String of columns rewriting and confluence of the jeu de taquin

Nohra Hage, Philippe Malbos

String of columns rewriting and plactic-like data structures

Nohra Hage Philippe Malbos

Abstract – In this article, we prove combinatorial and algebraic properties on the plactic and hypoplactic congruences via a new rewriting approach on tableaux, that we apply to the jeu de taquin and the right-bottom rectification algorithms. The jeu de taquin is an algorithm that transforms a skew tableau into a Young tableau by local transformation rules on the columns of the tableaux. This algorithm has remarkable combinatorial and algorithmic properties, in particular it is confluent, that is the resulting Young tableau does not depend on the way the local transformations rules are applied. We introduce the notion of string of columns rewriting system in order to study rewriting properties of algorithms defined on glued sequences of columns. We prove the convergence of the jeu de taquin and we show how to deduce algebraic properties on the plactic congruence. We define the right-bottom rectification algorithm that transforms a Young tableau into a quasi-ribbon tableau by a string of columns rewriting system. We prove its convergence and we deduce algebraic properties on the hypoplactic congruence.

Keywords – Jeu de taquin, plactic monoids, hypoplactic monoids, string data structures, string of columns rewriting.

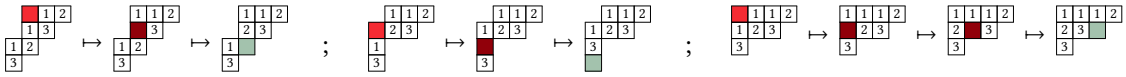
M.S.C. 2010 – Primary: 05E99. **Secondary:** 20M05, 68Q42, 20M35.

1 Introduction **2**
2 String data structures **5**
2.1 String data structures 5
2.2 String data structures and crystal structures 7
3 String of columns rewriting **9**
3.1 Strings of columns 9
3.2 Examples of plactic-like structures 11
3.3 Crystal structures on strings of columns 13
3.4 String of columns rewriting 17
3.5 Terminations orders 20
4 Convergence of the jeu de taquin **21**
4.1 Jeu de taquin 21
4.2 Jeu de taquin as rewriting 22
4.3 Jeu de taquin as morphism 27
5 Convergence of the right-bottom rectification **30**
5.1 Convergence of the right-bottom rectification 30
5.2 Right-bottom rectification as morphism 34

1. Introduction

The *jeu de taquin* was introduced by Schützenberger as an algorithm on the structure of tableaux in order to give one of the first correct proofs of the Littelwood-Richardson rule on the multiplicity of a Schur polynomial in a product of Schur polynomials, [21]. A *Young tableau* is a collection of boxes in left-justified rows filled with elements of the totally ordered alphabet $[n] := \{1 < \cdots < n\}$, where the entries weakly increase along each row and strictly increase down each column. A *skew tableau* is obtained by eliminating boxes from the rows of a Young tableau starting from top to bottom and from right to left. The eliminated boxes are called *inner corners* of the skew tableau. We read tableaux column-wise from left to right and from bottom to top: the following tableaux

are respectively skew tableau and Young tableau whose readings are 3121312 and 3213112, where the red empty boxes denote the inner corners. The jeu taquin consists in applying successively *forward sliding operations* on a skew tableau that move an inner corner into an outer position by keeping the rows weakly increasing and the columns strictly increasing, until no more inner corners remain in the initial skew tableau, as follows



Schützenberger proved remarkable properties of the jeu de taquin on skew tableaux, [21]. He proved that the rectification $\pi_{tq}(w)$ of a skew tableau w by the jeu de taquin is a Young tableau whose reading is equivalent to the reading of w with respect the *plactic congruence* relation generated by the following *Knuth relations*, [12]:

$$zxy = xzy, \quad \text{for } 1 \leqslant x \leqslant y < z \leqslant n, \text{ and } \quad yzx = yxz, \quad \text{for } 1 \leqslant x < y \leqslant z \leqslant n.$$

As a consequence, he proved that the resulting Young tableau does not depend on the order in which we choose inner corners in the forward slidings. This is the *confluence* property of the jeu de taquin. Here, this property follows the cross-section property of Young tableaux with respect the plactic congruence, proved by Knuth in [12], namely two words on $[n]$ are plactic congruent if and only if they lead to the same Young tableau after applying Schensted's insertion algorithm, [20]. Explicitly, if there are two sequences of sliding operations that transform a tableau w into two different tableaux w_1 and w_2, then we continue applying sliding operations until we reach normal forms $\widetilde{w}_1$ and $\widetilde{w}_2$, that is tableaux without inner corners:

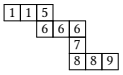
$$\begin{array}{ccccc} & & w_1 & \longrightarrow & \widetilde{w}_1 \\ w & \nearrow & & & \\ & \searrow & & & \\ & & w_2 & \longrightarrow & \widetilde{w}_2 \end{array}$$

Since $\widetilde{w}_1$ and $\widetilde{w}_2$ are two Young tableaux such that their readings are plactic congruent, following the cross-section property, we deduce that $\widetilde{w}_1 = \widetilde{w}_2$.

In this article, we introduce a rewriting machinery to prove in a direct way the confluence of the jeu de taquin. We define the sliding operations as rewriting rules on *strings of columns*, that is strings

composed by glued sequences of columns, with a gluing map that describes the relative positions of columns. In this way, the jeu de taquin is described by a rewriting system on strings of columns. Our first main result, Theorem 4.2.3, proves that this rewriting system is confluent and terminating and that the normal form of any skew tableau is a Young tableau. We show how to deduce from the confluence and normalization property of this rewriting system the cross-section property of Young tableaux with respect the plactic congruence, and the commutation of right and left Schensted's algorithms. Moreover, using this rewriting description of Schützenberger's sliding operations, Theorem 4.3.1 shows that the rectification defines a morphism of string data structures from the string data structure of skew tableaux to the string data structure of Young tableaux. Finally, we show that this morphism induces a Kashiwara crystal isomorphism between skew tableaux and Young tableaux.

In a second part of this article, using the same machinery we give a rewriting description of the *right-bottom rectification* algorithm that transforms a Young tableau into a *quasi-ribbon tableau*. This structure of tableaux was used by Krob and Thibon in [13] and plays a similar role to that of the structure of Young tableaux in the theory of non-commutative symmetric functions and quasi-symmetric functions. Recall that a quasi-ribbon tableau is a collection of boxes filled with elements of $[n]$, where the entries weakly increase along each row and strictly increase down each column, and where the columns are arranged from left to right so that the bottom box in each column aligns with the top box of the next column. We read a quasi-ribbon tableau column-wise from left to right and from bottom to top: the following tableau



is a quasi-ribbon tableau whose reading is 1165687689.

We define an algorithm, the *right-bottom rectification* algorithm, that transforms a Young tableau w into a quasi-ribbon tableau $\pi_{rb}(w)$ whose reading is equivalent to the reading of w with respect the congruence relation generated by the *hypoplactic relations* defined as the Knuth relations together with the following relations, [13]:

$$zxty = xzyt \text{ for } 1 \leqslant x \leqslant y < z \leqslant t \leqslant n, \qquad tyzx = ytxz \text{ for } 1 \leqslant x < y \leqslant z < t \leqslant n.$$

Our rectification algorithm is defined by a string of columns rewriting system whose rules describe the elementary right and bottom slidings of columns. The second main result of this article, Theorem 5.1.3, proves that this rewriting system is confluent and terminating and that the normal form of any Young tableau is a quasi-ribbon tableau. Theorem 5.2.3 shows that the right-bottom rectification π_{rb} defines a morphism of string data structures from the string data structure of Young tableaux to the string data structure of quasi-ribbon tableaux. We deduce from this morphism that the set of quasi-ribbon tableaux satisfies the cross-section property for the hypoplactic congruence. Finally, we show that this morphism induces a quasi-Kashiwara crystal isomorphism between Young tableaux and quasi-ribbon tableaux.

To summarize the constructions introduced in this article, we construct two surjective morphisms of string data structures

$$d\mathbb{S}_n^r \xrightarrow{\pi_{tq}} \mathbb{Y}_n^r \xrightarrow{\pi_{rb}} \mathbb{Q}_n^r$$

from string data structure of skew tableaux $d\mathbb{S}_n^r$, to string data structure of Young tableaux $\mathbb{Y}_n^r$, and to string data structure of quasi-ribbon tableaux $\mathbb{Q}_n^r$, that induce surjective morphisms of monoids

$[n]^* \twoheadrightarrow \mathbf{P}_n \twoheadrightarrow \mathbf{Hp}_n$, where $\mathbf{P}_n$ and $\mathbf{Hp}_n$ are respectively the *plactic monoid* and the *hypoplactic monoid*, and that also induce crystal isomorphisms

$$\Gamma(\mathcal{K}, d) \simeq \Gamma(\mathcal{K}, \pi_{tq}(d)), \qquad \text{and} \qquad \Gamma(q\mathcal{K}, d') \simeq \Gamma(q\mathcal{K}, \pi_{rb}(d')),$$

for every skew tableau d and Young tableau d', with respect the Kashiwara crystal structure $\mathcal{K}$ [11] and the quasi-Kashiwara crystal structure $q\mathcal{K}$ [3].

We expect that the convergent string of columns rewriting systems introduced in this article can be used to explicit the *relations among the relations* for the jeu de taquin and the right-bottom rectification algorithms. Indeed, convergent presentations allow us to compute *coherent presentations* of monoids, that is presentations by generators, relations, and relations between the relations, [5, 7, 8]. A coherent presentation constitutes the first step in an explicit construction of a cofibrant approximation of the monoid in the category of $(\infty, 1)$-categories, [6].

Organization of the article. In Section 2 we recall the notion of *string data structures* from [9], that correspond to presentations of monoids defined by data structures and whose relations are formulated by an insertion algorithm. We introduce the notion of *morphisms of string data structures* and we show how the property of commutation of insertion algorithms can be preserved by such a morphism. We recall in (2.2) the notion of crystal structure on words from [10, 11], and we define a compatibility condition between a crystal structure and a string data structure so that the corresponding crystal monoid is isomorphic to the monoid presented by the string data structure.

In Section 3, we study two-dimensional strings defined by glued sequences of columns that we call *string of columns*. We define a *plactic-like structure* as a string data structure whose data are string of columns with respect a fixed gluing sequence. We define in (3.4) the notion of *string of columns rewriting system* as a binary relation on the set of strings of columns, whose rules are applied with respect to right and left positions. We show how to present a plactic-like structure by such a rewriting system.

In Section 4, we describe the jeu de taquin by a string of columns rewriting system, whose the rules describe forward slidings. We prove its confluence and termination. We deduce that the jeu de taquin defines a morphism of string data structures from skew tableaux to Young tableaux, and thus a crystal isomorphism between skew tableaux and Young tableaux.

In Section 5, we define the *right-bottom rectification* algorithm that transforms a Young tableau into a quasi-ribbon tableau. The algorithm is defined as a string of columns rewriting system, whose the rules describe right and bottom slidings. We prove that this rewriting system is confluent and terminating. Finally, we deduce that the right-bottom rectification defines a morphism of string data structures from Young tableaux to quasi-ribbon tableaux, and thus a crystal isomorphism between Young tableaux and quasi-ribbon tableaux.

Conventions and notations. We will consider the totally ordered set $[n] := \{1 < \cdots < n\}$, for $n \in \mathbb{Z}_{>0}$, as ground alphabet. We denote by $[n]^*$ the free monoid of *words over* $[n]$, whose empty word is denoted by λ. The congruence generated by a set S of relations on $[n]^*$ is denoted by $\approx_S$. The length of a word w in $[n]^*$ is denoted by $|w|$, and the number of times the element i of $[n]$ appears in w is denoted by $|w|_i$. The *weight map* is the map $\mathrm{wt} : [n]^* \to \left(\mathbb{N} \cup \{0\}\right)^n$, defined by $\mathrm{wt}(w) = (|w|_1, \ldots, |w|_n)$, for all w in $[n]^*$. Let $w = x_1 x_2 \ldots x_k$ be a word in $[n]^*$. We will denote by $\underline{w} = x_k \ldots x_2 x_1$ its *mirror word*. We denote by $\ell(w)$ the leftmost letter of w and by $\mathrm{Rem}(w)$ the subword of w such that $w = \ell(w)\,\mathrm{Rem}(w)$. We will denote by $<_{\mathrm{lex}}$ the lexicographic order on $[n]^*$ induced by the order on $[n]$, and by $\preccurlyeq_{lex}$ (resp. $\preccurlyeq_{revlex}$) the lexicographic (resp. reverse lexicographic) order on tuples of natural numbers.

2. String data structures

This section recalls the notions of string data structure from [9] and of crystal structure from [10, 11]. We define a compatibility condition between a crystal structure and a string data structure so that the corresponding crystal monoid and structure monoid are isomorphic.

2.1. String data structures

2.1.1. String data structures. A *right* (resp. *left*) *string data structure* $\mathbb{S}$ (over $[n]$) is a triple (D, I, R) made of a set D with a distinguished element $\emptyset$, and two maps $R : D \to [n]^*$ and $I : D \times [n] \to D$ satisfying the three following conditions:

i) $R(I(\emptyset, x)) = x$ for all x in $[n]$,

ii) the relation $(\emptyset \leftsquigarrow_I R(d)) = d)$ (resp. $(R(d) \rightsquigarrow_I \emptyset)$= d) holds for any d in D, where $\leftsquigarrow_I : D \times [n]^* \to D$, (resp. $\rightsquigarrow_I$: $D \times [n]^* \to D$) is the map defined by

$$(d \leftsquigarrow_I u) = (I(d, \ell(u)) \leftsquigarrow_I \mathrm{Rem}(u))$$

$$\big(\text{resp. } (u \rightsquigarrow_I d) = (\underline{\mathrm{Rem}(\underline{u})} \rightsquigarrow_I I(d, \ell(\underline{u})))\big)$$

for all d in D and u in $[n]^*$, and $d \leftsquigarrow_I \lambda = d$ (resp. $\lambda \rightsquigarrow_I d = d$) for all $d \in D$,

iii) the map R is injective and $R(\emptyset) = \lambda$.

We say that R is the *reading map* of $\mathbb{S}$, and that I *inserts* an element of $[n]$ into an element of D. The map $\leftsquigarrow_I$ (resp. $\rightsquigarrow_I$) is called the *insertion map* of words in $[n]^*$ into elements of D. The map $C_{\mathbb{S}} : [n]^* \to D$ sending a word w to $(\emptyset \leftsquigarrow_I w)$ (resp. $(w \rightsquigarrow_I \emptyset)$), is called the *constructor* of $\mathbb{S}$ from words in $[n]^*$, and it is surjective as a consequence of **ii)**. We will denote by $\iota_D : [n] \to D$ the map that sends a letter x in $[n]$ on the single element data $I(\emptyset, x)$, that we write simply x when no confusion can arise. For a given string data structure $\mathbb{S}$, we will denote by $D_{\mathbb{S}}$ its underlying set, and by $I_{\mathbb{S}}$ and $R_{\mathbb{S}}$ its insertion and reading maps. A string data structure $\mathbb{S}$ is called *weight-preserving* if, for every u in $[n]^*$, the condition $\mathrm{wt}(u) = \mathrm{wt}(R(C_{\mathbb{S}}(u)))$ holds. In the sequel, if there is no possible confusion all string data structures are right. The proofs of stated properties for left ones are similar.

In this article, all string data structures are also *associative*, that is the product $\star_I$ on D, defined by $d \star_I d' := (d \leftsquigarrow_I R(d'))$, for all d, d' in D, is associative. As a consequence, the set D with the product $\star_I$ is a monoid called the *structure monoid* of $\mathbb{S}$, and denoted by $\mathbf{M}(D, I)$ or $\mathbf{M}(\mathbb{S})$. A string data structure $\mathbb{S}$ satisfies the *cross-section property* for a congruence relation $\approx$ on $[n]^*$, if $u \approx v$ holds if and only if $C_{\mathbb{S}}(u) = C_{\mathbb{S}}(v)$ holds for all u, v in $[n]^*$. In that case, the quotient monoid $[n]^*/\approx$ is isomorphic to the monoid $\mathbf{M}(\mathbb{S})$, [9]. We say that $\mathbb{S}$ satisfies the *cross-section property* for a monoid $\mathbf{M}$ if $\mathbb{S}$ satisfies the cross-section property for the congruence $\approx_{\mathbf{M}}$ on $[n]^*$ defined by $\mathbf{M}$.

Let (D, I, R) and (D, J, R) be right and left string data structures. We say that the insertion maps I and J *commute* if

$$y \rightsquigarrow_J (d \leftsquigarrow_I x) \;=\; (y \rightsquigarrow_J d) \leftsquigarrow_I x, \tag{2.1.2}$$

holds for all d in D and x, y in $[n]$. In this case, the compositions $\star_I$ and $\star_J$ are associative, and the relation $d \star_I d' = d' \star_J d$, holds for all d, d' in D, [9].

2.1.3. Morphisms of string data structures. Given two string data structures $\mathbb{S} = (D, I, R)$ and $\mathbb{S}' = (D', I', R')$, a *morphism of string data structures* from $\mathbb{S}$ to $\mathbb{S}'$ is a map $f : D \to D'$ that maps $\emptyset$ to $\emptyset'$ and satisfies the following two conditions:

i) $f(d \leftsquigarrow_I x) = f(d) \leftsquigarrow_{I'} x$, for all $d \in D$ and $x \in [n]$,

ii) for every d in D, $C_{\mathbb{S}'}(R(d)) = C_{\mathbb{S}'}(R'(f(d)))$ holds.

2.1.4. Lemma. *Any morphism of string data structures $f : \mathbb{S} \to \mathbb{S}'$ induces a morphism of monoids $\mathbf{M}(f) : \mathbf{M}(\mathbb{S}) \to \mathbf{M}(\mathbb{S}')$.*

Proof. Following condition **i)** in (2.1.3), we prove $f(d \leftsquigarrow_I u) = f(d) \leftsquigarrow_{I'} u$, for all d in D and u in $[n]^*$ by induction on $|u|$. Suppose the equality holds when $|u| = k - 1$, then for y in $[n]$ we have

$$f(d \leftsquigarrow_I uy) = f((d \leftsquigarrow_I u) \leftsquigarrow_I y) = f((d \leftsquigarrow_I u)) \leftsquigarrow_{I'} y = (f(d) \leftsquigarrow_{I'} u) \leftsquigarrow_{I'} y) = f(d) \leftsquigarrow_{I'} uy.$$

In an other hand, following condition **ii)**, we have $C_{\mathbb{S}'}(R(d')) = C_{\mathbb{S}'}(R'(f(d')))$, for all d' in D, hence $C_{\mathbb{S}'}(R'(f(d))R(d')) = C_{\mathbb{S}'}(R'(f(d))R'(f(d')))$. As a consequence, we have

$$f(d \star_I d') = f(d \leftsquigarrow_I R(d') = f(d) \leftsquigarrow_{I'} R(d') = f(d) \leftsquigarrow_{I'} R'(f(d') = f(d) \star_{I'} f(d'),$$

for all d, d' in D, proving that $\mathbf{M}(f)$ is a morphism of monoids. □

The commutation of insertions of a string data structure can be proved using the following result.

2.1.5. Lemma. *Let (D, I, R) and (D, J, R) be right and left string data structures such that I and J commute. Let (D', I', R') and (D', J', R') be right and left string data structures. If there exists a surjective morphism of string data structures from D to D' with respect to insertions I and I' and to insertions J and J', then I' and J' commute.*

Proof. Suppose that $f : D \to D'$ is a surjective map that extends into morphisms of string data structures $(D, I, R) \to (D', I', R')$ and $(D, J, R) \to (D', J', R')$. We have

$$f(y \rightsquigarrow_J (d \leftsquigarrow_I x)) = y \rightsquigarrow_{J'} (f(d \leftsquigarrow_I x)) = y \rightsquigarrow_{J'} (f(d) \leftsquigarrow_{I'} x),$$

and

$$f((y \rightsquigarrow_J d) \leftsquigarrow_I x) = f((y \rightsquigarrow_J d)) \leftsquigarrow_{I'} x = (y \rightsquigarrow_{J'} f(d)) \leftsquigarrow_{I'} x,$$

for all d in D and x, y in $[n]$. By commutation of I and J, we deduce the following equality

$$y \rightsquigarrow_{J'} (f(d) \leftsquigarrow_{I'} x) = (y \rightsquigarrow_{J'} f(d)) \leftsquigarrow_{I'} x.$$

The map f being surjective, we deduce that I' and J' commute. □

2.2. String data structures and crystal structures

2.2.1. Crystals. Recall from [11], that a *crystal structure* on a subset S of $[n]^*$ is a data $C = \big((e_i)_{i\in[n-1]}, (f_i)_{i\in[n-1]}\big)$ made of families of *crystal operators*, that is maps

$$e_i, f_i : S \to S \cup \{0\},$$

satisfying the following properties:

i) e_i and f_i are *mutually inverse*, *i.e.* for all $u, v \in S$, $v = f_i(u)$ holds if and only if $u = e_i(v)$ holds,

ii) if $u \in S$ satisfies $e_i(u) \neq 0$ (resp. $f_i(u) \neq 0$), then

$$\varepsilon_i(e_i(u)) = \varepsilon_i(u) - 1,\ \varphi_i(e_i(u)) = \varphi_i(u) + 1 \ \text{ and } \ \mathrm{wt}(u) \prec_{\mathrm{weight}} \mathrm{wt}(e_i(u))$$

$$\big(\text{resp. } \varepsilon_i(f_i(u)) = \varepsilon_i(u) + 1,\ \varphi_i(f_i(u)) = \varphi_i(u) - 1 \text{ and } \mathrm{wt}(f_i(u)) \prec_{\mathrm{weight}} \mathrm{wt}(u)\big),$$

$$\text{where } \varepsilon_i(u) = \max\{k \in \mathbb{N} \cup \{0\} \mid e_i^k(u) \neq 0\} \text{ and } \varphi_i(u) = \max\{k \in \mathbb{N} \cup \{0\} \mid f_i^k(u) \neq 0\}.$$

A word u in S is of *highest-weight* if $e_i(u) = 0$, for all i in $[n-1]$.

