Coherence of monoids by insertions and Chinese syzygies

Nohra Hage Philippe Malbos

Abstract – A data structure describes a way to organize and store a collection of data, and defines primitive efficient functions and operations on data such as constructors, modifications and access maps. In this work, we consider data structures on strings as combinatorial descriptions of structured words having a theory of normal forms defined by insertion algorithms. We show that an insertion map induces a product on data and we give necessary conditions making this product associative. We deduce a rewriting description of the cross-section property for the structure monoid associated to a string data structure. We show how to compute a coherent presentation of the structure monoid made of rewriting rules defined by insertion on words and whose syzygies are defined as relations among insertion algorithms. As an illustration, we show how our constructions can be applied to Chinese monoids, and we make explicit the shape of syzygies of the Chinese congruence.

Keywords – String rewriting systems, data structures, normal forms, plactic monoids, Chinese monoids.

M.S.C. 2010 – Primary: 20M35, 68Q42. **Secondary:** 20M05, 18D05.

1. Introduction

String data structures and syzygies

A data structure describes a way to organize, manage and store a collection of structured data. It defines primitive efficient functions and operations on data such as constructors, modification and access maps. In this article, we introduce the notion of string data structure as a combinatorial description of structured words on ordered alphabets. Such data structures appear in many contexts in combinatorial algebra, combinatorics and fundamental computer science through combinatorial data structures, such as arrays, tableaux, staircases or binary search trees. For instance, array data structures can be used to describe normal forms for plactic monoids of type A with Young tableaux [15, 33, 38, 45], plactic monoids of classical types with symplectic and orthogonal tableaux, [35, 36], Chinese monoids with staircases, [10, 14], hypoplactic monoids with quasi-ribbon tableaux, [40], left and right patience sorting monoids with left and right patience sorting tableaux, [9, 44], and stalactic monoids with stalactic tableaux [26, 41]. Binary search trees, binary search trees with multiplicities and pairs of twin binary search trees can be used to describe normal forms for sylvester monoids, [25], taiga monoids, [41], and Baxter monoids, [17].

Cross-section by insertion. In all of these situations, structured data are constructed using insertion algorithms, and give interpretations of congruence relations by a characterization of a cross-section property for the presented monoids. Explicitly, given a string data structure $\mathbb{S}$ over an alphabet A defined by a right insertion algorithm I, to each word $w = x_1x_2\dots x_k$ on A it is associated a structured data $C_{\mathbb{S}}(w)$ obtained by insertion of the word w in the empty data $\emptyset$ by application of the insertion I step by step:

$$C_{\mathbb{S}}(w) := (\emptyset \leftsquigarrow_I w) = ((((\emptyset \leftsquigarrow_I x_1) \leftsquigarrow_I x_2) \leftsquigarrow_I \dots) \leftsquigarrow_I x_{k-1}) \leftsquigarrow_I x_k$$

where $t \leftsquigarrow_I x$ denotes the insertion of the letter x in the data t. Structured data form a cross-section property for a congruence relation $\approx$ on the free monoid A^*: for any words w, w' on A, $w \approx w'$ if and only if the insertion algorithm yields the same structured data: $C_{\mathbb{S}}(w) = C_{\mathbb{S}}(w')$.

In this work, we explain how insertion algorithms induce a product on the structured data, and we give necessary conditions to have an associative product. We relate the cross-section property with respect to a string data structure to a confluence property of a string rewriting system whose rules are defined by insertion. Finally, we show how to compute a reduced coherent presentation of a monoid presented by a string data structure, made of generators, rewriting rules describing the insertion of letters in words, and syzygies of the presentation interpreted in terms of relations among the insertion algorithms. This is the first step in an explicit construction of a cofibrant approximation of the monoid in the category of $(\infty, 1)$-categories, [19, 20], whose acyclicity is proved by an iterative construction of a normalisation reduction strategy.

Tableaux and plactic congruence. String data structures appear naturally in the study of the plactic congruence of type A on the free monoid over $[n] := \{1, \dots, n\}$, generated by the *Knuth relations* $zxy = xzy$ for all $1 \leqslant x \leqslant y < z \leqslant n$ and $yzx = yxz$ for all $1 \leqslant x < y \leqslant z \leqslant n$. This congruence emerged from the works of Schensted [42] and Knuth [29] on the combinatorial study of Young tableaux, and the *plactic monoid* of type A of rank n, denoted by $\mathbf{P}_n$, introduced by Lascoux and Schützenberger in [33] is the monoid generated on $[n]$ and submitted to the plactic congruence. Knuth proved in [29] that the set Yt_n of Young tableaux over $[n]$ satisfies the cross-section property for the plactic congruence.

