ$ and $A_l = A^\circ - \lambda$ then the sequence of roots $(\beta_1, \ldots, \beta_l)$ is a $\lambda$-chain (of roots).
Example 1.2.3. Consider the dominant weight \( \lambda = (3, 2, 0) \). Then the diagram below gives an alcove path with \( \lambda \)-chain given by \((\alpha_{23}, \alpha_{13}, \alpha_{23}, \alpha_{13}, \alpha_{12}, \alpha_{13})\).

Given a finite sequence of roots \( \Gamma = (\beta_1, \ldots, \beta_l) \), we define the sequence of integers \((l_1, \ldots, l_l)\) by \( l_i := \#\{j < i \mid \beta_j = \beta_i\} \), for \( i = 1, \ldots, l \). We shall refer to these \( l_i \)'s as the level of \( \beta_i \). Note that \( \hat{r}_i = s_{\beta_i, l_i} \). The following two conditions on \( \Gamma \) are equivalent to the definitions of \( \lambda \)-chains and alcove paths.

(R1) The number of occurrences of any positive root \( \alpha \) in \( \Gamma \) is \( \langle \lambda, \alpha^\vee \rangle \).

(R2) For each triple of positive roots \((\alpha, \beta, \gamma)\) with \( \gamma^\vee = \alpha^\vee + \beta^\vee \), the subsequence of \( \Gamma \) consisting of \( \alpha, \beta, \gamma \) is a concatenation of pairs \((\alpha, \gamma)\) and \((\beta, \gamma)\) (in any order).

Definition 1.2.4. An admissible subset is a subset of \([n] := \{1, \ldots, n\}\) (possibly empty), that is, \( J = \{j_1 < j_2 < \ldots < j_s\} \), such that we have the following saturated chain in the Bruhat order on \( W \):

\[ 1 \leq r_{j_1} \leq r_{j_1}r_{j_2} \leq \ldots \leq r_{j_1}r_{j_2} \ldots r_{j_s}. \]

Here \( r_i = s_{\beta_i} \). We denote by \( \mathcal{A} = \mathcal{A}(\Gamma) \) the collection of all admissible subsets corresponding to our fixed \( \lambda \)-chain \( \Gamma \). Given an admissible subset \( J \), we use the
notation

\[ \mu(J) := -\widehat{r}_{j_1} \ldots \widehat{r}_{j_s}(-\lambda), \quad w(J) := r_{j_1} \ldots r_{j_s}. \]

We call \( \mu(J) \) the weight of the admissible subset \( J \). Note that \( \mathcal{A}(\Gamma) \) serves an indexing set for a basis.

We have the following character formula in terms of admissible subsets:

\[ \text{ch}(V_\lambda) = \sum_{J \in \mathcal{A}} e^{\mu(J)}. \]

Let \( U(g) \) be the enveloping algebra of the Lie algebra \( g \), which is generated by \( E_i, F_i, H_i \), for \( i = 1, \ldots, r \), subject to the Serre relations. Let \( \mathcal{B} \) be the canonical basis of \( U(n^-) \), and let \( \mathcal{B}_\lambda := \mathcal{B} \cap V_\lambda \) be the canonical basis of the irreducible representation \( V_\lambda \) with highest weight \( \lambda \). Let \( v_\lambda \) and \( v_\lambda^{\text{low}} \) be the highest and lowest weight vectors in \( \mathcal{B}_\lambda \), respectively. Let \( E_i, F_i \), for \( i = 1, \ldots, r \), be Kashiwara’s operators; these are also known as raising and lowering operators, respectively. The crystal graph of \( V_\lambda \) is the directed colored graph on \( \mathcal{B}_\lambda \) defined by arrows \( x \to y \) colored \( i \) for each \( F_i(x) = y \).

We now describe the action of root operators on the collection \( \mathcal{A}(\Gamma) \) of admissible subsets corresponding to our fixed \( \lambda \)-chain. They are associated with a fixed simple root \( \alpha_p \), and are traditionally denoted by \( F_p \) (also called a lowering operator) and \( E_p \) (also called a raising operator).

Let \( J \) be a fixed admissible subset, and let

\[ \gamma(J) = (F_0, A_0, F_1, \ldots, F_l, A_l, F_\infty), \quad \Gamma(J) = ((\gamma_1, \gamma'_1), \ldots, (\gamma_l, \gamma'_l), \gamma_\infty). \]

Fix a permutation \( w \) in \( S_n \) and a subset \( J = \{j_1 < \ldots < j_l\} \) of \([m]\) (not necessarily \( w \)-admissible). Let \( \Pi \) be the alcove path corresponding to \( \Gamma \), and define the alcove walk \( \Omega \) by

\[ \Omega := \phi_{j_1} \ldots \phi_{j_l}(w(\Pi)). \]

Given \( k \) in \([m]\), let \( i = i(k) \) be the largest index in \([s]\) for which \( j_i < k \). Let \( \gamma_k := \ldots \)
$w r_{j_1} \ldots r_{j_i} (\beta_k)$, and let $H_{\gamma_k, m_k}$ be the hyperplane containing the face $F_k$ of $\Omega$. In other words

$$H_{\gamma_k, m_k} = w \hat{r}_{j_1} \ldots \hat{r}_{j_i} (H_{\beta_k, l_k}).$$

One of our goals will be to describe $m_k$ purely in terms of the filling associated to $(w, J)$.

Let $\hat{t}_k$ be the affine reflection in the hyperplane $H_{\gamma_k, m_k}$. Note that

$$\hat{t}_k = w \hat{r}_{j_1} \ldots \hat{r}_{j_i} \hat{r}_{k} \hat{r}_{j_i} \ldots \hat{r}_{j_1} w^{-1}.$$ 

Thus, we can see that

$$w \hat{r}_{j_1} \ldots \hat{r}_{j_i} = \hat{t}_{j_i} \ldots \hat{t}_{j_1} w.$$

It will be useful for us to view the roots $\gamma_k$ as representing hyperplanes after foldings across hyperplanes that are given by a $\beta_k$ where foldings are only performed if the $\beta_k$ was selected in the admissible subsequence. Referring to the subsequence after folding will be understood in this fashion from this point onward.

Let us also fix a simple root $\alpha_p$. We associate with $J$ the sequence of integers $L(J) = (m_1, \ldots, m_l)$ defined by $F_i \subset H_{-|\gamma_i|, m_i}$ for $i = 1, \ldots, l$. We also define $m_{\infty}^p := \langle \mu(J), \alpha_p^\lor \rangle$, which means that $F_{\infty} \subset H_{-\alpha_p, m_{\infty}^p}$. Finally, we let

$$I(J, p) := \{i \mid \delta_i = -\alpha_p\}, \quad L(J, p) := (\{m_i\}_{i \in I(J, p)}, l_{\infty}^p), \quad M(J, p) := \max L(J, p).$$

(1.2.5)

We first consider $F_p$ on the admissible subset $J$. This is defined whenever $M(J, p) > 0$. Let $m = m_F(J, p)$ be defined by

$$m_F(J, p) := \begin{cases} \min \{i \in I(J, p) \mid m_i = M(J, p)\} & \text{if this set is nonempty} \\ \infty & \text{otherwise} \end{cases}.$$
Let $k = k_F(J, p)$ be the predecessor of $m$ in $I(J, p) \cup \{\infty\}$, which always exists. It turns out that $m \in J$ if $m \neq \infty$, but $k \not\in J$. Finally, we set

$$F_p(J) := (J \setminus \{m\}) \cup \{k\}. \quad (1.2.6)$$

Let us now define a partial inverse $E_p$ to $F_p$. The operator $E_p$ is defined on the admissible subset $J$ whenever $M(J, p) > \langle \mu(J), \alpha_p^\vee \rangle$. Let $k = k_E(J, p)$ be defined by

$$k_E(J, p) := \max \{i \in I(J, p) \mid m_i = M(J, p)\};$$

the above set turns out to be always nonempty. Let $m = m_E(J, p)$ be the successor of $k$ in $I(J, p) \cup \{\infty\}$. It turns out that $k \in J$ but $m \not\in J$. Finally, we set

$$E_p(J) := (J \setminus \{k\}) \cup (\{m\} \setminus \{\infty\}). \quad (1.2.7)$$
1.3 Root Systems and Weyl Groups in Types $A_{n-1}, B_n, C_n$ and $D_n$

In each type the root system, Weyl group, and Bruhat order will be critical for the discussion that follows. In this vein we give this background and descriptions for each type.

1.3.1 Type $A_{n-1}$

In type $A_{n-1}$ the Weyl group is $S_n$, i.e. the group of permutations on $n$ elements.

Consider the $n-1$ dimensional subspace of $\mathbb{R}^n$ orthogonal to the vector $e_1 + ... + e_n$ where $e_i$ for $i \in [n]$ are the basis vectors of $\mathbb{R}^n$. Then the root system is given by:

$$\Phi = \{e_i - e_j, i \neq j, i \text{ and } j \in [n]\}$$

We shall also need the following notions for the Bruhat order on $S_n$:

Let $t_{ab}$ be the transposition sending $(a, b)$ to $(b, a)$. The covering relations in the Bruhat order are $\sigma \trianglelefteq \tau = \sigma \cdot t_{ab}$, where $\ell(\tau) = \ell(\sigma) + 1$. We denote this by

$$\sigma \xrightarrow{t_{ab}} \tau.$$ 

Here $\ell(\sigma)$ is given by the equation:

$$\ell(\sigma) = \# \{1 \leq i < j \leq n : \sigma(i) > \sigma(j) \} \quad (1.3.1)$$

In type $A$ we have the following cover condition:

**Theorem 1.3.2.** A permutation $\sigma$ admits a cover $\sigma \trianglelefteq \sigma \cdot t_{ab}$ with $a < b$ and $\sigma(a) < \sigma(b)$ if and only if whenever $a < c < b$, then either $\sigma(c) < \sigma(a)$ or else $\sigma(b) < \sigma(c)$. 
1.3.2 Type $C_n$

We order the letters in type $C_n$ as follows: $[\pi] := \{1 < 2 < \ldots < n - 1 < n < \pi < n - \bar{1} < \ldots < \bar{2} < \bar{1}\}$ (one should note that this is not the standard order). The group $C_n$ is the group of signed permutations. An element of $C_n$ is such that $\sigma(\bar{i}) = \bar{\sigma(i)}$ for all $i \in [n]$, here we use the convention that $\bar{i} = i$. It is therefore sufficient to write only the first $n$ entries of any permutation. We write an element $\sigma \in C_n$ as $\sigma = (\sigma(1) \ldots \sigma(n))$.

Consider the space $\mathbb{R}^n$ with basis vectors $e_i$ for $i \in [n]$. Then the root system for type $C_n$ is given by:

$$\Phi = \{\pm e_i \pm e_j, i, j \in [n]\}.$$  

For $i < j$ we shall make the following identifications:

- $(i, j)$ with $e_i - e_j$ and $s_{e_i-e_j} = t_{ij}t_{\bar{j}}$,
- $(i, \bar{j})$ with $e_i + e_j$ and $s_{e_i+e_j} = t_{i\bar{j}}t_{\bar{j}}$,
- $(i, \bar{i})$ with $2e_i$ and $s_{2e_i} = t_{i\bar{i}}$.

Let $\sigma \in C_n$ be a permutation.

**Theorem 1.3.3.** $\pi$ covers $\sigma$ in the strong Bruhat order on $C_n$ if and only if there exist $i, j \in [1, \bar{1}]$ with $i \leq n$, $i < j \leq \bar{1}$ such that:

1. $\sigma(i) < \sigma(j)$
2. if $j > n$ then either $\sigma(j) \leq n$ or $\sigma(i) \geq \bar{n}$
3. $\pi = \sigma(i, j)$
4. There is no $i < l < j$ such that $\sigma(i) < \sigma(l) < \sigma(j)$. 

14
Let \( \ell(\sigma) \) be used to denote the length of the element \( \sigma \in C_n \). This is then given by the formula

\[
\ell(\sigma) = \# \{ 1 \leq i < j \leq n : \sigma(i) > \sigma(j) \} + \sum_{i \leq n: \sigma(i) \geq \pi} (n + 1 - \sigma(i)) \quad (1.3.4)
\]

### 1.3.3 Type \( B_n \)

The Weyl group for type \( B_n \) for \( n \geq 2 \) is identical to that of type \( C_n \), consequently the cover conditions and length formulas are the same for both types.

Consider the space \( \mathbb{R}^n \) with basis vectors \( e_i \) for \( i \in [n] \). Then the root system for type \( B_n \) is given by:

\[
\Phi = \{ \pm e_i, \pm (e_i \pm e_j), i \neq j \in [n] \}.
\]

### 1.3.4 Type \( D_n \)

In type \( D_n \) for \( n \geq 4 \) the Weyl group is the group consisting of signed permutations on \( n \) elements where only an even number of negative signs are permitted.

Consider the space \( \mathbb{R}^n \) with basis vectors \( e_i \) for \( i \in [n] \). Then the root system for type \( D_n \) is given by:

\[
\Phi = \{ \pm (e_i \pm e_j), i \neq j \in [n] \}.
\]

We now consider the cover condition in type \( D_n \). Let \( \sigma \in D_n \) be a permutation.

**Theorem 1.3.5.** \( \pi \) covers \( \sigma \) in the strong Bruhat order on \( D_n \) if and only if there exist \( i, j \in [1, \overline{1}] \) such that:

1. \( \sigma(i) < \sigma(j) \)
2. \( \pi = \sigma(i, j) \) with \( i \leq n, \ i < j \leq \bar{1} \)

3. There is no \( i < l < j, \ l \neq \bar{i}, \bar{j} \) such that \( \sigma(i) < \sigma(l) < \sigma(j) \).

Note that this cover condition is nearly the same as in type C, the difference being that if \( a \) and \( b \) are the values exchanged, in positions \( i \) and \( j \) respectively, and either \( \bar{a} \) (or \( \bar{b} \)) is such that \( a < \bar{a} < b \) (or \( a < \bar{b} < b \)), then this no longer violates the cover condition as it would have in type C. Note that \( a \) and \( b \) here are not necessarily positive values.
1.4 Tableaux and Crystal Graphs

In this section we recall the background on semistandard Young tableaux, and we refer the reader to [2, 18, 22] for more details. We also recall the tableaux of Kashiwara and Nakashima in types $B$, $C$ and $D$ [19]. Additionally we give the basic background for the crystal graphs in each type.

1.4.1 Type $A_n$

A Young diagram is a sequence of left justified boxes in rows with the lengths of rows weakly decreasing. A Young tableau is then a filling of a Young diagram where numbers are placed in each box such that the entries are weakly increasing across rows and strictly increasing down columns.

The shape $\lambda$ of a tableau $T$ is given by $\lambda = (\lambda_1, ..., \lambda_k)$ where $\lambda_i$ is the length of the $i^{th}$ row and $k$ is the number of rows. The shape of the conjugate tableau is given by $\lambda' = (\lambda'_1, ..., \lambda'_m)$ where $\lambda'_i$ is the length of the $i^{th}$ column and $m$ is the number of columns, [2].

The column word $w$ for a tableau $T$ is the listing of the entries from the bottom to the top in each column, starting with the leftmost column. This definition of column word is type independent, as are shapes of tableaux, and will be applied freely to other types throughout.

Example 1.4.1. The following is an example of a Young tableau of shape $\lambda = (4, 3, 1)$:

\[
\begin{array}{cccc}
1 & 1 & 2 & 4 \\
2 & 3 & 3 & \\
& 4 & \\
\end{array}
\]

In this case $\lambda' = (3, 2, 2, 1)$. The column word associated to $T$ is $w = 42131324$. 

17
Consider now the column word $w$ of a tableau $T$ in type $A$. Then consider the subword of $w$ consisting of only the entries of the form $i$ or $i+1$. Call this word $w_i$. Adjacent pairs of the form $i + 1$ $i$ may be ignored. Repeat this until a word of the form $\rho(w) = i^*i + 1^*$ is reached.

**Definition 1.4.2.** In the context of tableaux $e_i$ and $f_i$ are then defined as follows:

If $s > 0$, $e_i(w)$ is obtained by changing the leftmost $i + 1$ to an $i$ and all other letters remain unchanged. If $s = 0$ then $e_i(w) = 0$. Then $f_i(w)$ is defined as the inverse.

If $r > 0$, $f_i(w)$ is obtained by changing the rightmost $i$ to an $i + 1$ and all other letters remain unchanged. If $r = 0$ then $f_i(w) = 0$. Then $f_i(w)$ is defined as the inverse.

**Example 1.4.3.** Consider the following tableau $T$ and the action of $f_2$ on $T$:

$$T = \begin{array}{ccc}
1 & 2 & 3 \\
2 & 3 & 5
\end{array}.$$

In this case we have $w = 52132$ and $w_2 = 232$ which after pairing gives $\rho(w) = 2$. Thus $f_2$ changes the 2 in the first column into a 3 giving us the following picture:

$$\begin{array}{ccc}
1 & 2 & 3 \\
2 & 3 & 5
\end{array} \rightarrow \begin{array}{ccc}
1 & 2 & 3 \\
2 & 3 & 5
\end{array}.$$

**Definition 1.4.4.** We define the content of a filling $\sigma$ of a tableau $T$, $ct(\sigma)$, as $ct(\sigma) = (#1, \ldots, #n)$.

Therefore $ct(\sigma)$ is the ordered $n$-tuple counting the number of occurrences of each entry in the filling $\sigma$ of the tableau $T$.

**Example 1.4.5.** Following the previous example where $\sigma$ is the filling of the tableau $T$ we have $ct(\sigma) = (1, 2, 1, 0, 1)$.

This definition of content will be the same in future types.
1.4.2 Type $C_n$

In type $C_n$ Kashiwara-Nakashima or KN tableaux will be one of our primary objects of interest, much of the background here comes from [5]. We begin by defining columns:

A column is a Young diagram $C$ of column shape filled with letters from $[\bar{n}]$ strictly increasing from top to bottom.

Here the set $[\bar{n}]$ has the following order:

$$1 < 2 < \cdots < n < \bar{n} < \cdots < \bar{2} < 1. \quad (1.4.6)$$

The $i$th entry of a column $C$ shall be denoted $C(i)$.

**Definition 1.4.7.** For a column $C$ let $I = \{z_1 > \cdots > z_r\}$ be the set of unbarred letters $z$ such that the pair $(z, \overline{z})$ occurs in $C$. The column $C$ is said to split when there exists a set of $r$ unbarred letters $J = \{t_1 > \cdots > t_r\} \subset [\bar{n}]$ such that:

1. $t_1$ is the greatest letter of $[\bar{n}]$ satisfying: $t_1 < z_1, t_1 \notin C$ and $\overline{t_1} \notin C$

2. for $i = 2, \ldots, r$, $t_i$ is the greatest letter of $[\bar{n}]$ satisfying: $t_i < \min(t_{i-1}, z_i), t_i \notin C$ and $\overline{t_i} \notin C$.

In the case where the column $C$ may be split we write:

$rC$ for the column obtained by changing in $C$, $z_i$ into $\overline{t}_i$ for each letter $z_i \in I$ and by reordering if necessary,

$lC$ for the column obtained by changing in $C$, $z_i$ into $t_i$ for each letter $z_i \in I$ and by reordering if necessary.

A column is KN-admissible if and only if it can be split.
Example 1.4.8.

\[
C = \begin{array}{ccc}
2 & 3 & 5 \\
3 & 5 & 2 \\
5 & 2 & 3
\end{array}
\]

Then \( I = \{5, 2\} \) and \( J = \{4, 1\} \) and

\[
lC = \begin{array}{ccc}
1 & 3 & 4 \\
4 & 5 & 2
\end{array}, \quad rC = \begin{array}{ccc}
2 & 3 & 5 \\
4 & 5 & 1
\end{array}.
\]

We place a partial order on columns, let \( C \) and \( D \) be columns of length \( k_C \) and \( k_D \) respectively. We then say that \( C \leq D \) provided that \( k_C \geq k_D \) and \( rC(i) \leq lD(i) \) for \( 1 \leq i \leq k_D \).