2.2.2. Kashiwara and quasi-Kashiwara crystal structures. We recall from [11] the Kashiwara crystal structure for type A on $[n]^*$, that we will denote by $\mathcal{K} = (\widetilde{e_i}, \widetilde{f_i})$. For $i \in [n-1]$, the *Kashiwara operators*

$$\widetilde{e_i}, \widetilde{f_i} : [n]^* \to [n]^* \cup \{0\}$$

are defined as follows. A word w in $[n]^*$ is transformed into a word in $\{+,-\}^*$, by replacing each letter i of w by $+$, each letter $i+1$ by $-$, and every other letter by λ, and by keeping a record of the original letter replaced by each symbol. Then we delete the subwords $-+$ until no such subwords remain. The final word is of the form $+^r-^l$. Note that the method given in [11] consists of eliminating sub-words $+-$, this is because the choice of convention for the reading map on Young tableaux. If $r > 0$, then $\widetilde{f_i}(w)$ is obtained by replacing in w the rightmost element i by $i+1$ and the others elements of w stay unchanged. If $r = 0$, then $\widetilde{f_i}(w) = 0$. If $l > 0$, then $\widetilde{e_i}(w)$ is obtained by replacing in w the leftmost element $i+1$ by i and the others elements of w stay unchanged. If $l = 0$, then $\widetilde{e_i}(w) = 0$.

Recall from [3] the quasi-Kashiwara crystal structure on $[n]^*$, that we denote by $q\mathcal{K} = (\ddot{e}_i, \ddot{f}_i)$. For $i \in [n-1]$, the *quasi-Kashiwara operators*

$$\ddot{e}_i, \ddot{f}_i : [n]^* \to [n]^* \cup \{0\}$$

are defined as follows. Let w in $[n]^*$. If i appears to the right of $i+1$ in w, then $\ddot{e}_i(w) = \ddot{f}_i(w) = 0$. If i does not appear to the right of $i+1$ in w, and w contains at least one element i, then $\ddot{f}_i(w)$ is obtained from w by replacing the rightmost i by $i+1$. If w does not contain i, then $\ddot{f}_i(w) = 0$. If i does not appear to the right of $i+1$ in w, and w contains at least one element $i+1$, then $\ddot{e}_i(w)$ is obtained from w by replacing the leftmost $i+1$ by i. If w does not contain $i+1$, then $\ddot{e}_i(w) = 0$. Note that the quasi-Kashiwara operators act as restriction of the Kashiwara operators, that is for u in $[n]^*$, if $\ddot{e}_i(u)$ (resp. $\ddot{f}_i(u)$) is defined, so is $\widetilde{e_i}(u)$ (resp. $\widetilde{f_i}(u)$), and $\widetilde{e_i}(u) = \ddot{e}_i(u)$ (resp. $\widetilde{f_i}(u) = \ddot{f}_i(u)$), [3, Prop. 5.2].

2.2.3. Crystal graphs. The *crystal graph* of C is the oriented graph, denoted by $\Gamma(C)$, whose vertex set is S, and for all u, u' in S, there is a labelled arrow $u \xrightarrow{i} u'$ if and only if $f_i(u) = u'$, or equivalently $e_i(u') = u$. For u in S, we will denote by $\Gamma(C, u)$, or $\Gamma(u)$ for short, the connected component of the crystal graph $\Gamma(C)$ containing u. A *crystal isomorphism* from $\Gamma(u)$ to $\Gamma(u')$ is a bijective map $\psi : \Gamma(u) \to \Gamma(u')$ satisfying the following two conditions:

i) it is *weight-preserving*, that is $\mathrm{wt}(\psi(v)) = \mathrm{wt}(v)$, for every v in $\Gamma(u)$,

ii) for every arrow $v \xrightarrow{i} v'$ in $\Gamma(u)$, there is an arrow $\psi(v) \xrightarrow{i} \psi(v')$ in $\Gamma(u')$.

2.2.4. Crystal congruence. We define a relation $\approx_C$ on S by setting, for all $u, u' \in S$, $u \approx_C u'$ if there is a crystal isomorphism $\psi : \Gamma(u) \to \Gamma(u')$ such that $\psi(u) = u'$.

For $i \in [n-1]$, we say that the operator e_i (resp. f_i) is *monomial* if for all $u, v, u', v' \in S$ such that $u \approx_C v$ and $u' \approx_C v'$, then $e_i(uu') \neq 0$ (resp. $f_i(uu') \neq 0$) if and only if $e_i(vv') \neq 0$ (resp. $f_i(vv') \neq 0$), and if both are non-zero, one of the following properties holds:

i) $e_i(uu') = ue_i(u')$ and $e_i(vv') = ve_i(v')$ (resp. $f_i(uu') = uf_i(u')$ and $f_i(vv') = vf_i(v')$);

ii) $e_i(uu') = e_i(u)u'$ and $e_i(vv') = e_i(v)v'$ (resp. $f_i(uu') = f_i(u)u'$ and $f_i(vv') = f_i(v)v'$).

A crystal structure is called *monomial* if its operators are.

2.2.5. Compatibility with a string data structure. Let $\mathbb{S} = (D, I, R)$ be a string data structure. For $i \in [n-1]$, we say that the crystal operator e_i (resp. f_i) *commutes with* I if, for $u \in S$, such that $e_i(u) \neq 0$ (resp. $f_i(u) \neq 0$), we have

$$e_i(R(C_{\mathbb{S}}(u))) = R(C_{\mathbb{S}}(e_i(u))) \qquad \big(\text{resp. } f_i(R(C_{\mathbb{S}}(u))) = R(C_{\mathbb{S}}(f_i(u)))\big).$$

In that case, we say that the crystal structure C *commutes with the string data structure* $\mathbb{S}$.

2.2.6. Theorem. *Let $\mathbb{S}$ be a weight-preserving string data structure, and C be a monomial crystal structure that commutes with $\mathbb{S}$. Then the relation $\approx_C$ is a congruence on $[n]^*$. Moreover, if there is at most one element of $R_{\mathbb{S}}(D_{\mathbb{S}})$ in each $\approx_C$-class, then $\mathbb{S}$ satisfies the cross-section property for $\approx_C$.*

Proof. We first prove that $\approx_C$ is a congruence on $[n]^*$. Consider u, u', v, v' in $[n]^*$ such that $u \approx_C v$ and $u' \approx_C v'$, that is, there exist crystal isomorphisms $\psi : \Gamma(u) \to \Gamma(v)$ and $\psi' : \Gamma(u') \to \Gamma(v')$. We prove that $uu' \approx_C vv'$ by showing that the following map

$$\psi'' : \Gamma(uu') \to \Gamma(vv')$$

sending any $w = f_{i_1} \dots f_{i_k}(uu')$ in $\Gamma(uv)$ to $f_{i_1} \dots f_{i_k}(vv')$, where $f_{i_1}, \dots, f_{i_k} \in \{(f_i)_{i\in[n-1]}, (e_i)_{i\in[n-1]}\}$ is a crystal isomorphism. This result is a consequence from the facts that C is monomial and ψ and ψ' are crystal isomorphisms, see [3] for the quasi-Kashiwara case. Let us now show that $\mathbb{S}$ satisfies the cross-section property for $\approx_C$. Define first the map

$$\Theta : \Gamma(u) \to \Gamma(R(C_{\mathbb{S}}(u))) \tag{2.2.7}$$

for all u in $[n]^*$, sending any w in $\Gamma(u)$ to $R(C_{\mathbb{S}}(w))$. For every arrow $w \xrightarrow{i} w'$ in $\Gamma(u)$, there is an arrow $\Theta(w) \xrightarrow{i} \Theta(w')$ in $\Gamma(R(C_{\mathbb{S}}(w)))$. Indeed, the equality $f_i(R(C_{\mathbb{S}}(w))) = R(C_{\mathbb{S}}(f_i(w))) = R(C_{\mathbb{S}}(w'))$

(resp. $e_i(R(C_{\mathbb{S}}(w'))) = R(C_{\mathbb{S}}(e_i(w'))) = R(C_{\mathbb{S}}(w))$) holds. Since $\mathbb{S}$ is weight-preserving, we conclude that Θ is a crystal isomorphism. Suppose now that $u \approx_C v$, for all $u, v \in [n]^*$. By isomorphism (2.2.7), we have $u \approx_C R(C_{\mathbb{S}}(u))$ and $v \approx_C R(C_{\mathbb{S}}(v))$. Hence $R(C_{\mathbb{S}}(u)) \approx_C R(C_{\mathbb{S}}(v))$ holds. Since there is at most one element of $R_{\mathbb{S}}(D_{\mathbb{S}})$ in each $\approx_C$-class, the equality $R(C_{\mathbb{S}}(u)) = R(C_{\mathbb{S}}(v))$ holds in $[n]^*$. Since R is injective, we deduce that $C_{\mathbb{S}}(u) = C_{\mathbb{S}}(v)$. Conversely, suppose that $C_{\mathbb{S}}(u) = C_{\mathbb{S}}(v)$. By (2.2.7), we have $u \approx_C R(C_{\mathbb{S}}(u))$ and $v \approx_C R(C_{\mathbb{S}}(v))$, showing that $u \approx_C v$. This proves the result. □

The quotient of the free monoid $[n]^*$ by the congruence $\approx_C$ is called the *crystal monoid of* C, denoted by $\mathbf{M}(C)$. A monomial crystal structure C is called *compatible* with a string data structure $\mathbb{S}$ if the monoids $\mathbf{M}(C)$ and $\mathbf{M}(\mathbb{S})$ are isomorphic.

3. String of columns rewriting

In this section, we study two-dimensional strings defined by gluing columns and called *string of columns*. This combinatorial structure generalizes many structures of tableaux such as skew tableaux, [21], Young tableaux of type A, [22], Young tableaux of type B, C, D and G_2, [11], quasi-ribbon tableaux, [19], and Patience Sorting structures [1]. We give explicitly in (3.2) the plactic-like structures on skew, Young and quasi-ribbon tableaux. We formulate in (3.3) Kashiwara (resp. quasi-Kashiwara) crystal operators in terms of columns by introducing a crystal structure on Scol_n that restricts the Kashiwara (resp. quasi-Kashiwara) crystal structure defined on $[n]^*$ with respect to the reading R_{SW}. Finally, we define in (3.4) the notion of string of columns rewriting system as a binary relation on the set of string of columns, whose rules are applied with respect to right and left positions, and we show how to present a plactic-like structure by such a rewriting system.

3.1. Strings of columns

3.1.1. Columns. A *column* (*over* $[n]$) is a decreasing string $c^k \dots c^1$ over $[n]$, *i.e.*, with $c^{i+1} > c^i$, for $1 \leqslant i < k$. It is represented by a collection of boxes in left-justified rows, filled with elements of $[n]$, whose each row contains only one box, and the entries strictly increase down. A column is pictured by

$$c \;=\; \begin{array}{|c|}\hline c^1 \\ \hline c^2 \\ \hline \vdots \\ \hline c^k \\ \hline \end{array}$$

where $1 \leqslant c^1 < \dots < c^k \leqslant n$, and is also denoted by $(c^1; \dots ; c^k)$. We say that the *i-th box* contains the entry c^i, and that k is the *length* of c, that we will denote by $|c|$. We denote by Col_n the set of columns over $[n]$. A column of length 0 is the *empty column* denoted by λ_c.

3.1.2. String of columns. Two columns c_1, c_2 in Col_n can be *glued at position p* in $\mathbb{Z}$ as follows:

$$c_1 \big|_p c_2 \;=\; \begin{array}{|c|}\hline c_2^1 \\ \hline \vdots \\ \hline \end{array}\!\!\begin{array}{|c|}\hline c_2^i \\ \hline \end{array} \\ \begin{array}{|c|}\hline c_1^1 \\ \hline c_1^2 \\ \hline \vdots \\ \hline c_1^p \\ \hline \vdots \\ \hline c_1^k \\ \hline \end{array}\!\!\begin{array}{|c|}\hline c_2^{i+1} \\ \hline \vdots \\ \hline c_2^{i+p} \\ \hline \end{array}$$

For $1 \leqslant j \leqslant p$, we say that (c_1^j, c_2^{i+j}) is a *full row of length* 2 in $c_1|_p c_2$. A pair $(\square, c_2^j)$, for $1 \leqslant j \leqslant i$, and a pair $(c_1^j, \square)$, for $p+1 \leqslant j \leqslant k$, is called a *row of length* 2, where $\square$ denotes the *empty box*. A *string of columns* is a sequence of glued columns:

$$w = c_1|_{p_1} c_2|_{p_2} \dots c_m|_{p_m} c_{m+1}. \tag{3.1.3}$$

The sequence $(p_1, p_2, \dots, p_m)$ in $\mathbb{Z}^m$ is called the *gluing sequence* of w. Gluing sequences can be defined in a consistent way by considering a *gluing map* $g : \mathrm{Col}_n \times \mathrm{Col}_n \to \mathbb{Z}$ that associates to columns c and c', a gluing position $g(c, c')$. Given a gluing map g, we define the set of strings of columns constructed with respect to g as the set of string of columns of the form (3.1.3), where for any $1 \leqslant i \leqslant m$, $p_i = g(c_i, c_{i+1})$. The *total length of* w is the tuple $tl(w) = (|c_1|, \dots, |c_{m+1}|) \in \mathbb{N}^{m+1}$. We will denote by $||w||$ the number of columns of w. We denote by w^i the i-th row of w, by $|w^i|$ the number of non-empty boxes in w^i, and by w^i_j the element in the j-th box of w^i, where the rows are ordered from top to bottom and their boxes are ordered from left to right.

For $3 \leqslant k \leqslant m+1$, a *connected row of length* k is a sequence $(c_{i_1}^{j_1}, \dots, c_{i_k}^{j_k})$ such that $(c_{i_l}^{j_l}, c_{i_{l+1}}^{j_{l+1}})$ is a full row of length 2 in $c_{i_l}|_{p_{i_l}} c_{i_{l+1}}$, for $1 \leqslant l \leqslant k$. A connected row $(c_{i_1}^{j_1}, \dots, c_{i_k}^{j_k})$ is *increasing* if $c_{i_1}^{j_1} \leqslant \dots \leqslant c_{i_k}^{j_k}$. We call a *row* (*over* $[n]$) a string of columns whose gluing sequence is constant equal to 1 and columns are of length 1. A string of columns is *row connected* (resp. *row increasing*) if all its rows are connected (resp. increasing).

Given a fixed gluing map g, we define the *shape* of a row connected string of columns, constructed with respect the gluing g, and containing k rows as the sequence $(\lambda_1, \dots, \lambda_k) \in \mathbb{N}^k$ such that the i-th row contains λ_i boxes for $1 \leqslant i \leqslant k$.

3.1.4. Monoids of string of columns. We will denote by Scol_n the set of strings of columns over $[n]$ and by $\mathrm{Scol}_n^{\leqslant}$ the set of row connected and row increasing string of columns over $[n]$. We denote by String_n the subset of Scol_n made of strings of columns of length 1 and the gluing sequence contains only 1.

Given a fixed gluing map g, we define a concatenation operation with respect to g by the map $\cdot|_g \cdot : \mathrm{Scol}_n \times \mathrm{Scol}_n \to \mathrm{Scol}_n$, by setting

$$(c_1|_{p_1} \dots |_{p_m} c_{m+1}) \,|_g\, (c'_1|_{q_1} \dots |_{q_n} c'_{n+1}) = c_1|_{p_1} \dots |_{p_m} c_{m+1}|_{g(c_{m+1}, c'_1)} c'_1|_{q_1} \dots |_{q_n} c'_{n+1}.$$

The operation $|_g$ is associative and unitary, where the identity is the *empty string of columns* denoted by λ_c. We denote by Scol_n^g the set of string of columns in Scol_n whose gluing sequence is given by the gluing map g. In other words, Scol_n^g is the free monoid on Col_n with respect the product $|_g$.

3.1.5. The four corner readings. The *south-west reading* is the map $R_{SW} : \mathrm{Scol}_n \to [n]^*$ that reads a string of columns, column-wise from left to right and from bottom to top. There are three other corner readings R_{NW}, R_{NE}, R_{SW} and R_{SE} defined in a similar way and that read a string of columns, column by column, with respect right or left and top or bottom directions.

Define the map $\mathrm{Fac} : [n]^* \to [n]^*$ sending a word w into the factorization $w = w_1 \dots w_k$, where each w_i, for $i = 1, \dots, k$, is a maximal strictly decreasing sequence, that is, the R_{SW}-reading of a column in Col_n. For a fixed gluing map $g \in \mathbb{Z}^n$, consider the map

$$[\,]_g : [n]^* \to \mathrm{Scol}_n \tag{3.1.6}$$

that transforms each word w in $[n]^*$ into a string of columns $(c_1, \dots, c_k)$ where each column c_i is filled by the elements of w_i in $\mathrm{Fac}(w)$ from bottom to top, for $i = 1, \dots, k$, with respect the gluing map g.

3.1.7. Properties of strings of columns. A row connected string of columns w as in (3.1.3) is called

i) *left-justified* (resp. *right-justified*) if $|c_i| \geqslant |c_{i+1}|$ and $|c_{i+1}| \leqslant p_i \leqslant |c_i|$ (resp. $|c_{i+1}| \geqslant |c_i|$ and $|c_i| \leqslant p_i \leqslant |c_{i+1}|$), for all $1 \leqslant i \leqslant m$.

ii) *top-justified*, (resp. *bottom-justified*) if $p_i = |c_{i+1}|$ (resp. $p_i = |c_i|$), for all $1 \leqslant i \leqslant m$.

iii) *decreasing* (resp. *increasing*) if its gluing sequence is decreasing (resp. increasing).

3.1.8. Plactic-like structures. A *plactic-like structure* (*over* $[n]$) is a string data structure $\mathbb{S}$, whose underlying set $D_{\mathbb{S}}$ is a subset of $\mathrm{Scol}_n^{\leqslant}$, with a fixed gluing map, and whose reading is a four corner reading. A monoid is called *plactic-like* if it is isomorphic to the structure monoid of a plactic-like structure. In the rest of this article we suppose that any plactic-like structure is defined with respect the south-west reading R_{SW}.

3.2. Examples of plactic-like structures

3.2.1. Skew tableaux. A *skew tableau* with $m+1$ columns is a string of columns $w = c_1|_{p_1} \dots |_{p_m} c_{m+1}$ in $\mathrm{Scol}_n^{\leqslant}$, whose gluing sequence satisfies $p_k \leqslant |c_{k+1}|$, for all $1 \leqslant k \leqslant m$. A *diagonal skew tableau* is a skew tableau $c_1|_{p_1} \dots |_{p_m} c_{m+1}$ whose gluing sequence satisfies $p_k = 1$, for all $1 \leqslant k \leqslant m$. We denote by s the gluing map for diagonal skew tableaux. We will denote by Sk_n (resp. dSk_n) the set of skew (resp. diagonal skew) tableaux over $[n]$. Any string u over $[n]$ is the R_{SW}-reading of a unique diagonal skew tableau, thus the map R_{SW} defines a bijection from dSk_n to $[n]^*$. Consider a skew tableau w. An *inner corner* in w is an empty box located above and to the left of two non-empty boxes. An *outer corner* in w is an empty box located to the end of a row or at the bottom of a column or to the right and beyond of two non-empty boxes.

3.2.2. Young and quasi-ribbon tableaux. A *Young* (resp. *quasi-ribbon*) *tableau* with $m+1$ columns is a string of columns $w = c_1|_{p_1} \dots |_{p_m} c_{m+1}$ in $\mathrm{Scol}_n^{\leqslant}$ whose gluing sequence is decreasing and satisfies

$$p_k = |c_{k+1}| \quad (\text{resp. } p_k = |c_k| + |c_{k+1}| - 1), \quad \text{for all } 1 \leqslant k \leqslant m. \tag{3.2.3}$$

We denote by $\mathcal{Y}$ the gluing map for Young tableaux, and by Yt_n (resp. Qr_n) the set of Young (resp. quasi-ribbon) tableaux over $[n]$.

3.2.4. Top and bottom concatenation. We define the *top* (resp. *bottom*) *concatenation* of an element x in a column $c = (c^1; \dots; c^k)$ as the skew tableau defined by

$$c \leftsquigarrow_a x = \begin{cases} c|_1 x & \text{if } x \geqslant c^1, \\ (x; c^1; \dots; c^k) & \text{else.} \end{cases} \qquad \Big(\text{resp. } x \rightsquigarrow_a c = \begin{cases} (c^1; \dots; c^k; x) & \text{if } x > c^k, \\ x|_1 c & \text{else.} \end{cases}\Big).$$

We extend these concatenations into insertions maps on skew tableaux, defined for $x \in [n]$ and $w = c_1|_{p_1} \dots |_{p_m} c_m$ in Sk_n by setting

$$w \leftsquigarrow_{I_r^a} x = c_1|_{p_1} \dots |_{p_m} (c_m \leftsquigarrow_a x), \quad (\text{resp. } x \rightsquigarrow_{I_l^a} w = (x \rightsquigarrow_a c_1)|_{p_1} \dots |_{p_m} c_m\ ). \tag{3.2.5}$$

The top and bottom concatenation on diagonal skew tableaux given in (3.2.5) define two plactic-like structures on diagonal skew tableaux: a right one $d\mathbb{S}_n^r := (\mathrm{dSk}_n, I_r^a)$ and a left one $d\mathbb{S}_n^c := (\mathrm{dSk}_n, I_l^a)$.