Schensted introduced two algorithms to insert an element x of $[n]$ into a tableau t of Yt_n, [42]: the *right insertion algorithm* S_r, we denote $t \leftsquigarrow_{S_r} x$ and the *left insertion algorithm* S_l, we denote $x \rightsquigarrow_{S_l} t$. Denote by $R_{col} : \mathrm{Yt}_n \to [n]^*$ the map that reads a tableau column by column from left to right and from bottom to top. The insertion algorithms allow to define two internal products on Yt_n by setting

$$t \star_{S_r} t' = (t \leftsquigarrow_{S_r} R_{col}(t')), \qquad t \star_{S_l} t' = (R_{col}(t') \rightsquigarrow_{S_l} t),$$

for all tableaux t, t' in Yt_n. Knuth showed in [29] that these products define on Yt_n a structure of monoid that is isomorphic to the plactic monoid $\mathbf{P}_n$, see also [34]. The associativity of these products is an immediate consequence of the commutation of the two insertion algorithms, that is

$$y \rightsquigarrow_{S_l} (t \leftsquigarrow_{S_r} x) \;=\; (y \rightsquigarrow_{S_l} t) \leftsquigarrow_{S_r} x$$

holds for all tableau t and x, y in $[n]$, as shown by Schensted in [42]. We show that these insertion algorithms define a string data bistructure on the set Yt_n. We explain how the cross-section property can be deduced from this structure and how to relate it to the confluence property of a rewriting system defined on the set Yt_n. The study of plactic monoids of type A using string rewriting systems on Knuth generators is not straightforward. Indeed, for $n \geqslant 4$ they do not admit finite completion with respect to the lexicographic order, [31]. However, finite completions can be obtained by adding new generators in the quasi-center of the monoid, such as column generators or row generators, [2, 4], see also [4, 6, 23] for classical types. Such confluent and terminating presentations can be used to make explicit coherent presentations of plactic monoids giving all the relations among the relations of the presentations, [24]. In Section 4, we explain that the confluence of the column and row presentations are direct consequences of the commutation of Schensted's insertion algorithms.

Main results and organization of the article

Let us present the main results of this article. Section 2 gives some preliminaries on presentations of monoids by string rewriting systems and coherent presentations.

String data structures. In Section 3 we introduce the notion of *string data structure* over a totally ordered alphabet A as a set of combinatorial data describing structured words equipped with insertion and reading maps. Explicitly, a string data structure $\mathbb{S}$ over A is quadruple (D, ℓ, I, R) made of a set D of data, a reading map ℓ of words in the free monoid A^* on A, a reading map $R : D \to A^*$ and a one-element insertion map $I : D \times A \to D$ satisfying conditions given by Definition 3.1.1. The map I extends into a map $I_\ell : D \times A^* \to D$ defined recursively by

$$I_\ell(d, u) = I_\ell(I(d, x_1), x_2 \dots x_k) \quad \text{and} \quad I_\ell(d, \lambda) = d$$

for all d in D and u in A^* with $x_1 \dots x_k = \ell(u)$. The map $C_{\mathbb{S}} := I_\ell(\emptyset, -)$ is called the *constructor* of $\mathbb{S}$. We say that $\mathbb{S}$ is *right* (resp. *left*) if the insertion map I_ℓ is defined with respect to the left-to-right reading ℓ_l (resp. right-to-left reading ℓ_r). In that case, $I_{\ell_l}(d, u)$ (resp. $I_{\ell_r}(d, u)$) will be denoted by $d \leftsquigarrow_{I_{\ell_l}} u$ (resp. $u \rightsquigarrow_{I_{\ell_r}} d$).