Let \( \sigma \) be a permutation of length \( 2n \) in type \( C_n \) and let \( C \) be a column of a tableau of this type of length \( l \) for some \( 1 \leq l \leq n \) such that \( \sigma[l] = C \). Here we define \( \sigma[k] \) to be the restriction of \( \sigma \) to it’s first \( k \) entries. Similarly define \( \sigma[i, j] \) to be the restriction of \( \sigma \) to positions \( i \) through \( j \).

Start with a partition \( \lambda = (\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_n > 0) \) and the conjugate partition \( \lambda' = (\lambda'_1 \geq \lambda'_2 \geq ... \geq \lambda'_m > 0) \). Then a KN tableau \( T \) of shape \( \lambda \) will be a tableau with \( n \) rows where for \( 1 \leq i \leq n \) the length of row \( i \) is \( \lambda_i \), or equivalently \( T \) is a tableau with columns such that for \( 1 \leq j \leq m \) the column \( j \) has length \( \lambda'_j \). In order for \( T \) to be a KN tableau the column \( j \) which we shall denote \( C_j \) must be a KN-admissible column, i.e. the column \( C_j \) must split, and \( C_{j-1} \leq C_j \) for \( 1 < j \leq m \).

It will be convenient for us to view \( T \) as having \( 2m \) columns rather than \( m \) columns. The way this is done is by replacing each column \( C_j \) with the pair of columns \( lC_j \) and \( rC_j \). We shall refer to such a tableau as the split tableau \( spl(T) \). This terminology shall also apply to types \( B \) and \( D \). Note that the partial order on columns given above will guarantee that the entries are weakly increasing from left to right in rows
in the split tableau $spl(T)$.

**Example 1.4.9.** The example to follow shows the correspondence between $T$ and $spl(T)$.

\[
T = \begin{array}{ccc}
1 & 2 & \\
4 & 4 & \\
4 & 4 & \\
2 & \\
\end{array},
\quad spl(T) = \begin{array}{ccc}
1 & 1 & 1 & 2 & \\
3 & 4 & 4 & 4 & \\
4 & 3 & 2 & 1 & \\
2 & 2 & \\
\end{array}.
\]

Recall the construction of column words from type $A$, in particular the word is read starting at the bottom of the column working toward the top of the column, starting with the leftmost column. Note that this word is read from the unsplit tableau, not the split tableau. In type $C$ the same process is used as was in type $A$. Call this word $w$. Then consider the subword of $w$ consisting of the entries of the form, $i$, $i + 1$, $i$ or $i + 1$, for $i < n$, and call this word $w_i$. Denote $i + 1$ or $i$ by a $+$ and denote $i$ or $i + 1$ by a $-$. Adjacent pairs of the form $-+$ may be ignored. This may be repeated until a subword of the form $\rho(w) = +r-s$ is reached.

If $s > 0$ $e_i(w)$ is obtained by changing the leftmost $-$ to a $+$ (i.e. changing $i + 1$ into $i$ and $\overline{i + 1}$ into $\overline{i + 1}$) and all other letters remain unchanged. If $s = 0$ then $e_i(w) = 0$.

Then $f_i(w)$ is defined as the inverse.

If $r > 0$ $f_i(w)$ is obtained by changing the rightmost $+$ to a $-$ (i.e. changing $i$ into $i + 1$ and $\overline{i + 1}$ into $\overline{i}$) and all other letters remain unchanged. If $r = 0$ then $f_i(w) = 0$. Then $e_i(w)$ is defined as the inverse.

In the case where $i = n$ consider the subword of $w$ consisting of entries of the form $n$ or $\overline{n}$, call this word $w_n$. Denote $n$ by a $+$ and denote $\overline{n}$ by a $-$. Adjacent pairs of the form $+-$ may be ignored. This may be repeated until a subword of the form $\rho(w) = +r-s$ is reached as before. Then $e_n(w)$ and $f_n(w)$ are defined as above.

Now here in type $C$ as well as in the future for types $B$ and $D$ we shall need to consider the column word of the split tableau. We shall call this word $\overline{w}$. This is read in exactly the same way as the column word only on the split tableau. Then define
ρ(\tilde{w}) by exactly the same pairing process as for ρ(w). Then as discussed in [19] we have that $f_i$ acts as $f_i^2$ on $\rho(\tilde{w})$.

Here we give an example that shows how $f_i$ acts in the context of type $C$ using both the split tableau and the unsplit tableau. We note however that the action in terms of the split tableau will be of greater use in the discussion which follows.

**Example 1.4.10.** Begin by considering the following tableau $T$ with split tableau $\text{spl}(T)$:

\[
T = \begin{array}{cccc}
1 & 2 & 2 & 4 \\
2 & 2 & 2 & 3 \\
4 & 4 & 3 & 3 \\
3 & 3 & 3 & 3
\end{array}
\quad \text{spl}(T) = \begin{array}{cccc}
1 & 1 & 1 & 2 \\
2 & 2 & 2 & 1 \\
4 & 4 & 3 & 3 \\
3 & 3 & 3 & 3
\end{array}.
\]

These have corresponding column words:

\[
w = \overline{3421} \overline{2}, \quad \tilde{w} = \overline{3421} \overline{3421} \overline{21} \overline{12}.
\]

We then consider $f_2(T)$ and $f_2(\text{spl}(T))$. Begin by considering $\rho(w)$ and $\rho(\tilde{w})$.

\[
\rho(w) = ++ \quad \rho(\tilde{w}) = ++++
\]

So then we have:

\[
f_2(\rho(w)) = + - \quad f_2(\rho(\tilde{w})) = + + --
\]

yielding

\[
f_2(w) = \overline{3431} \overline{22} \quad f_2(\tilde{w}) = \overline{3421} \overline{3431} \overline{21} \overline{12}.
\]

This gives us the following for $f_2(T)$ and $f_2(\text{spl}(T))$:

\[
f_2(T) = \begin{array}{cccc}
1 & 2 & 2 & 4 \\
3 & 2 & 2 & 3 \\
4 & 4 & 3 & 3 \\
3 & 3 & 3 & 3
\end{array}
\quad f_2(\text{spl}(T)) = \begin{array}{cccc}
1 & 1 & 1 & 2 \\
2 & 3 & 2 & 1 \\
4 & 4 & 3 & 3 \\
3 & 3 & 3 & 3
\end{array}.
\]
1.4.3 Type \( B_n \)

In type \( B_n \) the Kashiwara-Nakashima tableaux are as follows[6]:

A column \( C \) is a Young Diagram of column shape filled with letters from \([\tilde{n}]\) increasing from top to bottom and 0 is the only letter that may appear more than once. Here we shall take \([\tilde{n}]\) to be the set \([\pi]\cup\{0\}\) with the following order:

\[
1 < 2 < \cdots < n < 0 < \tilde{n} < \cdots < 2 < 1.
\]

For a column \( C \) of type \( B \) let \( I = \{z_1 = 0, \ldots, z_r = 0 > z_{r+1} > \ldots > z_s\}\) be the set of unbarred letters \( z \) such that the pair \((z, \overline{z})\) occurs in \( C \). The column \( C \) is said to split when there exists a set of \( s \) unbarred letters \( J = \{t_1 > \ldots > t_s\}\subset[\tilde{n}]\) such that:

1. \( t_1 \) is the greatest letter of \( \pi \) satisfying: \( t_1 < z_1, t_1 \notin C \) and \( \overline{t}_1 \notin C \)

2. for \( i = 2, \ldots, s, \) \( t_i \) is the greatest letter of \( \pi \) satisfying: \( t_i < \min(t_{i-1}, z_i), t_i \notin C \) and \( \overline{t}_i \notin C \).

As in type \( C \), in the case where the column \( C \) may be split we write:

\( rC \) for the column obtained by changing in \( C \), \( \overline{z}_i \) into \( \overline{t}_i \) for each letter \( z_i \in I \) and by reordering if necessary,

\( lC \) for the column obtained by changing in \( C \), \( z_i \) into \( t_i \) for each letter \( z_i \in I \) and by reordering if necessary [6].

Also as in type \( C \), a column in type \( B \) is admissible if and only if it can be split.

Once an admissible column has been defined the tableaux setup in type \( B \) is the same as in type \( C \).
Example 1.4.11. Let

\[
C = \begin{pmatrix}
2 \\
4 \\
0 \\
5 \\
4
\end{pmatrix}
\]

Then \( I = \{0, 4\} \) and \( J = \{3, 1\} \) and

\[
lC = \begin{pmatrix}
1 \\
2 \\
3 \\
5 \\
4
\end{pmatrix},
\quad
rC = \begin{pmatrix}
2 \\
4 \\
5 \\
3 \\
1
\end{pmatrix}.
\]

A tableau \( T \) in type \( B \) is then defined in the same fashion as in type \( C \). The columns of \( T \) must all split and the weakly increasing across rows condition must be satisfied in \( spl(T) \).

Definitions for \( e_i \) and \( f_i \) are precisely the same in type \( B \) as they were in type \( C \).

1.4.4 Type \( D_n \)

In type \( D_n \) the Kashiwara-Nakashima Tableaux are as follows[6]:

A column \( C \) is a Young Diagram of column shape filled with letters from \([\overline{n}]\) such that \( C(i+1) \not< C(i) \) for \( i \in [l-1] \) where \( l \) is the length of the column \( C \). The ordering on \([\overline{n}]\) in type \( D \) is as follows:

\[
1 < 2 < \cdots < \overline{n} < \cdots < \overline{2} < 1
\]

The entries \( n \) and \( \overline{n} \) may be repeated in a column, however they must be different in adjacent positions.

As in types \( B \) and \( C \) a column \( C \) is admissible if and only if it splits. The definition however for splitting in type \( D \) is heavily reliant on type \( B \). Particularly we have the following definition for a splitting in type \( D \):
Let $C$ be a column of type $D$. Define $\hat{C}$ to be the column of type $B$ obtained by replacing each factor of $\overline{n}n$ in $C$ with 00. We then say that $C$ splits when $\hat{C}$ splits. Here $lC = l\hat{C}$ and $rC = r\hat{C}$ [6].

**Example 1.4.12.** Let

$$C = \begin{pmatrix}
2 \\
3 \\
\overline{5} \\
\overline{5} \\
3
\end{pmatrix}.$$

Then

$$\hat{C} = \begin{pmatrix}
2 \\
3 \\
0 \\
0 \\
3
\end{pmatrix},$$

$I = \{0, 0, 3\}$, $J = \{5, 4, 1\}$ and

$$lC = l\hat{C} = \begin{pmatrix}
1 \\
2 \\
4 \\
\overline{5} \\
\overline{3}
\end{pmatrix}, \quad rC = r\hat{C} = \begin{pmatrix}
2 \\
3 \\
\overline{5} \\
\overline{4} \\
1
\end{pmatrix}.$$

A tableau $T$ in type $D$ is also defined in the same fashion as in types $B$ and $C$. The columns of $T$ must all split and the weakly increasing across rows condition must be satisfied in $spl(T)$. It is not enough that the columns of a tableaux be weakly increasing across columns but $a$-odd-configurations and $a$-even-configurations of a particular type are not allowed and are defined below.

**Definition 1.4.13.** Let $C_1$ and $C_2$ be admissible columns of type $D$ and $p, q, r, s$ integers such that $1 \leq p \leq q < r \leq s \leq k$. The entries of $C_1$ are $x_1$ through $x_k$, the entries of $C_2$ are $y_1$ through $y_k$.

$C_1C_2$ contains an $a$-odd-configuration (with $a \notin \{n, \overline{n}\}$) when:
• $a = x_p, \overline{a} = x_r$ are letters of $C_1$ and $\overline{a} = y_s, n = y_q$ letters of $C_2$ such that $r - q + 1$ is odd

or

• $a = x_p, n = x_r$ are letters of $C_1$ and $\overline{a} = y_s, \overline{a} = y_q$ letters of $C_2$ such that $r - q + 1$ is odd.

$C_1C_2$ contains an $a$-even-configuration (with $a \in \{ n, \overline{n} \}$) when:

• $a = x_p, n = x_r$ are letters of $C_1$ and $\overline{a} = y_s, n = y_q$ letters of $C_2$ such that $r - q + 1$ is even

or

• $a = x_p, \overline{a} = x_r$ are letters of $C_1$ and $\overline{a} = y_s, \overline{a} = y_q$ letters of $C_2$ such that $r - q + 1$ is even.

Then define $\mu(a)$ to be the positive integer given by $\mu(a) = s - p$.

**Definition 1.4.14.** In type $D$ a tableau is given by $C_1, \ldots, C_r$ admissible columns of type $D$ with $C_i \leq C_{i+1}$ for $1 \leq i < r$ with the additional restriction that $rC_i|C_{i+1}$ does not contain an $a$-configuration (even or odd) such that $\mu(a) = n - a[6]$. Let an $a$-configuration refer to an $a$-odd-configuration or an $a$-even configuration.

**Example 1.4.15.** The following is an example of a pair of columns containing a containing a 3-odd configuration:

\[
\begin{array}{ccc}
3 & 5 \\
4 & 4 \\
5 & 3
\end{array}
\]

Define $e_i$ and $f_i$ as they were in type $C$ for $i < n$. For the case where $i = n$ this is now slightly different in terms of the definition on column words. Consider the column word $w$. Then consider the subword of $w$ consisting of the entries of the form,
$n - 1$, $n$, $n - 1$ or $\overline{n}$ and call this word $w_n$. Denote $n - 1$ or $n$ by a + and denote $\overline{n}$ or $n - 1$ by a −. Adjacent pairs of the form −+ may be ignored. This may be repeated until a subword of the form $\rho(w) = +^r a$ is reached.

If $r > 0$ $f_n(w)$ is obtained by changing the rightmost + to a − and all other letters remain unchanged. This has the effect of changing either an $n$ into an $n - 1$ or changing an $n - 1$ into an $\overline{n}$. If $r = 0$ then $f_n(w) = 0$. Then $e_n(w)$ is defined as the inverse.

Since $f_n$ is the only case in which the behavior differs we give the following example:

**Example 1.4.16.** Begin by considering the following tableau $T$ with split tableau $spl(T)$ in the case where $n = 4$:

\[
T = \begin{array}{c}
2 \\
3 \\
4 \\
4
\end{array} \quad spl(T) = \begin{array}{c}
1 \\
2 \\
3 \\
4 \\
4
\end{array}
\]

These have corresponding column words:

\[
w = 4432 \quad \tilde{w} = 4321 1432.
\]

We then consider $f_4(T)$ and $f_4(spl(T))$. Begin by considering $\rho(w)$ and $\rho(\tilde{w})$.

\[
\rho(w) = + \quad \rho(\tilde{w}) = ++.
\]

So then we have:

\[
f_4(\rho(w)) = - \quad f_4(\rho(\tilde{w})) = -,
\]

yielding

\[
f_4(w) = 4412 \quad f_4(\tilde{w}) = 4321 1342.
\]
This gives us the following for $f_4(T)$ and $f_4(spl(T))$:

\[
\begin{align*}
\begin{array}{c}
2 \\
4 \\
4 \\
4 \\
\end{array}
&= \begin{array}{c}
1 \\
2 \\
4 \\
3 \\
\end{array} \\
\begin{array}{c}
2 \\
4 \\
4 \\
4 \\
\end{array}
&= \begin{array}{c}
1 \\
2 \\
4 \\
3 \\
\end{array}.
\end{align*}
\]
Chapter 2

Type A

2.1 Specializing the Alcove Path Model to $A_{n-1}$

An alcove path can be viewed as a sequence of reflections in walls of alcoves, and therefore be viewed as a sequence of transpositions, in this vein we make the following definitions:

Fix $n$ throughout. We will use the notation $(a, b)$ to refer to the root $\epsilon_a - \epsilon_b$ and the corresponding reflection/transposition. We shall also use the notation $(a, b)_l$ to refer to the affine reflection in the hyperplane given by $H_{\epsilon_a - \epsilon_b, l} = \{ v \in \mathbb{R}^n \mid \langle v, \epsilon_a - \epsilon_b \rangle = l \}$.

Define $\Gamma(k)$ for $k \leq n$ to have the following form:

\[
\begin{align*}
(k, k + 1), & \quad (k, k + 2), \quad \ldots, \quad (k, n), \\
(k - 1, k + 1), & \quad (k - 1, k + 2), \quad \ldots, \quad (k - 1, n), \\
\vdots & \quad \vdots \\
(1, k + 1), & \quad (1, k + 2), \quad \ldots, \quad (1, n) .
\end{align*}
\]

The chain $\Gamma(\lambda)$ is then defined as:

\[
\Gamma(\lambda) = \Gamma_1(\lambda'_1)\Gamma_2(\lambda'_2)\ldots\Gamma_m(\lambda'_m) \quad (2.1.1)
\]
Example 2.1.2. Let $n = 6$ and $k = 3$: Then $\Gamma(k)$ is as follows:

\[
(3,4), (3,5), (3,6), \\
(2,4), (2,5), (2,6), \\
(1,4), (1,5), (1,6)
\]

Lemma 2.1.3. $\Gamma(k)$ is an $\omega_k$-chain.

Proof. We use the criterion for $\lambda$-chains in [11][Definition 4.1, Proposition 4.4], cf. [11][Proposition 10.2]. This criterion says that a chain of roots $\Gamma$ is a $\lambda$-chain if and only if it satisfies the following conditions:

(R1) The number of occurrences of any positive root $\alpha$ in $\Gamma$ is $\langle \lambda, \alpha^\vee \rangle$.

(R2) For each triple of positive roots $(\alpha, \beta, \gamma)$ with $\gamma^\vee = \alpha^\vee + \beta^\vee$, the subsequence of $\Gamma$ consisting of $\alpha, \beta, \gamma$ is a concatenation of pairs $(\alpha, \gamma)$ and $(\beta, \gamma)$ (in any order).

Letting $\lambda = \omega_k = \varepsilon_1 + \ldots + \varepsilon_k$, condition (R1) is easily checked; a root $(a, b)$ appears once in $\Gamma(k)$ if $a \leq k < b$, and zero times otherwise. For condition (R2), we use a case by case analysis, as follows, where $a < b < c$ and $\alpha = (a, b)$, $\beta = (b, c)$ and $\gamma = (a, c)$:

(1) $k \geq c$

(2) $b \leq k < c$

(3) $a \leq k < b$.

Case (1) the subsequence is empty. Case (2) the subsequence is $(b, c)$ followed by $(a, c)$, i.e. $\beta$ followed by $\gamma$. Case (3) the subsequence is $(a, b)$ followed by $(a, c)$, i.e. $\alpha$ followed by $\gamma$. 

30
Definition 2.1.4. We define a $w$-admissible subsequence $A$ to be a subsequence of $\Gamma(\lambda)$ such that it is the labels of the covers of a saturated chain in the Bruhat order of $S_n$ starting at $w$. In the particular case where $w$ is the identity permutation this is merely an admissible subsequence.

A $w$-admissible subsequence may then be represented by using the following construction:

$$\Gamma(\lambda) = (a_1, b_1) \ldots (a_N, b_N)$$

where $a_i < b_i$. Here $N = \sum_i \lambda'_i (n - \lambda'_i)$. The $\sigma$-admissible subsequence $\gamma$ is then viewed as a subset $J := \{j_1 < \ldots < j_s\}$ of $[N]$. So $A = (a_{j_1}, b_{j_1}) \ldots (a_{j_s}, b_{j_s})$.