The top (resp. bottom) concatenation on a diagonal skew tableau w acts only on the last (resp. first) column of w and do not change the others columns. As a consequence, for all $x, y \in [n]$, we have the following commutation property:

$$y \rightsquigarrow_{I_l^a} (w \leftsquigarrow_{I_r^a} x) = (y \rightsquigarrow_{I_l^a} w) \leftsquigarrow_{I_r^a} x. \tag{3.2.6}$$

Hence the insertions maps I_r^a and I_l^a commute, and the structure monoid $\mathbf{M}(d\mathbb{S}_n^r)$ is isomorphic to $[n]^*$.

3.2.7. Schensted's insertion algorithms, [20]. Given a row r (resp. a column c), we denote by RowInsert(r, x) (resp. ColumnInsert(c, x)) the procedure that inserts an element x in a row r (resp. column c) and returns a pair (r', y) (resp. (c', y)) made of the resulting row r' (resp. column c') and the bumping element y that can be empty, as follows. If x is bigger or equal (resp. strictly bigger) than all the elements of r (resp. c), then r' (resp. c') is obtained by adding x to the end (resp. the bottom) of r (resp. c) and y is empty. Otherwise, let y be the smallest element of r (resp. c) such that $x < y$ (resp. $x \leqslant y$), then r' (resp. c') is obtained from r (resp. c) by replacing y by x. The *right* (resp. *left*) *insertion algorithm* computes a tableau $(t \leftsquigarrow_{S_r} x)$ (resp. $(x \rightsquigarrow_{S_l} t)$) as follows:

RightInsertYT(t, x)
Input: A Young tableau t and x in $[n]$.
Output: The Young tableau $(t \leftsquigarrow_{S_r} x)$.

$y := x$; $t' := \emptyset$;
while $y \neq \lambda$ **do**
  $r := t[1]$;
  $t := t/r$;
  $(r', y) :=$ RowInsert(r, y)
  $t' := (t'; r')$
end
return $(t'; t)$

Algorithm 1: Schensted's right algorithm

LeftInsertYT(t, x)
Input: A Young tableau t and x in $[n]$.
Output: The Young tableau $(x \rightsquigarrow_{S_l} t)$.

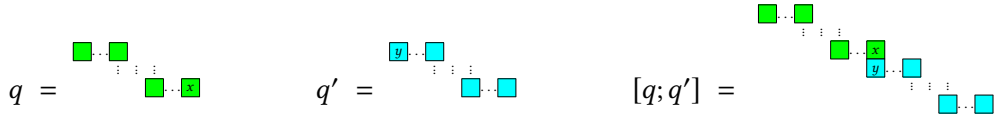
$y := x$; $t' := \emptyset$;
while $y \neq \lambda$ **do**
  $c := t_{[1]}$;
  $t := t_{/c}$;
  $(c', y) :=$ ColumnInsert(c, y)
  $t' := [t'; c']$
end
return $[t'; t]$

Algorithm 2: Schensted's left algorithm

where $t[i]$ (resp. $t_{[i]}$) denotes the i-th row (resp. column) of the tableau t, and $t/t[1]$ (resp. $t_{/t_{[1]}}$) the Young tableau obtained from t by removing its first row (resp. column), and where $(t; t')$ (resp. $[t; t']$) denotes the Young tableau obtained by concatenating t over (resp. to the right of) a Young tableau t' when the concatenation defines a Young tableau.

These two algorithms define two plactic-like structures on Young tableaux: a right one $\mathbb{Y}_n^r := (\mathrm{Yt}_n, S_r)$ and a left one $\mathbb{Y}_n^c := (\mathrm{Yt}_n, S_l)$.

3.2.8. Insertion algorithms in quasi-ribbon tableaux. Given two quasi-ribbon tableaux q and q', we denote by $[q; q']$ the quasi-ribbon tableau obtained by concatenating q over q' by gluing the rightmost and bottommost box b of q over the leftmost and topmost box b' of q', when the element of b is strictly smaller than the element of b', as illustrated in the following diagrams:



where $1 \leqslant x < y \leqslant n$. In this way, we will represent a quasi-ribbon tableau q with k rows by $[q^1; \ldots; q^k]$.

Recall from [19, Algorithm 4.4] (resp. [2, Algorithm 4.4]) the right (resp. left) insertion algorithm, denoted by $(q \leftsquigarrow_{H_r} x)$ (resp. $(x \rightsquigarrow_{H_l} q)$) that inserts an element x in $[n]$ into a quasi-ribbon tableau q.

RightInsertQR(q, x)
Input: A quasi-ribbon q with R rows and x in $[n]$.
Output: The quasi-ribbon tableau $q \leftsquigarrow_{H_r} x$.

$q' := \emptyset$; $q'' := \emptyset$; $r := \emptyset$; $r' := \emptyset$;
if $x < q^1_1$ **then**
  $q' = (x)$;
  return $[q'; q]$
end
for $i = R, \ldots, 1$ **do**
  for $j = |q^i|, \ldots, 1$ **do**
    if $x \geqslant q^i_j$ **then**
      $r = (q^i_1, \ldots, q^i_j, x)$;
      $r' = (q^i_{(j+1)}, \ldots, q^i_{|q^i|})$;
      $q' = [q^1; \ldots; q^{i-1}; r]$;
      $q'' = [r'; q^{i+1}; \ldots; q^R]$;
      return $[q'; q'']$
    end
  end
end

Algorithm 3: Right quasi-ribbon insertion

LeftInsertQR(q, x)
Input: A quasi-ribbon q with R rows and x in $[n]$.
Output: The quasi-ribbon tableau $x \rightsquigarrow_{H_l} q$.

$q' := \emptyset$; $q'' := \emptyset$; $r := \emptyset$; $r' := \emptyset$;
if $x > q^R_{|q^R|}$ **then**
  $q' = (x)$;
  return $[q; q']$
end
for $i = 1, \ldots, R$ **do**
  for $j = 1, \ldots, |q^i|$ **do**
    if $x \leqslant q^i_j$ **then**
      $r = (x, q^i_j, \ldots, q^i_{|q^i|})$;
      $r' = (q^i_1, \ldots, q^i_{(j-1)})$;
      $q' = [q^1; \ldots; q^{i-1}; r']$;
      $q'' = [r; q^{i+1}; \ldots; q^R]$;
      return $[q'; q'']$
    end
  end
end

Algorithm 4: Left quasi-ribbon insertion

Left and right insertion H_r and H_l induce two string data structures on Qr_n: a right one $\mathbb{Q}^r_n := (\mathrm{Qr}_n, H_r)$ and a left one $\mathbb{Q}^l_n := (\mathrm{Qr}_n, H_l)$.

3.3. Crystal structures on strings of columns

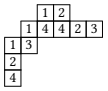
3.3.1. Kashiwara crystal operators on Scol_n. For i in $[n-1]$, we define operators

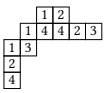
$$\widetilde{\mathbf{e}}_i, \widetilde{\mathbf{f}}_i : \mathrm{Scol}_n \to \mathrm{Scol}_n \cup \{0\}, \tag{3.3.2}$$

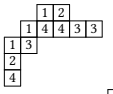
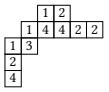
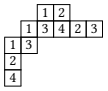
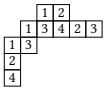
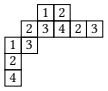
as follows. A string of columns w in Scol_n is transformed into a string over $\{+, -\}$ by replacing each column of w, from left to right, by the symbol + if the column contains the letter i and does not contain the letter $i+1$, the symbol $-$ if the column contains the letter $i+1$ and does not contain the letter i, the empty string λ, in the other cases, and by keeping a record of the original column replaced by each symbol. Then we eliminate the substrings $-+$ until no such substrings remain. The final string will be of the form $+^r -^l$.

i) If $r > 0$, then $\widetilde{\mathbf{f}}_i(w)$ is obtained from w by considering the column that was replaced by the rightmost symbol + and replace its element i by $i+1$. If $r = 0$, then $\widetilde{\mathbf{f}}_i(w) = 0$.

ii) If $l > 0$, then $\widetilde{\mathbf{e}}_i(w)$ is obtained from w by considering the column that was replaced by the leftmost symbol $-$ and replace its element $i+1$ by i. If $l = 0$, then $\widetilde{\mathbf{e}}_i(w) = 0$.

For instance, consider $w =$  in Scol_4. Compute $\widetilde{\mathbf{f}}_2(w)$ and $\widetilde{\mathbf{e}}_2(w)$.

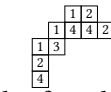
 $\longmapsto$ $+-\lambda++-$ $\longmapsto$ $++- = +^2-^1$

Then $\widetilde{\mathbf{f}}_2(w) =$  and $\widetilde{\mathbf{e}}_2(w) =$  . In the same way, we obtain $\widetilde{\mathbf{f}}_3(w) = 0$, $\widetilde{\mathbf{e}}_3(w) =$  , $\widetilde{\mathbf{e}}_1(w) =$  and $\widetilde{\mathbf{f}}_1(w) =$  .

3.3.3. Quasi-Kashiwara crystal operators on Scol_n. For i in $[n-1]$, we define operators

$$\mathbf{e}_i, \mathbf{f}_i : \mathrm{Scol}_n \to \mathrm{Scol}_n \cup \{0\}, \tag{3.3.4}$$

as follows. Let w in Scol_n. If a column of w contains both i and $i+1$, or it contains i and is situated to the right of a column containing $i+1$, then $\mathbf{e}_i(w) = \mathbf{f}_i(w) = 0$. In the other cases, suppose that at least one column of w contains the letter $i+1$ (resp. i), then $\mathbf{e}_i(w)$ (resp. $\mathbf{f}_i(w)$) is obtained from w by considering its leftmost (resp. rightmost) column containing $i+1$ (resp. i) and by replacing the corresponding $i+1$ (resp. i) by i (resp. $i+1$).

For instance, consider $w =$  in Scol_4. Since the first column of w contains 1 and 2, then $\mathbf{f}_1(w) = \mathbf{e}_1(w) = 0$. Since the fourth column of w contains 2 and is situated to the right of the second column containing 3, then $\mathbf{f}_2(w) = \mathbf{e}_2(w) = 0$. In the same way, we obtain $\mathbf{f}_3(w) = \mathbf{e}_3(w) = 0$.

3.3.5. Remarks. By definition, the Kashiwara and quasi-Kashiwara operators on Scol_n preserve gluing sequences. Note also that the Kashiwara (resp. quasi-Kashiwara) crystal structure on Scol_n is a restriction of the Kashiwara (resp. quasi-Kashiwara) crystal structure on $[n]^*$ with respect the reading R_{SW}. Indeed, by definition of the operators $\widetilde{\mathbf{e}}_i$, $\widetilde{\mathbf{f}}_i$, $\mathbf{e}_i$, $\mathbf{f}_i$, for a fixed gluing map g, we have

$$\widetilde{\mathbf{e}}_i(w) = [\widetilde{e}_i(R_{SW}(w))]_g \quad \text{and} \quad \widetilde{\mathbf{f}}_i(w) = [\widetilde{f}_i(R_{SW}(w))]_g, \tag{3.3.6}$$

$$\big(\text{resp. } \mathbf{e}_i(w) = [\ddot{e}_i(R_{SW}(w))]_g \quad \text{and} \quad \mathbf{f}_i(w) = [\ddot{f}_i(R_{SW}(w))]_g\big), \tag{3.3.7}$$

for every w in Scol_n^g, and $i \in [n-1]$. As a consequence, for every w in Scol_n^g, we have

$$R_{SW}(\widetilde{\mathbf{e}}_i(w)) = \widetilde{e}_i(R_{SW}(w)) \quad \text{and} \quad R_{SW}(\widetilde{\mathbf{f}}_i(w)) = \widetilde{f}_i(R_{SW}(w)),$$

$$\big(\text{resp. } R_{SW}(\mathbf{e}_i(w)) = \ddot{e}_i(R_{SW}(w)) \quad \text{and} \quad R_{SW}(\mathbf{f}_i(w)) = \ddot{f}_i(R_{SW}(w))\big),$$

that is, the following diagrams commute

$$\begin{array}{ccc} \mathrm{Scol}_n & \xrightarrow{R_{SW}} & [n]^* \\ {\scriptstyle \widetilde{\mathbf{e}}_i}\downarrow & & \downarrow{\scriptstyle \widetilde{e}_i} \\ \mathrm{Scol}_n & \xrightarrow[R_{SW}]{} & [n]^* \end{array} \qquad \begin{array}{ccc} \mathrm{Scol}_n & \xrightarrow{R_{SW}} & [n]^* \\ {\scriptstyle \widetilde{\mathbf{f}}_i}\downarrow & & \downarrow{\scriptstyle \widetilde{f}_i} \\ \mathrm{Scol}_n & \xrightarrow[R_{SW}]{} & [n]^* \end{array} \qquad \Big(\text{resp. } \begin{array}{ccc} \mathrm{Scol}_n & \xrightarrow{R_{SW}} & [n]^* \\ {\scriptstyle \mathbf{e}_i}\downarrow & & \downarrow{\scriptstyle \ddot{e}_i} \\ \mathrm{Scol}_n & \xrightarrow[R_{SW}]{} & [n]^* \end{array} \qquad \begin{array}{ccc} \mathrm{Scol}_n & \xrightarrow{R_{SW}} & [n]^* \\ {\scriptstyle \mathbf{f}_i}\downarrow & & \downarrow{\scriptstyle \ddot{f}_i} \\ \mathrm{Scol}_n & \xrightarrow[R_{SW}]{} & [n]^* \end{array} \Big).$$

Conversely, for every u in $[n]^*$, we have

$$\widetilde{e}_i(u) = R_{SW}(\widetilde{\mathbf{e}}_i([u]_g)) \quad \text{and} \quad \widetilde{f}_i(u) = R_{SW}(\widetilde{\mathbf{f}}_i([u]_g))$$

$$(\text{resp. } \ddot{e}_i(u) = R_{SW}(\mathbf{e}_i([u]_g)) \quad \text{and} \quad \ddot{f}_i(u) = R_{SW}(\mathbf{f}_i([u]_g)).$$

Indeed, suppose that $\mathrm{Fac}(u) = u_1 \dots u_k$. For a fixed i in $[n-1]$, compute $\widetilde{f}_i(u)$ and $\widetilde{e}_i(u)$. If u_j, for $j = 1, \dots, k$, contains both $i+1$ and i, then $i+1$ and i appear only once in u_j, and $i+1$ is located to the left of i. Otherwise, i or $i+1$ appears only once in u_j. Then, following the definition of $\widetilde{f}_i$ and $\widetilde{e}_i$ recalled in 2.2.2, we replace u_j, for $j = 1, \dots, k$, by $-$ if it contains only $i+1$, by $+$ if it contains only i, and by λ in the other cases. After, ones eliminates the subwords $-+$ until no such subwords remain. We obtain a word of the form $+^r -^l$. If $r > 0$, then $\widetilde{f}_i(u)$ is obtained from w by considering the strictly decreasing subword in $\mathrm{Fac}(w)$ that was replaced by the rightmost symbol $+$ and replace its element i by $i+1$. If $r = 0$, then $\widetilde{f}_i(u) = 0$. If $l > 0$, then $\widetilde{e}_i(u)$ is obtained from u by considering the strictly decreasing subword in $\mathrm{Fac}(u)$ that was replaced by the leftmost symbol $-$ and replace its element $i+1$ by i. If $l = 0$, then $\widetilde{e}_i(u) = 0$. Hence, $\widetilde{e}_i(u) = R_{SW}(\widetilde{\mathbf{e}}_i([u]_g))$ and $\widetilde{f}_i(u) = R_{SW}(\widetilde{\mathbf{f}}_i([u]_g))$, for a fixed gluing map g. Similarly, we show that $\ddot{e}_i(u) = R_{SW}(\mathbf{e}_i([u]_g))$ and $\ddot{f}_i(u) = R_{SW}(\mathbf{f}_i([u]_g))$.

As a consequence, the crystal operators (3.3.2) (resp. (3.3.4)) define a crystal structure on Scol_n, denoted by $\overline{\mathcal{K}}$ (resp. $\overline{q\mathcal{K}}$), that coincides with the crystal structure $\mathcal{K}$ (resp. $q\mathcal{K}$) on $[n]^*$, in the sense that the following diagrams commute

$$\begin{array}{ccc} \mathrm{Scol}_n & \xleftarrow{[\,]_g} & [n]^* \\ {\scriptstyle \widetilde{\mathbf{e}}_i}\downarrow & & \downarrow{\scriptstyle \widetilde{e}_i} \\ \mathrm{Scol}_n & \xrightarrow[R_{SW}]{} & [n]^* \end{array} \qquad \begin{array}{ccc} \mathrm{Scol}_n & \xleftarrow{[\,]_g} & [n]^* \\ {\scriptstyle \widetilde{\mathbf{f}}_i}\downarrow & & \downarrow{\scriptstyle \widetilde{f}_i} \\ \mathrm{Scol}_n & \xrightarrow[R_{SW}]{} & [n]^* \end{array} \qquad \left(\text{resp. } \begin{array}{ccc} \mathrm{Scol}_n & \xleftarrow{[\,]_g} & [n]^* \\ {\scriptstyle \mathbf{e}_i}\downarrow & & \downarrow{\scriptstyle \ddot{e}_i} \\ \mathrm{Scol}_n & \xrightarrow[R_{SW}]{} & [n]^* \end{array} \right. \qquad \left. \begin{array}{ccc} \mathrm{Scol}_n & \xleftarrow{[\,]_g} & [n]^* \\ {\scriptstyle \mathbf{f}_i}\downarrow & & \downarrow{\scriptstyle \ddot{f}_i} \\ \mathrm{Scol}_n & \xrightarrow[R_{SW}]{} & [n]^* \end{array} \right).$$

Denoting $\Gamma(\overline{\mathcal{K}}, d)$ (resp. $\Gamma(\overline{q\mathcal{K}}, d)$) the connected component of d in Scol_n with respect $\widetilde{\mathbf{e}}_i$ and $\widetilde{\mathbf{f}}_i$ (resp. $\mathbf{e}_i$ and $\mathbf{f}_i$), we have a set-theoretical isomorphism of connected component $\Gamma(\overline{\mathcal{K}}, d) \simeq \Gamma(q\mathcal{K}, R_{SW}(d))$ (resp. $\Gamma(q\overline{\mathcal{K}}, d) \simeq \Gamma(q\mathcal{K}, R_{SW}(d))$). As a consequence, the following two properties hold:

i) The vertex set of every connected component of $\Gamma(\overline{\mathcal{K}}, d)$ is a union of vertex sets of connected component of $\Gamma(\overline{q\mathcal{K}}, d)$, for every $d \in \mathrm{Scol}_n$, [3, Proposition 8.4].

ii) Let $\psi : \Gamma(\overline{\mathcal{K}}, d) \to \Gamma(\overline{\mathcal{K}}, \psi(d))$ be a crystal isomorphism. Then, ψ restricts to a quasi-crystal isomorphism $\psi : \Gamma(\overline{q\mathcal{K}}, d') \to \Gamma(\overline{q\mathcal{K}}, \psi(d'))$, for every d' in $\Gamma(\overline{\mathcal{K}}, d)$, [3, Proposition 8.5].

3.3.8. Proposition. *Let $\mathbb{S}$ and $\mathbb{S}'$ be two plactic-like structures. Let C be a crystal structure on $[n]^*$ compatible with $\mathbb{S}'$. Any morphism of string data structures $f : \mathbb{S} \to \mathbb{S}'$ induces a crystal isomorphism from $\Gamma(C, R(d))$ to $\Gamma(C, R'(f(d)))$, for every d in $D_{\mathbb{S}}$.*

Proof. By definition of f, we have $C_{\mathbb{S}'}(R(d)) = C_{\mathbb{S}'}(R'(f(d)))$, for every d in $D_{\mathbb{S}}$. In an other hand, the crystal structure C being compatible with $\mathbb{S}'$, the crystal monoid $\mathbf{M}(C)$ is isomorphic to the structure monoid $\mathbf{M}(\mathbb{S}')$. As a consequence, $R(d)$ and $R'(f(d))$ are equal in the crystal monoid $\mathbf{M}(C)$, and thus the connected components $\Gamma(C, R(d))$ and $\Gamma(C, R'(f(d)))$ are isomorphic, for every d in $D_{\mathbb{S}}$. □

3.3.9. Crystal operators on $\mathrm{Scol}_n^{\leqslant}$. We define a restriction of crystal operators on $\mathrm{Scol}_n^{\leqslant}$. For i in $[n-1]$ and w in $\mathrm{Scol}_n^{\leqslant}$, we define the operators $\widetilde{\mathbf{e}}_i, \widetilde{\mathbf{f}}_i : \mathrm{Scol}_n^{\leqslant} \to \mathrm{Scol}_n^{\leqslant} \cup \{0\}$, as in (3.3.1), except for conditions **i)** and **ii)**, defined as follows

i)' If $r > 0$, consider the diagram d obtained from w by considering the column that was replaced by the rightmost symbol + and replace its element i by $i+1$. If $d \in \mathrm{Scol}_n^{\leqslant}$, then $\widetilde{\mathbf{f}}_i(w) = d$. Otherwise, $\widetilde{\mathbf{f}}_i(w) = 0$. If $r = 0$, then $\widetilde{\mathbf{f}}_i(w) = 0$.

ii)' If $l > 0$, consider the diagram d' obtained from w by considering the column that was replaced by the leftmost symbol − and replace its element $i+1$ by i. If $d' \in \mathrm{Scol}_n^{\leqslant}$, then $\widetilde{\mathbf{e}}_i(w) = d'$. Otherwise, $\widetilde{\mathbf{e}}_i(w) = 0$. If $l = 0$, then $\widetilde{\mathbf{e}}_i(w) = 0$.

Note that by restricting this crystal structure to the set of quasi-ribbon tableaux, we recover the Krob–Thibon quasi-crystal introduced in [14].