Structure monoid. We define an internal product $\star_I$ on D by setting $d \star_I d' := I_\ell(d, R(d'))$ for all d, d' in D. By definition the product $\star_I$ is unitary, and the string data structure $\mathbb{S}$ is called *associative* if $\star_I$ is associative. The set D with the product $\star_I$ is a monoid called the *structure monoid* of $\mathbb{S}$, denoted by $\mathbf{M}(D, I)$. We say that an associative string data structure *presents* a monoid $\mathbf{M}$ if its structure monoid is isomorphic to $\mathbf{M}$. We will consider the rewriting system $\mathcal{R}(D, \mathbb{S})$ on D, whose rules are

$$\gamma_{d,d'} : d \cdot d' \to d \star_I d'$$

for any d, d' in D. It is terminating, moreover it is confluent when $\mathbb{S}$ is associative. It is thus a convergent presentation of the monoid $\mathbf{M}(D, I)$, and the set of $\mathcal{R}(D, \mathbb{S})$-normal forms satisfies the cross-section property for $\mathbf{M}(D, I)$. We say that an associative string data structure $\mathbb{S}$ over A satisfies the *cross-section property* for a congruence relation $\approx$ on A^*, if $u \approx v$ holds if and only if $C_{\mathbb{S}}(u) = C_{\mathbb{S}}(v)$ holds for all u, v in A^*. A string data structure $\mathbb{S}$ is *compatible* with a congruence relation $\approx$ on A^*, if for all d in D and u, v in A^*, $u \approx v$ implies $I_\ell(d, u) = I_\ell(d, v)$, and $RC_{\mathbb{S}}(u) \approx u$.

Theorem 3.1.13 states that a right (resp. left) associative $\mathbb{S}$, is compatible with a congruence relation $\approx$ on A^* if, and only if, it satisfies the cross-section property for $\approx$ if, and only if, it presents the quotient monoid $A^*/\approx$ (resp. the opposite of the quotient monoid $A^*/\approx$).

Moreover, Proposition 3.1.18 states that when $\mathbb{S}$ is right (resp. left) associative, the rules

$$\gamma_{d,d'} : R(d)R(d') \to R(d \star_I d')$$

for all d, d' in D such that $R(d \star_I d') \neq R(d)R(d')$, form a convergent rewriting system $\mathcal{R}(R)$ on A, and that $\mathbb{S}$ presents the monoid (resp. opposite monoid) presented by $\mathcal{R}(R)$. It follows that $\mathcal{R}(R)$ is a convergent presentation of the structure monoid $\mathbf{M}(D, I)$. As a consequence, one can prove that an associative string data structure $\mathbb{S}$ satisfies the cross-section property for a congruence relation $\approx$ just by showing that $\mathcal{R}(R)$ presents the quotient monoid $A^*/\approx$.

Commutation of insertions. In Subsection 3.2, we define a *string data bistructure* over A as a quadruple (D, I, J, R) such that I and J are one-element insertions on D that define a right and left string data structure over A respectively and *commute*, that is the following condition

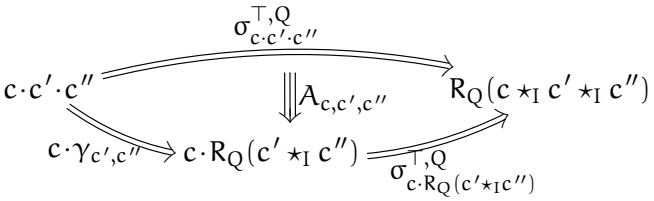
$$(x \rightsquigarrow_I d) \leftsquigarrow_J y \;=\; x \rightsquigarrow_I (d \leftsquigarrow_J y)$$

holds for all d in D and x, y in A. Theorem 3.2.3 proves that commutation of insertion induces the associativity of products $\star_I$ and $\star_J$ and the commutation relation $d \star_I d' = d' \star_J d$ for all d, d' in D. Moreover, in that case the structure monoids $\mathbf{M}(D, I)$ and $\mathbf{M}(D, J)$ are anti-isomorphic. As an example, we show in Subsection 3.3, that right and left Schensted's insertion algorithms equip the set of Young tableaux with a string data bistructure that presents the plactic monoid $\mathbf{P}_n$. We illustratre also that Lecouvey's left insertion algorithms define left string data structures on symplectic tableaux, [35] and orthogonal tableaux, [36] used to characterize cross-section properties for plactic monoids of classical type C, B and D respectively. However, the existence of a right insertion algorithm on symplectic and orthogonal tableaux that commutes with Lecouvey's left insertion, and thus a string data bistructure on these tableaux is still an open problem. In Subsection 3.4 we give other instances of string data structure on quasi-ribbon tableaux, [30, 40], binary search trees, [25], and patience sorting tableaux, [9, 44]. Note that the existence of a string data bistructure on these structured data is also still an open problem.