It is then immediate that a $w$-admissible subsequence, $A$, is an increasing sequence of transpositions extracted from $\Gamma(\lambda)$. We may then think of the transpositions contained in the $w$-admissible subsequence $A$ as being marked or underlined positions in $\Gamma(\lambda)$ as below:

$$(a_1, b_1) \ldots (a_{j_1}, b_{j_1}) \underline{(a_{j_2}, b_{j_2})} \ldots (a_N, b_N).$$

Definition 2.1.5. We shall refer to the underlined positions as foldings from this point onward.

Recall $(a, b)$ is the transposition given by sending the root $e_a - e_b$ to its negative. Recall also $(a, b; l)$, also written $(a, b)_l$, the corresponding affine transposition. Here we shall also think of $(a, b; l)$ as a map $(a, b; l) : \mathbb{Z}^n \to \mathbb{Z}^n$ given by $(a, b; l)(\lambda) = (\lambda_1, \ldots, \lambda_b + l, \ldots, \lambda_a - l, \ldots, \lambda_n)$ for $\lambda \in \mathbb{Z}^n$. We see that we may write the chain $A$ as a product of transpositions $A = (a_{j_1}, b_{j_1}) \ldots (a_{j_s}, b_{j_s})$.

We now attach a notion of level to the chain, this is merely a specialization to the case of type $A$:
Specialization 2.1.6. The level $l_i$ of $(a_i, b_i)$ in column $j$ is given by:

$$l_i = \left| \{ k < j \mid \lambda'_k < b_i \} \right|.$$ 

Note that $l_i$ is merely the number of columns of length less than $b_i$, or the number of times that transposition has occurred previously, that is

$$l_i = |\{ j < i : (a_j, b_j) = (a_i, b_i) \}|.$$

Example 2.1.7. Consider the tableau $T$ with $\lambda' = (3, 2, 2)$ and $n = 4$, then the associated chain with levels is as follows:

$$(3, 4)_0 (2, 4)_0 (1, 4)_0 (2, 3)_0 (2, 4)_1 (1, 3)_0 (1, 4)_1 (2, 3)_1 (2, 4)_2 (1, 3)_1 (1, 4)_2$$

Letting $\alpha_i$ be the root associated to $(a_i, b_i)$, allows for viewing $A$ as the chain of roots $\alpha_j_1...\alpha_j_κ$.

We define $\beta_i$ to be the root given by $\beta_i = (a_{j_1}, b_{j_1})...(a_{j_k}, b_{j_k})(\alpha_i)$ where $k$ is the largest such that $j_k < i$. Just as $\alpha_i$ was associated to $(a_i, b_i)$ associate $\beta_i$ with $(c_i, d_i)$, i.e. $\beta_i = \epsilon_{c_i} - \epsilon_{d_i}$ and $(c_i, d_i)$ is the transposition sending $\beta_i$ to its negative. We may view $\alpha_i$ and $\beta_i$ in terms of unfolded and folded chains respectively. We then will have levels $l_i$ associated to the unfolded chains and levels $m_i$ associated to the corresponding folded chain. We define $s_{j_i} = (a_{j_i}, b_{j_i}; l_{j_i})...(a_{j_k}, b_{j_k}; l_{j_k})...(a_{j_1}, b_{j_1}; l_{j_1})$. The object to be considered is the hyperplane $(a_{j_1}, b_{j_1}; l_{j_1})...(a_{j_k}, b_{j_k}; l_{j_k})(H_{\alpha_i, t_i}) = s_{j_k}...s_{j_1}(H_{\alpha_i, t_i}) = H_{\beta_i, m_i}$. We shall wish to observe the effect of each $s_{j_i}$ on the previous hyperplane. In a later lemma we shall find that $m_i$ may be found easily by a straightforward counting at the tableaux level.

The weight of a admissible subsequence $A$, $\mu(\mathcal{A})$, after specializing our context,
is given by:

\[
\mu(\mathcal{A}) = (a_1, b_1)_{t_1} \cdots (a_s, b_s)_{t_s}(\lambda).
\]

This comes from the fact that \( \mu = -\hat{r}_1 \cdots \hat{r}_s(-\lambda) \).
2.2 The Bijection Between Admissible Subsequences and Tableaux

The goal of this section is to find a bijection between admissible subsequences and tableau. We begin by giving a filling map that takes us from admissible subsequences to tableaux and then construct an inverse to this filling map.

2.2.1 The Filling Map

Consider the partition $\lambda$ and consider $A$. We may break this down as follows:

$A = A_1A_2 \ldots A_m$

Here $A_i$ is the portion of $A$ associated to $\Gamma_i(\lambda'_i)$ in

$\Gamma(\lambda) = \Gamma_1(\lambda'_1)\Gamma_2(\lambda'_2) \ldots \Gamma_m(\lambda'_m)$

We then consider the following:

$\sigma_0 \xrightarrow{A_1} \sigma_1 \xrightarrow{A_2} \ldots \xrightarrow{A_m} \sigma_m$

where $\sigma_0$ is the identity permutation and $A_i$ is a $\sigma_{i-1}$-admissible subset. Here $\sigma_i$ is given by $\sigma_i = \sigma_{i-1}(a_{j_1}, b_{j_1}) \ldots (a_{j_s}, b_{j_s})$ where $A_i = (a_{j_1}, b_{j_1}) \ldots (a_{j_s}, b_{j_s})$.

Let $C_i$ be column $i$ of the tableau $T$ of shape $\lambda$.

We then have the filling map given by setting the entries of $C_i$ equal to $\sigma_i[k]$ where $\sigma_i[k]$ is the restriction of $\sigma_i$ to the first $k$ positions.
2.2.2 An Inverse to the Filling Map

The primary goal of this subsection is to prove that the filling map is in fact a bijection. This will be done by constructing an explicit inverse to the filling map. We begin with the theorem we wish to prove.

Theorem 2.2.1. The filling map is a bijection between semi-standard Young tableaux of shape \( \lambda \) with entries in \([n]\) and admissible subsequence in \( S_n \).

We shall first need the following lemma.

Lemma 2.2.2. For \( i \leq k < j \leq n \), \( \pi(j) = b \) and \( \pi(l) > b \) for \( i < l \leq k \), there exists a unique sequence \( k < j_1 < \ldots < j_p = j \) such that

\[
\ell(\pi(i,j_1)\ldots(i,j_r)) = \ell(\pi(i,j_1)\ldots(i,j_{r-1})) + 1 \quad \text{for} \quad 0 < r \leq p 
\]  

(2.2.3)

Proof. We shall prove this by giving an algorithm that produces the unique sequence of transpositions of the given form. We begin by showing existence. Define \( j_0 = i \). Let \( j_1 > i \) be the first position such that \( \pi(j_0) < \pi(j_1) \leq b \). We see that \( \ell(\pi_{i,j_1}) = \ell(\pi) + 1 \) since by construction either \( \pi(l) < \pi(j_0) \) or \( \pi(l) > b \) for \( i < l < j_1 \). We then repeat this process with \( \pi(i,j_1) \) finding \( j_2 \) using \( j_1 \) instead of \( j_0 \) and then repeat as necessary, and since \( j - k \) is finite this process will terminate. It is not hard to see that the sequence of \( j_1 \)'s will be increasing. To show uniqueness, we need to show that if \( k < l_1 < \ldots < l_q = j \) is another such sequence then \( j_1 = l_1 \) (here we shall let \( l_0 = i \) as well). Once this has been shown the same argument may be iterated so that we obtain \( p = q \) and \( j_m = l_m \) for \( 1 \leq m \leq p \). Thus we examine the situation for \( j_1 \) and assume that \( l_1 \neq j_1 \). We first note that \( l_1 \geq j_1 \), otherwise length would decrease. There exists a position \( m \) such that \( \pi(l_{m-1}) < \pi(j_1) \) and \( \pi(l_m) > \pi(j_1) \). We then observe that \( \ell(\pi(i,j_1)\ldots(i,j_m)) > \ell(\pi(i,j_1)\ldots(i,j_{m-1})) + 1 \). But this violates the condition that the length increase by exactly one.
The following algorithm serves as the foundation of constructing an inverse map to the filling map:

**Algorithm 2.2.4.** set $\rho = \pi$

set $i = k$

while $i \geq 1$ do

set $j = k + 1$

while $j < n$ do

if ($\rho(j) > \rho(i)$ and $\rho(j) \leq C'(i)$)

return $\rho = \rho(i, j)$

end if

set $j = j + 1$

end while

set $i = i - 1$

end while

Here for a particular column $C_i$ we have $\pi = \sigma_{i-1}$ and $\rho = \sigma_i$ and the transpositions selected in the process of the algorithm correspond to $A_i$.

\[\square\]

**Proof.** (of Theorem 2.2.1) We now find a unique admissible subsequence associated to a tableaux $T$ by giving an explicit algorithm, resulting in an explicit inverse to the above defined map from admissible subsequences to tableaux. Here we assume $T$ has entries in the set $[n]$. We shall begin by observing the shape of the conjugate tableaux $\tilde{T}$, that is by noting the length of each column given by $(\lambda'_1, ..., \lambda'_m)$. Begin with the first column of length $\lambda_1$ and with the sequence $s = (1, 2, ..., n)$ and using the above lemma find the unique chain that replaces the $\lambda^{st}_1$ entry of $s$ with the associated value of that box in the tableaux $T$. We then repeat this for the $\lambda_1 - 1$ entry. It suffices to check that this does not effect the entries further down the column, which is clear
since transpositions of the type would violate the above lemma, in particular the length condition would not be preserved. Then repeat this throughout the column until the appropriate entry is in each box of the column, giving a subsequence $A_i$. Then repeat this procedure on the remaining columns, resulting in a subsequence $A = A_1...A_m$ (we note that it is possible for $A_i$ to be the empty subsequence). Due to the uniqueness in each step we have that this in fact a well define inverse, thus establishing the claim that the map from admissible subsequences to tableaux is in fact a bijection.

Let us also define $T_i$ to be the tableaux corresponding, via the bijection given the filling map, to the first $i$ terms of the admissible subsequence associated to $T$ having been applied, so $T_0$ would be the tableau with the entries in each row being the number of that row and $(a_i, b_i)(T_i)$ is the result of replacing the entries $a_i$ with the entry $b_i$ in tableau $T_{i-1}$ from column $q$ onward, where $(a_i, b_i)$ is in the column or block $q$ of the $w$-admissible subsequence $A$ associated to tableau $T$.

**Example 2.2.5.** Consider $\lambda = (4, 3, 1)$


The tableau to which $A$ maps would then be the rightmost tableau above. It can also be easily seen in this example how the map from tableaux to chains is constructed.
2.3 A Level Counting Formula For Type A

Our goal in this section is to produce a formula for the levels $m_i$ as this will be critical in the next section where we prove that the bijection given preserves the crystal graph structure.

We shall require the following well known formula for reflections of hyperplanes which we shall reference in the proof of the lemma below:

$$t_{a,k}(H_{\beta,l}) = H_{t_{a,k}(\beta,l-k)}; \langle (a,b), (c,d) \rangle = \langle e_a - e_b, e_c - e_d \rangle; \langle e_i, e_j \rangle = \delta_{ij}. \tag{2.3.1}$$

Recall the $\lambda$-chain $\Gamma$ and let us write $\Gamma = (\beta_1, \ldots, \beta_m)$. As such, we recall the hyperplanes $H_{\beta_k,l_k}$ and the corresponding affine reflections $\widehat{r}_k = s_{\beta_k,l_k}$. If $\beta_k = (a,b)$ falls in the segment $\Gamma_p$ of $\Gamma$ (upon the factorization $\Gamma = \Gamma_1 \ldots \Gamma_{\lambda_1}$ of the latter), then it is not hard to see that

$$l_k = |\{i : 1 \leq i < p, \lambda_i' \geq a\}|.$$

Let $T = ((a_1,b_1), \ldots, (a_s,b_s))$ be the subsequence of $\Gamma$ indexed by the positions in $J$. Let $T^i$ be the initial segment of $T$ with length $i$, let $w_i := wT^i$, and $\sigma_i := f(w, T^i)$. In particular, $\sigma_0$ is the filling with all entries in row $i$ equal to $w(i)$, and $\sigma := \sigma_s = f(w, T)$. The columns of a filling of $\lambda$ are numbered, as usual, from left to right by $\lambda_1$ to $\lambda_l$. Note that, if $\beta_{i+1} = (a_{i+1}, b_{i+1})$ falls in the segment $\Gamma_p$ of $\Gamma$, then $\sigma_{i+1}$ is obtained from $\sigma_i$ by replacing the entry $w_i(a_{i+1})$ with $w_i(b_{i+1})$ in the columns $p, \ldots, \lambda_1$ (and the row $a_{i+1}$) of $\sigma_i$.

Given a fixed $k$, let $\beta_k = (a,b)$, $c := w_i(a)$, and $d := w_i(b)$, where $i = i(k)$ is defined as above. Then $\gamma := \gamma_k = (c,d)$, where we might have $c > d$. Recall $\beta$ correspond to the root prior to folding, $\gamma$ to the root after folding. Let $\Gamma_q$ be the segment of $\Gamma$ where $\beta_k$ falls. Given a filling $\sigma$, we denote by $\sigma(p)$ and $\sigma[p,q]$ the parts
of $\sigma$ consisting of the columns $1, 2, \ldots, p - 1$ and $p, p + 1, \ldots, q - 1$, respectively. We use the notation $N_e(\sigma)$ to denote the number of entries $e$ in the filling $\sigma$.

**Proposition 2.3.2.** With the above notation, we have

$$m_k = \langle \text{ct}(\sigma(q)), \gamma^\vee \rangle = N_c(\sigma(q)) - N_d(\sigma(q)).$$

**Proof.** We apply induction on $i$, which starts at $i = 0$. We will now proceed from $j_1 < \ldots < j_i < k$, where $i = s$ or $k \leq j_{i+1}$, to $j_1 < \ldots < j_{i+1} < k$, and we will freely use the notation above. Let

$$\beta_{j_{i+1}} = (a', b'), \quad c' := w_i(a'), \quad d' := w_i(b').$$

Let $\Gamma_p$ be the segment of $\Gamma$ where $\beta_{j_{i+1}}$ falls, where $p \geq q$.

We need to compute

$$w_{\hat{r}_{j_1}} \cdots \hat{t}_{j_{i+1}}(H_{\beta_k, \ell_k}) = \hat{t}_{j_{i+1}} \cdots \hat{t}_{j_1} w(H_{\beta_k, \ell_k}) = \hat{t}_{j_{i+1}}(H_{\gamma, m}),$$

where $m = \langle \text{ct}(\sigma_i(q)), \gamma^\vee \rangle$, by induction. Note that $\gamma' := \gamma_{j_{i+1}} = (c', d')$, and $\hat{t}_{j_{i+1}} = s_{\gamma', m'}$, where $m' = \langle \text{ct}(\sigma_i(p)), (\gamma')^\vee \rangle$, by induction. We will use the following formula:

$$s_{\gamma', m'}(H_{\gamma, m}) = H_{s_{\gamma'}(\gamma), m - m'\langle \gamma', \gamma^\vee \rangle}.$$

Thus, the proof is reduced to showing that

$$m - m'\langle \gamma', \gamma^\vee \rangle = \langle \text{ct}(\sigma_{i+1}(q)), s_{\gamma'}(\gamma^\vee) \rangle.$$

An easy calculation, based on the above information, shows that the latter equality
is non-trivial only if $p > q$, in which case it is equivalent to

$$\langle \text{ct}(\sigma_{i+1}[p, q]) - \text{ct}(\sigma_i[p, q]), \gamma' \rangle = \langle \gamma', \gamma' \rangle \langle \text{ct}(\sigma_{i+1}[p, q]), (\gamma')' \rangle. \quad (2.3.3)$$

This equality is a consequence of the fact that

$$\text{ct}(\sigma_{i+1}[p, q]) = s_{\gamma'}(\text{ct}(\sigma_i[p, q])), $$

which follows from the construction of $\sigma_{i+1}$ from $\sigma_i$ explained above.

What needs to be taken away from this is that the level $m$ for the transposition $(a, b)_m$ in column $p$ is simply the number of occurrences of $a$ minus the number of occurrences of $b$ in columns prior to $p$. \qed
2.4 The Crystal Graph Structure and Root Operators

In this section we use the level counting formula from the previous section to show that the bijection given between admissible subsequences and tableaux preserves crystal graph structure. This will come down to showing that the root operators $E_i$ and $F_i$ commute with the bijection for $1 \leq i < n$. We will need to determine what $\sigma$-admissible subsequences look like for an individual column when restricted to $i$ and $i+1$ and then use this to compare the operator $F_i$'s effect in the context of tableaux(particularly column words) and in the context of admissible subsequences. We will then merely observe that the effect is in fact the same establishing the result.

We must however first define $E_i$ and $F_i$ in the context of the alcove path model.

Define $\Gamma(J)^i$ to be the portion of $\Gamma(J)$ consisting of only the transpositions which exchange the values $i$ and $i+1$, that is of the form $(i,i+1)$ or $(i,i+1)$ or $(i+1,i)$. Where $(i,i+1)$ represents a position of a folding, such a position will be referred to as marked.

Write $\Gamma(J)^i = (a_1,b_1) ... (a_{m},b_{m})$ and $J \subseteq [m]$ be the subset of marked indices $J = \{i_1 < ... < i_k\}$. So then $\Gamma(J)^i = (a_1,b_1) ... (\overline{a_{i_1},b_{i_1}}) ... (\overline{a_{i_k},b_{i_k}}) ... (a_{m},b_{m})$.

Let $p$ be the final position in $[m] \setminus J$ in column $j-1$, that is the last unmarked position in column $j−1$ where $j$ is the column in which the highest level occurs.

We now specialize the definition of $F_i$ to the context of type $A$.

**Specialization 2.4.1.** In the context of admissible subsequences $F_i$ is defined as follows:

$$F_i(\Gamma(J)^i) = (a_1,b_1) ... (\overline{a_{i_1},b_{i_1}}) ... (a_p,b_p)(a_{p+1},b_{p+1}) ... (\overline{a_{i_k},b_{i_k}}) ... (a_{m},b_{m}),$$

i.e. the position before the highest level becomes marked and the position of the highest
level ceases to be marked. In the case where the highest level is never reached we have

\[ F_i(\Gamma(J)^i) = (a_1, b_1)...(a_{i_k}, b_{i_k})...(a_m, b_m). \]

i.e. the final position becomes marked.

We define \( E_i \) similarly, however for our purposes we shall need only \( F_i \).

We begin by defining the function \( g(i) \). Start at \((0, -1/2)\). Then \((i, i+1), (i+1, i)\), and \((i, i+1)\) are represented as follows:

- \((i, i+1)\) by a linear segment going up by one and to the right by one (up-up)
- \((i+1, i)\) by a linear segment going down by one and to the right by one (down-down)
- \((i, i+1)\) by a linear segment going up a half, right a half, followed by a linear segment going down a half, to the right by a half (up-down).

**Theorem 2.4.2.** The bijection between semi-standard Young tableaux of shape \( \lambda \) with entries in \([n]\) and admissible subsequences in \( S_n \) commutes with the root operators \( E_i \) and \( F_i \).

*Proof.* The goal will be to show that \( F_i \) commutes with the bijection as \( E_i \) is merely the inverse of \( E_i \). This will break into two cases, one where the highest level actually occurs. The second where the highest level does not but rather the final column ends with only an \( i \). The proof for the first case has the following structure assuming \( q \) is the column where the highest level occurs:

1. Observe that the level \( m_i \) does not change on a column.

2. The first highest level occurs in a column \( q \) after a column \( p \) containing only an \( i \).
3. The $i$ in the column $p$ is the final $i$ in the $\rho(w)$ after pairing. That is the $i$ which is changed to an $i+1$ by $F_i$.