3.3.10. Proposition. *Let $\mathbb{S}$ be a plactic-like structure, and C be the crystal structure on $\mathrm{Scol}_n^{\leqslant}$ presented in* (3.3.9)*. Let w be in $\mathrm{Scol}_n^{\leqslant}$ with k rows, whose i-th row consists entirely of i, for $1 \leqslant i \leqslant k$, then w is of highest-weight.*

Proof. Let w be in $\mathrm{Scol}_n^{\leqslant}$ with k rows, where each i-th row consists entirely of i, for $1 \leqslant i \leqslant k$. The symbols i and $i+1$ appear together in each column of w, for $1 \leqslant i \leqslant k-1$. Then $\widetilde{\mathbf{e}}_i(w) = 0$, for all $1 \leqslant i \leqslant k-1$. Moreover, for $i = k, \ldots, n-1$, the symbol $i+1$ do not appear in w, and then $\widetilde{\mathbf{e}}_i(w) = 0$, showing the claim. □

Note that the converse of this property is not true in general. For example, for the following skew tableau $w = \begin{array}{|c|c|}\hline 1 & 1 \\ \hline 1 & 2 \\ \hline \end{array}$ in Sk_3, we have $\widetilde{\mathbf{e}}_i(w) = 0$ but its second row does not consist entirely of 2. However, the converse of this property is true in the case of Young (resp. quasi-ribbon) tableaux.

3.3.11. Proposition. *The set of Young (resp. quasi-ribbon) tableaux of a given shape λ forms single connected components with a unique highest-weight string of columns with respect to the crystal structure on $\mathrm{Scol}_n^{\leqslant}$ presented in* (3.3.9)*.*

Proof. Let w be a highest-weight Young (resp. quasi-ribbon) tableau of shape λ, that is, $\widetilde{\mathbf{e}}_i(w) = 0$, for all i in $[n-1]$. By definition, $\widetilde{\mathbf{e}}_i(w) = 0$ if and only if w consists of columns which contain both i and $i+1$, columns which contain neither i or $i+1$, columns containing i without $i+1$ and columns containing $i+1$ without i, such that each column containing only the symbol $i+1$ is followed by a column that contains only the symbol i. Since w is a Young (resp. quasi-ribbon) tableau with a fixed shape λ, then w is unique. Moreover, by definition of the crystal operators on $\mathrm{Scol}_n^{\leqslant}$, applying these operators does not change the gluing sequence of a given tableau and then preserves its shape. Hence, starting from a unique highest-weight Young (resp. quasi-ribbon) tableau of shape λ and by applying the crystal operators on it we recover all the tableaux of the same shape, showing the claim. □

3.3.12. Proposition. *Let $\mathbb{S}$ be a weight-preserving string data structure, and C be a monomial crystal structure that commutes with $\mathbb{S}$, such that there is at most one element of $R_{\mathbb{S}}(D_{\mathbb{S}})$ in each $\approx_C$-class. If each connected component $\Gamma(C, R(d))$, for $d \in D_{\mathbb{S}}$, contains a unique highest-weight string of columns, then in every connected component in $\Gamma(C)$ there is a unique highest-weight word.*

Proof. Consider u in $[n]^*$. By Theorem 2.2.6, the monoids $[n]^*/\approx_C$ and $\mathbf{M}(\mathbb{S})$ are isomorphic. Since $C_{\mathbb{S}}(u) = C_{\mathbb{S}}(R(C_{\mathbb{S}}(u)))$, the connected component $\Gamma(C, u)$ is isomorphic to $\Gamma(C, R(C_{\mathbb{S}}(u)))$ which contains a unique highest-weight string of columns. Hence, every connected component in $\Gamma(C)$ contains a unique highest-weight word. □

3.3.13. Example:Kashiwara and quasi-Kashiwara crystals. Consider the Kashiwara crystal structure $\mathcal{K} = (\widetilde{e_i}, \widetilde{f_i})$ on $[n]^*$. This crystal structure is monomial and commutes with $\mathbb{Y}_n^r$, [11, 15]. Moreover, there is at most one element of $R_{SW}(\mathrm{Yt}_n)$ in each $\approx_{\mathcal{K}}$-class. Indeed, if two Young tableaux are at the same place in two isomorphic connected components of the crystal graph of $\mathcal{K}$, then they have the same weight and thus they are equal. Note also that the plactic-like structure $\mathbb{Y}_n^r$ is weight-preserving. Then, by Theorem 2.2.6, the crystal structure $\mathcal{K}$ is compatible with $\mathbb{Y}_n^r$, and its crystal monoid is isomorphic to $\mathbf{M}(\mathbb{Y}_n^r)$. Moreover, we recover from Proposition 3.3.12 that in every connected component in $\Gamma(\mathcal{K})$ there is a unique highest-weight word. Consider now the quasi-Kashiwara crystal structure $q\mathcal{K} = (\ddot{e}_i, \ddot{f}_i)$ on $[n]^*$. This crystal structure is monomial, [3, Lemma 5.4], and commutes with $\mathbb{Q}_n^r$, [3, Lemma 6.9]. Moreover, there is at most one element of $R_{SW}(\mathrm{Qr}_n)$ in each $\approx_{q\mathcal{K}}$-class, [3, Proposition 6.7]. Note also that $\mathbb{Q}_n^r$ is weight-preserving. Then, by Theorem 2.2.6, the crystal structure $q\mathcal{K}$ is compatible with $\mathbb{Q}_n^r$, and its crystal monoid is isomorphic to $\mathbf{M}(\mathbb{Q}_n^r)$. Moreover, we recover from Proposition 3.3.12 that in every connected component in $\Gamma(q\mathcal{K})$ there is a unique highest-weight word, [3, Proposition 6.13].

3.4. String of columns rewriting

3.4.1. Rewriting steps. Let define $\mathrm{Scol}_n^{\mathcal{P}} = \mathbb{Z}\times\mathrm{Scol}_n\times\mathbb{Z}$, whose elements are triples (p, u, q) where p, q are positions in $\mathbb{Z}$ and u is a string of columns, that we will denote by $|_p u|_q$. We define a *string of columns rewriting system*, called *rewriting system* for short in the sequel, as a binary relation on $\mathrm{Scol}_n^{\mathcal{P}}$. That is, a set of *rules* of the form

$$\alpha : |_{p_s} u|_{q_s} \Rightarrow |_{p_t} v|_{q_t}, \tag{3.4.2}$$

where u, v are strings of columns in Scol_n and p_s, q_s, p_t, q_t are positions in $\mathbb{Z}$. The pair (p_s, q_s) (resp. (p_t, q_t)) is called the *source* (resp. *target*) *positions*, and u (resp. v) is called the *string of columns source* (resp. *target*) of the rule α, denoted by $s(\alpha)$ (resp. $t(\alpha)$).

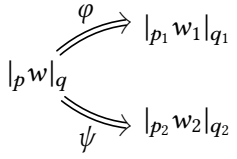
A string of columns w is said to be *reducible with respect to* α, if there is a factorization $w = w_1|_{p_s} s(\alpha)|_{q_s} w_2$ in Scol_n. In that case, w *reduces into* $w' = w_1|_{p_t} t(\alpha)|_{q_t} w_2$. Such a *reduction* is denoted by $w_1|_{p_s}\alpha|_{q_s} w_2$, or α if there is no possible confusion. Given a rewriting system $\mathcal{R}$, the set of all reductions defines a binary relation on Scol_n, called the $\mathcal{R}$*-rewrite relation* that we will denote by $\Rightarrow_{\mathcal{R}}$, or $\Rightarrow$ if there is no possible confusion. The elements of $\Rightarrow_{\mathcal{R}}$ are called $\mathcal{R}$*-rewriting steps*, and have the form

$$|_{p_1} w_1|_{p_s} s(\alpha)|_{q_s} w_2|_{p_2} \Rightarrow |_{p_1} w_1|_{p_t} t(\alpha)|_{q_t} w_2|_{p_2}, \tag{3.4.3}$$

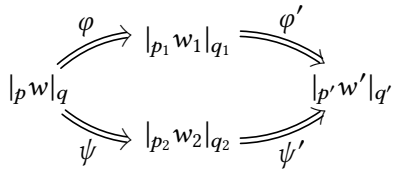
for all α in $\mathcal{R}$ and w_1, w_2 in Scol_n. In (3.4.3) the data $|_{p_1} w_1|_{p_s} - |_{q_s} w_2|_{p_2}$ is called the *context* of the rule α. If we denote by C this context, the reduction (3.4.3) can be also denoted by $C[\alpha]$.

We denote by $\Rightarrow_{\mathcal{R}}^*$ the reflexive and transitive closure of the relation $\Rightarrow_{\mathcal{R}}$, whose elements are called $\mathcal{R}$*-rewriting paths*.

3.4.4. Rewriting properties. We say that a rewriting system $\mathcal{R}$ is *terminating* if there is no infinite $\mathcal{R}$-rewriting path. A *local branching* (resp. *branching*) of $\mathcal{R}$ is a pair (φ, ψ) of $\mathcal{R}$-rewriting steps (resp. $\mathcal{R}$-rewriting paths) having the same source as depicted in the following reduction diagram:



Such a branching is *confluent* if there exist $\mathcal{R}$-rewriting paths φ' and ψ' with a common target as follows:



(3.4.5)

We say that $\mathcal{R}$ is *locally confluent* (resp. *confluent*) if any local branching (resp. branching) of $\mathcal{R}$ is confluent, and that $\mathcal{R}$ is convergent if it is confluent and terminating. A string of columns w is in *normal form with respect to* $\mathcal{R}$, if there is no rule that reduces w. When $\mathcal{R}$ is convergent, any string of columns w has a unique normal form, denoted by $\mathrm{Nf}(w, \mathcal{R})$.

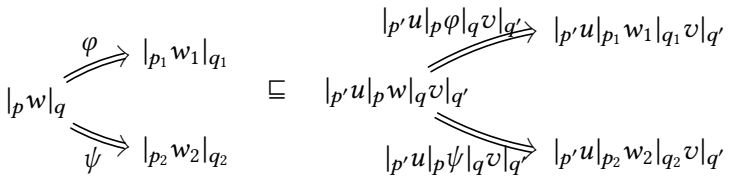
3.4.6. Critical branching. A local branching of the form (φ, φ) is called *aspherical*. A local branching (φ, ψ) is called *orthogonal* if the source of φ does not *overlap* with the source of ψ, that is the source of the branching is of the form $|_{p_0} w_1 |_{p_1} s(\varphi) |_{q_1} w_2 |_{p_2} s(\psi) |_{q_2} w_3 |_{p_3}$, with w_1, w_2, w_3 in Scol_n. A local branching that is neither aspherical nor orthogonal is called *overlapping*. There are three shapes of overlapping branchings (φ, ψ), where $s(\varphi) = |_{p_0} w_1 |_{p_s} s(\alpha) |_{q_s} w_2 |_{p_2}$, and $s(\psi) = |_{p'_0} w'_1 |_{p'_s} s(\beta) |_{q'_s} w'_2 |_{p'_2}$, with $\alpha, \beta \in \mathcal{R}$, described by the following situations:

i) (*position overlapping*) $q_s = q'_s$,

ii) (*string overlapping*) $s(\alpha) = |_{p_s} u_\alpha |_{p'_s} v_\alpha |_{q_s}$ and $s(\beta) = |_{p'_s} v_\alpha |_{q_s} u_\beta |_{q'_s}$,

iii) (*inclusion*) $s(\beta) = |_{p'_s} u_\beta |_{p_s} s(\alpha) |_{q_s} v_\beta |_{q'_s}$.

An overlapping branching that is minimal for the relation $\sqsubseteq$ on branchings generated by



for any branching (φ, ψ) and context $u|_p - |_q v$ of reductions φ, ψ, is called a *critical branching*.

3.4.7. Lemma. *A rewriting system $\mathcal{R}$ is locally confluent if and only if all its critical branchings are confluent. Moreover, if $\mathcal{R}$ is terminating with all its critical branchings are confluent, then it is confluent.*

Proof. The first statement is the critical branching lemma. Suppose that all the critical branchings of $\mathcal{R}$ are confluent and prove that any local branching of $\mathcal{R}$ is confluent. By definition, every aspherical branching is trivially confluent, and every orthogonal local branching is confluent. Consider an overlapping but not minimal local branching (φ, ψ), there exist factorizations $\varphi = C[\varphi']$ and $\psi = C[\psi']$, where (φ', ψ') is a critical branching of $\mathcal{R}$. By hypothesis, this branching is confluent, and there are reductions paths $\varphi'' : t(\varphi') \to w$ and $\psi'' : t(\psi') \to w$ that reduce targets of φ' and ψ' to the same string of columns w. It follows that the reductions paths $C[\varphi'']$ and $C[\psi'']$ make the branching (φ, ψ) confluent.

The second statement is an immediate consequence of Newman's lemma, [18], that proves that any locally confluent terminating rewriting system is confluent. □

3.4.8. Normalization strategies. A *reduction strategy* for a rewriting system $\mathcal{R}$ specifies a way to apply the rules in a deterministic way. When $\mathcal{R}$ is normalizing, a *normalization strategy* is a mapping σ of every string of columns $|_p w|_q$ to a rewriting path $\sigma_{|_p w|_q}$ with source $|_p w|_q$ and target a chosen normal form of $|_p w|_q$ with respect to $\mathcal{R}$. For a reduced rewriting system, we distinguish the leftmost reduction strategy and the rightmost one, according to the way we apply first the rewriting rule that reduces the leftmost or the rightmost string of columns. They are defined as follows. For every string of columns $|_p w|_q$, the set of rewriting steps with source $|_p w|_q$ can be ordered from left to right by setting $\varphi \prec \psi$, for rewriting steps $\varphi = |_p w_1|_{p_s} \alpha |_{q_s} w_2|_q$ and $\varphi = |_p w_1'|_{p_s} \beta |_{q_s} w_2'|_q$ such that $||w_1|| < ||w_1'||$. If $\mathcal{R}$ is finite, then the order $\prec$ is total and the set of rewriting steps of source $|_p w|_q$ is finite. Hence this set contains a smallest element $\sigma_{|_p w|_q}$ and a greatest element $\eta_{|_p w|_q}$, respectively called the *leftmost* and the *rightmost rewriting steps on* $|_p w|_q$. If, moreover, the rewriting system terminates, the iteration of σ (resp. η) yields a normalization strategy for $\mathcal{R}$ called the *leftmost* (resp. *rightmost*) *normalization strategy of* $\mathcal{R}$:

$$\rho^{\top}_{\mathcal{R}}(|_p w|_q) = \sigma_{|_p w|_q} | \rho^{\top}_{\mathcal{R}}(t(\sigma_{|_p w|_q})) \qquad (\text{resp. } \rho^{\perp}_{\mathcal{R}}(|_p w|_q) = \eta_{|_p w|_q} | \rho^{\perp}_{\mathcal{R}}(t(\eta_{|_p w|_q})) \ ).$$

The *leftmost* (resp. *rightmost*) *rewriting path* on a string of columns $|_p w|_q$ is the rewriting path obtained by applying the leftmost (resp. rightmost) normalization strategy $\rho^{\top}_{\mathcal{R}}$ (resp. $\rho^{\perp}_{\mathcal{R}}$). We refer the reader to [6] and [7] for more details on rewriting normalization strategies.

3.4.9. Projections by reductions. Let $\mathbb{S} = (D, I)$ and $\mathbb{S}' = (D', I')$ be two plactic-like structures, and let g be the gluing map of $\mathbb{S}$. Let $S' \subset S$ be two sets of relations on $[n]^*$ such that $\mathbb{S}$ satisfies the cross-section property for $\approx_S$. Let $\mathcal{R}$ be a convergent rewriting system satisfying the following conditions:

i) ($\mathcal{R}$ *computes* $\mathbb{S}'$) for any $w \in [n]^*$, $C_{\mathbb{S}'}(w) = \mathrm{Nf}([w]_g, \mathcal{R})$,

ii) ($\mathcal{R}$ *is compatible with* $\approx_{S'}$) for any rule $d \Rightarrow d'$ in $\mathcal{R}$, we have $R_{SW}(d) \approx_{S'} R_{SW}(d')$,

iii) ($\mathcal{R}$ *computes* $\approx_{S'}$) for all $w, w' \in [n]^*$, $w \approx_{S'} w'$ implies $\mathrm{Nf}([w]_g, \mathcal{R}) = \mathrm{Nf}([w']_g, \mathcal{R})$.

We deduce from these three conditions that $\mathbb{S}'$ satisfies the cross-section property for $\approx_{S'}$. Moreover, we prove that the mapping

$$f : d \mapsto \mathrm{Nf}(d, \mathcal{R}) \tag{3.4.10}$$

for all $d \in D$, defines a morphism of string data structures from $\mathbb{S}$ to $\mathbb{S}'$. The fact that f is compatible with the readings follows from Condition **i)**. Indeed, for any d in D, we have $f(d) = C_{\mathbb{S}'}(R_{SW}(d))$, showing that $C_{\mathbb{S}'}(R_{SW}(d)) = C_{\mathbb{S}'}(R_{SW}(f(d)))$. Let us prove that f commutes with insertions. By **ii)**,

we have $R_{SW}(f(d \leftsquigarrow_I x)) \approx_{S'} R_{SW}(d \leftsquigarrow_I x)$. Moreover, we have $R_{SW}(d \leftsquigarrow_I x) \approx_S R_{SW}(d)x$, hence $R_{SW}(f(d \leftsquigarrow_I x)) \approx_{S'} R_{SW}(d)x$. On the other hand, the following equalities holds in D'

$$f(d) \leftsquigarrow_{I'} x = C_{\mathbb{S}'}(R_{SW}(f(d))x) \overset{\text{i)}}{=} \mathrm{Nf}([R_{SW}(f(d))x]_g, \mathcal{R}).$$

Then $f(d) \leftsquigarrow_{I'} x \approx_{\mathcal{R}} [R_{SW}(f(d))x]_g$, and thus by **ii)**, we deduce

$$R_{SW}(f(d) \leftsquigarrow_{I'} x) \approx_{S'} R_{SW}(f(d))x \approx_{S'} R_{SW}(d)x,$$

showing that $R_{SW}(f(d \leftsquigarrow_I x)) \approx_{S'} R_{SW}(f(d) \leftsquigarrow_{I'} x)$. Finally, following Condition **iii)**, we obtain

$$f(d \leftsquigarrow_I x) = f(d) \leftsquigarrow_{I'} x.$$

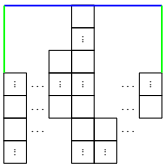
3.5. Terminations orders

A way to prove termination of a string of columns rewriting system $\mathcal{R}$ is to consider a map $f : \mathrm{Scol}_n \to (X, \prec)$, where $(X, \prec)$ is a well-ordered set satisfying, for all $w, w' \in \mathrm{Scol}_n$,

$$w \Rightarrow_{\mathcal{R}} w' \quad \text{implies} \quad f(w') \prec f(w).$$

In this subsection, we define two termination orders that will be useful in the following sections.

3.5.1. Top-left sliding order. Let $w = u_1|_{p_1} \dots |_{p_{m-1}} u_m$ be in Scol_n. We denote by h_{u_k} the number of empty boxes between the top box of the column u_k and the *top position of* w, shown by the blue line in the following picture



Define the *top deviation of* w as the sequence $d^{\top}(w) = (h_{u_1}, \dots, h_{u_m}) \in \mathbb{N}^m$. We will denote by $\preceq_{lex}$ the total order on Scol_n defined by $w \preceq_{lex} w'$ if and only if

$$tl(w) \prec_{revlex} tl(w') \text{ or } \big(tl(w) = tl(w') \text{ and } d^{\top}(w) \prec_{lex} d^{\top}(w')\big).$$

In order to prove the termination of top-left sliding operations presented in 4.2.1, we define the total order $\ll_{tl}$ on Scol_n by setting, for w, w' in Scol_n, $w \ll_{tl} w'$ if and only if

$$||w|| < ||w'|| \text{ or } \big(||w|| = ||w'|| \text{ and } w \preceq_{lex} w'\big).$$

3.5.2. Right-bottom sliding order. Let $w = u_1|_{p_1} \dots |_{p_{m-1}} u_m$ be in Scol_n. The *deviation* of a string of two columns $u_1|_p u_2$ is defined by $d(u_1|_p u_2) = |u_1| + |u_2| - p$. The *total deviation of* w is the tuple

$$d(w) = (d(u_1|_{p_1} u_2), d(u_2|_{p_2} u_3), \dots, d(u_{m-1}|_{p_{m-1}} u_m)) \in \mathbb{N}^{m-1}.$$

We will denote by $\sqsubseteq_{lex}$ the total order on Scol_n defined by $w \sqsubseteq_{lex} w'$ if and only if

$$||w|| < ||w'|| \text{ or } \big(||w|| = ||w'|| \text{ and } d(w) \prec_{lex} d(w')\big)$$

for all w, w' in Scol_n. In order to prove the termination of right-bottom sliding operations presented in 5.1.1, we define the total order $\ll_{rb}$ on Scol_n by setting, for $w, w' \in \mathrm{Scol}_n$, $w \ll_{rb} w'$ if and only if

$$R_{SW}(w) <_{\text{lex}} R_{SW}(w') \text{ or } \big(R_{SW}(w) = R_{SW}(w') \text{ and } w \sqsubseteq_{lex} w'\big).$$

4. Convergence of the jeu de taquin

In this section, we study the confluence of the jeu de taquin through a rewriting system defined by column sliding. We show that this rewriting system is convergent and we describe the jeu de taquin as a morphism of string data structures from skew to Young tableaux. We show how to deduce that the Young tableaux satisfy the cross-section property for plactic monoid, that left and right Schensted's algorithms commute, and that the jeu de taquin induces a crystal isomorphism between the sets of skew and Young tableaux.