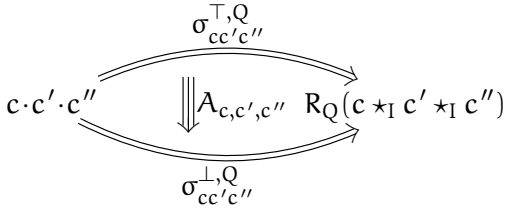
Generating string data structures and coherence by insertion. In general the rewriting system $\mathcal{R}(D, \mathbb{S})$ on D is infinite. In some situations, we can reduce the set of generators to a finite subset Q of D in order to have a finite rewriting system $\mathcal{R}(Q, \mathbb{S})$ on Q that is Tietze equivalent to $\mathcal{R}(D, \mathbb{S})$. This is the motivation of the notion of a *generating set* of a string data structure $\mathbb{S}$ defined in Subsection 4.1 as a subset Q of D such that any element d in D can be decomposed as $d = c_1 \star_I c_2 \star_I \ldots \star_I c_k$, where $c_1, \ldots, c_k \in Q$, and that there exists a unique decomposition $d = c_1 \star_I \ldots \star_I c_l$, with $c_1, \ldots, c_l$ in Q satisfying $c_i \star_I c_{i+1} \notin Q$ for all $1 \leqslant i \leqslant l-1$, and $R(d) = R(c_1) \ldots R(c_l)$ holds in A^*. For instance, the set of columns over $[n]$ and the set of rows over $[n]$ generate the set of Young tableaux Yt_n equipped with Schensted's insertions. We define the rewriting system $\mathcal{R}(Q, \mathbb{S})$ on Q whose rules are

$$\gamma_{c,c'} : c \cdot c' \to R_Q(c \star_I c')$$

for all c, c' in Q, whenever $c \cdot c' \neq R_Q(c \star_I c')$. In most applications, the termination of $\mathcal{R}(Q, \mathbb{S})$ can be showed by introducing a well-founded order on the free monoid on Q. A generating set Q of $\mathbb{S}$ is called *well-founded* if the rewriting system $\mathcal{R}(Q, \mathbb{S})$ is terminating. When $\mathbb{S}$ is right associative having a well-founded generating set Q, Proposition 4.1.8 states that $\mathcal{R}(Q, \mathbb{S})$ is a convergent presentation of the structure monoid $\mathbf{M}(D, I)$. As a consequence, the set of $\mathcal{R}(Q, \mathbb{S})$-normal forms satisfies the cross-section property for $\mathbf{M}(D, I)$. The last result of Section 4, Theorem 4.2.1, shows how to extend $\mathcal{R}(Q, \mathbb{S})$ into a coherent convergent presentation of the monoid $\mathbf{M}(D, I)$ when $\mathbb{S}$ is a right associative string data structure, generated by a well-founded set Q. The generating 3-cells have shapes



for any c, c', c'' in Q such that $c \cdot c' \neq R_Q(c \star_I c')$ and $c' \cdot c'' \neq R_Q(c' \star_I c'')$, where $\sigma^{\top,Q}$ is the leftmost reduction strategy of $\mathcal{R}(Q, \mathbb{S})$. In particular, we show that when D is equipped by a bistructure (D, I, J, R) and generated by a well-founded set Q, then the generating 3-cells $A_{c,c',c''}$ can be written



where $\sigma^{\top,Q}$ and $\sigma^{\bot,Q}$ are the leftmost and rightmost normalisation strategy corresponding to the application of the insertions I and J respectively.

String data structures on Chinese staircases. As an illustration, we construct in Section 5 a string data bistructure that presents the Chinese monoid introduced in [14] by Duchamp and Krob in their classification of monoids with growth similar to that of the plactic monoid. The *Chinese monoid* of rank n, denoted by $\mathbf{C}_n$, is the monoid generated by $[n]$ and submitted to the relations $zyx = zxy = yzx$