4. $\Gamma(\lambda)^i$ restricted to columns $p+1, \ldots, q-1$ is empty.

5. The entries $i$ and $i+1$ do not occur in columns $p+1, \ldots, q-1$.

6. When $\Gamma(J)^i$ is considered restricted to column $p$ it ends in a $(i, i+1)$ and $\Gamma(J)^i$ restricted to column $q$ begins with $(i, i+1)$.

7. When $F_i$ changes the $i$ in column $p$ to an $i+1$ in the column word context this makes the final $(i, i+1)$ in column $p$ an $(i, i+1)$ as well as changing the $(i, i+1)$ in column $q$ to a $(i+1, i)$ in the context of $\Gamma(J)^i$.

8. Since $F_i$ has the same effect in both contexts $F_i$ commutes with the bijection and therefore preserves the crystal graph.

Now we fill in the details. It is first of all clear that the level $m_i$ does not change in a given column because $m_i$ is given by the counting formula in terms of only the entries in the previous columns.

To verify that the first highest level in column $q$ implies that column $p$ contains an $i$ and no $i+1$ is straightforward as this is the only way for the level in a column which follows to be greater. If there is no $i$ the level either remains the same if there is no $i+1$ or decreases by 1 if there is an $i+1$.

To verify that the $i$ in the column $p$ is the final $i$ in the $\rho(w)$ after pairing we go back to the counting formula again. Merely observe that $i+1$ followed by an $i$ results in a net change of 0 as far as $m_i$ is concerned. This corresponds to the pairings and cancellation of $i+1$ followed by $i$ in the pairing process.

To verify that when $\Gamma(J)^i$ is considered restricted to column $p$ it ends in a $(i, i+1)$ we consider the graph $g(i)$ mentioned above. What we are looking for is the first highest peak of this graph. First off the peak must be of the form $(i, i+1)$ or it isn’t
a peak. In order for this to be the first highest peak the end of the first nonempty column prior to \( q \) must then be of the form \((i, i+1)\) corresponding to an up-up on \( g(i) \). For either of the other two possibilities the level \( m_i \) of column \( q \) will have been achieve either in \( p \) (perhaps even an earlier column.) Note also that there is no other \((i, i+1)\) in this column or there would be an increase in level within the column \( p \). This also establishes that \( \Gamma(\lambda)^i \) restricted to column \( q \) begins with \((i, i+1)\).

To verify that \( \Gamma(J)^i \) restricted to columns \( p + 1, \ldots, q - 1 \) is empty merely note that an \((i, i+1)\) cannot be followed by an \((i+1, i)\) and if the next transposition that occurs in \( \Gamma(J)^i \) is of the form \((i, i+1)\) then the highest level has not actually been reached. This gives us that the next transposition that occurs in \( \Gamma(J)^i \) is of the form \((i, i+1)\) which is in a later column by the counting formula and must therefore be in column \( q \) as this is where the first highest level occurs.

To verify that the entries \( i \) and \( i+1 \) do not occur in columns \( p + 1, \ldots, q - 1 \) note that if the level does not change then either both \( i \) and \( i+1 \) occur or neither. We are only interested in the case where both occur. If both did occur then the effect of \( F_i \) would be to trade the positions of \( i \) and \( i+1 \) in columns \( p + 1, \ldots, q - 1 \). This however is not possible as \( F_i \) acting on a valid tableau produces another valid tableau and if there were an \( i \) and an \( i+1 \) in a column then they would be out of order in that column after \( F_i \) has been applied.

To verify that when \( F_i \) changes the \( i \) in column \( p \) to an \( i+1 \) in the column word context this makes the final \((i, i+1)\) in column \( p \) an \((i, i+1)\) in the context of \( \Gamma(J)^i \) we merely examine what these mean. First changing an \( i \) to an \( i+1 \) corresponds to selecting an \((i, i+1)\) transposition that was not previously selected, i.e. making it underlined. Since this is the \( i \) in column \( q - 1 \) as shown earlier this must make the final \((i, i+1)\) of column \( p \) an \((i, i+1)\). To observe that the \((i, i+1)\) of column \( q \) becomes a \((i+1, i)\) note that the contribution of \( F_i \) in terms of levels for columns \( q \) onward is that these levels are 2 less than they were previously as an \( i \) has been
changed to an $i + 1$ so where once a $+1$ was counted now a $−1$ is counted resulting in a level decrease of two. The only way this can possible happen is if the restriction of $\Gamma(J)^i$ to column $q$ begins with a $(i + 1, i)$.

It is then immediately clear that $F_i$ has the same result on both sides of the bijection, therefore $F_i$ commutes with the bijection, establishing the claim.

Now for the case where the no highest level is actually reached. This is the case only when there is an $i$ in a column of the tableau after which there are no $i$’s or $i + 1$’s and it is not cancelled by a previous $i + 1$. In this case this $i$ will be the final term of $\rho(i)$ in which case $F_i$ makes this an $i + 1$. Alternatly this is a final $(i, i + 1)$ of $\Gamma(J)^i$ in which case $F_i$ makes this an $(i, i + 1)$. Thus the bijection holds in this case as well.

We thus have the following immediate corollary to the above theorem:

**Theorem 2.4.3.** The bijection between semi-standard Young tableaux of shape $\lambda$ with entries in $[n]$ and $w$-admissible subsequences in $S_n$ preserves the crystal graph structure for Young tableaux of shape $\lambda$ with entries in $[n]$.

As an immediate corollary to this we have the following:

**Corollary 2.4.4.** The bijection between semi-standard Young tableaux of shape $\lambda$ with entries in $[n]$ and $w$-admissible subsequences in $S_n$ preserves weight for Young tableaux of shape $\lambda$ with entries in $[n]$.

*Proof.*

**Example 2.4.5.** Consider the case where $\lambda = (4, 3, 1)$ with tableau as below:

```
1 1 2 5
3 4 4
4 5
```

In this case the admissible subsequence after foldings is
\[
\begin{align*}
\end{align*}
\]
\[
\]
\[
\Gamma(\lambda)^4 \text{ in this case is } (45)||&(45)||&(45)
\]

The column word in this case is 45445, when restricted to 4 and 5, which becomes 445 after pairing.

**Example 2.4.6.** Here we give the graph of \(g(4)\) based on the previous example.

![Graph of g(4)](image)

The levels \(m_4\) are then easily read off of the graph of \(g(4)\) above to be, 0 in the first column, 1 in the second column, 1 in the third, and 2 in the fourth.
2.5 An Example

The example in this section will serve as an example of the correspondence between alcove paths, admissible subsequences and tableaux. This is done here as the type $A$ case is the most easily visualized due to the fact that we can visualize the alcove walks in only two dimensions in the case where $n = 3$. We start with the following alcove path prior to folding.

Associated with this alcove walk is the chain of roots given below:

$$(2, 3)_0 (1, 3)_0 (2, 3)_1 (1, 3)_1 (1, 2)_0 (1, 3)_2 .$$

The transpositions can be read off by starting at the fundamental alcove at the beginning of the alcove path towards the top of the diagram. These are read off by observing which alcove wall is being crossed by the alcove path from start to end.
Note that the first wall crossed is the wall orthogonal to the root $\alpha_{2,3}$ and it is the first wall crossed in that direction, consequently the level belonging to this is 0. So the root is read off by giving the root orthogonal to the wall crossed and the level is given by counting how many walls in that direction from the corresponding wall on the fundamental alcove. Do this from the beginning of the alcove walk to the end listing each according to this scheme working towards the end. For another example consider the last root in the chain above, this corresponds to crossing the wall orthogonal to the root $\alpha_{1,3}$ and this is the second wall orthogonal to $\alpha_{1,3}$ from the fundamental alcove.

Also observe that the weight $\lambda$ of the chain may be read off by reading the weight $-\lambda$ at the base of the alcove at which the alcove path ends. In the case of the example above $\lambda = (3, 2, 0)$.

The tableau associated with this example is the following:

$$T = \begin{array}{ccc}
1 & 1 & 1 \\
2 & 2 \\
\end{array}$$

Observe that the chain of roots associated to this tableau is precisely the same as that of the alcove path above.
We shall now consider the same alcove path only with two foldings introduced, one at the third wall and one at the sixth wall. This yields the following picture:

This yields the following chain of roots after the foldings:

\((2,3)_0 (1,3)_0 (2,3)_1 (1,2)_0 (1,3)_1 (1,2)_1\).

The underlined transpositions correspond to where the foldings occurred. The tableau associated to this is the following:

\[
T = \begin{array}{ccc}
1 & 1 & 2 \\
2 & 3
\end{array}
\]
The weight $\lambda$ of the alcove path is $\lambda = (2, 2, 1)$. This is also the same as the weight of the tableau, easily read by reading the content of the filling of the tableau, i.e. counting the number of ones in the filling, the number of twos and so on.
Chapter 3

Type $C$

3.1 Specializing the Alcove Path Model to $C_n$

We shall fix $n$ from this point onward.

Define an $\Gamma^l_i(k)$ for $i \leq k$ chain to have the following form:

$((i,k+1), (i,k+2), \ldots, (i,n), (i,i), (i,n), (i,n-1), \ldots, (i,k+1), (i,i-1), (i,i-2), \ldots, (i,1))$.

$\Gamma^l_i(k)$ is then defined as

$\Gamma^l_i(k) = \Gamma^l_i(k)\Gamma^l_{i-1}(k)\ldots\Gamma^l_1(k)$

Define an $\Gamma^r_i(k)$ chain to have the following form:
The chain \( \Gamma(k) \) is then defined as:

\[
\Gamma(k) = \Gamma^l(k) \Gamma^r(k).
\] (3.1.1)

**Example 3.1.2.** For \( k = 3 \) and \( n = 5 \) we have the following:

\[
\Gamma^l(k) = (3, 4) \ (3, 5) \ (3, 3) \ (3, 5) \ (3, 4) \ (3, 2) \ (3, 1),
\]

\[
(2, 4) \ (2, 5) \ (2, 2) \ (2, 5) \ (2, 4) \ (2, 1),
\]

\[
(1, 4) \ (1, 5) \ (1, 1) \ (1, 5) \ (1, 1).
\]

and

\[
\Gamma^r(k) = (3, 2) \ (3, 1) \ (2, 1).
\]

Recall that \( \lambda \) represents a partition and \( \lambda' \) the conjugate partition. The chain \( \Gamma(\lambda) \) is then defined as:

\[
\Gamma(\lambda) = \Gamma_1(\lambda'_1) \Gamma_2(\lambda'_2) \ldots \Gamma_m(\lambda'_m),
\] (3.1.3)

and breaks down as

\[
\Gamma(\lambda) = \Gamma^l_1(\lambda'_1) \Gamma^r_1(\lambda'_1) \Gamma^l_2(\lambda'_2) \Gamma^r_2(\lambda'_2) \ldots \Gamma^l_m(\lambda'_m) \Gamma^r_m(\lambda'_m).
\] (3.1.4)
Lemma 3.1.5. $\Gamma(k)$ is an $\omega_k$-chain.

Proof. We use the criterion for $\lambda$-chains in [11][Definition 4.1, Proposition 4.4], cf. [11][Proposition 10.2]. This criterion says that a chain of roots $\Gamma$ is a $\lambda$-chain if and only if it satisfies the following conditions:

(R1) The number of occurrences of any positive root $\alpha$ in $\Gamma$ is $\langle \lambda, \alpha^\vee \rangle$.

(R2) For each triple of positive roots $(\alpha, \beta, \gamma)$ with $\gamma^\vee = \alpha^\vee + \beta^\vee$, the subsequence of $\Gamma$ consisting of $\alpha, \beta, \gamma$ is a concatenation of pairs $(\alpha, \gamma)$ and $(\beta, \gamma)$ (in any order).

Letting $\lambda = \omega_k = \varepsilon_1 + \ldots + \varepsilon_k$, condition (R1) is easily checked; for instance, a root $(a, b)$ appears twice in $\Gamma(k)$ if $a < b \leq k$, once if $a \leq k < b$, and zero times otherwise. For condition (R2), we use a case by case analysis, as follows, where $a < b < c$:

1. $\alpha = (a, b)$, $\beta = (b, c)$, $\gamma = (a, c)$;
2. $\alpha = (a, b)$, $\beta = (b, \overline{c})$, $\gamma = (a, \overline{c})$;
3. $\alpha = (a, c)$, $\beta = (b, \overline{c})$, $\gamma = (a, \overline{b})$;
4. $\alpha = (b, c)$, $\beta = (a, \overline{c})$, $\gamma = (a, \overline{b})$;
5. $\alpha = (a, b)$, $\beta = (b, \overline{b})$, $\gamma = (a, \overline{a})$;
6. $\alpha = (a, \overline{a})$, $\beta = (b, \overline{b})$, $\gamma = (a, \overline{b})$.

Case (1) is the type $A$ case. Each of the cases (2)-(4) has the following three sub cases: $k \geq c$, $b \leq k < c$, and $a \leq k < b$, while each of the cases (5)-(6) has the following two sub cases: $k \geq b$, and $a \leq k < b$. For instance, if $b \leq k < c$ in Case (3), then the subsequence of $\Gamma(k)$ consisting of $\alpha, \beta, \gamma$ is $((a, \overline{b}), (a, c), (a, \overline{b}), (b, \overline{c}))$. □

Example 3.1.6. Consider the case of a column of length $k = 3$ and entries taken from $[\overline{n}]$ where $n = 4$. In this case $\Gamma^l(3)$ and $\Gamma^r(3)$ have the following forms:
Definition 3.1.7. We then define a \( w \)-left admissible subsequence \( \mathcal{A}^l \) to be a subsequence of \( \Gamma^l(k) \) such that it is the labels of the covers of a saturated chain in the Bruhat order of \( C_n \) starting at \( w \). We denote the mentioned saturated chain in the Bruhat order by \( w \xrightarrow{\mathcal{A}^l} w' \), where \( w' \) is the permutation where the chain ends. We shall identify admissible subsequences with corresponding chains called admissible chains. Similarly define a \( w \)-right admissible subsequence \( \mathcal{A}^r \) to be a subsequence of \( \Gamma^r(k) \) such that it is the labels of the covers of a saturated chain in the Bruhat order of \( C_n \) starting at \( w \).

Definition 3.1.8. Define a \( w \)-admissible subsequence \( \mathcal{A} \) to be a subsequence of \( \Gamma(\lambda) \) such that it is the labels of the covers of a saturated chain in the Bruhat order of \( C_n \) starting at \( w \). In the particular case where \( w \) is the identity permutation we shall just say admissible subsequence.

3.2 The Filling Map

Our primary goal at this point shall be to create a bijection from admissible sequences to KN tableaux. We begin by examining the left portion of the chain.

Suppose we have an admissible subsequence which is a subsequence of

\[
\Gamma(\lambda) = \Gamma^l_1(\lambda')_1 \Gamma^r_1(\lambda')_1 \Gamma^l_2(\lambda')_2 \Gamma^r_2(\lambda')_2 \ldots \Gamma^l_m(\lambda')_m \Gamma^r_m(\lambda')_m.
\]  

Then we may view this as having the following splitting on the corresponding Bruhat chain:

\[
\begin{align*}
\text{id} & \rightarrow \mathcal{A}^l_0 \rightarrow w^l_1 \rightarrow \mathcal{A}^r_1 \rightarrow w'^l_1 \rightarrow \mathcal{A}^l_1 \rightarrow w^l_2 \rightarrow \mathcal{A}^r_2 \rightarrow \ldots \rightarrow \mathcal{A}^r_m \rightarrow w'^r_m.
\end{align*}
\]  

Then for \( 1 \leq i \leq m \) the column \( lC_i = w^l_i[\lambda'_i] \) and \( rC_i = w'^r_i[\lambda'_i] \). This will provide the desired filling map from admissible sequences to doubled KN tableaux. From this
we have a filling of the split tableau $spl(T)$ as follows:

Consider the column $i$ of $spl(T)$, call it $D_i$, the for $i$ odd $D_i = lC_{i+1}$ and for $i$ even $D_i = lC_i$.

### 3.3 A Partial Inverse to the Filling Map

Our goal is to now create an inverse to the above map, i.e. a map from KN tableaux to admissible sequences. We begin by creating the inverse for left admissible subsequences and will address how to do this for right admissible subsequences in the following section.

**Theorem 3.3.1.** Given the pair $(w, C')$, where $C = w[n] \leq C'$ are Kashiwara-Nakashima columns in type $C_n$ and $w$ a signed permutation, there exists a unique $w$-left admissible subsequence $A^l$ from $w$ to $w'$ such that $w'[k] = C'$.

This is done via an algorithm which explicitly constructs said $\sigma$-left admissible subsequence.

**Algorithm 3.3.2.** set $\pi = \sigma$;  
set $i = k$;  
while $i \geq 1$ do  
exchange($k + 1, n$);  
if ($\pi(\bar{i}) > \pi(i)$ and $\pi(\bar{i}) \leq C'(i)$)  
return $\pi = \pi(i, \bar{i})$;  
end if;  
exchange($\bar{i}, k + 1$);  
exchange($i - 1, 1$);
set $i = i - 1$;
end while;

exchange$(a, b)$;

set $j = a$;

while $j < b$ do

if ($\pi(j) > \pi(i)$ and $\pi(j) \leq C'(i)$)

return $\pi = \pi(i,j)$;

end if

set $j = j + 1$
end while

**Example 3.3.3.** Consider the left admissible subsequence associated to the left column $C$ given below:

$C = \begin{bmatrix} 1 \\ 3 \\ 2 \\ 4 \\ 3 \\ 1 \end{bmatrix}$

In this case $w$ is the identity permutation. The admissible subsequence shall be given in two forms, before and after foldings:

**Before Folding:**

$(3, 4)(3, 3)(3, 4)(3, 2)(3, 1)(2, 4)(2, 3)(2, 1)(2, 1, 4)(1, 4)(1, 1)$

**After Folding:**


In order to prove that the above algorithm produces the desired bijection we shall
first need a few lemmas similar to those used in the case of type $A$.

**Lemma 3.3.4.** For $i \leq k < j \leq n$, $\pi(i) = a$, $\pi(j) = b$ and $\pi(l) \notin [a,b]$ for $i < l \leq k$, there exists a unique sequence $k < j_1 < \ldots < j_p = j$ such that

$$\ell(\pi, j_1)(i, j_r) = \ell(\pi, j_1)(i, j_{r-1}) + 1 \quad \text{for} \quad 1 \leq r \leq p \quad (3.3.5)$$

**Proof.** This particular lemma is nearly identical to that in type $A$ and the proof is the same. □

**Lemma 3.3.6.** For $i \leq k < j \leq n$, $\pi(i) = a$, $\pi(j) = b$ and $\pi(l) \notin [a,b]$ for $i < l \leq n$, there exists a unique sequence $\overline{i} < \overline{j}_1 < \ldots < \overline{j}_p = \overline{j}$ such that

$$\ell(\pi, \overline{j}_1)(i, \overline{j}_r) = \ell(\pi, \overline{j}_1)(i, \overline{j}_{r-1}) + 1 \quad \text{for} \quad 1 \leq r \leq p \quad (3.3.7)$$

**Lemma 3.3.8.** For $i \leq k \leq n, j < k$, $\pi(i) = a$, $\pi(j) = b$ and $\pi(l) \notin [a,b]$ for $i < l \leq \overline{j}$, there exists a unique sequence $i > j_1 > \ldots > j_p = j$ such that

$$\ell(\pi, \overline{j}_1)(i, \overline{j}_r) = \ell(\pi, \overline{j}_1)(i, \overline{j}_{r-1}) + 1 \quad \text{for} \quad 1 \leq r \leq p \quad (3.3.9)$$

**Proof.** (Proof of 3.3.1) What needs to be shown is that the algorithm given above produces such a subsequence without the cover condition being violated at any point in time. Let $a$ be the value $w(i)$ and $b$ the value $w'(i)$, where $1 \leq i \leq j$. Note that $a \leq b$, however only the case where $a < b$ is of interest. We shall assume that algorithm has been properly executed in positions $k$ through position $i + 1$ without violations of the cover condition. What we are really showing then is that in any position $i$
the algorithm may be executed without violating the cover condition provided that positions \( i + 1 \) through \( k \) have already been completed. It is worth noting that the base case of position \( k \) must be shown as this is essentially an induction argument. However the argument given for an arbitrary position \( i \) will be sufficient for the initial position \( k \). Particularly the proof for position \( k \) will not require the assumption that preceding portion of the algorithm has been executed successfully, all other portions will be the same. So we now continue on considering position \( i \) and the the execution of the algorithm in that position.