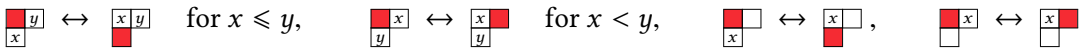
4.1. Jeu de taquin

4.1.1. Plactic monoids. Recall that the *plactic monoid (of type A) of rank n*, introduced in [16], and denoted by $\mathbf{P}_n$, is generated on $[n]$ and submitted to the following *Knuth relations*, [12]:

$$zxy = xzy, \quad \text{for } 1 \leqslant x \leqslant y < z \leqslant n, \quad \text{and} \quad yzx = yxz, \quad \text{for } 1 \leqslant x < y \leqslant z \leqslant n. \tag{4.1.2}$$

Knuth in [12] described the congruence $\approx_{\mathbf{P}_n}$ generated by these relations in terms of Young tableaux and proved the cross-section property for the monoid $\mathbf{P}_n$.

4.1.3. Forward sliding, [21]. A *forward sliding* is a sequence of the following slidings:



starting from a skew tableau and one of its inner corners, and moving the empty box until it becomes an outer corner. The *jeu de taquin* on a skew tableau w consists in applying successively the forward slide algorithm starting from w until we get a string of columns without inner corners denoted $\pi_{tq}(w)$, which is shown to be a Young tableau. In this way, the jeu de taquin defines a map

$$\pi_{tq} : \mathrm{Sk}_n \longrightarrow \mathrm{Yt}_n,$$

also called the *rectification* of skew tableaux. Schützenberger proved in [21] many properties of the jeu de taquin. These properties are also presented by Fulton in [4], as follows. For any $w \in \mathrm{Sk}_n$, the following conditions hold

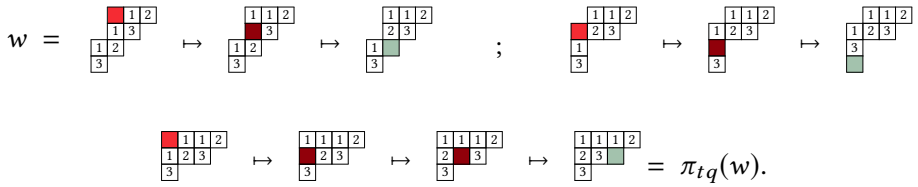
i) [4, Proposition 2]. $R_{SW}(w) \approx_{\mathbf{P}_n} R_{SW}(\pi_{tq}(w))$,

ii) [4, Corollary 1]. The rectification $\pi_{tq}(w)$ is the unique Young tableau satisfying **i)**,

iii) [4, Claim 2]. The map π_{tq} does not depend on the order in which the inner corners are chosen.

Note that condition **ii)** is a consequence of the cross-section property for $\mathbf{P}_n$ proved in [12] and condition **i)**. Moreover, condition **iii)** is a consequence of conditions **i)** and **ii)**. In the rest of this section, we show that conditions **i)** and **ii)** are direct consequence of a confluence property of a rewriting system that computes the map π_{tq}, and without supposing the cross-section property for $\mathbf{P}_n$ which is also consequence of this confluence property.

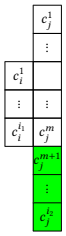
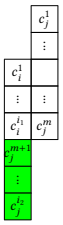
4.1.4. Example. The jeu de taquin on the following skew tableau w starting with the inner corner ■ applies three occurrences of forward sliding, where ■ denotes the empty box, and ■ the outer corner:

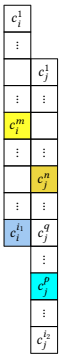


4.2. Jeu de taquin as rewriting

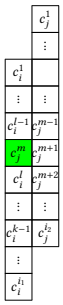
4.2.1. Rules of the jeu de taquin. The jeu de taquin map π_{tq} is described by the union of rewriting systems $\mathcal{FS}_n = \mathcal{LS}_n \cup \mathcal{IS}_n \cup \mathcal{TS}_n$ whose sets of rules are defined as follows.

i) $\mathcal{LS}_n$ the set of *left-sliding* rules that move sub-columns to the left in the following two situations:

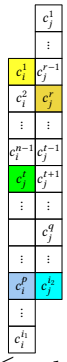
a) $\alpha_{c_i,c_j} : |_{p'}$  $|_{q'} \Rightarrow |_{p'}$  $|_{q'}$, indexed by columns c_i, c_j, and positions p', q', such that $(c_i^{i_1}, c_j^m)$ is a row, $i_1 \leqslant m < i_2$ and $c_i|_{(i_1+i_2-m)}c_j \in \mathrm{Scol}_n^{\leqslant}$.

b) $\delta^{\alpha}_{c_i,c_j} : |_{p'}$  $|_{q'} \Rightarrow |_{p'}$  $|_{q'+(p-q)}$, indexed by columns c_i, c_j, and positions p', q', such that $c_i|_{(i_1+i_2-q)}c_j \notin \mathrm{Scol}_n^{\leqslant}$, and m is maximal such that $c_i|_{(i_1+i_2-p)}c_j \in \mathrm{Scol}_n^{\leqslant}$ and $i_1 \leqslant p$.

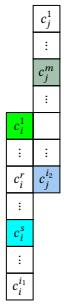
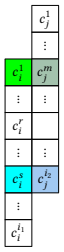
ii) $\mathcal{IS}_n$ is the set of *insert-sliding* rules performing insertions in the following two situations:

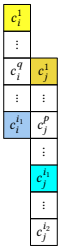
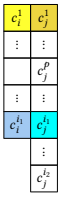
a) $\beta_{c_i,c_j} : |_{p'}$  $|_{q'} \Rightarrow |_{p'}$  $|_{q'}$, indexed by columns c_i, c_j, and positions p', q', such

that $c_i|_k c_j \in \mathrm{Scol}_n^{\leqslant}$ with $1 \leqslant k \leqslant i_1$ and $k < i_2$, and l is minimal such that (c_i^l, c_j^{m+1}) is a row and $c_i^{l-1} < c_j^m < c_i^l$.

b) $\delta^{\beta}_{c_i,c_j} : |_{p'}$  $|_{q'} \Rightarrow |_{p'}$  $|_{q'+(i_1+i_2-p-q)}$, indexed by columns c_i, c_j, and positions p', q', such that $c_i|_{(i_1+i_2-q)}c_j \notin \mathrm{Scol}_n^{\leqslant}$, where p is maximal such that $c_i|_p c_j \in \mathrm{Scol}_n^{\leqslant}$, and n is minimal such that (c_i^n, c_j^{t+1}) is a row with $c_i^{n-1} < c_j^t < c_i^n$.

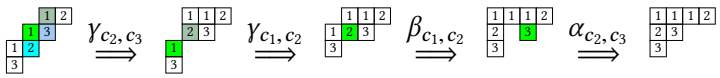
iii) $\mathcal{TS}_n$ is the set of *left top sliding* rules that move columns to the top as follows

a) $\gamma_{c_i,c_j} : |_{p'}$  $|_{q'} \Rightarrow |_{p'-(s-r)}$  $|_{q'}$, indexed by columns c_i, c_j and positions p', q', such that $c_i|_r c_j \in \mathrm{Scol}_n^{\leqslant}$ with $1 \leqslant r < i_2$, or $c_i|_r c_j$ is not row connected with $c_i^1 \leqslant c_j^{i_2}$, and s is maximal such that $c_i|_s c_j \in \mathrm{Scol}_n^{\leqslant}$ and $s \leqslant i_2$.

b) $\delta_{c_i,c_j} : |_{p'}$  $|_{q'} \Rightarrow |_{p'}$  $|_{q'+(i_1-p)}$, indexed by columns c_i, c_j, and positions p', q', such that $c_i|_{(i_1+i_2-p)}c_j \notin \mathrm{Scol}_n^{\leqslant}$ and $c_i|_{i_2}c_j \in \mathrm{Scol}_n^{\leqslant}$, or $c_i|_{(i_1+i_2-p)}c_j \in \mathrm{Scol}_n^{\leqslant}$ and $i_1 > p$, or $c_i|_k c_j$ is not row connected with $k > i_1$ and $c_i^{i_1} \leqslant c_j^1$.

In the sequel, if there is no possible confusion, we will omit the subscripts c_i, c_j in the notation of the rules. Moreover, for any rule μ in $\mathcal{FS}_n$, we will denote by μ^* any composition of rewriting sequences involving the rules μ and ending on a normal form with respect to μ.

4.2.2. Example. The rectification of the skew tableau w from Example 4.1.4 can be computed with the following reduction of $\mathcal{FS}_n$:

$w =$  $= \pi_{tq}(w)$

4.2.3. Theorem. *The rewriting system $\mathcal{FS}_n$ is convergent. In particular, the normal form of any skew tableau with respect to $\mathcal{FS}_n$ is a Young tableau. Moreover, $\mathcal{FS}_n$ computes $\mathbb{Y}_n^r$ (resp. $\mathbb{Y}_n^c$), that is, the equality*

$$C_{\mathbb{Y}_n^r}(w) = \mathrm{Nf}([w]_s, \mathcal{FS}_n) \quad \big(\textit{resp. } C_{\mathbb{Y}_n^c}(w) = \mathrm{Nf}([w]_s, \mathcal{FS}_n)\big) \tag{4.2.4}$$

holds, for any $w \in [n]^*$.

The rest of this section is devoted to the proof of this result. Lemmata 4.2.10 and 4.2.11 show that the rewriting system $\mathcal{FS}_n$ is convergent. As a consequence, we obtain that the normal form of any skew tableau with respect to $\mathcal{FS}_n$ is a Young tableau. Lemma 4.2.7 together with the convergence of the rewriting system $\mathcal{FS}_n$ yield that $\mathcal{FS}_n$ computes $\mathbb{Y}_n^r$ (resp. $\mathbb{Y}_n^c$).

4.2.5. Lemma. *For any rule $|c_{i_1}|c_{i_2}| \Rightarrow |c_{j_1}|c_{j_2}|$ in $\mathcal{FS}_n$, we have $|c_{i_1} \star_{S_r} c_{i_2}| = \rho^{\top}_{\mathcal{FS}_n}(|c_{j_1}|c_{j_2}|)$. Moreover, for any $c_i = (c_i^1, \ldots, c_i^{i_1})$ and $c_j = (c_j^1, \ldots, c_j^{i_2})$ in Col_n, we have*

$$c_i \star_{S_r} c_j = \rho^{\top}_{\mathcal{FS}_n}(c_i|_1 c_j), \qquad c_j \star_{S_l} c_i = \rho^{\perp}_{\mathcal{FS}_n}(c_i|_1 c_j). \tag{4.2.6}$$

Proof. Prove that for any rule $|c_{i_1}|c_{i_2}| \Rightarrow |c_{j_1}|c_{j_2}|$ in $\mathcal{FS}_n$, we have $|c_{i_1} \star_{S_r} c_{i_2}| = \rho^{\top}_{\mathcal{FS}_n}(|c_{j_1}|c_{j_2}|)$. The rules α_{c_i,c_j}, $\delta^{\alpha}_{c_i,c_j}$, β_{c_i,c_j} and $\delta^{\beta}_{c_i,c_j}$ followed by β^* yield to the Young tableau $|c_i \star_{S_r} c_j|$. Consider now the rule γ_{c_i,c_j}. If $|c_j| \leqslant |c_i|$ and $c_i|_{|c_j|}c_j \in \mathrm{Scol}_n$ then the target of γ_{c_i,c_j} is equal to $|c_i \star_{S_r} c_j|$. If $|c_i| < |c_j|$ and $c_i|_{|c_j|}c_j \in \mathrm{Scol}_n$, then the rule γ_{c_i,c_j} is followed by the rule α_{c_i,c_j} in order to obtain $|c_i \star_{S_r} c_j|$. If $c_i|_{|c_j|}c_j \notin \mathrm{Scol}_n$ then the rule γ_{c_i,c_j} followed by α_{c_i,c_j} and then by β^*, or only followed by β^* yield to $|c_i \star_{S_r} c_j|$. Consider finally the rule δ_{c_i,c_j}. If $|c_j| \leqslant |c_i|$ then the target δ_{c_i,c_j} is equal to $|c_i \star_{S_r} c_j|$. Otherwise, if $|c_i| < |c_j|$ then δ_{c_i,c_j} is followed by α_{c_i,c_j} in order to obtain $|c_i \star_{S_r} c_j|$.

Prove the first equality of (4.2.6) by induction on $|c_j|$, the proof being similar for $\star_{S_l}$. Suppose that $|c_j| = 1$, we consider the following two cases. If $c_i^1 \leqslant c_j^1$, then $c_i|_1 c_j$ is equal to $c_i \star_{S_r} c_j$. If $c_j^1 < c_i^1$, then by applying δ^{β} on $c_i|_1 c_j$ we obtain $c_i \star_{S_r} c_j$. Suppose the equality holds when $|c_j| = i_2 - 1$, and prove it when $|c_j| = i_2$. First consider the case when $c_i^1 \leqslant c_j^{i_2}$. Suppose that by induction we have

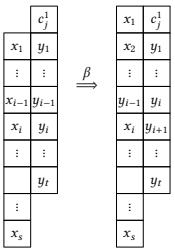
$$\rho^{\top}_{\mathcal{FS}_n}\Big(\begin{array}{|c|c|}\hline & c_j^2 \\ \hline & \vdots \\ \hline c_i^1 & c_j^{i_2} \\ \hline \vdots & \\ \hline c_i^{i_1} & \\ \hline\end{array}\Big) \quad = \quad \begin{array}{|c|c|}\hline x_1 & y_1 \\ \hline \vdots & \vdots \\ \hline & y_t \\ \hline \vdots & \\ \hline x_s & \\ \hline\end{array} \quad = c_i \star_{S_r} (c_j^2, \ldots, c_j^{i_2}),$$

where x_i and y_j are elements of c_i and c_j with $t \leqslant s$. We prove that $\rho^{\top}_{\mathcal{FS}_n}\Big(\begin{array}{|c|c|}\hline & c_j^1 \\ \hline x_1 & y_1 \\ \hline \vdots & \vdots \\ \hline & y_t \\ \hline \vdots & \\ \hline x_s & \\ \hline\end{array}\Big) = c_i \star_{S_r} c_j$, by considering the following two cases.

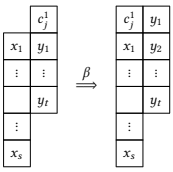
Case 1. $x_1 \leqslant c_j^1$ and $x_{k+1} \leqslant y_k$, for all $k = 1, \ldots, t-1$:

$$\begin{array}{|c|c|}\hline & c_j^1 \\ \hline x_1 & y_1 \\ \hline \vdots & \vdots \\ \hline x_s & y_t \\ \hline\end{array} \overset{\gamma}{\Longrightarrow} \begin{array}{|c|c|}\hline x_1 & c_j^1 \\ \hline x_2 & y_1 \\ \hline \vdots & \vdots \\ \hline x_s & y_{t-1} \\ \hline & y_t \\ \hline\end{array} \overset{\alpha}{\Longrightarrow} \begin{array}{|c|c|}\hline x_1 & c_j^1 \\ \hline x_2 & y_1 \\ \hline \vdots & \vdots \\ \hline x_s & y_{t-1} \\ \hline y_t & \\ \hline\end{array} = c_i \star_{S_r} c_j, \quad \text{for } s = t \qquad \text{or} \qquad \begin{array}{|c|c|}\hline & c_j^1 \\ \hline x_1 & y_1 \\ \hline \vdots & \vdots \\ \hline & y_t \\ \hline \vdots & \\ \hline x_s & \\ \hline\end{array} \overset{\gamma}{\Longrightarrow} \begin{array}{|c|c|}\hline x_1 & c_j^1 \\ \hline x_2 & y_1 \\ \hline \vdots & \vdots \\ \hline & y_t \\ \hline \vdots & \\ \hline x_s & \\ \hline\end{array} = c_i \star_{S_r} c_j, \quad \text{for } s > t.$$

Case 2. $x_1 \leqslant c_j^1$ (resp. $c_j^1 < x_1$) and let x_i be minimal such that $x_{i-1} < y_{i-1} < x_i$:

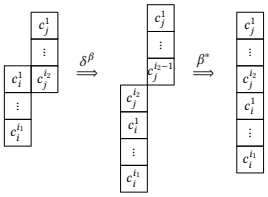


$= c_i \star_{S_r} c_j.$ (resp.



$= c_i \star_{S_r} c_j.$)

Suppose finally that $c_i^1 > c_j^{i_2}$. We obtain:



$= c_i \star_{S_r} c_j.$ □

The following result shows that the right and left Schensted insertions correspond respectively to the leftmost and rightmost reduction paths with respect the rewriting system $\mathcal{FS}_n$.

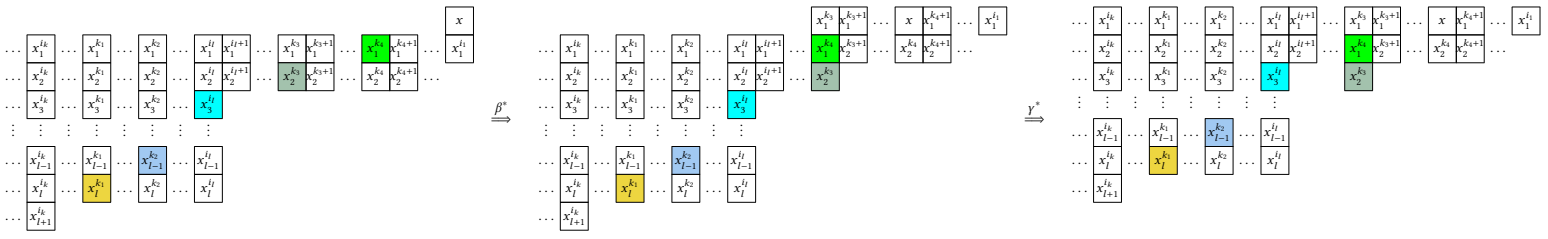
4.2.7. Lemma. *For any word w in $[n]^*$, we have*

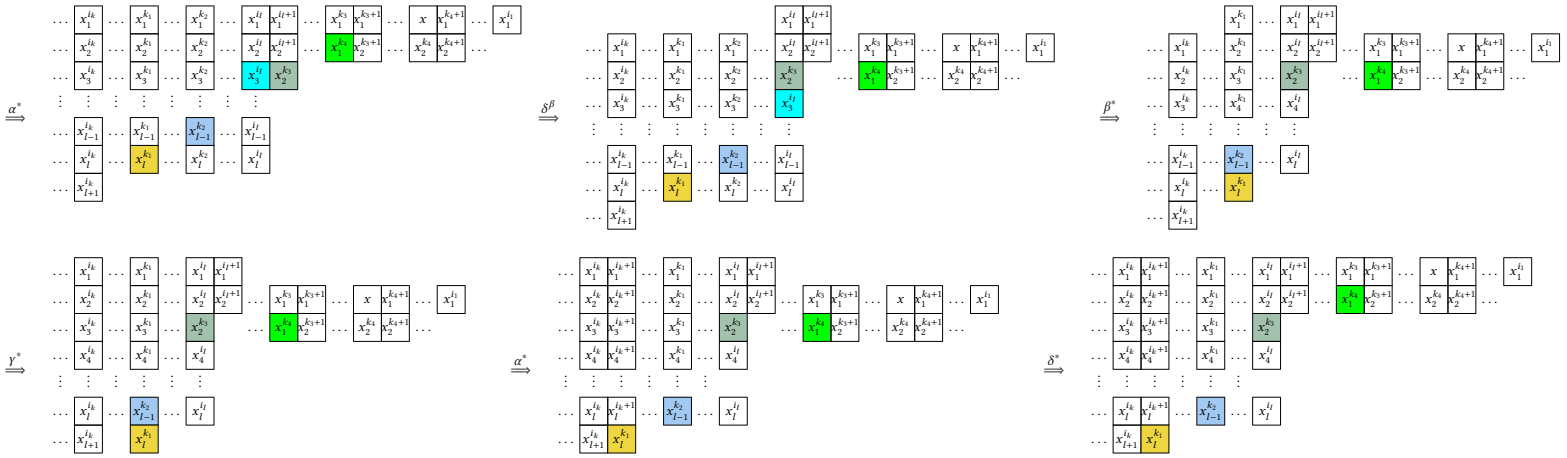
$$(\emptyset \leftsquigarrow_{S_r} w) = \rho^{\top}_{\mathcal{FS}_n}([w]_s), \qquad (w \rightsquigarrow_{S_l} \emptyset) = \rho^{\perp}_{\mathcal{FS}_n}([w]_s). \tag{4.2.8}$$

Proof. We prove the first equality in (4.2.8) by induction on the number of columns in $[w]_s$, the proof being similar for the insertion S_l. When $[w]_s$ is of length 2, then the equality is a consequence of Lemma 4.2.5. For $k \geqslant 3$, suppose that the equality holds for words of length $k-1$, and consider $[w]_s = c_1|_1 \dots |_1 c_k$. By induction hypothesis, we have $(\emptyset \leftsquigarrow_{S_r} w) = \rho^{\top}_{\mathcal{FS}_n}(c_1|_1 \dots |_1 c_{k-1}) \star_{S_r} c_k$. Let us show that

$$\rho^{\top}_{\mathcal{FS}_n}(c_1|_1 \dots |_1 c_{k-1}) \star_{S_r} c_k = \rho^{\top}_{\mathcal{FS}_n}(c_1|_1 \dots |_1 c_k). \tag{4.2.9}$$

Since inserting c_k into $\rho^{\top}_{\mathcal{FS}_n}(c_1|_1 \dots |_1 c_{k-1})$ consists into inserting its elements one by one from bottom to top, it suffices to prove (4.2.9) for $c_k = (x)$. If x is bigger or equal to the last element $x_1^{i_1}$ of the first row of $\rho^{\top}_{\mathcal{FS}_n}(c_1|_1 \dots |_1 c_{k-1})$, then $\rho^{\top}_{\mathcal{FS}_n}(c_1|_1 \dots |_1 c_{k-1})|_1 \iota(x)$ is a Young tableau that is equal to $\rho^{\top}_{\mathcal{FS}_n}(c_1|_1 \dots |_1 c_{k-1}) \star_{S_r} \iota(x)$. Otherwise, if $x < x_1^{i_1}$, we first apply a rule δ^{β} in order to slide the box containing x to the top of the one containing $x_1^{i_1}$. We then apply the following reduction rules as shown in the following reduction diagrams. Note that, the elements in the colored boxes represent the ones that are bumped when inserting x into the tableau $\rho^{\top}_{\mathcal{FS}_n}(c_1|_1 \dots |_1 c_{k-1})$.