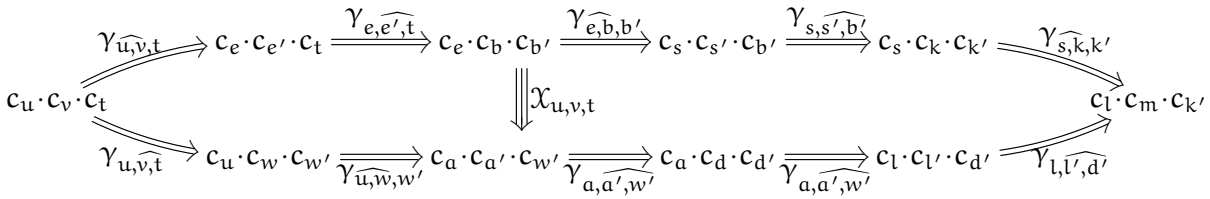
for all $1 \leqslant x \leqslant y \leqslant z \leqslant n$. This Chinese congruence was interpreted in [10] by *Chinese staircases* and the authors prove that the set Ch_n of Chinese staircases over $[n]$ satisfies the cross-section property for the monoid $\mathbf{C}_n$. We recall in Subsection 5.1 the structure of Chinese staircase, the right insertion algorithm C_r and the left insertion algorithm C_l in Chinese staircases introduced in [10] and [5] respectively. Theorem 5.1.4, shows that these two insertions commute, that is, for all staircase t in Ch_n and x, y in $[n]$, the following equality holds in Ch_n:

$$y \rightsquigarrow_{C_l} (t \leftsquigarrow_{C_r} x) \;=\; (y \rightsquigarrow_{C_l} t) \leftsquigarrow_{C_r} x.$$

As a consequence, the right and left insertions with the row reading R_r induce a string data bistructure on Chinese staircases over $[n]$, that implies, by Theorem 3.2.3, that the compositions $\star_{C_r}$ and $\star_{C_l}$ are associative.

In Subsection 5.2 we construct a finite semi-quadratic convergent presentation $\mathcal{R}(Q_n, \mathbb{C}_n)$ of the monoid $\mathbf{C}_n$ induced by the right string data structure $\mathbb{C}_n := (\mathrm{Ch}_n, C_r, \ell_l, R_r)$, and whose set of generators Q_n is made of columns over $[n]$ of length at most 2 and square generators. We deduce that the set of normal forms with respect to $\mathcal{R}(Q_n, \mathbb{C}_n)$, called *Chinese normal forms*, satisfies the cross-section property for the monoid $\mathbf{C}_n$. Note that finite convergent presentations of Chinese monoids were already obtained in [11, 22], by completion of Chinese relations, and in [5] by adding column generators. However, these presentations are not semi-quadratic, and thus it is difficult to extend these presentations into a coherent one.

Finally, Theorem 5.3.12 extends the rewriting system $\mathcal{R}(Q_n, \mathbb{C}_n)$ into a finite convergent coherent presentation of the monoid $\mathbf{C}_n$ by adjunction of generating 3-cells with the following degagonal form



for any c_u, c_v, c_t in Q_n such that $c_u \cdot c_v$ and $c_v \cdot c_t$ are not Chinese normal forms, and where the 2-cells $\gamma_{-,-}$ denote either a rewriting rule of $\mathcal{R}(Q_n, \mathbb{C}_n)$ or an identity.

2. Preliminaries on rewriting

This preliminary section recalls the basic notions of rewriting we use in this article. For a fuller account of the theory, we refer the reader to [3]. In Subsection 2.2 we will recall from [16, 21] the notion of coherent presentation of a monoid that extends the notion of a presentation by globular homotopy generators taking into account all the relations amongst the relations.

We will denote by X^* the free monoid of *words* written in the alphabet X, the product being concatenation of words, and the identity being the empty word, denoted by λ. We will denote by $u = x_1 \dots x_k$ a word in X^* of *length* k, where $x_1, \dots, x_k$ belong to X. The length of a word u will be denoted by $|u|$.