We begin by dividing the permutation \( w \) into several regions. The regions are as follows:

Region \( I \) is positions \( k + 1 \) through \( n \) of \( w \).

Region \( II \) is positions \( \pi \) through \( \overline{k+1} \) of \( w \).

Region \( III \) is positions \( \overline{i} \) through \( \overline{1} \) of \( w \).

Region \( X_u \) is positions \( i + 1 \) through \( k \) of \( w \).

Region \( X_l \) is positions \( \overline{k} \) through \( \overline{i+1} \) of \( w \).

These regions are more explicitly seen in the following diagram:

Let position \( j \) be the position in which the value \( b \) occurs in \( w \), i.e \( j \) is the position such that \( w(j) = b \). We then continue with a few observations.

First note that \( j \) is in one of the regions \( I, II \) or \( III \). This is because entries \( 1 \) through \( k \) of \( w, w' \) and every intermediate permutation in the sequence from \( w \) to
$w'$ are increasing in value. If $j$ is in $X_u$ then this order will be violated after the values $a$ and $b$ are exchanged as $a < b$. If $j$ is in $X_l$ then the value $\bar{b}$ is in position $\bar{j}$ which is the in region $X_u$ so that when values $a$ and $b$ are exchanged so are values $\bar{a}$ and $\bar{b}$. However by our assumptions $X_u$ is identical to $w'$ restricted to positions $i + 1$ through $k$ since the algorithm has been executed for those positions, consequently no such exchange can be made. This establishes that $j$ is in one of regions $I$, $II$ or $III$. (Note that if $i = k$ this observation is trivial.)

We shall then regard the algorithm as taking place in several steps, step 1 exchanges with region $I$, step 2 exchanging an entry with it’s negative, step 3 exchanges with region $II$ and step 4 exchanges with region $III$.

Next observe that at the end of step 1 the value in position $I$ will be the value $a_I$ where $a_I$ is defined as $a_I = \max\{w(j), j \in I\}$. (Note here that $a \leq a_I \leq b$.) This observation is clear since each entry in region $I$ is examined and if the value in that entry is larger than the current value in position $i$ and less than $b$ then the exchange is made. It is also clear that during step 1 the cover condition was never violated as covered in the preceding lemma. This observation is needed since if this were not the case then the cover condition would be violated in a later step since there would exist a value $d$ in regions $II$ or $III$ such that $a_I < d \leq b$ and a value $c$ in region $I$ such that such that $a_I < c < d \leq b$, forcing the cover condition to be violated prior to reaching the value $b$ in position $i$.

An immediate consequence of the previous observation is that if $a \leq n$ and $b \geq \bar{n}$ then $a_I \geq n$. This is because all entries in $X_u$ would be greater than $\bar{n}$ since the algorithm had been successfully completed for those positions. Then note that if the value in this position is not $n$ to begin with then either $n$ or $\bar{n}$ is in region $I$ and that whichever of the two is in that region is greater than $a$ and less than or equal to $b$. If $a_I = n$ and $b > n$ then step 2 will be the exchange of $n$ with $\bar{n}$ otherwise nothing occurs in step 2.
As a result of the previous observation we have that all exchanges with regions II or III will be positives for positives or negatives for negatives, as the previous observation really tells us that the correct sign is achieved by the end of step 2. Note that this is precisely what is needed for the cover condition to not be violated in steps 3 and 4, that together with the previous lemmas.

It remains at this to to check that during step 4 there is no entry \( e \in X_l \) such that \( a_{II} < c \leq b \) where \( a_{II} \) is the value in position \( i \) at the end of step 3, i.e. after exchanges with region II. Observe that \( a_{II} \) is the largest value between \( a \) and \( b \) that is not in \([1, i-1]\). Consequently during step 4 any exchange of a value \( j \) will be with the value \( j + 1 \). Since there are no values between \( j \) and \( j + 1 \) there cannot be any value in \( X_l \) that will cause the cover condition to be violated. Note also that any entry in positions 1 through \( i - 1 \) will increase by at most 1 in step 4. This will ensure that switches in position \( i \) will not cause the value in position \( j \) for \( j < l \) to exceed \( w'(j) \) as a result of the algorithms execution.

We have at this point established the desired result. \( \square \)
3.4 Splittings and Right-Admissible Subsequences

The right chain shall next be examined. The goals here are firstly, to construct the admissible subsequence from a left column to a right column and secondly, to show that this chain's existence is equivalent to a column being admissible (i.e. splitting). This will be the second part to the inverse to the filling map given earlier.

We first give an algorithm that produces the desired right admissible subsequence. Here we let $w$ be the permutation associated to $lC$.

**Algorithm 3.4.1.** set $\pi = w$;

set $i = k$;

while $i \geq 2$ do

set $j = i - 1$;

while $j \geq 1$ do

if $(\pi(j) > \pi(i) \text{ and } \pi(j) \leq rC(i))$

$\pi = \pi(i, j)$;

end if

set $j = j - 1$;

end while

set $i = i - 1$;

end while

Uniqueness of the chain is clear by construction. It remains only to show that the cover condition is not violated as the algorithm is executed. We first check that $w(i) \notin [z_j, \bar{t}_j]$ for $1 \leq j \leq r$ and $i \in [k + 1, k + 1]$ as otherwise the cover condition would be violated. However this is immediate by the construction of $t_j$ as defined in the splitting, as $t_j$ is the largest possible entry such that $t_j$ and $\bar{t}_j$ are not already in $C[1, k]$. This gives us that no entry in the positions $[k + 1, k + 1]$ are an issue. Positions $[i + 1, k]$ as these are of larger value than any entry in positions $[1, i]$. These
together along with the symmetry of the permutation show the algorithm used above does not violate the cover condition.

Example 3.4.2.

\[ C = \begin{align*}
2 \\
3 \\
5 \\
5 \\
2
\end{align*} \]

Then \( I = \{5, 2\} \) and \( J = \{4, 1\} \) and

\[
lC = \begin{align*}
1 \\
3 \\
4 \\
5 \\
2
\end{align*} , \quad rC = \begin{align*}
2 \\
3 \\
5 \\
4 \\
1
\end{align*} .
\]

We have the following explicit chain from \( lC \) to \( rC \):

\[
\begin{align*}
1 & \rightarrow (5,1) \\
3 & \rightarrow (4,3) \\
4 & \rightarrow
\end{align*}
\]

In this case the tableaux \( \tilde{D} \) and \( \tilde{E} \) are initially as follows:

\[
\tilde{D} = \begin{align*}
* \\
* \\
5 \\
2
\end{align*} , \quad \tilde{E} = \begin{align*}
2 \\
3 \\
5 \\
*
\end{align*} .
\]

Here the *'s represent positions which are left unfilled.

The goal at present is to prove that there is a one-to-one correspondence between splittings of columns in type \( C \) and right-admissible subsequences. Leading to the following theorem:

**Theorem 3.4.3.** Given a KN-column \( C \) in type \( C \) the following are equivalent:

- There exists a right admissible subsequence from a permutation \( w \) to a permutation \( w' \) such that \( w[k] = lC \) and \( w'[k] = rC \)
• *The column C has a splitting.*

The essential idea behind the proof is the use of an intermediate construct in the form of a sort of game, though not technically a game in the mathematical sense. We shall call this construct the Pebble Game throughout this section. The idea will be to show that for a legal initial configuration in The Pebble Game there is a one-to-one correspondence between the initial configuration and a particular right admissible subsequence as well as a one-to-one correspondence between the initial configuration and a splitting of a column in type \( C \).

The construction of the Pebble Game as well as details of each of these come in the sections which follow. The primary reason for this Pebble Game is that it will make the visualization of the correspondence between right admissible subsequences and splittings explicit in a very visual fashion. Let \( C \) be an unsplit admissible column of type \( C \) with entries in \([n]\).

### 3.4.1 The Pebble Game

The Pebble Game is like a game to be played by the moving of pebbles were the initial configuration is dependent upon the entries in a column \( C \) where \( C \) in an arbitrary column, not necessarily a KN-tableau. The setup is as follows:

**The Board**

The setup for this game is a board consisting of two adjacent columns of length \( n \) with all cells empty. We will call the left column \( \overline{S} \) and the right column \( S \). We refer to cell \( n \) as the bottom and cell 1 as the top of either \( S \) or \( \overline{S} \). Notions of up and down are defined accordingly.

**Placement**

Next is the initial placement of pebbles on the board. This initial placement
will be based upon a column $C$. If the entry $i$ occurs in $C$ then a pebble is placed in cell $i$ of $\overline{S}$ and if the entry $i$ occurs in $C$ the a pebble is placed in cell $i$ of $S$. A cell with a pebble shall be referred to as occupied.

**Moves**

Actual play of the game is merely a moving of the pebbles in $S$. A move at cell $i$ occurs when cell $i$ is occupied in both $S$ an $\overline{S}$. The move consists of moving the pebble from cell $i$ of $S$ to the next empty cell of $S$ going up. Note that a pebble in $\overline{S}$ is never moved.

**Gameplay**

The game is then merely to begin at the bottom of $S$ and make a move if called for working up the column $S$ one cell at a time. The idea being that you always pick the first possible move from the bottom. If at any point a move is called for but is unable to be made then the game cannot be completed.

Note that this is trivial in the case of a column $C$ where there are no entries $i$ in $C$ such that $i$ is also an entry of $C$, i.e. the game has no moves.

A *legal configuration* will be a initial placement of pebbles on the board such that there never occurs an instance during gameplay where a move is required yet unable to be executed.

Observe that each move in the game corresponds to a transposition in a right admissible subsequence.

**3.4.2 The Diagram of the Pebble Game**

Based on the above game we shall create a diagram which records all of the moves that occurred in the process of The Pebble Game. The diagram will consist of the two columns $S$ and $\overline{S}$ together with an arrows. An arrow is placed for every move
that is made with the initial point of the arrow at the initial placement of the stone at the beginning of the move and the terminal point of the arrow at the position of the stone at the end of the move. For purposes of the diagram we will actually draw an arrow from the initial position of the pebble in $S$ to the same position in $\overline{S}$ and then an arrow to the next position of the pebble in $S$, we shall however refer to this as one arrow. This will be seen in the example that follows.

An arrow sequence is then a collection of arrows where the terminal point of one arrow is the initial point of the next. An arrow sequence can easily be seen to represent the path that a particular stone takes throughout the duration of the game.

This diagram will be used to go back and forth between admissible subsequences and splittings for columns of type $C$.

**Example 3.4.4.** Consider the following tableau:

$$C = \begin{array}{c}
2 \\
3 \\
7 \\
\overline{7} \\
\overline{6} \\
4 \\
3
\end{array}$$

From this we get the following diagram:

This example will be continued throughout this section.
3.4.3 Bijection Between the Pebble Game and KN Columns

In this section we show that there exists a well defined mapping between the Pebble Game and KN columns. This is done by showing given a column $C$ and the associated pebble game, that the game finished, and that given a pebble game that finishes we may produce a KN column $C$.

Recall the definition of a splitting of a column $C$ from the initial background 1.4.7. We then begin with a lemma that established that a column corresponds to a pebble game that finishes. The lemma in it’s process also shows us precisely how the pebble game may be used to produce the associated KN column $C$.

**Lemma 3.4.5.** Consider a KN column $C$ and the associated pebble game. In the case of a KN column the game finishes. Moreover, the set $I$ is the set consisting of $i$ where $i$ is a position in $S$ that is the initial point of an arrow sequence and the set $J$ is the set consisting of $j$ where $j$ is a position of $S$ that is the terminal point of an arrow sequence 1.4.7.

**Proof.** The fact that $I$ is the set of initial points of arrow sequences is immediate as $I$ is by definition the set of entries $i$ such that both $i$ and $\bar{i}$ occur in $C$ which is exactly the initial point of an arrow sequence in the pebble game as a move will certainly be required in each of these cases. It is also clear that no move can lead into these positions since a move of a pebble always ends in an unoccupied position of $S$.

To show that the set $J$ is the set of terminal points of arrow sequences is slightly more difficult. Recall the initial splitting construction with $I = \{z_1 > ... > z_r\}$ and $J = \{t_1 > ... > t_r\}$. Define $J_k = \{t_1 > ... > t_k\}$ for $1 \leq k \leq r$. Also define $O_S$ to be the unbarred entries of $S$ and $O_S$ the set of barred entries of $C$. Then by definition $t_k$ is the largest member of the set $[z_k] - [O_S \cup O_S \cup J_{k-1}]$. There are then two possibilities, either the arrow sequence for the pebble in position $z_k$ goes through $t_k$ or the arrow sequence for $z_l$ goes through $t_k$ for $l < k$. All that concerns us is that some arrow
sequence goes through $t_k$ thus suppose that the arrow sequence of $z_l$ does not go through $t_k$ and we shall show that in this case the arrow sequence for $z_k$ must. This is however clear since any value less than $z_k$ and greater than $t_k$ cannot be the end of the arrow sequence as it is either already occupied as the end of the arrow sequence of a member of $J_{k-1}$ or occupied in either $S$ or $\overline{S}$. If any of these positions are occupied no move can take a pebble to that position. If that position is occupied in $\overline{S}$ then it cannot be the ending location for the pebble. Thus for any position between $z_k$ and $t_k$ either there is no move that takes a pebble to that position or the pebble must continue to move. That being the case the pebble must be moved to position $t_k$ at some point. Thus we have that some move brings a pebble to position $t_k$. Now note that if a pebble is in position $t_k$ then is is not moved further as there is no move required since this position is not occupied in $\overline{S}$. Thus each $t_k$ is the terminus of an arrow sequence establishing that the set $J$ is equivalent to the collection of terminal points of arrow sequences. Now note that termination of the game is immediate as the set $J$ is guaranteed to exist by the definition of a KN column.

\[ \square \]

**Example 3.4.6.** In our example $I = \{7, 3\}$ and $J = \{5, 1\}$

From the above lemma the construction of $lC$ and $rC$ fairly easy to see and are covered in the corollaries which follow. The first tells us how to find $lC$ from the the pebble game. The second gives how to find $rC$ from the pebble game. It follows immediately that if both $rC$ and $lC$ can be found that $C$ can be as well. Both corollaries assume that the pebble game finishes.

**Corollary 3.4.7.** The column $lC$ is read off of the pebble game as follows:

The bottom(barred) values of $lC$ read from the bottom are the placements of pebbles in $\overline{S}$ read from the top of the diagram.
The top (unbarred) values of $lC$ read from the bottom are the final locations of pebbles in $S$ read from the bottom up.

Proof. To verify that the bottom values of $lC$ read from the bottom are the placements of pebbles in $\overline{S}$ read from the top of the diagram merely note that the bottom of $lC$ is the same as the bottom of $C$ which is precisely the initial placement of stones in $\overline{S}$. Then just put these in the correct order.

To verify that the top values of $lC$ read from the bottom are the final locations of pebbles in $S$ read from the bottom up note that the ends of arrow sequences form the set $J$. The set $J$ together with the unmoved pebbles of $S$ form the unbarred entries of $lC$ as the unmoved pebbles are merely the unbarred entries that are not changed by the splitting procedure. Then merely put these in the correct order, i.e. read from bottom to top.

Example 3.4.8. Referring back to our example we get:

$$lC = \begin{bmatrix} 1 \\ 2 \\ 5 \\ 7 \\ 6 \\ 4 \\ 3 \end{bmatrix}$$

Corollary 3.4.9. The column $rC$ is read off or the pebble game as follows:

The bottom (barred) values of $rC$ read from the bottom are the positions that are either the end of an arrow sequence or are positions where $\overline{S}$ is occupied initially and $S$ is not occupied initially read from the top of the diagram.

The top (unbarred) values of $rC$ read from the bottom are the initial locations of pebbles in $S$ read from the bottom up.

Proof. To verify that the bottom positions of $rC$ read from the bottom are the positions that are either the end of an arrow sequence or are positions where $\overline{S}$ is occupied
initially and $S$ is not occupied initially read from the top of the diagram note that $J$ is the set of endpoints of arrow sequences and that the barred entries not affected by the splitting procedure are precisely those where $\overline{S}$ is occupied initially and $S$ is not occupied initially. Then it is merely putting these in the correct order.

To verify that the top(unbarred) values of $rC$ read from the bottom are the initial locations of pebbles in $S$ read from the bottom up note that this is merely the unbarred values in $C$ which are precisely the initial positions of pebbles in $S$.  

**Example 3.4.10.** Referring back to our example we get:

\[
\begin{array}{c}
2 \\
3 \\
7 \\
6 \\
5 \\
4 \\
1
\end{array}
\]

We now have that the pebble game associated to a KN column finishes as well as a means to construct a KN column from an pebble game that finishes. Uniqueness in both directions is clear from the construction, we therefore have the following theorem:

**Theorem 3.4.11.** There exists a bijection between the pebble games which finish, i.e. have legal initial configurations, and KN columns in type $C$.

This then establishes the link between KN columns and the pebble game, thus we now move on to establish the link between the pebble game and right admissible subsequences.
3.4.4 Bijection Between the Pebble Game and Right Admissible Subsequences

We start with showing how to go from the pebble game to a right admissible subsequence. Define a step to be one move of a pebble which is seen in the diagram as a horizontal arrow from the pebbles starting position in $S$ together with the from that position to the ending position of that move in $S$.

Starting at the top of the diagram find the first step. Call the top of the step position $i$ and the bottom of the step position $j$. Then the portion of the right admissible subsequence associated to this step will be of the form:

$$(j - 1, j)(j - 2, j - 1) \ldots (i, i + 1)$$

Call this portion of the right admissible subsequence $c_{i,j}$. Then let $j_1 < \ldots < j_k$ be the positions of the tops of steps and let $i_1 < \ldots < i_k$ be the positions of the bottom of steps. Then using this terminology we have the following lemma:

**Lemma 3.4.12.** The right admissible subsequence in terms of entries will then be the concatenation $c_{i_1,j_1} \ldots c_{i_k,j_k}$.

**Proof.** Note that the top step of the diagram corresponds to finding the largest barred value $j_1$ of $lC$ such that $j_1 \in J$ and changing that to $i_1$, the value in the same position of $rC$. This forces $(j - 1, j)(j - 2, j - 1) \ldots (i, i + 1)$ to be the associated part of the right admissible subsequence as all intermediate values between $i$ and $j$ occur in the column in either barred or unbarred form and are thus exchanged ‘across’, otherwise the cover condition will be violated in the process. (This is merely the result of an analog the lemmas used for producing exchanges at a particular position in type A). Now simply repeat this procedure working from the top of the diagram starting with the column that results from the previous stage and the top step having been removed. \(\square\)
Example 3.4.13. Referring back to our example we get the following right admissible subsequence in terms of entries:

\[(3, 2)(2, 1) (6, 5) (7, 6)\]

Where the spaces are inserted to indicate the separation between steps.