The resulted Young tableau is equal to $\rho^{\top}_{\mathcal{FS}_n}(c_1|_1 \dots |_1 c_{k-1}) \star_{S_r} \iota(x)$, showing the claim. □

4.2.10. Lemma. *The rewriting system $\mathcal{FS}_n$ is terminating.*

Proof. We prove that for any reduction $w \Rightarrow w'$ with respect to $\mathcal{FS}_n$, we have $w \ll_{tl} w'$ for the order $\ll_{tl}$ defined 3.5.1. If $w \Rightarrow w'$ is a reduction with respect to $\mathcal{LS}_n$, then $||w|| = ||w'||$ and $tl(w) \prec_{revlex} tl(w')$, showing that $w \ll_{tl} w'$. Suppose now that the reduction is with respect to $\mathcal{IS}_n$. There are two cases depending on the number of columns in the targets of the rules β and δ^{β}. If the targets consist only of one column then $||w|| < ||w'||$. If they consist of two columns then $||w|| = ||w'||$ and $tl(w) \prec_{revlex} tl(w')$. Then, if $w \Rightarrow w'$ is a reduction with respect to $\mathcal{IS}_n$, we obtain $w \ll_{tl} w'$. Finally, for any reduction $w \Rightarrow w'$ with respect to $\mathcal{TS}_n$, we have $||w|| = ||w'||$, $tl(w) = tl(w')$ and $d^{\top}(w) \prec_{lex} d^{\top}(w')$, showing that $w \ll_{tl} w'$. □

4.2.11. Lemma. *The rewriting system $\mathcal{FS}_n$ is confluent.*

Proof. Following Lemma 3.4.7, we prove that the rewriting system $\mathcal{FS}_n$ is confluent by showing the confluence of all its critical branchings. Consider first the rewriting system $\mathcal{R}(\mathrm{Col}_n, \mathbb{Y}^r_n)$ whose rules are of the form $\gamma_{c,c'} : |c|c'| \Rightarrow |c \star_{S_r} c'|$, for all c, c' in Col_n such that $c|_y c' \neq c \star_{S_r} c'$. Prove that starting from a string of columns consisting of three columns $|c_i|c_j|c_k|$, we lead to the Young tableau $|c_i \star_{S_r} c_j \star_{S_r} c_k|$ after applying at most three steps of reductions with respect $\mathcal{R}(\mathrm{Col}_n, \mathbb{Y}^r_n)$ starting from the left or from the right. We prove this result using Schützenberger's involution on columns as shown in [8, Remark 3.2.7]. Let c be a column of length p, the *Schützenberger involution* of c, denoted by c^*, is the column of length $n - p$ obtained by taking the complement of the elements of c. This involution is extended to string of columns by setting $(c_1| \dots |c_r)^* = c_r^*| \dots |c_1^*$, for all $c_1, \dots, c_r$ in Col_n. If $c_1|_y \dots |_y c_r$ is a Young tableau, then $(c_1|_y \dots |_y c_r)^* = c_r^*|_y \dots |_y c_1^*$ is also a Young tableau. Moreover, we have $(c_1 \star_{S_r} \dots \star_{S_r} c_r)^* = (c_r^* \star_{S_r} \dots \star_{S_r} c_1^*)$, for all $c_1, \dots, c_r$ in Col_n. In particular, for three columns c_i, c_j and c_k in Col_n, we have $(c_i \star_{S_r} c_j \star_{S_r} c_k)^* = (c_k^* \star_{S_r} c_j^* \star_{S_r} c_i^*)$, see [17]. In one hand, by definition of Schensted's insertion S_r, starting from $|c_i|c_j|c_k|$, we lead to $|c_i \star_{S_r} c_j \star_{S_r} c_k|$ after applying at most three steps of reductions with respect $\mathcal{R}(\mathrm{Col}_n, \mathbb{Y}^r_n)$ starting from the left. That is, we have

$$|c_i|c_j|c_k| \overset{\gamma_{c_i,c_j}|c_k}{\Longrightarrow} |c_n|_y c_{n'}|c_k| \overset{c_n|\gamma_{c_{n'},c_k}}{\Longrightarrow} |c_n|c_s|_y c_{s'}| \overset{\gamma_{c_n,c_s}|c_{s'}}{\Longrightarrow} |c_i \star_{S_r} c_j \star_{S_r} c_k|.$$

In an other hand, we have

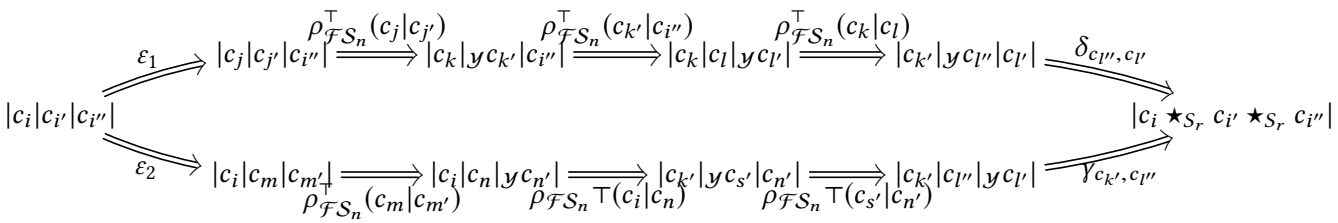
$$|c_i|c_j|c_k| \overset{c_i|\gamma_{c_j,c_k}}{\Longrightarrow} |c_i|(c_j\star_{S_r}c_k)| = |c_i|c_l|_{\mathcal{Y}}c_{l'}| \overset{\gamma_{c_i,c_l}|c_{l'}}{\Longrightarrow} |(c_i\star_{S_r}c_l)|c_{l'}| = |c_m|_{\mathcal{Y}}c_{m'}|c_{l'}| \overset{c_m|\gamma_{c_{m'},c_{l'}}}{\Longrightarrow} |c_m|(c_{m'}\star_{s_r}c_{l'})|.$$

Let us show that $|c_m|(c_{m'}\star_{s_r}c_{l'})| = |c_i\star_{S_r}c_j\star_{S_r}c_k|$. By applying the involution on tableaux, we obtain

$$|c_k^*|c_j^*|c_i^*| \Longrightarrow |(c_k^*\star_{S_r}c_j^*)|c_i^*| = |c_{l'}^*|_{\mathcal{Y}}c_l^*|c_i^*| \Longrightarrow |c_{l'}^*|(c_l^*\star_{S_r}c_i^*)| = |c_{l'}^*|c_{m'}^*|_{\mathcal{Y}}c_m^*| \Longrightarrow |(c_{l'}^*\star_{S_r}c_{m'}^*)|c_m^*|.$$

By definition of S_r, we have $|c_k^*\star_{S_r}c_j^*\star_{s_r}c_i^*| = |(c_{l'}^*\star_{S_r}c_{m'}^*)|c_m^*|$. Since $(c_k^*\star_{s_r}c_j^*\star_{S_r}c_i^*) = (c_i\star_{S_r}c_j\star_{S_r}c_k)^*$, we deduce that $|(c_i\star_{S_r}c_j\star_{S_r}c_k)^*| = |(c_{l'}^*\star_{S_r}c_{m'}^*)|c_m^*|$. Finally, by applying the involution on tableaux, we obtain $|(c_i\star_{S_r}c_j\star_{S_r}c_k)| = |c_m|(c_{m'}\star_{S_r}c_{l'})|$.

Following Lemma 4.2.5, for any rule $|c_{i_1}|c_{i_2}| \Rightarrow |c_{j_1}|c_{j_2}|$ in $\mathcal{FS}_n$, we have $|c_{i_1}\star_{S_r}c_{i_2}| = \rho^{\top}_{\mathcal{FS}_n}(|c_{j_1}|c_{j_2}|)$. Hence, any critical branching of $\mathcal{TS}_n$ has the following confluence diagram



where ε_1 and ε_2 are $\mathcal{TS}_n$-reductions and where some indicated rules can correspond to identities, such that $c_k|_{\mathcal{Y}}c_{k'} = c_j\star_{S_r}c_{j'}, c_l|_{\mathcal{Y}}c_{l'} = c_{k'}\star_{S_r}c_{i''}, c_{k'}|_{\mathcal{Y}}c_{l''} = c_k\star_{S_r}c_l, c_n|_{\mathcal{Y}}c_{n'} = c_m\star_{S_r}c_{m'}, c_{k'}|_{\mathcal{Y}}c_{s'} = c_i\star_{S_r}c_n$ and $c_{l''}|_{\mathcal{Y}}c_{l'} = c_{s'}\star_{S_r}c_{n'}$. □

4.3. Jeu de taquin as morphism

4.3.1. Theorem. *The rectification map $\pi_{tq} : \mathrm{dSk}_n \to \mathrm{Yt}_n$ extends into a surjective morphism of string data structures:*

$$\pi^r_{tq} : d\mathbb{S}^r_n \to \mathbb{Y}^r_n \quad (\textit{resp. } \pi^c_{tq} : d\mathbb{S}^c_n \to \mathbb{Y}^c_n). \tag{4.3.2}$$

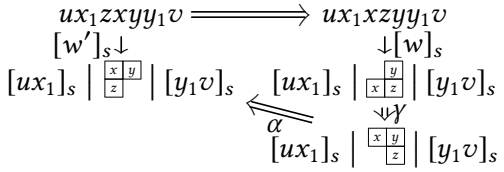
Proof. The surjectivity of π_{tq} is a consequence of the relation $\pi_{tq}([R_{SW}(d)]_s) = d$, that holds for any d in Yt_n. Prove that π^r_{tq} (resp. π^c_{tq}) is a morphism of string data structures by showing Conditions **i)**, **ii)** and **iii)** of 3.4.9. Condition **i)** is a consequence of Theorem 4.2.3. Prove Condition **ii)**, that is, for any rule $d \Rightarrow d'$ in $\mathcal{FS}_n$, we have $R_{SW}(d) \approx_{\mathbf{P}_n} R_{SW}(d')$. It suffices to show that $R_{SW}(s(\eta)) \approx_{\mathbf{P}_n} R_{SW}(t(\eta))$, for every rule η in $\mathcal{FS}_n$. This is obvious for γ and δ. For the rule α, consider $R_{SW}(s(\alpha)) = c_i^{i_1}\dots c_i^1c_j^{i_2}\dots c_j^{m+1}c_j^m\dots c_j^1$ and $R_{SW}(t(\alpha)) = c_j^{i_2}\dots c_j^{m+1}c_i^{i_1}\dots c_i^1c_j^m\dots c_j^1$. On one hand, we have

$$\begin{aligned} c_i^{i_1}\dots\mathbf{c_i^2c_i^1c_j^{i_2}}\dots c_j^{m+1}c_j^m\dots c_j^1 &\overset{(4.1.2)}{=}\dots\overset{(4.1.2)}{=} c_i^{i_1}c_j^{i_2}c_i^{i_1-1}\dots\mathbf{c_i^2c_i^1c_j^{i_2-1}}\dots c_j^{m+1}c_j^m\dots c_j^1 \\ &\overset{(4.1.2)}{=}\dots\overset{(4.1.2)}{=}\mathbf{c_i^{i_1}c_j^{i_2}c_j^{i_2-1}}c_i^{i_1-1}\dots c_i^2c_i^1c_j^{i_2-2}\dots c_j^{m+1}c_j^m\dots c_j^1\overset{(4.1.2)}{=} \\ c_j^{i_2}c_i^{i_1}c_j^{i_2-1}c_i^{i_1-1}\dots\mathbf{c_i^2c_i^1c_j^{i_2-2}}\dots c_j^{m+1}c_j^m\dots c_j^1 &\overset{(4.1.2)}{=}\dots\overset{(4.1.2)}{=} c_j^{i_2}\dots c_j^{m+1}c_j^mc_i^{i_1}\dots c_i^1c_j^{m-1}\dots c_j^1. \end{aligned}$$

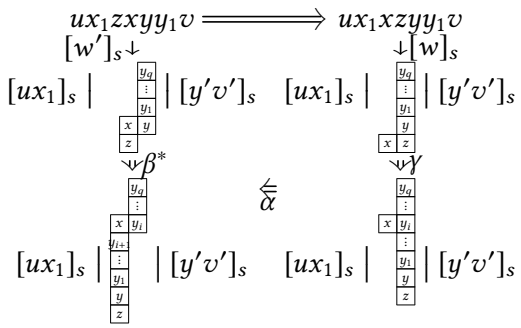
In an other hand, we have $c_j^{i_2}\dots c_j^{m+1}c_i^{i_1}\dots\mathbf{c_i^2c_i^1c_j^m}\dots c_j^1 \overset{(4.1.2)}{=}\dots\overset{(4.1.2)}{=} c_j^{i_2}\dots c_j^{m+1}c_j^mc_i^{i_1}\dots c_i^1c_j^{m-1}\dots c_j^1$. Hence, $R_{SW}(s(\alpha)) \approx_{\mathbf{P}_n} R_{SW}(t(\alpha))$. Similarly, we show the property for rules β, δ^α and δ^β.

Prove Condition **iii)**, that is, for all $w, w' \in [n]^*$, $w \approx_{\mathbf{P}_n} w'$ implies $\mathrm{Nf}([w]_s, \mathcal{FS}_n) = \mathrm{Nf}([w']_s, \mathcal{FS}_n)$. Since $\mathcal{FS}_n$ is convergent, we show that for all $w, w' \in [n]^*$, $w \approx_{\mathbf{P}_n} w'$ implies $[w]_s \approx_{\mathcal{FS}_n} [w']_s$. Suppose first that $w = ux_1\mathbf{xzy}y_1v$ and $w' = ux_1\mathbf{zxy}y_1v$, for all $1 \leqslant x \leqslant y < z \leqslant n$, $u, v \in [n]^*$ and $x_1, y_1 \in [n]$, and show that $[w]_s \approx_{\mathcal{FS}_n} [w']_s$. We consider the following cases:

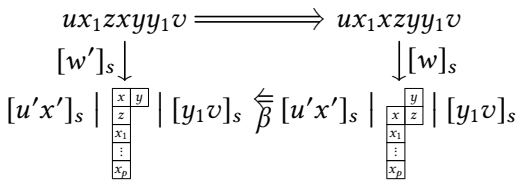
Case 1. $x_1 \leqslant x$ and $y \leqslant y_1$.



Case 2. $x_1 \leqslant x$ and $y > y_1$. Suppose that $v = y_2 \dots y_q y' v'$ such that $y_1 > y_2 > \dots > y_q$ and $y_q \leqslant y'$.

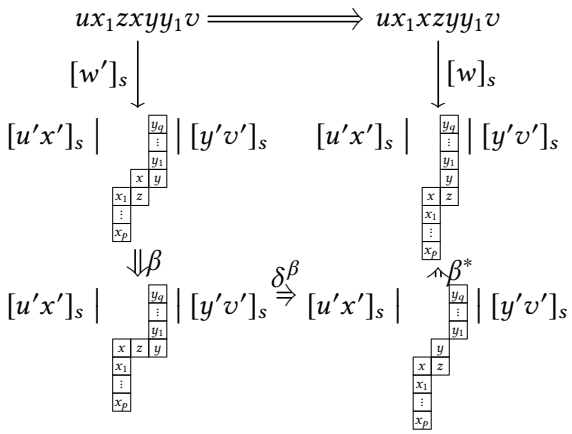


Case 3. $x < z < x_1$ and $y \leqslant y_1$. Suppose that $u = u'x'x_p \dots x_1$ such that $x' \leqslant x_p$ and $x_p > \dots > x_1$.



The case $x < z < x_1$ and $y > y_1$ is studied in the same way.

Case 4. $x < x_1 \leqslant z$ and $y > y_1$. We study similarly the case $x < x_1 \leqslant z$ and $y \leqslant y_1$. Suppose that $u = u'x'x_p \dots x_1$ and $v = y_2 \dots y_q y' v'$ such that $x' \leqslant x_p > \dots > x_1$ and $y_1 > \dots > y_q \leqslant y'$.



Suppose now that $w = uy_1\mathbf{yzx}x_1v$ and $w' = uy_1\mathbf{yxz}x_1v$, for all $1 \leqslant x < y \leqslant z \leqslant n$, $u, v \in [n]^*$ and $x_1, y_1 \in [n]$, and show that $[w]_s \approx_{\mathcal{FS}_n} [w']_s$. If $y_1 \leqslant y$ and $x \leqslant x_1$, then

$$\begin{array}{ccc} uy_1yzxx_1v & \Longrightarrow & uy_1yxzx_1v \\ [w]_s\downarrow & & \downarrow[w']_s \\ [uy_1]_s \,\Big|\, \begin{array}{|c|c|}\hline & x \\ \hline y & z \\ \hline \end{array} \,\Big|\, [x_1v]_s & \overrightarrow{\beta} & [uy_1]_s \,\Big|\, \begin{array}{|c|c|}\hline x & z \\ \hline y \\ \cline{1-1} \end{array} \,\Big|\, [x_1v]_s \end{array}$$

The cases ($y_1 > y$ and $x > x_1$), ($y_1 \leqslant y$ and $x > x_1$) and ($y_1 > y$ and $x \leqslant x_1$) are studied similarly. □

As a direct consequence of this result, we deduce

4.3.3. Corollary. *The string data structure $\mathbb{Y}^r_n$ (resp. $\mathbb{Y}^c_n$) satisfies the cross-section property for the monoid $\mathbf{P}_n$.*

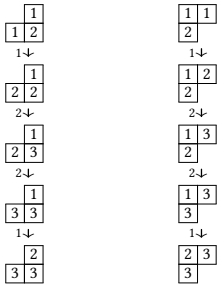
The commutation of S_r and S_l, proved by Schensted by a direct method, [20, Lemma 6], is a consequence of Lemma 2.1.5 and Theorem 4.3.1:

4.3.4. Corollary. *The insertions S_r and S_l commute, and the structure monoid $\mathbf{M}(\mathbb{Y}^r_n)$ is isomorphic to the monoid $\mathbf{P}_n$.*

As a consequence of Proposition 3.3.8 and Theorem 4.3.1, we deduce

4.3.5. Corollary. *The morphism of string data structures $\pi^r_{tq} : d\mathbb{S}^r_n \to \mathbb{Y}^r_n$ (resp. $\pi^c_{tq} : d\mathbb{S}^c_n \to \mathbb{Y}^c_n$) induces a crystal isomorphism from $\Gamma(\mathcal{K}, d)$ to $\Gamma(\mathcal{K}, \pi_{rb}(d))$, for every d in dSk_n.*

4.3.6. Example. The following two connected components are isomorphic with respect to Kashiwara's crystal structure defined in 3.3.9:



where the skew tableaux and Young tableaux situated at the same place in this crystal isomorphism can be related by the jeu de taquin slidings.

4.3.7. Remark. Note that Schensted's insertions are related to the jeu de taquin by the following formulas

$$t \leftsquigarrow_{S_r} x \;=\; \pi_{tq}\big([R_{SW}(t)]_s \leftsquigarrow_{I^a_r} x\big), \qquad\qquad x \rightsquigarrow_{S_l} t \;=\; \pi_{tq}\big(x \rightsquigarrow_{I^a_l} [R_{SW}(t)]_s\big),$$

for all t in Yt_n and x in $[n]$. Note also that the associativity of $\star_{S_r}$ can be deduced from the morphism π_{tq}. Indeed, for all t in Yt_n, and x in $[n]$, we have $t \leftsquigarrow_{S_r} x = \pi_{tq}(t|_1\iota(x))$, and thus $t \star_{S_r} t' \;=\; \pi_{tq}(t|_1t')$, for all $t, t' \in \mathrm{Yt}_n$. By Theorem 4.2.3, we obtain

$$(t \star_{S_r} t') \star_{S_r} t'' \;=\; \pi_{tq}(t|_1t'|_1t'') \;=\; t \star_{S_r} (t' \star_{S_r} t''),$$

for any $t, t', t'' \in \mathrm{Yt}_n$.

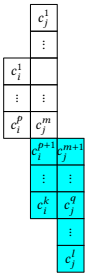
5. Convergence of the right-bottom rectification

In this section, we introduce the right-bottom rectification as a morphism of string data structures from Young tableaux to quasi-ribbon tableaux using a rewriting system made of right-bottom sliding rules. We show that this rewriting system is convergent and we deduce that the set of quasi-ribbon tableaux satisfies the cross-section property for the hypoplactic monoid and that the right-bottom rectification induces a crystal isomorphism between the sets of Young and quasi-ribbon tableaux.

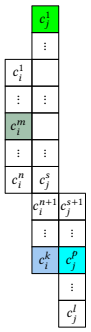
5.1. Convergence of the right-bottom rectification

5.1.1. Right-bottom rectification algorithm. The *right-bottom sliding* $\pi_{rb} : \mathrm{Yt}_n \to \mathrm{Scol}_n^{\leqslant}$ is the union of rewriting systems $\mathcal{RBT}_n = \mathcal{RS}_n \cup \mathcal{BS}_n \cup \mathcal{TS}_n$ whose sets of rules are defined as follows.

i) $\mathcal{RS}_n$ the set of *right-sliding* rules that move sub-columns to the right in the following two situations:

a) $\alpha_{c_i,c_j} : |_{p'}$  $|_{q'} \Rightarrow |_{(p'-k+p)}$  $|_{(q'-m)}$, indexed by columns c_i, c_j, and positions p', q',

such that $c_i|_{(k+l-q)}c_j \in \mathrm{Scol}_n^{\leqslant}$, and where p is minimal such that $c_i^p \leqslant c_j^m < c_i^{p+1}$. Note that the elements c_j^n, for $m+1 \leqslant n \leqslant l$, can be empty. Note also that when $c_i^p = c_j^m$, the elements c_j^n, for $1 \leqslant n \leqslant m-1$, are empty.