2.1. String rewriting

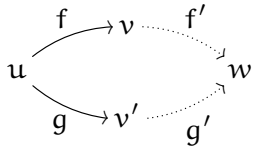
2.1.1. String rewriting systems. A *(string) rewriting system* on X is a subset R of $X^* \times X^*$. An element (u, v) of R is called a *rule* and will be denoted by $u \to v$. A *one step reduction* is defined by $wuw' \to wvw'$ for all words w, w' in X^* and rule $\beta : u \to v$ in R, and will be denoted by $w\beta w'$. One step reductions form the *reduction relation* on X^* denoted by $\to_R$. A *rewriting path* with respect to R is a finite or infinite sequence $u_0 \to_R u_1 \to_R u_2 \to_R \cdots$. This corresponds to the reflexive and transitive closure of the relation $\to_R$, that we denote by $\to_R^*$. A word u in X^* is R-*reduced* if there is no reduction with source u. A R-*normal form* for a word u in X^* is a R-reduced word v such that u reduces into v. The rewriting system R *terminates* if it has no infinite rewriting path, and it is *(weakly) normalizing* if every word u in X^* reduces to some R-normal form. A rewriting system R is *reduced* if, for every rule $\beta : u \to v$ in R, the source u is $(R \setminus \{\beta\})$-reduced and the target v is R-reduced. The reflexive, symmetric and transitive closure of $\to_R$ is the congruence on X^* generated by R, that we denote by $\approx_R$. The monoid presented by R is the quotient of the free monoid X^* by the congruence $\approx_R$. A presentation of a monoid $\mathbf{M}$ is a rewriting system whose presented monoid is isomorphic to $\mathbf{M}$. Two rewriting systems are *Tietze equivalent* if they present isomorphic monoids. Recall that a *Tietze transformation* between two rewriting systems is a sequence of *elementary Tietze transformations*, defined on a rewriting system R on X by the following operations:

i) adjunction or elimination of an element x in X and of a rule $\beta : u \to x$, where u is an element in X^* that does not contain x,

ii) adjunction or elimination of a rule $\beta : u \to v$ such that u and v are equivalent by the congruence generated by $R \setminus \{\beta\}$.

One shows that two rewriting systems are Tietze equivalent if, and only if, there exists a Tietze transformation between them. We refer the reader to [16, Subsection 2.1] for more details on Tietze transformations.

2.1.2. Confluence. A *branching* (resp. *local branching*) of a rewriting system R on X is a non ordered pair (f, g) of reductions (resp. *one step reductions*) of R on the same word. A branching is *aspherical* if it is of the form (f, f), for a rewriting step f and *Peiffer* when it is of the form (fv, ug) for rewriting steps f and g with source u and v respectively. The *overlapping* branchings are the remaining local branchings. An overlapping local branching is *critical* when it is minimal for the order $\sqsubseteq$ generated by the relations $(f, g) \sqsubseteq (wfw', wgw')$, given for all local branching (f, g) and words w, w' in X^*. A branching (f, g) is *confluent* if there exist reductions f' and g' reducing to the same word:



(2.1.3)

The rewriting system R is *confluent* if all of its branchings are confluent, and *convergent* if it is both confluent and terminating. If R is convergent, then every word u in X^* has a unique R-normal form.

2.1.4. Normalization strategies. Recall that a *reduction strategy* for a rewriting system R on X specifies a way to apply the rules in a deterministic way. It is defined as a mapping ϑ of every word u in X^* to a rewriting step ϑ_u with source u. When R is normalizing, a *normalization strategy* is a mapping σ of every word u to a rewriting path σ_u with source u and target a chosen R-normal form of u. For a reduced rewriting system, we distinguish two canonical reduction strategies to reduce words: the leftmost one and the rightmost one, according to the way we apply first the rewriting rule that reduces the leftmost or the rightmost subword. They are defined as follows. For every word u of X^*, the set of rewriting steps with source u can be ordered from left to right by setting $f \prec g$, for rewriting steps $f = v\gamma v'$ and $g = w\beta w'$ with some source u, and such that $|v| < |w|$. If R is finite, then the order $\prec$ is total and the set of rewriting steps of source u is finite. Hence this set contains a smallest element ρ_u and a greatest element η_u, respectively called the *leftmost* and the *rightmost rewriting steps on* u. If, moreover, the rewriting system terminates, the iteration of ρ (resp. η) yields a normalization strategy for R called the *leftmost* (resp. *rightmost*) *normalization strategy of* R:

$$\sigma_u^{\top} = \rho_u \star_1 \sigma_{t(\rho_u)}^{\top} \qquad (\text{resp. } \sigma_u^{\bot} = \eta_u \star_1 \sigma_{t(\eta_u)}^{\bot}). \tag{2.1.5}$$

The *leftmost* (resp. *rightmost*) *rewriting path* on a word u is the rewriting path obtained by applying the leftmost (resp. rightmost) normalization strategy $\sigma_u^{\top}$ (resp. $\sigma_u^{\bot}$). We refer the reader to [19] for more details on rewriting normalization strategies.