In the other direction we need to produce the initial configuration of the pebble game given the right admissible subsequence. Suppose that \(\sigma\) is the initial permutation taken to \(\sigma'\) after the right admissible subsequence. Let \(l\) be the last unbarred position of \(\sigma[k]\). Then we have the following:

Lemma 3.4.14. The initial configuration of the pebble game is given by placing pebbles in the positions of \(S\) given by the values of \(\sigma'[l]\) and placing pebbles in the positions of \(\overline{S}\) given by the values of \(\sigma[l + 1, k]\).

Proof. This follows from observing that \(\sigma'[l]\) is the top of \(rC\) and \(\sigma[l + 1, k]\) is the bottom of \(lC\). Then note that this will give the initial configuration of a column with top the same as the top of \(rC\) and bottom the same as the bottom of \(lC\). This is precisely \(C\) thus the initial configuration is the desired configuration. \(\Box\)

The previous lemmas give us the following theorem:

Theorem 3.4.15. There exists a bijection between right admissible subsequences and initial configurations of the pebble game such that the initial configuration corresponding to a KN column \(C\) is identified with a right admissible from a permutation \(\sigma\) to a permutation \(\sigma'\) such that \(\sigma[k] = lC\) and \(\sigma'[k] = rC\).

This theorem together with the theorem of the previous subsection give us the result that given a KN-column \(C\) in type \(C\) the following are equivalent:

- There exists a right admissible subsequence from a permutation \(\sigma\) to a permutation \(\sigma'\) such that \(\sigma[k] = lC\) and \(\sigma'[k] = rC\)
• The column $C$ has a splitting.
3.5 A Level Counting Formula for Type C

The arguments in this section will be analogous to those in the case of type A. This level counting formula will serve as the basis for showing the bijection given in type C also preserves the crystal graph. In fact as was evident earlier any time such a formula can be found the argument given in type A will be able to extend with slight modification.

Recall $\Gamma$ and let us write $\Gamma = (\beta_1, \ldots, \beta_m)$. As such, we recall the hyperplanes $H_{\beta_k, l_k}$ and the corresponding affine reflections $\hat{r}_k = s_{\beta_k, l_k} = s_{\beta_k} + l_k \beta_k$.

Now fix a signed permutation $w$ in $C_n$ and a subset $J = \{j_1 < \ldots < j_s\}$ of $[m]$ (not necessarily $w$-admissible). Let $\Pi$ be the alcove path corresponding to $\Gamma$, and define the alcove walk $\Omega$ by

$$\Omega := \phi_{j_1} \ldots \phi_{j_s}(w(\Pi)).$$

Given $k$ in $[m]$, let $i = i(k)$ be the largest index in $[s]$ for which $j_i < k$, and let $\gamma_k := wr_{j_1} \ldots r_{j_i}(\beta_k)$. Then the hyperplane containing the face $F_k$ of $\Omega$ is of the form $H_{\gamma_k, m_k}$; in other words

$$H_{\gamma_k, m_k} = w\hat{r}_{j_1} \ldots \hat{r}_{j_i}(H_{\beta_k, l_k}).$$

As in type A our goal is to describe $m_k$ purely in terms of the filling associated to $(w, J)$.

Let $\hat{t}_k$ be the affine reflection in the hyperplane $H_{\gamma_k, m_k}$. Note that

$$\hat{t}_k = w\hat{r}_{j_1} \ldots \hat{r}_{j_i} \hat{r}_k \hat{r}_{j_i} \ldots \hat{r}_{j_1} w^{-1}.$$ 

Thus, we can see that

$$w\hat{r}_{j_1} \ldots \hat{r}_{j_i} = \hat{t}_{j_i} \ldots \hat{t}_{j_1} w.$$
Let \( T = ((a_1, b_1), \ldots, (a_s, b_s)) \) be the subsequence of \( \Gamma \) indexed by the positions in \( J \). Let \( T^i \) be the initial segment of \( T \) with length \( i \), let \( w_i := wT^i \), and \( \sigma_i := \overline{f(w_i)} \), see \((\ref{eq:74})\). In particular, \( \sigma_0 \) is the filling with all entries in row \( i \) equal to \( w(i) \), and \( \sigma := \sigma_s = \overline{f(w, T)} \). The columns of a filling of \( 2\lambda \) are numbered left to right by 1 to \( 2\lambda_1 \). If \( \beta_{j_i+1} = (a_{i+1}, b_{i+1}) = (a, b) \) falls in the segment of \( \Gamma \) corresponding to column \( p \) of \( 2\lambda \), then \( \sigma_{i+1} \) is obtained from \( \sigma_i \) by replacing the entry \( w_i(a) \) with \( w_i(b) \) in the columns \( 1, \ldots, p-1 \) of \( \sigma_i \), as well as, possibly, the entry \( w_i(b) \) with \( w_i(a) \) in the same columns.

Now fix a position \( k \), and consider \( i = i(k) \) and the roots \( \beta_k, \gamma := \gamma_k \), as above, where \( \gamma_k \) might be negative. Assume that \( \beta_k \) falls in the segment of \( \Gamma \) corresponding to column \( q \) of \( 2\lambda \). Given a filling \( \phi \), we denote by \( \phi(p) \) and \( \phi[p, q) \) the parts of \( \phi \) consisting of the columns \( 1, \ldots, 2p - 1 \) and \( p, \ldots, q - 1 \), respectively.

**Proposition 3.5.1.** With the above notation, we have

\[
m_k = \langle \text{ct}(\sigma[q]), \gamma^\vee \rangle.
\]

Let us first define \( \text{ct}(\sigma) = (c_1, \ldots, c_n) \), where \( c_i \) is half the difference between the number of occurrences of the entries \( i \) and \( \bar{i} \) in \( \sigma \). Sometimes, this vector is written in terms of the coordinate vectors \( \varepsilon_i \), as

\[
\text{ct}(\sigma) = c_1\varepsilon_1 + \ldots + c_n\varepsilon_n = \frac{1}{2} \sum_{b \in \sigma} \varepsilon_{\sigma(b)};
\]

where the last sum is over all boxes \( b \) of \( \sigma \), and we set \( \varepsilon_{\bar{i}} := -\varepsilon_i \).

**Proof.** We apply induction on \( i \), which starts at \( i = 0 \), when the verification is straightforward. We will now proceed from \( j_1 < \ldots < j_i < k \), where \( i = s \) or \( k \leq j_{i+1} \), to \( j_1 < \ldots < j_{i+1} < k \), and we will freely use the notation above. Assume that \( \beta_{j_{i+1}} \) falls in the segment of \( \Gamma \) corresponding to column \( p \) of \( 2\lambda \), where \( p \geq q \).
We need to compute

\[ w \hat{r}_{j_1} \ldots \hat{r}_{j_{i+1}} (H_{\beta_k,l_k}) = \hat{t}_{j_{i+1}} \ldots \hat{t}_{j_i} w (H_{\beta_k,l_k}) = \hat{t}_{j_{i+1}} (H_{\gamma,m}), \]

where \( m = \langle \text{ct} (\sigma_i[q]), \gamma^\vee \rangle \), by induction. Let \( \gamma' := \gamma_{j_{i+1}} \), and \( \hat{t}_{j_{i+1}} = s_{\gamma', m'} \), where \( m' = \langle \text{ct} (\sigma_i[p]), (\gamma')^\vee \rangle \), by induction. We will use the following formula:

\[ s_{\gamma', m'} (H_{\gamma,m}) = H_{s_{\gamma'} (\gamma), m - m' (\gamma', \gamma^\vee)}. \]

Thus, the proof is reduced to showing that

\[ m - m' (\gamma', \gamma^\vee) = \langle \text{ct} (\sigma_{i+1} [q]), s_{\gamma'} (\gamma^\vee) \rangle. \]

An easy calculation, based on the above information, shows that the latter equality is non-trivial only if \( p > q \), in which case it is equivalent to

\[ \langle \text{ct} (\sigma_{i+1} (p, q)) - \text{ct} (\sigma_i (p, q)), \gamma^\vee \rangle = \langle \gamma', \gamma^\vee \rangle \langle \text{ct} (\sigma_{i+1} (p, q)), (\gamma')^\vee \rangle. \]

This equality is a consequence of the fact that

\[ \text{ct} (\sigma_{i+1} (p, q)) = s_{\gamma'} (\text{ct} (\sigma_i (p, q))), \]

which follows from The Pebble Game of \( \sigma_{i+1} \) from \( \sigma_i \) explained above.

We next wish to translate what this counting formula means in the type \( C \) case.

Let \( N_c (\sigma(q)) \) and \( N_d (\sigma(q)) \) be as defined the number of occurrences of \( c \) minus the number of occurrence of \( \overline{c} \) in columns 1, \ldots, \( q - 1 \) of the split tableau. This means that we are counting the occurrence in both left and right columns of the splitting.

**Corollary 3.5.3.** Given \( (c, d)_m \) a transposition in column \( q \) where \( d \neq \overline{c} \) and \( c < d \)
we have

\[ m = \frac{N_c(\sigma(q)) - N_d(\sigma(q))}{2} \]  

(3.5.4)

**Proof.** Consider the formula from the previous result and perform the calculation.

We take the inner product of the content \( \sigma \) with the root \( \epsilon_c - \epsilon_d \), that is \((N_1(\sigma(q)), \ldots, N_n(\sigma(q)))\) with \((0, \ldots, 0, 1, 0, \ldots, 0, 1, 0, \ldots, 0)\) where the 1’s are in positions \( c \) and \( d \). The formula is then immediate.

\[ \Box \]

In the special case where \( d = c \) we have \( m_i = N_c(\sigma(q)) \).

From the above commentary we see that we do not get as clean of a result as in type \( A \), where the level did not change in a given column. We do however have that the level \( m_k \) does not change in a left column or a right column if we view the split tableaux. We also have that if a half integer value is calculated in a particular column then the transposition does not occur in that column.
3.6 The Crystal Graph Structure and Root Operators

In this section we wish to establish that the bijection given by the filling map and its inverse preserves crystal graph structure.

We shall let \((i + 1, i)\) be represented by \((i, i + 1)\) to simplify notation.

Write \(\Gamma(\lambda) = (a_1, b_1) \ldots (a_m, b_m)\) and let \(K \subseteq [m]\) be the subset of marked indices \(K = \{i_1 < \ldots < i_k\}\). So then \(\Gamma(\lambda)^i = (a_1, b_1) \ldots (a_{i_1}, b_{i_1}) \ldots (a_{i_k}, b_{i_k}) \ldots (a_m, b_m)\).

Let \(p\) be the final position in \([m] \backslash K\) in column \(j - 1\), that is the last unmarked position in column \(j - 1\).

\[f_i(\Gamma(\lambda)^i) = (a_1, b_1) \ldots (a_{i_1}, b_{i_1}) \ldots (a_p, b_p)(a_{p+1}, b_{p+1}) \ldots (a_{i_k}, b_{i_k}) \ldots (a_m, b_m),\]
i.e. the position before the highest level becomes marked. In the case where the highest level occurs in the first column \(f_i(\Gamma(\lambda)^i) = (a_1, b_1) \ldots (a_{i_1}, b_{i_1}) \ldots (a_{i_k}, b_{i_k}) \ldots (a_m, b_m)\).

**Theorem 3.6.1.** The bijection between \(KN\)-tableaux of shape \(\lambda\) with entries in \([n]\) and \(\sigma\)-admissible subsequences in \(C_n\) commutes with the root operators \(e_i\) and \(f_i\).

**Proof.** The proof in this case mimics the proof in the case of type \(A\) with the notable exception that we are using the column word \(\rho(\tilde{w})\) of the split tableau rather than the column word of the unsplit tableau. Here we let \(q\) be the column where the highest level \(m_i\) occurs as before. The outline of the proof will then be essentially the same as in type \(A\) with minor modification as follows:

We first replace \(i\) and \(i + 1\) with a + and replace \(i + 1\) and \(i\) with \(\) as in the pairing procedure. We also identify \((i, i + 1)\) with \((\tilde{i} + 1, \tilde{i})\) and \((i + 1, i)\) with \((\tilde{i}, \tilde{i} + 1)\) as they are effectively the same transpositions. Additionally it will be convenient to go back and forth between the split and unsplit tableau, so for a column \(p\) of the unsplit let \(p_l\) and \(p_r\) correspond to the left and right columns of the split tableau respectively.
1. Observe that the level $m_i$ does not change on a column of the split tableau. The level may be different in the left than in the right however.

2. The first highest level occurs in a column $q$ of the unsplit tableau after a column $p$ of the unsplit tableau containing only a $(+$(after pairing). In the case of the split tableau this will correspond to a ++ pair in columns $p_l$ and $p_r$.

3. The + in the column $p$ is the final + in $\rho(w)$ after pairing. That is the + which is changed to an $-$ by $f_i$. In the context of the split this is the final ++ pair of $\rho(\tilde{w})$.

4. $\Gamma(\lambda)^i$ restricted to columns $p + 1, \ldots, q - 1$ is empty.

5. The entries $i$, $i + 1$, $\overline{i + 1}$ and $\overline{i}$ do not occur in columns $p + 1, \ldots, q - 1$.

6. When $\Gamma(\lambda)^i$ is considered restricted to column $p$ it ends in a $(i, i + 1)$ and $\Gamma(\lambda)^i$ restricted to column $q$ begins with $(i, i + 1)$.

7. When $f_i$ changes the + in column $p$ to an $-$ in the column word context this makes the final $(i, i + 1)$ in column $p$ an $(i, i + 1)$ as well as changing the $(i, i + 1)$ in column $q$ to a $(i + 1, i)$ in the context of $\Gamma(\lambda)^i$. In the context of the split tableau changes the ++ in columns $p_l$ and $p_r$ to a -- and the effect on $\Gamma(\lambda)^i$ is identical.

8. Since $f_i$ has the same effect in both contexts $f_i$ commutes with the bijection and therefore preserves the crystal graph.

We shall at first only consider the case of $f_i$ for $1 \leq i < n$ and later examine the $f_n$ case.

The details of this remain similar to type $A$. First note that level cannot change in a column of the split tableau because of the counting formula, just as in type $A$.  

78
As far as nothing happening in columns $p + 1, \ldots, q - 1$ of the unsplit and the corresponding columns of the split these arguments are the same as in type $A$. By this we mean that $\Gamma(\lambda)^i$ is empty in these columns and the entries $i, i + 1, \tau$ and $\bar{i + 1}$ do not occur in these columns.

As far as the remainder of the proof goes we shall examine what occurs in the four cases in which $f_i$ is defined. We start with what these are in the case of the unsplit tableau, the cases are as follows:

2. An $\bar{i + 1}$ in column $p$.
3. An $i$, $i + 1$ and $\bar{i + 1}$ in column $p$.
4. An $i$, $\bar{i + 1}$ and $\bar{i}$ in column $p$.

It is also easy to see that these are the only cases as there must be one unpaired $+$ in column $p$ in order for $f_i$ to be defined. Keep in mind that in these cases $f_i$ acts as $f_i^2$ on the split tableau. It is also worth noting at this point that it is possible for an $i + 1$ to show up in the split tableau in a left column that was not there in the unsplit tableau as a result of the splitting procedure, however any time this happens there will be an $\bar{i + 1}$ showing up as well only in the right column leading to cancellation during the pairing process. Similarly an $i$ may show up in the left column of the split in the case that there was an $i + 1$ in that column to begin with, however an $\bar{i}$ will show up in the right column. Cancellation will occur in this case in the same way. Similarly cancellation occurs in the case where there was neither an $i$ or $i + 1$ and both show up as a result of the splitting process.

We shall now consider exactly what happens in each of these cases.

1. In the case of an $i$ in column $p$ we have that $p_l$ has an $i$ and $p_r$ has an $i$ prior to $f_i$. 

79
Here $\rho(\tilde{w})$ has a $+$ in $p_l$ and a $+$ in $p_r$. The action of $f_i$ is to change both of these to $-$. This effectively is switching the $i$ in $p_l$ to an $i+1$ and carrying that change over to $p_r$. So after $f_i$ we have that $p_l$ has an $i+1$ and $p_r$ has an $i+1$.

In the context of $\Gamma(\lambda)^i$ this corresponds to there being an $(i, i+1)$ at the end of the restriction to $p_l$ and this becomes $(i, i+1)$ after $f_i$. Note that $\Gamma(\lambda)^i$ restricted to $p_r$ in this case is empty. The $(i, i+1)$ in column $q$ of $\Gamma(\lambda)^i$ is then changed to an $(i+1, i)$ by the same argument as in type $A$. Thus in this case $f_i$ commutes with the bijection as desired.

2. The case of an $\bar{i} + \bar{1}$ in column $p$ follows by the same logic as the previous case.

3. The case of an $i, i+1$ and $\bar{i} + \bar{1}$ in column $p$ we have that $p_l$ has an $i$ and an $\bar{i} + \bar{1}$ and $p_r$ has an $i$ and an $i+1$ prior to $f_i$.

Here $\rho(\tilde{w})$ has a $++$ in $p_l$ and a $+-$ in $p_r$. This results in the $-+$ in $p_r$ canceling during the pairing process. The action of $f_i$ is then to change the $++$ in $p_l$ to a $--$. This effectively is switching the $i$ to an $i+1$ and switching the $\bar{i} + \bar{1}$ to an $\bar{1}$ in $p_l$ with no change to $p_r$. So after $f_i$ we have that $p_l$ has an $i+1$ and an $\bar{1}$ and $p_r$ has an $i$ and an $i+1$. This gives that $p$ unsplit has an $i$ an $i+1$ and an $\bar{1}$.

In the context of $\Gamma(\lambda)^i$ this corresponds to there being an $(i, i+1)$ at the end of the restriction to $p_l$ and this becomes $(i, i+1)$ after $f_i$. Note that $\Gamma(\lambda)^i$ restricted to $p_r$ in this case is empty. The $(i, i+1)$ in column $q$ of $\Gamma(\lambda)^i$ is then changed to an $(i+1, i)$ by the same argument as before. Thus in this case $f_i$ commutes with the bijection.

4. The case of an $i, \bar{i} + \bar{1}$ and $\bar{i}$ in column $p$ we have that $p_l$ has an $\bar{i} + \bar{1}$ and an $\bar{i}$ and $p_r$ has an $i$ and an $\bar{i} + \bar{1}$ prior to $f_i$.

Here $\rho(\tilde{w})$ has a $-+$ in $p_l$ and a $++$ in $p_r$. This results in the $-+$ in $p_l$ canceling during the pairing process. The action of $f_i$ is then to change the $++$ in $p_r$ to
This effectively is switching the $i$ to an $i + 1$ and switching the $i + 1$ to an $\bar{i}$ in $p_r$ with no change to $p_l$. So after $f_i$ we have that $p_l$ has an $\bar{i} + 1$ and an $i$ and $p_r$ has an $i + 1$ and an $\bar{i}$. This gives that $p$ unsplit has an $i + 1$ an $\bar{i} + 1$ and an $\bar{i}$.

In the context of $\Gamma(\lambda)^i$ this corresponds to there being an $(i, i + 1)$ at the end of the restriction to $p_r$ and this becomes $(i, i + 1)$ after $f_i$. The $(i, i + 1)$ in column $q$ of $\Gamma(\lambda)^i$ is then changed to an $(i + 1, i)$ by the same argument as before. Thus in this case $f_i$ commutes with the bijection.

The special case where the highest level never occurs follows in the same fashion as type $A$ together with the above commentary.

We now consider the special case of $f_n$. The only time this is defined is when in column $p$ there is only an $n$ in which case this will be changed to an $\bar{n}$. This follows exactly the same argument as the case of just an $i$ in column $p$.  

We thus have the following immediate corollary to the above theorem:

**Theorem 3.6.2.** The bijection between KN-tableaux of shape $\lambda$ with entries in $[\bar{n}]$ and admissible subsequences in $C_n$ preserves the crystal graph structure for KN-tableaux of shape $\lambda$ with entries in $[\bar{n}]$.