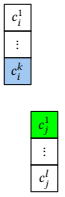
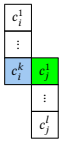
b) $\delta^{\alpha}_{c_i,c_j} : |_{p'}$  $|_{q'} \Rightarrow |_{p'}$  $|_{q'-(p-q)}$, indexed by columns c_i, c_j, and positions p', q',

such that $c_i|_{(k+l-q)}c_j \notin \mathrm{Scol}_n^{\leqslant}$, where m is maximal such that $c_i|_{(k+l-p)}c_j \in \mathrm{Scol}_n^{\leqslant}$, and where n is minimal such that $c_i^n \leqslant c_j^s < c_i^{n+1}$.

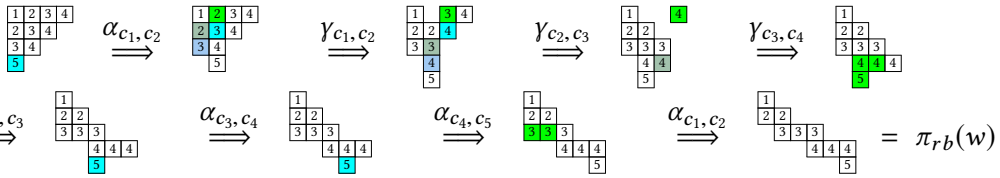
ii) $\mathcal{BS}_n$ the set of *bottom-sliding* rules that move columns to the bottom in the following situation:

$\gamma_{c_i,c_j} : |_{p'}$  $|_{q'} \Rightarrow |_{p'}$  $|_{(q'-p)}$, indexed by columns c_i, c_j, and positions p', q', where c_i and c_j are glued at position q, such that $q \geqslant 1$ and $c_i|_q c_j \in \mathrm{Scol}_n^{\leqslant}$, or $q < 1$ and $c_i^1 \leqslant c_j^l$, and where s is maximal such that $c_i|_{(k+l-s)}c_j \in \mathrm{Scol}_n^{\leqslant}$.

iii) $\mathcal{TS}_n$ the set of *top-sliding* rules that move columns to the top in the following situation:

$\delta_{c_i,c_j} : |_{p'}$  $|_{q'} \Rightarrow |_{p'}$  $|_{q'-p-1}$, indexed by columns c_i, c_j, and positions p', q', where c_i and c_j are glued at position $q > k$ such that $c_i|_q c_j$ is not row connected and $c_i^k \leqslant c_j^1$.

5.1.2. Example. The following reduction sequence reduces the Young tableau w into its right-bottom rectification $\pi_{rb}(w)$.

$w =$ 

5.1.3. Theorem. *The rewriting system $\mathcal{RBT}_n$ is convergent. In particular, the normal form of any Young tableau with respect to $\mathcal{RBT}_n$ is a quasi-ribbon tableau. Moreover, $\mathcal{RBT}_n$ computes $\mathbb{Q}_n^r$ (resp. $\mathbb{Q}_n^l$), that is, the equality*

$$C_{\mathbb{Q}_n^r}(w) = \mathrm{Nf}([w]_{\mathcal{Y}}, \mathcal{RBT}_n) \qquad (\textit{resp. } C_{\mathbb{Q}_n^l}(w) = \mathrm{Nf}([w]_{\mathcal{Y}}, \mathcal{RBT}_n)) \tag{5.1.4}$$

holds, for any $w \in [n]^*$.

The rest of this section is devoted to the proof of this result. Lemmata 5.1.9 and 5.1.10 show that the rewriting system $\mathcal{RBT}_n$ is convergent. As a consequence, we obtain that the normal form of any Young tableau with respect to $\mathcal{RBT}_n$ is a quasi-ribbon tableau. Lemma 5.1.7 together with the convergence of $\mathcal{RBT}_n$ yield that $\mathcal{RBT}_n$ computes $\mathbb{Q}_n^r$ (resp. $\mathbb{Q}_n^l$).

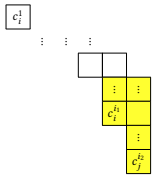
5.1.5. Lemma. *For any $c_i = (c_i^1, \ldots, c_i^{i_1})$ and $c_j = (c_j^1, \ldots c_j^{i_2})$ in Col_n, we have*

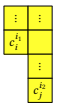
$$c_i \star_{H_r} c_j = \rho^{\top}_{\mathcal{RBT}_n}(c_i|_{|c_j|}c_j), \qquad c_j \star_{H_l} c_i = \rho^{\perp}_{\mathcal{RBT}_n}(c_i|_{|c_j|}c_j). \tag{5.1.6}$$

Moreover, for any rule $|c_{i_1}|c_{i_2}| \Rightarrow |c_{j_1}|c_{j_2}|$ in $\mathcal{RBT}_n$, we have $\rho^{\top}_{\mathcal{RBT}_n}(|c_{j_1}|c_{j_2}|) = |c_{i_1} \star_{H_r} c_{i_2}|$ and $\rho^{\perp}_{\mathcal{RBT}_n}(|c_{j_1}|c_{j_2}|) = |c_{i_2} \star_{H_l} c_{i_1}|$.

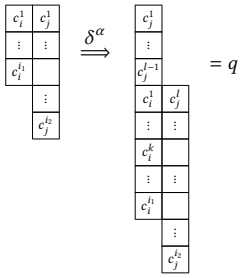
Proof. Prove the first equality of (5.1.6), the proof being similar for the second one. We consider two cases depending on whether or not $c_i|_{|c_j|}c_j$ belongs to $\mathrm{Scol}_n^{\leqslant}$.

Case 1. Suppose that $c_i|_{|c_j|}c_j \in \mathrm{Scol}_n^{\leqslant}$. If $c_i^{i_1} \leqslant c_j^1$, then applying the rule γ_{c_i,c_j} yields to $c_i \star_{H_r} c_j$. Otherwise, by applying a reduction sequence of rules α and γ starting in each step from the left, we reduce $c_i|_{|c_j|}c_j$ into a string of columns of the following form

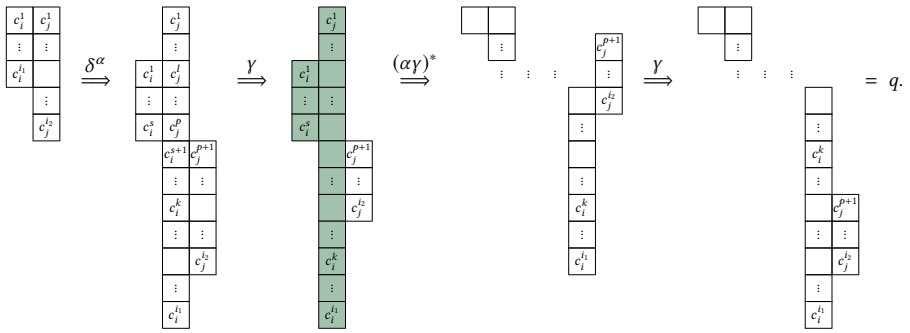


Similarly, we reduce $t =$  into a string of columns made of a quasi-ribbon tableau glued to the top of a new string of columns t' in $\mathrm{Scol}_n^{\leqslant}$. We reduce t' in the same way and we continue until a quasi-ribbon tableau appear which is equal by construction to $c_i \star_{H_r} c_j$.

Case 2. Suppose that $c_i|_{|c_j|}c_j \notin \mathrm{Scol}_n^{\leqslant}$. Suppose first that $c_j^{i_2} < c_i^1$, then the rule $\delta^{\alpha}_{c_i,c_j}$ yields to $c_i \star_{H_r} c_j$. Suppose now that k is maximal such that $c_i|_k c_j \in \mathrm{Scol}_n^{\leqslant}$. If (c_i^1, c_j^l) is a row such that $c_j^{l-1} < c_i^1$, then:



By **Case 1**, we obtain $\rho^{\top}_{\mathcal{RBT}_n}(q) = c_i \star_{H_r} c_j$. Otherwise, if s is minimal such that $c_i^s < c_j^p < c_i^{s+1}$, then we apply the following reduction sequence of rules α, δ^{α} and γ starting in each step from the left:



By **Case 1**, we obtain $\rho^{\top}_{\mathcal{RBT}_n}(q) = c_i \star_{H_r} c_j$.

Similarly, we show that for any rule $|c_{i_1}|c_{i_2}| \Rightarrow |c_{j_1}|c_{j_2}|$ in $\mathcal{RBT}_n$, we have $\rho^{\top}_{\mathcal{RBT}_n}(|c_{j_1}|c_{j_2}|) = |c_{i_1} \star_{H_r} c_{i_2}|$ (resp. $\rho^{\perp}_{\mathcal{RBT}_n}(|c_{j_1}|c_{j_2}|) = |c_{i_2} \star_{H_l} c_{i_1}|$). Indeed, the rules α_{c_i,c_j} and $\delta^{\alpha}_{c_i,c_j}$ followed by a reduction sequence of rules α and γ starting in each step from the left (resp. right) yield to $c_i \star_{H_r} c_j$ (resp. $c_j \star_{H_l} c_i$).

Moreover, if the first element of c_i is smaller or equal than the last element of c_j, then the target of the rule γ_{c_i,c_j} is equal to $c_i \star_{H_r} c_j$. Otherwise, the rule γ is followed by a sequence of rules α and γ starting in each step from the left (resp. right) in order to obtain $c_i \star_{H_r} c_j$ (resp. $c_j \star_{H_l} c_i$). Finally, the target of the rule δ_{c_i,c_j} is equal to $c_i \star_{H_r} c_j$. □

The following result shows that the right and left insertions H_r and H_l correspond respectively to the leftmost and rightmost reduction paths with respect the rewriting system $\mathcal{RBT}_n$.

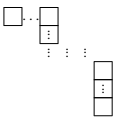
5.1.7. Lemma. *For any word w in $[n]^*$, we have*

$$(\emptyset \leftsquigarrow_{H_r} w) = \rho^{\top}_{\mathcal{RBT}_n}([w]_{\mathcal{Y}}), \qquad (w \rightsquigarrow_{H_l} \emptyset) = \rho^{\perp}_{\mathcal{RBT}_n}([w]_{\mathcal{Y}}). \tag{5.1.8}$$

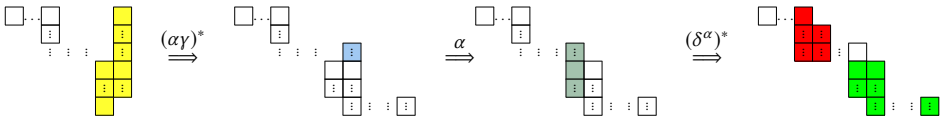
Proof. We prove the first equation of (5.1.8) by induction on the number of columns in $[w]_{\mathcal{Y}}$, the proof being similar for the second one. If $[w]_{\mathcal{Y}}$ is of length 2, then the equality is a consequence of Lemma 5.1.5. For $k \geqslant 3$, suppose that the equality holds for words of length $k-1$, and consider $[w]_s = c_1|_{\mathcal{Y}} \ldots |_{\mathcal{Y}} c_k$. By induction hypothesis, we have $(\emptyset \leftsquigarrow_{H_r} w) = \rho^{\top}_{\mathcal{RBT}_n}(c_1|_{\mathcal{Y}} \ldots |_{\mathcal{Y}} c_{k-1}) \star_{H_r} c_k$. Let us show that

$$\rho^{\top}_{\mathcal{RBT}_n}(c_1|_{\mathcal{Y}} \ldots |_{\mathcal{Y}} c_{k-1}) \star_{H_r} c_k = \rho^{\top}_{\mathcal{RBT}_n}(c_1|_{\mathcal{Y}} \ldots |_{\mathcal{Y}} c_k).$$

We will represent $\rho^{\top}_{\mathcal{RBT}_n}(c_1|_{\mathcal{Y}} \ldots |_{\mathcal{Y}} c_{k-1})$ by the following diagram:



The computation of $\rho^{\top}_{\mathcal{RBT}_n}(c_1|_{\mathcal{Y}} \ldots |_{\mathcal{Y}} c_{k-1}) \star_{H_r} c_k$ corresponds to the following reduction sequence, where some rewriting rules can be identities:



We then apply the leftmost reduction path with respect $\mathcal{RBT}_n$ on the red boxes until we obtain a quasi-ribbon tableau. After we apply rules γ in the same places where we have applied rules δ^α before. Finally, we apply the leftmost reduction path with respect $\mathcal{RBT}_n$ on the green boxes until a quasi-ribbon tableau appear which is equal by construction to $\rho^{\top}_{\mathcal{RBT}_n}(c_1|_{\mathcal{Y}} \ldots |_{\mathcal{Y}} c_{k-1}) \star_{H_r} c_k$, showing the claim. □

5.1.9. Lemma. *The rewriting system $\mathcal{RBT}_n$ is terminating.*

Proof. We prove that for any reduction $w \Rightarrow w'$ with respect to $\mathcal{RBT}_n$, we have $w \ll_{tl} w'$ for the order $\ll_{rb}$ defined in 3.5.2. Suppose first that $w \Rightarrow w'$ is a reduction with respect to $\mathcal{RS}_n$, that is, with respect to the rules α or δ^α. If the target of α consists of two or three columns (resp. one column), then we have $R_{SW}(w) <_{\text{lex}} R_{SW}(w')$ (resp. $R_{SW}(w) = R_{SW}(w')$ and $||w|| < ||w'||$). Similarly, for the reduction with respect to δ^α, we obtain $R_{SW}(w) <_{\text{lex}} R_{SW}(w')$ or $R_{SW}(w) = R_{SW}(w')$ and $||w|| < ||w'||$. Hence, for any reduction $w \Rightarrow w'$ with respect to $\mathcal{RS}_n$, we have $w \ll_{tl} w'$. Suppose now that $w \Rightarrow w'$ is a reduction with respect to $\mathcal{BS}_n \cup \mathcal{TS}_n$, we have $R_{SW}(w) = R_{SW}(w')$ and $w \sqsubseteq_{lex} w'$, and thus $w \ll_{rbl} w'$, showing the claim. □

5.1.10. Lemma. *The rewriting system $\mathcal{RBT}_n$ is confluent.*

Proof. Following 3.4.7, we show the confluence of the rewriting system $\mathcal{RBT}_n$ by showing that all its critical branchings are confluent. By Lemma 5.1.5, for any rule $|c_{i_1}|c_{i_2}| \Rightarrow |c_{j_1}|c_{j_2}|$ in $\mathcal{RBT}_n$, we have $\rho^{\top}_{\mathcal{RBT}_n}(|c_{j_1}|c_{j_2}|) = |c_{i_1} \star_{H_r} c_{i_2}|$ and $\rho^{\perp}_{\mathcal{RBT}_n}(|c_{j_1}|c_{j_2}|) = |c_{i_2} \star_{H_l} c_{i_1}|$. Then, any critical branching of $\mathcal{RBT}_n$ has the following form

$$|c_i|c_j|c_k| \begin{array}{l} \overset{\varepsilon_1}{\Longrightarrow} |c'_i|c'_j|c_k| \overset{\rho^{\top}_{\mathcal{RBT}_n}}{\Longrightarrow} |c_i \star_{H_r} c_j \star_{H_r} c_k| \\ \overset{\varepsilon_2}{\Longrightarrow} |c_i|c''_j|c'_k| \overset{\rho^{\perp}_{\mathcal{RBT}_n}}{\Longrightarrow} |c_k \star_{H_l} c_j \star_{H_l} c_i| \end{array}$$

where ε_1 and ε_2 are $\mathcal{RBT}_n$-reductions. Moreover, the right and left insertions H_r and H_l commute, [9]. Hence, the equality $c_i \star_{H_r} c_j \star_{H_r} c_k = c_k \star_{H_l} c_j \star_{H_l} c_i$, holds in Qr_n, showing the confluence of the critical branching. □

5.2. Right-bottom rectification as morphism

5.2.1. Hypoplactic monoids. The *hypoplactic monoid* $\mathbf{Hp}_n$ of rank n introduced in [13], is the monoid generated on $[n]$ and submitted to the Knuth relations (4.1.2) and the following relations:

$$zxty = xzyt \ \text{ for } \ 1 \leqslant x \leqslant y < z \leqslant t \leqslant n, \qquad tyzx = ytxz \ \text{ for } \ 1 \leqslant x < y \leqslant z < t \leqslant n. \tag{5.2.2}$$

The congruence generated by this presentation is called the *hypoplactic congruence* and is denoted by $\approx_{\mathbf{Hp}_n}$. The cross-section property for the hypoplactic monoid is proved in [13, 19].

5.2.3. Theorem. *The right-bottom rectification map $\pi_{rb} : \mathrm{Yt}_n \to \mathrm{Qr}_n$ extends into a surjective morphism of string data structures:*

$$\pi^r_{rb} : \mathbb{Y}^r_n \to \mathbb{Q}^r_n \quad (\textit{resp. } \pi^c_{rb} : \mathbb{Y}^c_n \to \mathbb{Q}^l_n). \tag{5.2.4}$$

Proof. We prove that π^r_{rb} (resp. π^c_{rb}) is a morphism of string data structures by showing Conditions **i)**, **ii)** and **iii)** of 3.4.9. Condition **i)** is a consequence of Theorem 4.2.3. Prove Condition **ii)**, that is, for any rule $d \Rightarrow d'$ in $\mathcal{RBT}_n$, we have $R_{SW}(d) \approx_{\mathbf{Hp}_n} R_{SW}(d')$. It suffices to show that $R_{SW}(s(\eta)) \approx_{\mathbf{Hp}_n} R_{SW}(t(\eta))$ for any η in $\mathcal{RBT}_n$. It is obvious for γ or δ. For rule α as in 5.1.1, we prove the property by induction on $|c_j|$, Suppose that $|c_j| = 1$, and consider $R_{SW}(s(\alpha)) = c_i^k \dots c_i^p \dots c_i^1 c_j^1$ and $R_{SW}(t(\alpha)) = c_i^p \dots c_i^1 c_i^k \dots c_i^{p+1} c_j^1$. Then we have

$$\begin{aligned} R_{SW}(t(\alpha)) &= c_i^p \dots \mathbf{c_i^1 c_i^k c_i^{k-1}} \dots c_i^{p+1} c_j^1 \overset{(4.1.2)}{=} \dots \overset{(4.1.2)}{=} c_i^p c_i^k \dots \mathbf{c_i^1 c_i^{k-1} c_i^{k-2}} \dots c_i^{p+1} c_j^1 \overset{(4.1.2)}{=} \dots \overset{(4.1.2)}{=} \\ &\mathbf{c_i^p c_i^k c_i^{p-1} c_i^{k-1}} \dots c_i^1 c_i^{k-2} \dots c_i^{p+1} c_j^1 \overset{(5.2.2)}{=} c_i^k c_i^p c_i^{k-1} c_i^{p-1} \dots \mathbf{c_i^1 c_i^{k-2} c_i^{k-3}} \dots c_i^{p+1} c_j^1 \overset{(4.1.2)}{=} \dots \overset{(4.1.2)}{=} \\ &c_i^k \mathbf{c_i^p c_i^{k-1} c_i^{p-1} c_i^{k-2}} \dots c_i^1 c_i^{k-3} \dots c_i^{p+1} c_j^1 \overset{(5.2.2)}{=} c_i^k c_i^{k-1} c_i^p c_i^{k-2} c_i^{p-1} \dots c_i^1 c_i^{k-3} c_i^{k-4} \dots c_i^{p+1} c_j^1. \end{aligned}$$

In the same way, we continue applying Relations (4.1.2) and (5.2.2) on $c_i^k c_i^{k-1} c_i^p c_i^{k-2} \dots \mathbf{c_i^1 c_i^{k-3} c_i^{k-4}} \dots c_i^{p+1} c_j^1$ and the equivalence $R_{SW}(t(\alpha)) \approx_{\mathbf{Hp}_n} R_{SW}(s(\alpha))$ holds. Suppose now that the property holds when $|c_j| = l-1$, and prove it when $|c_j| = l$. Consider $R_{SW}(s(\alpha)) = c_i^k \dots c_i^p \dots c_i^1 c_j^l \dots c_j^m \dots c_j^2 c_j^1$ and $R_{SW}(t(\alpha)) = c_i^p \dots c_i^1 c_i^k \dots c_i^{p+1} c_j^m \dots c_j^1 c_j^l \dots c_j^{m+1}$. By induction hypothesis,

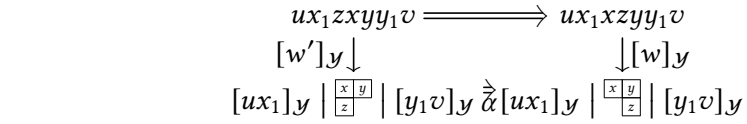
$$R_{SW}(s(\alpha)) \approx_{\mathbf{Hp}_n} c_i^p \dots c_i^1 c_i^k \dots c_i^{p+1} c_j^m \dots c_j^2 c_j^l \dots c_j^{m+1} c_j^1.$$

Moreover, we have $c_i^p \dots c_i^1 c_i^k \dots c_i^{p+1} c_j^m \dots \mathbf{c_j^2 c_j^l c_j^{l-1}} \dots c_j^{m+1} c_j^1 \overset{(4.1.2)}{=} \dots \overset{(4.1.2)}{=} c_i^p \dots c_i^1 c_i^k \dots c_i^{p+1} c_j^m \dots c_j^l c_j^{l-1} \dots \mathbf{c_j^2 c_j^{m+1} c_j^1} \overset{(4.1.2)}{=} \dots \overset{(4.1.2)}{=} R_{SW}(t(\alpha))$. Hence, we obtain that $R_{SW}(t(\alpha)) \approx_{\mathbf{Hp}_n} R_{SW}(s(\alpha))$. Similarly, we show the property for rule δ^α.