2.1.6. Semi-quadratic rewriting systems. A rewriting system R on X is *semi-quadratic* (resp. *quadratic*) if for all γ in R we have $|s(\gamma)| = 2$ and $|t(\gamma)| \leqslant 2$ (resp. $|s(\gamma)| = |t(\gamma)| = 2$). By definition, the sources of the critical branchings of a semi-quadratic rewriting system are of length 3. When R is reduced, there are at most two rewriting paths with respect to R with source a word of length 3. We will denote by $\rho_{l,p}(w)$ (resp. $\sigma_{r,p}(w)$) the word obtained by the rewriting path of length p with source a word w given by the leftmost (resp. rightmost) reduction strategy. Given a word w, we will denote by $\ell_l(w)$ (resp. $\ell_r(w)$) the length of the leftmost (resp. rightmost) rewriting path from w to its normal form.

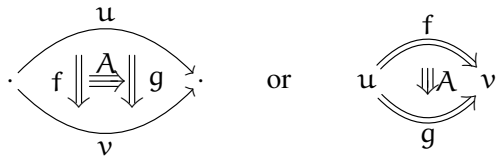
2.1.7. Cross-section property. Given a congruence $\approx$ on the free monoid X^*, we recall that a subset Y of X^* *satisfies the cross-section property* for the quotient monoid $X^*/\approx$ if each equivalence class with respect to $\approx$ contains exactly one element of Y. If R is a convergent rewriting system that presents the quotient monoid $X^*/\approx$, then the set of normal forms for R satisfies the cross-section property for $\approx$.

2.2. Coherent presentations

We recall the notion of coherent presentation of monoids formulated in terms of polygraphs in [16], and we refer the reader to [21] for a deeper presentation.

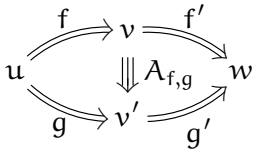
2.2.1. Two-dimensional polygraphs. Rewriting systems can be interpreted as 2-polygraphs with only one 0-cell. Such a 2-*polygraph* P is given by a pair (P_1, P_2), where P_1 is a set and P_2 is a *globular extension* of the free monoid P_1^* seen as a 1-category, that is a set of generating 2-cells $\beta : u \Rightarrow v$ relating 1-cells in P_1^*, with *source* u and *target* v, that we will denote respectively by $s_1(\beta)$ and $t_1(\beta)$. A rewriting system R on an alphabet X can be described by such a 2-polygraph whose generating 1-cells are given by X, and having a generating 2-cell $u \Rightarrow v$ for every rule $u \to v$ in R. Recall that a $(2,1)$-*category* is a category enriched in groupoids. We will denote by $P_2^{\top}$ the $(2,1)$-category freely generated by the 2-polygraph P, see [21, Section 2.4.] for expanded definitions.

2.2.2. Coherent presentations. A pair (f, g) of 2-cells of $P_2^\top$ such that $s_1(f) = s_1(g)$ and $t_1(f) = t_1(g)$ is called a 2-*sphere* of $P_2^\top$. A $(3,1)$-*polygraph* is a pair (P, P_3) made of a 2-polygraph P and a globular extension P_3 of the $(2,1)$-category $P_2^\top$, that is a set of 3-cells $A : f \Rrightarrow g$, where (f, g) is a 2-sphere of $P_2^\top$. The 2-cell f (resp. g) is called the source (resp. target) of A, and denoted by $s_2(A)$ (resp. $t_2(A)$). Such a 3-cell can be represented with the following globular shape:



where $\cdot$ denotes the unique 0-cell of P. We will denote by $P_3^\top$ the free $(3,1)$-category generated by the $(3,1)$-polygraph (P, P_3). An *extended presentation* of a monoid $\mathbf{M}$ is a $(3,1)$-polygraph whose underlying 2-polygraph is a presentation of $\mathbf{M}$. A *coherent presentation of* $\mathbf{M}$ is an extended presentation (P, P_3) of $\mathbf{M}$ such that the cellular extension P_3 of the $(2,1)$-category $P_2^\top$ is acyclic, that is, for every 2-sphere (f, g) of $P_2^\top$, there exists a 3-cell A in the $(3,1)$-category $P_3^\top$ such that $s_2(A) = f$ and $t_2(A) = g$.

2.2.3. Coherence from convergence. Recall that the coherent Squier's theorem from [43, Theorem 5.2], see also [21, Section 4.3], states that, any convergent rewriting system R on X presenting a monoid $\mathbf{M}$ can be extended into a coherent presentation of $\mathbf{M}$ having a generating 3-cell



for every critical branching (f, g) of R, where f' and g' are chosen confluent rewriting paths.