As an immediate corollary to this we have the following:

**Corollary 3.6.3.** The bijection between KN-tableaux of shape $\lambda$ with entries in $[\bar{n}]$ and admissible subsequences in $C_n$ preserves weight for KN-tableaux of shape $\lambda$ with entries in $[\bar{n}]$.  

81
Chapter 4

Type B

4.1 Specializing the Alcove Path Model to $B_n$

We shall fix $n$ from this point onward.

Define an $\Gamma_l(k)$ for $i \leq k$ chain to have the following form:

$$\left( (i, k+1), (i, k+2), \ldots, (i, n), (i, i), (i, \bar{n}), \ldots, (i, \bar{k}+1), (i, \bar{i}-1), \ldots, (i, \bar{1}) \right).$$

$\Gamma_l(k)$ is then defined as

$$\Gamma_l(k) = \Gamma_l^l(k)\Gamma_{k-1}^l(k)\ldots\Gamma_1^l(k)$$

Define an $\Gamma^r(k)$ chain to have the following form:
\((k, k), (k, k-1), \ldots, (k, 1)\),
\(\vdots, \vdots, \quad \\vdots, \vdots\),
(2, 2) (2, 1),
(1, 1) .

The chain \(\Gamma(k)\) is then defined as:

\[ \Gamma(k) = \Gamma^l(k)\Gamma^r(k). \quad (4.1.1) \]

The chain \(\Gamma(\lambda)\) is then defined as:

\[ \Gamma(\lambda) = \Gamma_1(\lambda'_1)\Gamma_2(\lambda'_2)\ldots\Gamma_m(\lambda'_m), \] 
\[ \quad (4.1.2) \]

and breaks down as

\[ \Gamma(\lambda) = \Gamma_1^l(\lambda'_1)\Gamma_1^r(\lambda'_1)\Gamma_2^l(\lambda'_2)\Gamma_2^r(\lambda'_2)\ldots\Gamma_m^l(\lambda'_m)\Gamma_m^r(\lambda'_m). \] 
\[ \quad (4.1.3) \]

Note that this is nearly identical to type \(C\). The only difference is in \(\Gamma^r(k)\) where here in type \(B\) transpositions of the form \((i, i)\) also occur.

One must also note that in the case where \(k = n\) \(\Gamma^r(k)\) will be identical to \(\Gamma^l(k)\). In this case \(lC = rC\) so the breakdown for \(\Gamma(\lambda)\) will be as follows:

\[ \Gamma(\lambda) = \Gamma_1(\lambda'_1)\Gamma_2(\lambda'_2)\ldots\Gamma_m(\lambda'_m) \]

as the distinction between left and right columns in no longer necessary. This particular case will be dealt with separately at the end of the section on type \(B\).
4.2 The Filling Map and its Inverse

The filling map in type B is identical to the filling map in type C. The inverse is also identical for the portion associated to left admissible subsequences. Note though that the algorithm used to produce a right admissible subsequence in type B is as follows:

Algorithm 4.2.1. \( \text{set } \pi = \sigma; \)
\[
\begin{align*}
\text{set } i &= k; \\
\text{while } i \geq 2 \text{ do} & \\
\text{set } j &= i; \\
\text{while } j \geq 1 \text{ do} & \\
\text{if } (\pi(j) > \pi(i) \text{ and } \pi(j) \leq rC(i)) & \\
\pi &= \pi(i, j); \\
\text{end if} & \\
\text{set } j &= j - 1; \\
\text{end while} & \\
\text{set } i &= i - 1; \\
\text{end while} & \end{align*}
\]

Note here that the only difference from the corresponding algorithm in type C is that \( j \) is set to \( i \) rather than \( i - 1 \) initially. This accounts for the allowing of exchanges of the form \( (i, \bar{i}) \) in right admissible subsequences in type B. It is also clear by the argument in type C that the cover condition is not violated at any point during the execution of the algorithm.
4.3 Splittings and Right-Admissible Subsequences

In this section we detail how to modify The Construct as used in type \( C \) for use in type \( B \) and then make note of the primary differences.

First recall the algorithm used to produce the right admissible subsequence where as before we let \( \sigma \) be the permutation associated to \( lC \).

**Algorithm 4.3.1.** set \( \pi = \sigma \);

set \( i = k \);

while \( i \geq 2 \) do

set \( j = i \);

while \( j \geq 1 \) do

if \((\pi(j) > \pi(i) \text{ and } \pi(j) \leq rC(i))\)

\( \pi = \pi(i, j) \);

end if

set \( j = j - 1 \);

end while

set \( i = i - 1 \);

end while

Here the only difference is that \( j \) is set to \( i \) rather than \( i - 1 \) at each stage as transposition of the form \( i, i - 1 \) may now occur.

The goal as before will be to show the following theorem:

**Theorem 4.3.2.** Given a KN-column \( C \) in type \( B \) the following are equivalent:

- There exists a right admissible subsequence from a permutation \( \sigma \) to a permutation \( \sigma' \) such that \( \sigma[k] = lC \) and \( \sigma'[k] = rC \)

- The column \( C \) has a splitting.
The proof here will be very similar to the proof in type C with some modification to the pebble game used there.

4.3.1 The Pebble Game

We now address the differences in the pebble game.

The Board

The setup for this game is a board similar to that of type C with one modification. For each 0 that occurs in a column $C$ there will be a cell added to the bottom of $S$ and $\overline{S}$. Such cells will be referred to as zero cells.

Placement

The initial placement of pebbles on the board will also be the same as in type C except that there will be pebbles placed in both $S$ and $\overline{S}$ in the zero cells, i.e. each cell representing a zero will be occupied in both $S$ and $\overline{S}$ in the initial configuration.

Moves and Gameplay

Once the board is setup the moves and gameplay are exactly the same as in type C.

We shall now examine the relationship between the contract and right admissible subsequences and column as we did in type C noting the primary differences.

4.3.2 Bijection Between the Pebble Game and KN Columns

We have as before the following lemma with proof identical to type C:

Lemma 4.3.3. Given a KN column $C$ and the associated pebble game. The set $I$ is the set consisting of $i$ where $i$ is a position in $S$ that is the initial point of an arrow
sequence. The set \( J \) is the set consisting of \( j \) where \( j \) is a position of \( S \) that is the terminal point of an arrow sequence.

This has the following consequences as in type \( C \):

**Corollary 4.3.4.** The column \( lC \) is read off or the pebble game as follows:

The bottom (barred) values of \( lC \) read from the bottom are the initial placements of pebbles in \( \overline{S} \) read from the top of the diagram excluding the zero cells.

The top (unbarred) values of \( lC \) read from the bottom are the final locations of pebbles in \( S \) read from the bottom up.

**Corollary 4.3.5.** The column \( rC \) is read off or the pebble game as follows:

The bottom (barred) values of \( rC \) read from the bottom are the positions that are either the end of an arrow sequence or are positions where \( \overline{S} \) is occupied initially and \( S \) is not occupied initially read from the top of the diagram.

The top (unbarred) values of \( rC \) read from the bottom are the initial locations of pebbles in \( S \) read from the bottom up excluding the zero cells.

It is worth noting that unlike type \( C \) the number of cells in the top or bottom of the column does not necessarily remain constant from the left column to the right column as zero cells will correspond to unbarred entries in \( lC \) and barred entries in \( rC \). However the reading of the columns \( lC \) and \( rC \) does not change significantly from how it had been done in type \( C \) as the only difference is ignoring zero cells.

Thus we have one direction of the correspondence between the pebble game and columns of type \( B \). In the other direction we have the following lemma as in type \( C \):

**Lemma 4.3.6.** Given a KN-column \( C \) there exists a unique legal initial configuration of The Pebble Game.

The proof here is also the same as it was in type \( C \). This gives us the following:
Theorem 4.3.7. There exists a bijection between legal initial configurations of the pebble game and KN columns in type B.

We now move on to establish the link between the pebble game and right admissible subsequences.

4.3.3 Bijection Between the Pebble Game and Right Admissible Subsequences

This will also be rather similar to type C. Starting at the top of the diagram find the first step. Call the top of the step position $i$ and the bottom of the step position $j$, assuming $j \neq 0$. Then the portion of the right admissible subsequence associated to this step will be of the form:

$$(j-1,j)(j-2,j-1)\ldots(i,i+1)$$

If however the bottom of the last step is in a zero position then let $i$ be the top of the given step, then the right admissible subsequence associated to this step will be of the form:

$$(n,\overline{n})(n-1,n)\ldots(i,i+1)$$

Note that in this case this differs as in these cases the value in the given position goes from being unbarred to barred as the passage is made from a left to right column.

Define this portion of the right admissible subsequence $c_{i,j}$ as in type C. Then let $j_1 < \ldots < j_k$ be the positions of the tops of steps and let $i_1 < \ldots < i_k$ be the positions of the bottom of steps as before. Here note that it is possible for $j_l$ for $1 \leq l \leq k$ to be 0, in this case we shall say that $j_l$ is $\overline{n}$ as this indicates that a value goes from being unbarred to barred. We then have the following lemma identical to type C:
Lemma 4.3.8. The right admissible subsequence in terms of entries will then be the concatenation $c_{i_1,j_1} \ldots c_{i_k,j_k}$.

In the other direction we need to produce the initial configuration of the pebble game given the right admissible subsequence as was done in type $C$. This will differ slightly as the unsplit column is arrived at differently than in type $C$. Suppose that $\sigma$ is the initial permutation taken to $\sigma'$ after the right admissible subsequence. Let $l_b$ be the first barred position of $\sigma[k]$ and let $l_t$ be the last unbarred position of $\sigma'[k]$(Here we define first and last by working from the first position of the permutation in question). Then we have the following:

Lemma 4.3.9. The initial configuration of the pebble game is given by placing pebbles in the positions of $S$ given by the values of $\sigma'[l_t]$, placing pebbles in the positions of $\bar{S}$ given by the values of $\sigma[l_b,k]$ and placing pebbles in both $S$ and $\bar{S}$ in zero positions where the number of zero positions is $l_b - l_t - 1$.

Proof. Here it helps to think of the initial unsplit column $C$ as consisting of a top, middle and bottom. The top the unbarred entries, the middle the zero entries and the bottom the barred entries. The top of $C$ is clearly the top of $rC$ by the splitting construction, this is however merely $\sigma'[l_t]$, likewise the bottom of $C$ is the bottom of $lC$ which is $\sigma[l_b,k]$. The difference here in type B is that there may be a middle portion filled with zeros. The number of zeros is merely the number of positions that are unbarred in $lC$ and barred in $C$ which is $l_b - l_t - 1$. \hfill \square

The previous lemmas give us the following theorem as in type $C$:

Theorem 4.3.10. There exists a bijection between right admissible subsequences and initial configurations of the pebble game such that the initial configuration corresponding to a $KN$ column $C$ is identified with a right admissible from a permutation $\sigma$ to a permutation $\sigma'$ such that $\sigma[k] = lC$ and $\sigma'[k] = rC$. 89
This theorem together with the theorem of the previous subsection give us the result that given a KN-column $C$ in type B the following are equivalent:

- There exists a right admissible subsequence from a permutation $\sigma$ to a permutation $\sigma'$ such that $\sigma[k] = lC$ and $\sigma'[k] = rC$
- The column $C$ has a splitting.
4.4 A Level Counting Formula for Type B

The level counting formula for type B follows by the same argument and calculation as for type C with minor exception in the special case.

Corollary 4.4.1. Given \((c, d)_m\) a transposition in column \(q\) where \(d \neq \overline{c}\) and \(c < d\) we have

\[ m = N_c(\sigma(q)) - N_d(\sigma(q)). \]  

(4.4.2)

We now consider the special case where \(d = \overline{c}\).

Corollary 4.4.3. Given \((c, \overline{c})_m\) a transposition in column \(q\) we have

\[ m = 2N_c(\sigma(q)). \]  

(4.4.4)

Proof. This comes from the fact that when taking the inner product \(\gamma\) is given by \(2\epsilon_c\) rather than \(\epsilon_c\). \qed

We are then able to easily see that this will have the same consequences as the formula in type C for the cases we are interested in.

4.5 The Crystal Graph Structure and Root Operators

Define \(e_i\) and \(f_i\) as they were in type C. We then have the following theorem as in type C:

Theorem 4.5.1. The bijection between KN-tableaux of shape \(\lambda\) with entries in \([n]\) and \(\sigma\)-admissible subsequences in \(B_n\) commutes with the root operators \(e_i\) and \(f_i\).

Proof. The proof in this case is entirely identical to that of type C in all cases. This is evident in that after splitting the structure of the tableau is the same as in type C.
The only difference in the chains on the alcove model side of things is that in going from a left to a right column there is now exchanges of the form \((i, \bar{7})\). Note however that the only time such an exchange is actually selected is in the case where \(i = n\) as was shown in the proof that the bijection existed. Consequently this will have no effect on the proof for \(i \neq n\).

The case of \(f_n\) commuting with the bijection is also the same as in type \(C\) as the only case where this is defined is in the case where there is a single \(n\) in the unsplit column \(p\) using the same setup as type \(C\).

This has the following consequences.

**Theorem 4.5.2.** The bijection between KN-tableaux of shape \(\lambda\) with entries in \([\bar{1}]\) and \(\lambda\)-increasing chains in \(B_n\) preserves the crystal graph structure for KN-tableaux of shape \(\lambda\) with entries in \([\bar{1}]\).

As an immediate corollary to this we have the following:

**Corollary 4.5.3.** The bijection between KN-tableaux of shape \(\lambda\) with entries in \([\bar{1}]\) and \(\lambda\)-increasing chains in \(B_n\) preserves weight for KN-tableaux of shape \(\lambda\) with entries in \([\bar{1}]\).
Chapter 5

Type $D$

5.1 Specializing the Alcove Path Model to $D_n$

This section will be very similar to that of types B and C with two notable exceptions. One that in the left column there is no longer an exchange of the form $(i, i)$, the righthand portion of the column is the same as it was in type C however. More importantly there is addressing that $a$-configurations are not allowed, a concern which only occurs in type $D$, and this is where we shall begin. We shall fix $n$ from this point onward.

Define an $\Gamma^i_l(k)$ for $i \leq k$ chain to have the following form:

\[
( (i, k + 1), (i, k + 2), \ldots, (i, n), \\
(i, \bar{n}), (i, \bar{n} - 1), \ldots, (i, \bar{k} + 1), \\
(i, \bar{i} - 1), (i, \bar{i} - 2), \ldots, (i, \bar{1}) ) .
\]

$\Gamma^i_l(k)$ is then defined as

\[
\Gamma^i_l(k) = \Gamma^i_{k}(k)\Gamma^i_{k-1}(k)\ldots\Gamma^i_{1}(k)
\]
Define an $\Gamma^r(k)$ chain to have the following form:

\[
\begin{align*}
&((k, k-1), (k, k-2), \ldots, (k, 1),\
&\vdots \quad \vdots\\
&(3, 2) \quad (3, 1),
&(2, 1) ) .
\end{align*}
\]

The chain $\Gamma(k)$ is then defined as:

\[
\Gamma(k) = \Gamma^l(k)\Gamma^r(k).
\] (5.1.1)

The chain $\Gamma(\lambda)$ is then defined as:

\[
\Gamma(\lambda) = \Gamma_1(\lambda'_1)\Gamma_2(\lambda'_2)\ldots\Gamma_m(\lambda'_m),
\] (5.1.2)

and breaks down as

\[
\Gamma(\lambda) = \Gamma^l_1(\lambda'_1)\Gamma^r_1(\lambda'_1)\Gamma^l_2(\lambda'_2)\Gamma^r_2(\lambda'_2)\ldots\Gamma^l_m(\lambda'_m)\Gamma^r_m(\lambda'_m).\] (5.1.3)

**Lemma 5.1.4.** $\Gamma(k)$ is an $\omega_k$-chain.

**Proof.** Follows in the same way as type C. \qed

**Example 5.1.5.** Consider the case of a column of length $k = 3$ and entries taken from $[\pi]$ where $n = 4$. In this case $\Gamma^l(3)$ and $\Gamma^r(3)$ have the following forms:

\[
\Gamma^l(3) = (3, 4)(3, \overline{4})(3, \overline{2})(3, \overline{1})(2, 4)(2, \overline{4})(2, \overline{1})(1, 4)
\] (5.1.6)
\[ \Gamma'(3) = (3, 2)(3, 1)(2, 1) \] (5.1.7)

5.1.1 The Filling Map

In type \( D \) the filling map is identical to that of types \( B \) and \( C \). The portion of interest in this case will be showing that the inverse to the filling map exists.

5.1.2 \( a \)-Configurations

We now address the main difference between type \( D \) and other types, particularly \( a \)-configurations. Here we shall use the term \( a \)-configuration to refer to an \( a \)-configuration in which \( n - a = r - q \). We wish to show that no alcove path exists in the case of an \( a \)-configuration, particularly we show that no sequence of covers in the bruhat order exists that satisfy the conditions of an \( a \)-configuration. The existence of an alcove path, or equivalently an admissible subsequence, will be taken care of explicitly as was done in previous types. Note that the existence of an \( a \)-configuration is only an issue when going form a right column to a left column. We start by considering columns \( C_1 \) and \( C_2 \) with \( C_1 \leq C_2 \) containing an \( a \)-configuration.

To a column \( C \) we associate a vector \( v_C \) of dimension \( n \) as follows for \( 1 \leq i \leq n \):

\[
v_C(i) = \begin{cases} 
1 & \text{if } i \in C \\
-1 & \text{if } i \not\in C \\
0 & \text{otherwise}
\end{cases}
\]

Lemma 5.1.8. If the columns \( C_1 \leq C_2 \) contain an \( a \)-configuration where \( s - p = n - a \) then both \( v_{C_1} \) and \( v_{C_2} \) are nonzero in position \( i \) for \( a \leq i \leq n \).

Proof. Recall from the definition of \( a \)-configurations that \( p \leq q < r \leq s \) where \( p \) is the position of \( a \) in \( C_1 \) and \( s \) is the position of \( \pi \) in \( C_2 \). This forces positions \( p \) through
s of $C_1$ and $C_2$ have values between $a$ and $\overline{a}$ without allowing the possibility of both $i$ and $\overline{i}$ occurring. Consequently for each $i$ with $a \leq i \leq n$ either $i$ or $\overline{i}$ is an entry of $C_1$ and likewise $C_2$ establishing the claim.

Furthermore by the same logic we have the following lemma:

**Lemma 5.1.9.** If $D$ is a column such that $C_1 \leq D \leq C_2$ where $C_1C_2$ contains an $a$-configuration where $s-p=n-a$ then $D$ is nonzero in position $i$ for $a \leq i \leq n$.

At this point it is our goal to show that no alcove path can exist in the case of an $a$-configuration. The way this is to be shown is by showing that it is not possible to find a sequence of covers in the Bruhat order that begins with $C_1$ and ends with $C_2$. Particularly we will pay attention to the position of the $n$ or $\overline{n}$ entry of $C_1$ and see how it compares to the position in $C_2$ provided that a sequence of covers in the Bruhat order actually exists. We will then see that this contradicts the assumptions of having an $a$-configuration.

First note what covers in the Bruhat order look like in the context of the vectors $v_C$ defined above. We say that $v_D$ covers $v_C$, written $v_C \preceq v_D$, if the permutation associated to $D$ is a cover of the permutation associated to $C$ in the Bruhat order. We can then say the following about $v_C$ and $v_D$ in this circumstance:

**Lemma 5.1.10.** If $v_C \preceq v_D$, where $C_1 \leq C \leq D \leq C_2$, then $v_D$ differs from $v_C$ as follows:

- $v_D$ differs from $v_C$ by $1, -1$ in positions $i, i+1$ being traded for $-1, 1$ where $a \leq i < n$
- $v_D$ differs from $v_C$ by $1, 1$ in positions $n-1, n$ being traded for $-1, -1$ where $a \leq i < n$.