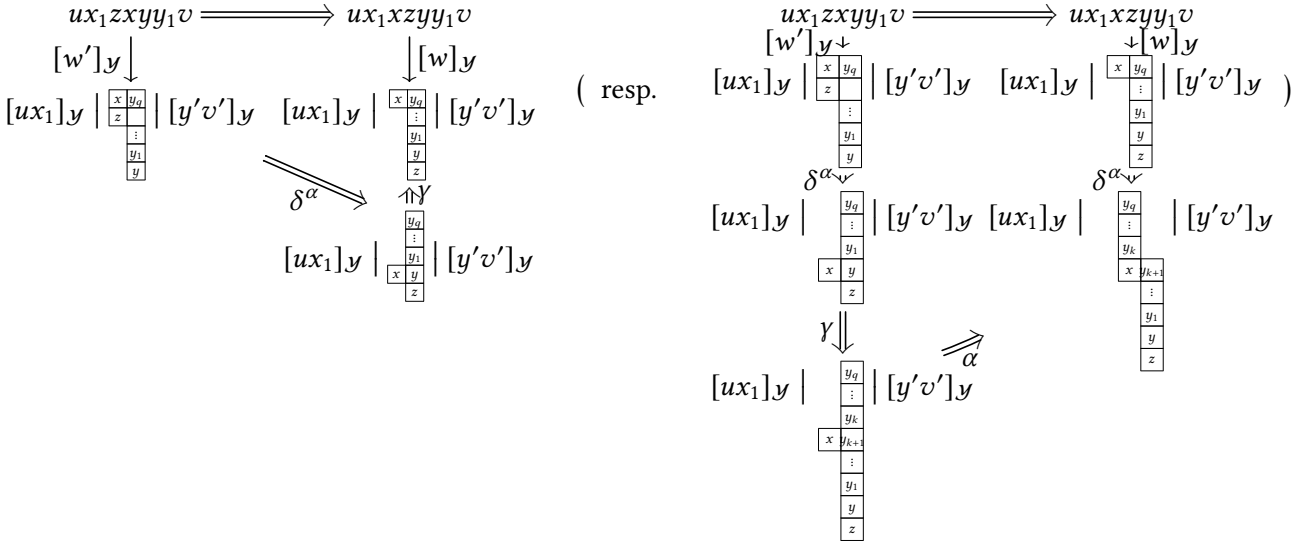
Prove Condition **iii)**, that is, for all w, w' in $[n]^*$, $w \approx_{\mathbf{Hp}_n} w'$ implies $\mathrm{Nf}([w]_{\mathcal{Y}}, \mathcal{RBT}_n) = \mathrm{Nf}([w']_{\mathcal{Y}}, \mathcal{RBT}_n)$. Since $\mathcal{RBT}_n$ is convergent, we show that for all $w, w' \in [n]^*$, $w \approx_{\mathbf{Hp}_n} w'$ implies $[w]_{\mathcal{Y}} \approx_{\mathcal{RBT}_n} [w']_{\mathcal{Y}}$.

Suppose first that $w = ux_1\mathbf{xzy}y_1v$ and $w' = ux_1\mathbf{zxy}y_1v$, for all $1 \leqslant x \leqslant y < z \leqslant n$, $u, v \in [n]^*$ and $x_1, y_1 \in [n]$, and prove that $[w]_{\mathcal{Y}} \approx_{\mathcal{RBT}_n} [w']_{\mathcal{Y}}$. We consider the following four cases:

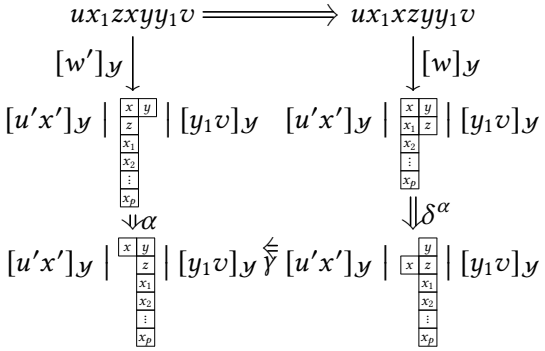
Case 1. $x_1 \leqslant x$ and $y \leqslant y_1$.



Case 2. $x_1 \leqslant x$ and $y > y_1$. Suppose that $v = y_2 \dots y_q y' v'$ such that $y_1 > y_2 > \dots > y_q \leqslant y'$. We consider the following sub-case: $x \leqslant y_i$, for all $1 \leqslant i \leqslant q$ (resp. $x \leqslant y_i$, for all $1 \leqslant i \leqslant k-1$ and $x > y_k$).

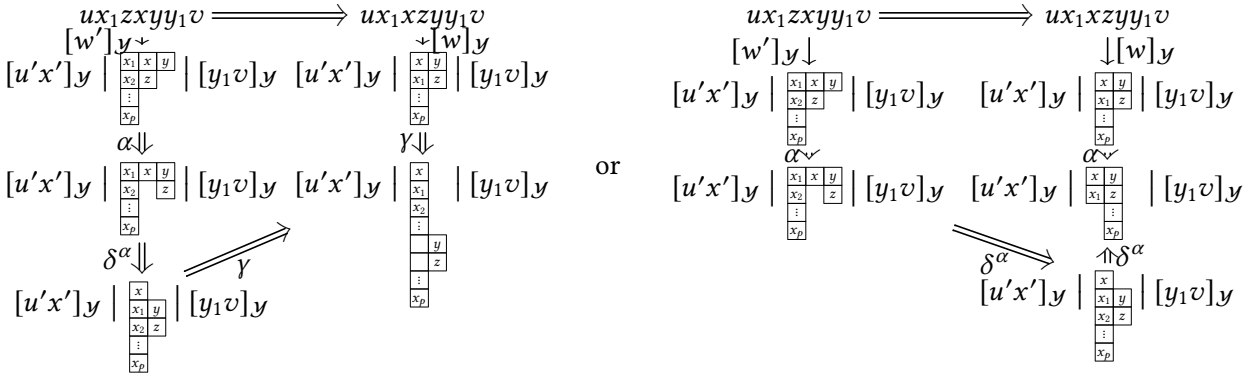


Case 3. $x < z < x_1$ and $y \leqslant y_1$. Suppose $u = u'x'x_p \dots x_1$ such that $x' \leqslant x_p > \dots > x_1$.



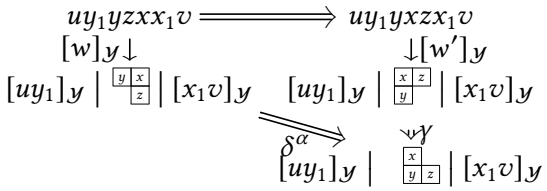
The case $x < z < x_1$ and $y > y_1$ is similar to **Case 2**.

Case 4. $x < x_1 \leqslant z$ and $y \leqslant y_1$. Suppose $u = u'x'x_p \dots x_1$ such that $x' \leqslant x_p > \dots > x_1$.



The case $x < x_1 \leqslant z$ and $y > y_1$ is similar to **Case 2**.

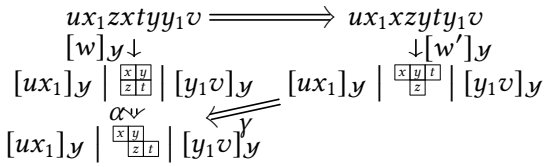
Suppose that $w = uy_1\mathbf{yzx}x_1v$ and $w' = uy_1\mathbf{yxz}x_1v$, for all $1 \leqslant x < y \leqslant z \leqslant n$, $u, v \in [n]^*$ and $x_1, y_1 \in [n]$, and prove that $[w]_{\mathcal{Y}} \approx_{\mathcal{RBT}_n} [w']_{\mathcal{Y}}$. If $y_1 \leqslant y$, $x \leqslant x_1$ and $z \leqslant x_1$, then



Similarly, we study the cases ($y_1 > y$ and $x > x_1$), ($y_1 \leqslant y$ and $x > x_1$) and ($y_1 > y$ and $x \leqslant x_1$).

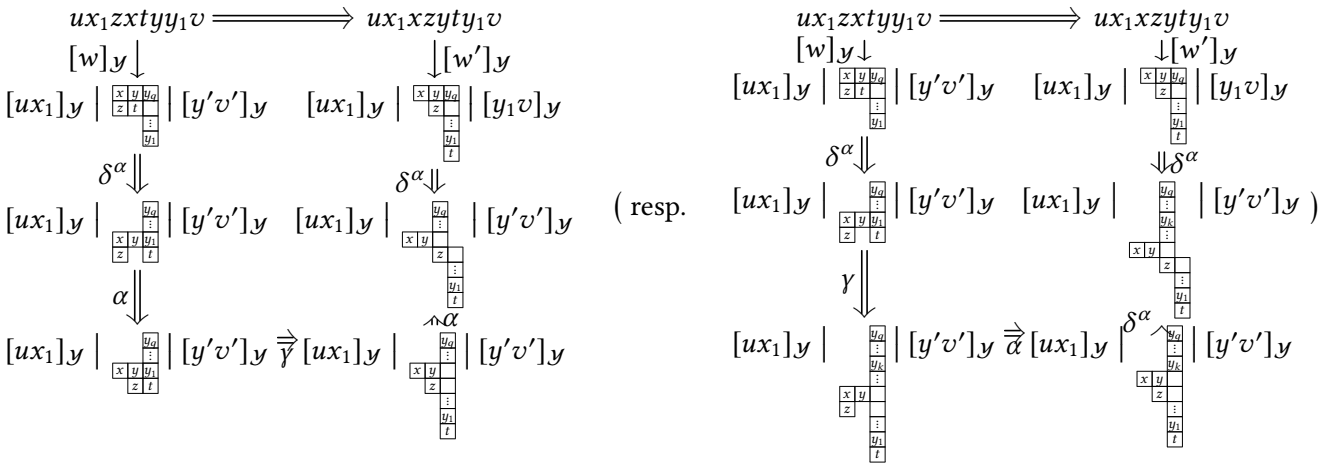
Suppose now $w = ux_1\mathbf{zxty}y_1v$ and $w' = ux_1\mathbf{xzyt}y_1v$, for all $1 \leqslant x \leqslant y < z \leqslant t \leqslant n$, $u, v \in [n]^*$ and $x_1, y_1 \in [n]$, and prove that $[w]_{\mathcal{Y}} \approx_{\mathcal{RBT}_n} [w']_{\mathcal{Y}}$. We will consider the following cases:

Case 1. $x_1 \leqslant x$ and $y \leqslant y_1$. If $t \leqslant y_1$, then

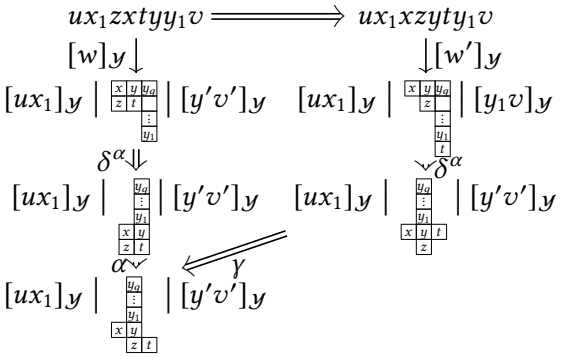


Otherwise, suppose that $v = y_2 \dots y_q y'v'$ such that $t > y_1 > \dots > y_q \leqslant y'$. If $y \leqslant y_i$, for all $1 \leqslant i \leqslant q$

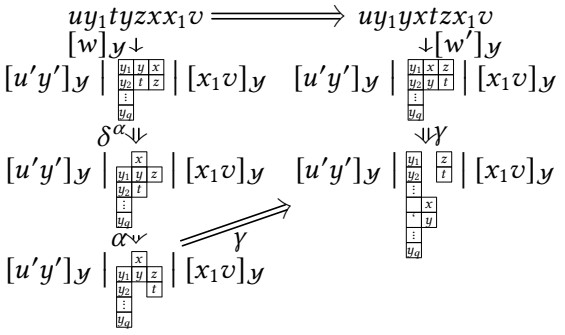
(resp. $y \leqslant y_i$, for all $1 \leqslant i \leqslant k-1$ and $y > y_k$), then



Case 2. $x_1 \leqslant x$ and $y > y_1$. In the same way, we study the cases ($x_1 > x$ and $y > y_1$) and ($x_1 > x$ and $y \leqslant y_1$). Suppose $v = y_2 \dots y_q y' v'$ such that $y_1 > \dots > y_q \leqslant y'$.



Suppose finally that $w = uy_1\mathbf{tyzx}x_1v$ and $w' = uy_1\mathbf{yxtz}x_1v$, for all $1 \leqslant x < y \leqslant z < t \leqslant n$, $u, v \in [n]^*$ and $x_1, y_1 \in [n]$, and prove that $[w]_{\mathcal{Y}} \approx_{\mathcal{RBT}_n} [w']_{\mathcal{Y}}$. We study the case ($y_1 \leqslant x < y$ and $x \leqslant x_1$), where $u = u'y'y_q \dots y_2$ such that $y_q > \dots y_2 > y_1$, $y' \leqslant y_q$ and $y_2 \leqslant y$, the others cases being similar.



Prove finally that the map π_{rb} is surjective. Recall from [3, Sect.8.2.] the *north-west rectification* map $\pi^{nw} : \mathrm{Qr}_n \to \mathrm{Yt}_n$ that transforms a quasi-ribbon q into a Young tableau by sliding all its columns

to the top until the topmost box of each is on the first row of q and then by sliding all its boxes leftwards along their rows until the leftmost box in each row is in the first column and such that all the rows remain connected. The equality $\pi^{nw}(q) = C_{\mathbb{Y}_n^r}(R_{SW}(q))$ holds, for any q in Qr_n, [3, Proposition 8.9], showing that $R_{SW}(\pi^{nw}(q)) \approx_{\mathbf{P}_n} R_{SW}(q) \approx_{\mathbf{Hp}_n} R_{SW}(q)$. Moreover, since the rewriting system $\mathcal{RBT}_n$ is compatible with $\approx_{\mathbf{Hp}_n}$, the equivalence $R_{SW}(\pi^{nw}(q)) \approx_{\mathbf{Hp}_n} R_{SW}(\pi_{rb}(\pi^{nw}(q)))$ holds, for any q in Qr_n, showing that $R_{SW}(q) \approx_{\mathbf{Hp}_n} R_{SW}(\pi_{rb}(\pi^{nw}(q)))$. Then, since $\mathcal{RBT}_n$ computes $\approx_{\mathbf{Hp}_n}$, the equality $\pi_{rb}(\pi^{nw}(q)) = q$ holds in Qr_n. Hence, the map π_{rb} is surjective. □

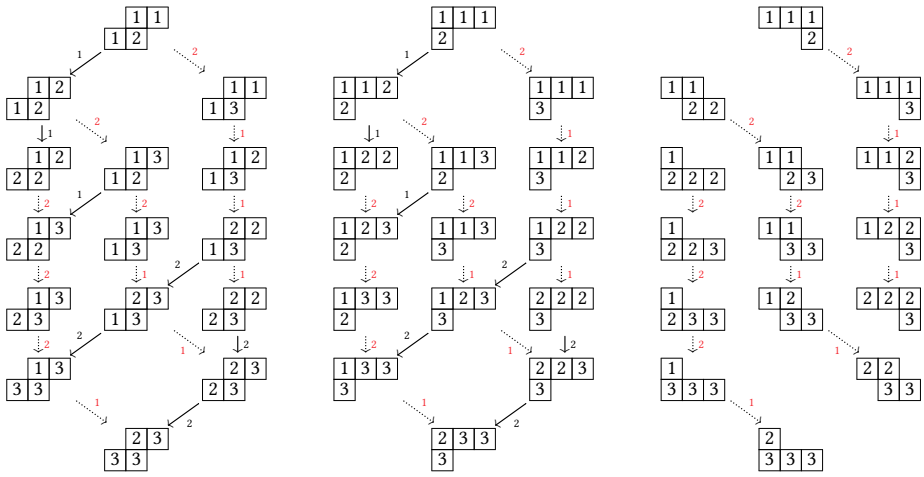
As a direct consequence of this result, we deduce

5.2.5. Corollary. *The string data structure $\mathbb{Q}_n^r$ (resp. $\mathbb{Q}_n^l$) satisfies the cross-section property for the monoid $\mathbf{Hp}_n$, and the structure monoid $\mathbf{M}(\mathbb{Q}_n^r)$ is isomorphic to $\mathbf{Hp}_n$.*

As a consequence of Proposition 3.3.8 and Theorem 5.2.3, we deduce

5.2.6. Corollary. *The morphism of string data structures $\pi_{rb}^r : \mathbb{Y}_n^r \to \mathbb{Q}_n^r$, (resp. $\pi_{rb}^c : \mathbb{Y}_n^c \to \mathbb{Q}_n^l$) induces a crystal isomorphism from $\Gamma(q\mathcal{K}, d)$ to $\Gamma(q\mathcal{K}, \pi_{rb}(d))$, for every d in Yt_n.*

5.2.7. Remark. Note finally that the procedure that transforms a skew tableau into a quasi-ribbon tableau by applying first the jeu de taquin followed by the right-bottom rectification can be also described by crystal isomorphisms using Kashiwara and quasi-Kashiwara crystals as shown in the following example:



where dotted arrows show action of the quasi-Kashiwara's operators and the other ones show action of the Kashiwara's operators. These connected components are isomorphic and every skew tableau, Young tableau and quasi-ribbon tableau situated at the same place in these crystal isomorphisms can be related by sliding.

References

[1] David Aldous and Persi Diaconis. Longest increasing subsequences: from patience sorting to the Baik-Deift-Johansson theorem. *Bull. Amer. Math. Soc. (N.S.)*, 36(4):413–432, 1999.

[2] Alan J. Cain, Robert D. Gray, and António Malheiro. Rewriting systems and biautomatic structures for Chinese, hypoplactic, and Sylvester monoids. *Internat. J. Algebra Comput.*, 25(1-2):51–80, 2015.

[3] Alan J. Cain and António Malheiro. Crystallizing the hypoplactic monoid: from quasi-Kashiwara operators to the Robinson-Schensted-Knuth-type correspondence for quasi-ribbon tableaux. *J. Algebraic Combin.*, 45(2):475–524, 2017.

[4] William Fulton. *Young tableaux*, volume 35 of *London Mathematical Society Student Texts*. Cambridge University Press, Cambridge, 1997. With applications to representation theory and geometry.

[5] Stéphane Gaussent, Yves Guiraud, and Philippe Malbos. Coherent presentations of Artin monoids. *Compos. Math.*, 151(5):957–998, 2015.

[6] Yves Guiraud and Philippe Malbos. Higher-dimensional normalisation strategies for acyclicity. *Adv. Math.*, 231(3-4):2294–2351, 2012.

[7] Yves Guiraud and Philippe Malbos. Polygraphs of finite derivation type. *Math. Structures Comput. Sci.*, 28(2):155–201, 2018.

[8] Nohra Hage and Philippe Malbos. Knuth's coherent presentations of plactic monoids of type A. *Algebr. Represent. Theory*, 20(5):1259–1288, 2017.

[9] Nohra Hage and Philippe Malbos. Coherence of monoids by insertions and Chinese syzygies. preprint, arXiv:1901.09879, February 2019.

[10] Masaki Kashiwara. Crystallizing the q-analogue of universal enveloping algebras. In *Proceedings of the International Congress of Mathematicians, Vol. I, II (Kyoto, 1990)*, pages 791–797. Math. Soc. Japan, Tokyo, 1991.

[11] Masaki Kashiwara and Toshiki Nakashima. Crystal graphs for representations of the q-analogue of classical Lie algebras. *J. Algebra*, 165(2):295–345, 1994.

[12] Donald E. Knuth. Permutations, matrices, and generalized Young tableaux. *Pacific J. Math.*, 34:709–727, 1970.

[13] Daniel Krob and Jean-Yves Thibon. Noncommutative symmetric functions iv: Quantum linear groups and hecke algebras at q = 0. *Journal of Algebraic Combinatorics*, 6(4):339–376, Oct 1997.

[14] Daniel Krob and Jean-Yves Thibon. Noncommutative symmetric functions v: A degenerate version of $\mathrm{U}_q(gl_n)$. *International Journal of Algebra and Computation*, 09(03n04):405–430, 1999.

[15] Alain Lascoux, Bernard Leclerc, and Jean-Yves Thibon. Crystal graphs and q-analogues of weight multiplicities for the root system A_n. *Lett. Math. Phys.*, 35(4):359–374, 1995.

[16] Alain Lascoux and Marcel-P. Schützenberger. Le monoïde plaxique. In *Noncommutative structures in algebra and geometric combinatorics (Naples, 1978)*, volume 109 of *Quad. "Ricerca Sci."*, pages 129–156. CNR, Rome, 1981.

[17] Cristian Lenart. On the combinatorics of crystal graphs. I. Lusztig's involution. *Adv. Math.*, 211(1):204–243, 2007.

[18] Maxwell Newman. On theories with a combinatorial definition of "equivalence". *Ann. of Math. (2)*, 43(2):223–243, 1942.

REFERENCES

[19] Jean-Christophe Novelli. On the hypoplactic monoid. volume 217, pages 315–336. 2000. Formal power series and algebraic combinatorics (Vienna, 1997).

[20] C. Schensted. Longest increasing and decreasing subsequences. *Canad. J. Math.*, 13:179–191, 1961.

[21] M.-P. Schützenberger. La correspondance de Robinson. pages 59–113. Lecture Notes in Math., Vol. 579, 1977.

[22] Alfred Young. On Quantitative Substitutional Analysis. *Proc. London Math. Soc. (2)*, 28(4):255–292, 1928.

Nohra Hage
nohra.hage@usj.edu.lb
École supérieure d'ingénieurs de Beyrouth (ESIB)
Université Saint-Joseph de Beyrouth (USJ), Liban

Philippe Malbos
malbos@math.univ-lyon1.fr
Univ Lyon, Université Claude Bernard Lyon 1
CNRS UMR 5208, Institut Camille Jordan
43 blvd. du 11 novembre 1918
F-69622 Villeurbanne cedex, France