**Proof.** First note that from the previous lemma that all positions $i$ where $a \leq i \leq n$ remain nonzero for $C$ and $D$ as $C_1 \leq C \leq D \leq C_2$. Thus the only possible exchanges
that are covers trade an $i$ for an $i + 1$ (consequently $\overline{i + 1}$ with $\overline{i}$) or $n - 1$ for $\overline{n}$ (consequently $n$ for $\overline{n - 1}$). These being precisely the two forms given above. 

At this point we need a way to keep track of the position of the $n$ or $\overline{n}$ entry in a column $C$ based on the vector $v_C$. This is done as follows:

**Lemma 5.1.11.** Consider a column $C$ and associated vector $v_C$ then:

- If $v_C(i) = 1$ then the position of the entry $i$ in the column $C$ is given by
  \[
  \{j; v_C(j) = 1, j \leq i\}
  \]

- If $v_C(i) = -1$ then the position of the entry $i$ in the column $C$ is given by
  \[
  \{j; v_C(j) = 1, or v_C(j) = -1, j \geq i\}.
  \]

**Proof.** This follows immediately from the fact that the entries in a column must be increasing as you go down the column. 

We now observe the effect of going from a column to a cover on the position of the $n$ or $\overline{n}$ entry. Note that each time the sign changes for $v_C(n)$ the positions of $n$ or $\overline{n}$ changes by one. This is immediate by merely counting as prescribed in the above lemma. Thus we have the following as a result:

**Lemma 5.1.12.** Consider columns $C_1C_2$ containing an $a$-configuration. Let $r$ be the position of $n$ or $\overline{n}$ in $C_1$ and $q$ the position of $n$ or $\overline{n}$ in $C_2$. Then we have the following:

- If $C_1(r) = n$ and $C_2(q) = \pi (or C_1(r) = \overline{n}$ and $C_2(q) = n$ then $r - q$ is odd.

- If $C_1(r) = n$ and $C_2(q) = \pi (or C_1(r) = \overline{n}$ and $C_2(q) = n$ then $r - q$ is even.

Note that the first case of the lemma above corresponds to an $a$-odd-configuration where $r - q + 1$ must be odd but is even by the lemma. The second case corresponds to an $a$-even-configuration where $r - q + 1$ must be even but is odd by the lemma. Thus as an immediate result we have the following theorem:
Theorem 5.1.13. Given \( \sigma \) and \( \sigma' \) such that \( \sigma[k] \) and \( \sigma'[k] \) contains an \( a \)-configuration, there does not exist a saturated chain in the Bruhat order from \( \sigma \) to \( \sigma' \), i.e. no admissible subsequence exists.

### 5.1.3 An Inverse to the Filling Map

Theorem 5.1.14. Given the pair \((\sigma, C')\), where \( C = \sigma[n] \leq C' \) are columns in \( D_n \) and \( C, C' \) does not have an \( a \)-configuration for \( a \in [n] \), there exists a unique \( \sigma \)-left admissible subsequence \( \gamma' \) from \( \sigma \) to a unique \( \sigma' \) such that \( \sigma'[k] = C' \).

Here right and left admissible are defined as they were in the case of type C.

We recall at this point that the filling map in type \( D \) is identical to the filling map of types B and C. We therefore shall begin to find an inverse to the filling map. Begin as in type C by giving an algorithm which explicitly constructs said \( \sigma \)-left admissible subsequence.

Algorithm 5.1.15. set \( \pi = \sigma \);

\[
\begin{align*}
&\text{set } i = k; \\
&\text{while } i \geq 1 \text{ do} \\
&\quad \text{exchange}(k + 1, n); \\
&\quad \text{exchange}(\pi, k + 1); \\
&\quad \text{exchange}(i - 1, 1); \\
&\quad \text{set } i = i - 1; \\
&\text{end while;} \\
&\quad \text{exchange}(a, b); \\
&\text{set } j = a; \\
&\text{while } j < b \text{ do} \\
&\quad \text{if } (\pi(j) > \pi(i) \text{ and } \pi(j) \leq C'(i)) \text{ end while;} \\
\end{align*}
\]
\[ \text{return } \pi = \pi(i, j); \]

\text{end if}

\text{set } j = j + 1;

\text{end while}

Note that this algorithm is nearly identical to that of type C. The difference being that here in type D the exchange of \(i\) with \(\overline{i}\) is no longer needed. We must first show that the algorithm given does not introduce \(a\)-configurations where there were none previously as this is a needed step in proving the theorem. We do this in the form of the lemma which follows. Here we shall abuse the definition of cover to refer to one column covering another if the corresponding permutations are covers.

**Lemma 5.1.16.** Consider columns \(C\) and \(D\) and corresponding admissible subsequence. This yields \(C_j\) for \(1 \leq j \leq m\) the sequence \(C = C_1 \leq \ldots \leq C_m = D\) such that \(C_{j+1}\) covers \(C_j\). Then if the column pair \(C, D\) does not contain an \(a\)-configuration then neither does the column pair \(C_j, D\).

**Proof.** For the purposes of this proof we only need observe that exchanges in a particular position \(i\) of the columns do not introduce new \(a\)-configurations. So we consider \(\overline{C} = C_j\) for some \(j\) and assume that the algorithm has been executed in positions \(i + 1\) through \(k\) without introducing any \(a\)-configurations. We then show that the algorithm as executed in position \(i\) will not introduce a \(a\)-configuration. This is done in three cases, each of which analyzes the algorithms effect on the vector \(v_{\overline{C}}\).

- \(\overline{C}(i) \geq \pi\) In this case the algorithm has the effect of moving a \(-1\) of \(v_{\overline{C}}\) to the left to a position with a 0, when viewing \(v_{\overline{C}}\) as a row vector, while not allowing passing of \(-1\) entries and then a series of exchanges such that an 1, \(-1\) pair becomes a \(-1, 1\) pair for adjacent positions. The fact that the algorithm proceeds this way is easily seen from the construction. We now note that such changes
cannot produce an $a$-configuration since anything to the right of the $\tilde{C}(i)$ entry in $v_\tilde{C}$ remains unchanged and the same position becomes a 0, consequently no $a$-configuration is created.

- $D(i) \leq n$ In this case a 1 in $v_\tilde{C}$ is moved to the right to positions with a 0 without passing a 1. The only possible $a$-configuration that could result would have $a$ as the final location of the 1 that has moved. However in this case we have that $D(i) = a$ which means that there cannot be an $a$-configuration since $\pi$ would have to be present in $D$.

- $\tilde{C}(i) \leq n, D(i) \geq \pi$ In this case a 1 in $v_\tilde{C}$ is moved to the right, noting that there are no other 1 to the right of it as the algorithm has been completed for positions further down the column. Consequently $v_\tilde{C}(n) = 0$ as the order condition on the column would be violated otherwise.

\[ \square \]

We now return to the proof of the theorem:

**Proof.** The proof here is also nearly identical to that of type C with a few minor changes, as such an outline of the proof with changes noted will be given here.

All regions are defined as they were in type C. The proof boils down to checking that the algorithm does not violate the cover condition at any point while being executed for a position $i$. Let $a$ and $b$ be as they were in the type C proof. We then make the following observations:

First, $b$ is in one of regions $I$, $II$ or $III$ for the same reasons as in type $C$.

Second, at the end of step one $a_I \geq n - 1$ if $b \geq \pi$. This second is slightly different than type $C$ as in that case we had $a_I \geq n$. This is because exchanges of the form $(i, \tilde{i})$ do not occur in type $D$. Particularly the exchange of $n$ for $\pi$ from step 2 in type C no longer occurs. However an exchange of $n - 1$ for $\pi$ can occur in type $D$ without violating the cover condition.
Third, the first exchange with regions II and III will be a positive for a negative if appropriate and all subsequent exchanges will be positive for positive or negative for negative. This also differs slightly from type C, this is because there is no longer a step 2 and step 3 is skipped to directly. Then continue with steps 3 and 4 as in type C.

Fourth, exchanges in step 4 are exchanges of \( j \) for \( j + 1 \) as in type C giving us that no cover condition violations occur in step 4 as well as any concerns over any effects on positions less than \( i \).

The only additional thing to be checked here is that when a position goes from having a unbarred entry to having a barred entry in position \( i \) that the entry in position \( i - 1 \) remains unbarred if \( C'(i - 1) \) is unbarred. The only way such a circumstance can occur is when via the algorithm we reach a column \( \tilde{C} \) such that \( \tilde{C}(i) = n \) and \( \tilde{C}(i - 1) = n - 1 \) as this is the only time where there is a cover that changes two unbarred entries to barred entries. Note that this Suppose \( C'(i) = \pi \). Note here that the exchange which makes two entries barred simultaneously happens in region III. This means that no value in the interval \([a, \pi]\) is in either region I or region II of the permutation corresponding to \( \tilde{C} \) as otherwise the value of \( \tilde{C}(i) \) would be greater than \( n \). We then note that there are two possibilities for the value of \( C'(i - 1) \) in this case:

- \( C'(i - 1) = n - 1 \)
- \( C'(i - 1) = n. \)

These are the only cases as the value in position \( i - 1 \) must be unbarred and greater than or equal to \( n - 1 \).

In the first case note that if \( C'(i - 1) = n - 1 \) then we have \( C'(i) = \pi \). This is not possible however since in this case the columns \( \tilde{C} \) and \( C' \) would not be comparable and the algorithm must have been violated at some previous point.

In the second case we now have that \( C'(i - 1) = n \) and \( C'(i) = \pi \) together with
\( \tilde{C}(i) = n \) and \( a \) an entry of \( \tilde{C} \). We have \( a \) an entry of \( \tilde{C} \) since neither \( a \) nor \( \overline{a} \) occurred in regions \( I \) or \( II \) together with the order condition on the column \( \tilde{C} \) guarantees that \( \overline{a} \) is not already there in a lower position as those position already correspond to their values in \( C' \). This gives us that the column pair \( \tilde{C}, C' \) contains an \( a \)-even configuration. However by the lemma we have that then \( C, C' \) must have contained some \( a \)-configuration. 

\( \square \)
5.2 Splittings and Right-Admissible Subsequences

In this section we detail how to modify The Construct as used in types B and C for use in type D and then make note of the primary differences as was done in type B.

First recall the algorithm used to produce the right admissible subsequence where as before we let $\sigma$ be the permutation associated to $lC$.

**Algorithm 5.2.1.** set $\pi = \sigma$;

set $i = k$;

while $i \geq 2$ do

set $j = i - 1$;

while $j \geq 1$ do

if $(\pi(j) > \pi(i) \text{ and } \pi(j) \leq rC(i))$

$\pi = \pi(i,j)$;

end if

set $j = j - 1$;

end while

set $i = i - 1$;

end while

Note that this is identical to type C.

The goal as before will be to show the following theorem:

**Theorem 5.2.2.** Given a KN-column $C$ in type D the following are equivalent:

- There exists a right admissible subsequence from a permutation $\sigma$ to a permutation $\sigma'$ such that $\sigma[k] = lC$ and $\sigma'[k] = rC$

- The column $C$ has a splitting.

The proof here will be very similar to the proof in types B and C with some modification to the pebble game used there. The game in this case will be more
similar to the one used in type $B$.

5.2.1 The Pebble Game

We now address the differences in the pebble game.

The Board

The setup for this game is a board similar to that of type $B$ with one modification. For each pair $n$ followed by $n$ that occurs in a column $C$ there will be two cells added to the bottom of $S$ and $\overline{S}$ in the form of zero cells as in type $B$.

Placement

The initial placement of pebbles on the board will also be the same as in type $B$ with pebbles placed in both $S$ and $\overline{S}$ in the zero cells. Note however that the only way a pebble is placed in position $n$ of $S$ is if there is a $n$ in $C$ that is unpaired with an $\overline{n}$, likewise the only way a pebble is placed in position $n$ of $\overline{S}$ is if there is an $\overline{n}$ in $C$ that is unpaired with an $n$ in $C$.

Moves and Gameplay

Once the board is setup the moves and gameplay are exactly the same as in types $B$ and $C$.

We shall now examine the relationship between the construct and right admissible subsequences and column as we did in types $B$ and $C$ noting the primary differences.

5.2.2 Bijection Between the Pebble Game and KN Columns

We have as before the following lemma:

Lemma 5.2.3. Given a KN column $C$ and the associated pebble game. The set $I$ is the set consisting of $i$ where $i$ is a position in $S$ that is the initial point of an arrow
sequence. The set \( J \) is the set consisting of \( j \) where \( j \) is a position of \( S \) that is the terminal point of an arrow sequence.

**Proof.** Identical to types B and C.

This has the following consequences as in types B and C:

**Corollary 5.2.4.** The column \( lC \) is read off or the pebble game as follows:

- The bottom(barred) values of \( lC \) read from the bottom are the initial placements of pebbles in \( S \) read from the top of the diagram excluding the zero cells.
- The top(unbarred) values of \( lC \) read from the bottom are the final locations of pebbles in \( S \) read from the bottom up.

**Corollary 5.2.5.** The column \( rC \) is read off or the pebble game as follows:

- The bottom(barred) values of \( rC \) read from the bottom are the positions that are either the end of an arrow sequence or are positions where \( S \) is occupied initially and \( S \) is not occupied initially read from the top of the diagram.
- The top(unbarred) values of \( rC \) read from the bottom are the initial locations of pebbles in \( S \) read from the bottom up excluding the zero cells.

Thus we have one direction of the correspondence between the pebble game and columns of type \( D \). In the other direction we have the following lemma as in types B and C:

**Lemma 5.2.6.** Given a KN-column \( C \) there exists a unique legal initial configuration of The Pebble Game.

The proof here is also similar to that of type C. This gives us the following:
Theorem 5.2.7. There exists a bijection between legal initial configurations of the pebble game and KN columns in type D.

We now move on to establish the link between the pebble game and right admissible subsequences.

5.2.3 Bijection Between the Pebble Game and Right Admissible Subsequences

This will also be rather similar to types B and C. Starting at the top of the diagram find the first step. Call the top of the step position \( i \) and the bottom of the step position \( j \), assuming \( j \neq 0 \). Then the portion of the right admissible subsequence associated to this step will be of the form:

\[(j - 1, j)(j - 2, j - 1) \ldots (i, i + 1)\]

If however the bottom of the last step is in a zero position then let \( i \) be the top of the given step, then the right admissible subsequence associated to this step will be of one of the following forms:

- \[(n, n - 1)(n - 2, n - 1) \ldots (i, i + 1)\]
- \[(n - 1, n) \ldots (i, i + 1)\]

The difference here is that steps with bottom zero are read off in pairs. The first reached in each pair, i.e. the one closer to the top, will be of the first form, the second in the pair will be of the second form.

Define the portion of the right admissible subsequence \( c_{i,j} \) as in type C. Then let \( j_1 < \ldots < j_k \) be the positions of the tops of steps and let \( i_1 < \ldots < i_k \) be the positions
of the bottom of steps as before. Here note that it is possible for \( j_l \) for \( 1 \leq l \leq k \) to be 0, in this case we shall say that \( j_l \) is \( \overline{n} \) as this indicates that a value goes from being unbarred to barred. We then have the following lemma identical to type C:

**Lemma 5.2.8.** The right admissible subsequence in terms of entries will then be the concatenation \( c_{i_1,j_1} \ldots c_{i_k,j_k} \). 

In the other direction we need to produce the initial configuration of the pebble game given the right admissible subsequence as was done in type C. This will differ slightly as the unsplit column is arrived at differently than in types B or C. Suppose that \( \sigma \) is the initial permutation taken to \( \sigma' \) after the right admissible subsequence. Let \( l_b \) be the first barred position of \( \sigma[k] \) and let \( l_t \) be the last unbarred position of \( \sigma'[k] \). (Here we define first and last by working from the first position of the permutation in question). Then we have the following:

**Lemma 5.2.9.** The initial configuration of the pebble game is given by placing pebbles in the positions of \( S \) given by the values of \( \sigma'[l_t] \), placing pebbles in the positions of \( \overline{S} \) given by the values of \( \sigma[l_b,k] \) and placing pebbles in both \( S \) and \( \overline{S} \) in zero positions where the number of zero positions is \( l_b - l_t - 1 \).

*Proof.* Same as types B and C. \( \square \)

The previous lemmas give us the following theorem as in types B and C:

**Theorem 5.2.10.** There exists a bijection between right admissible subsequences and initial configurations of the pebble game such that the initial configuration corresponding to a KN column \( C \) is identified with a right admissible from a permutation \( \sigma \) to a permutation \( \sigma' \) such that \( \sigma[k] = lC \) and \( \sigma'[k] = rC \).

This theorem together with the theorem of the previous subsection give us the result that given a KN-column \( C \) in type D the following are equivalent:
• There exists a right admissible subsequence from a permutation $\sigma$ to a permutation $\sigma'$ such that $\sigma[k] = lC$ and $\sigma'[k] = rC$

• The column $C$ has a splitting.
5.3 A Level Counting Formula for Type $D$

The level counting formula for type $D$ follows by the same argument as for type $C$ as well. This will differ from both types $B$ and $C$ only in the special case.

**Corollary 5.3.1.** Given $(c, d)_m$ a transposition in column $q$ where $d \neq \overline{c}$ and $c < d$ we have

$$m = N_c(\sigma(q)) - N_d(\sigma(q)).$$  \hfill (5.3.2)

We now simply note that the special case where $d = \overline{c}$ does not occur in type $D$.

5.4 The Crystal Graph Structure and Root Operators

Define $e_i$ and $f_i$ as they were in type $C$ for $i < n$. For the case where $i = n$ consider $\Gamma(\lambda)^n$, this will consist of the restriction of $\Gamma(\lambda)$ to transpositions of the form $(n, \overline{n - 1})$, $(\overline{n - 1}, n)$ or $(n, \overline{n - 1})$. Here we identify $(n, \overline{n - 1})$ with $(n - 1, \overline{n})$ as they are effectively the same transposition. $f_n$ is the defined in the same fashion as any other $f_i$ in this context.

We then have the following theorem as in types $B$ and $C$:

**Theorem 5.4.1.** The bijection between $KN$-tableaux of shape $\lambda$ with entries in $[\overline{n}]$ and $\sigma$-admissible subsequences in $D_n$ commutes with the root operators $e_i$ and $f_i$.

**Proof.** For the cases where $i < n - 1$ the proof is identical to type $C$.

When $i = n - 1$ the only way $f_{n-1}$ is defined is when the column $p$ of the unsplit column prior to the column $q$ of highest level contains an unpaired $n - 1$ or an unpaired $\overline{n}$. In each of these cases the proof is just as it was for just an $i$ or just an $\overline{i + 1}$ in type $C$. This is so since in each of these case which the entry shows up in the left and right columns and is changed in both.

109
When $i = n$ note that all pairs of $n$ and $\bar{n}$ cancel regardless of order. Thus the only way $f_n$ is defined is when there is an $n - 1$ or an $n$ after this cancellation. In this case $f_n$ acts in the same fashion as $f_i$ would for just an $i$ or just an $\bar{i} + 1$ as in type $C$.

□

This has the following consequences.

**Theorem 5.4.2.** The bijection between KN-tableaux of shape $\lambda$ with entries in $[\bar{n}]$ and $\lambda$-increasing chains in $D_n$ preserves the crystal graph structure for KN-tableaux of shape $\lambda$ with entries in $[\bar{n}]$.

As an immediate corollary to this we have the following:

**Corollary 5.4.3.** The bijection between KN-tableaux of shape $\lambda$ with entries in $[\bar{n}]$ and $\lambda$-increasing chains in $D_n$ preserves weight for KN-tableaux of shape $\lambda$ with entries in $[\bar{n}]$. 

110
Bibliography