3. String data structures, cross-section and confluence

This section introduces the notions of string data structure and string data bistructure, and presents examples of string data structures for well-known families of monoids. Throughout the article A denotes a totally ordered alphabet. For a natural number $n \geqslant 0$, we will denote the finite set $\{1, \dots, n\}$ with the natural order by $[n]$. A *reading of words on* A is a map $\ell : A^* \to A^*$ sending a word $x_1 \dots x_k$ in A^* on a word $x_{\sigma(1)} \dots x_{\sigma(k)}$ in A^*, where σ is a permutation on $[k]$. The identity on A^* will be called a *left-to-right* reading, denoted by ℓ_l. The *right-to-left* reading is the map, denoted by ℓ_r, that sends a word $x_1 x_2 \dots x_k$ to its *mirror image* $x_k \dots x_2 x_1$.

3.1. String data structures

3.1.1. String data structures. A *string data structure* $\mathbb{S}$ over an alphabet A is a quadruple (D, ℓ, I, R) made of a set D with a distinguished element $\emptyset$, a reading map ℓ of words on A and two maps $R : D \to A^*$ and $I : D \times A \to D$ satisfying the four following conditions:

i) $R(I(\emptyset, x)) = x$ for all x in A,

ii) the relation $I_\ell(\emptyset, -)R = \mathrm{Id}_D$ holds, where $I_\ell : D \times A^* \to D$ is the map defined by

$$I_\ell(d, u) = I_\ell(I(d, x_1), x_2 \dots x_k)$$

for all d in D and u in A^*, with $x_1 \dots x_k = \ell(u)$, and $I_\ell(d, \lambda) = d$ for all $d \in D$,

iii) the map $I_\ell(\emptyset, -) : A^* \to D$ is surjective,

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

One says that R is the *reading map* of $\mathbb{S}$, and that I *inserts* an element of A into an element of D. The map I_ℓ is called the *insertion map* of words in A^* into elements of D with respect to ℓ. The map

$$C_{\mathbb{S}} := I_\ell(\emptyset, -) : A^* \to D$$

is called the *constructor* of $\mathbb{S}$ from words in A^*. We will denote by $\iota_D : A \to D$ the map that sends a letter x in A on the single element data $I(\emptyset, x)$, that we write simply x when no confusion can arise.

A string data structure is called *right* (resp. *left*) if its insertion map is defined with respect to the reading ℓ_l (resp. ℓ_r). For u in A^* and d in D, we will denote $I_{\ell_l}(d, u)$ (resp. $I_{\ell_r}(d, u)$) by $d \leftsquigarrow_{I_{\ell_l}} u$ (resp. $u \rightsquigarrow_{I_{\ell_r}} d$). By definition, the relations

$$(d \leftsquigarrow_{I_{\ell_l}} uv) = (d \leftsquigarrow_{I_{\ell_l}} u) \leftsquigarrow_{I_{\ell_l}} v, \tag{3.1.2}$$

$$(uv \rightsquigarrow_{I_{\ell_r}} d) = u \rightsquigarrow_{I_{\ell_r}} (v \rightsquigarrow_{I_{\ell_r}} d), \tag{3.1.3}$$

hold for all d in D and u, v in A^*.

3.1.4. Associative insertion. Given a string data structure $\mathbb{S} = (D, \ell, I, R)$ we define an internal product $\star_I$ on D by setting

$$d \star_I d' := I_\ell(d, R(d')) \tag{3.1.5}$$

for all d, d' in D. By definition the relations $d \star_I \emptyset = d$ and $\emptyset \star_I d = d$ hold. Hence, the product $\star_I$ is unitary with respect to $\emptyset$. A string data structure $\mathbb{S}$ is called *associative* if the product $\star_I$ is associative. In that case, for a word $w = x_1 x_2 \dots x_k$ in A^*, we write $C_{\mathbb{S}}(w) = x_1 \star_I x_2 \star_I \dots \star_I x_k$.

3.1.6. Structure monoid. The set D with the product $\star_I$ is a monoid called the *structure monoid* of $\mathbb{S}$, and denoted by $\mathbf{M}(D, I)$. We will denote $u =_I v$ the equality of two words u and v in $\mathbf{M}(D, I)$. We say that an associative string data structure *presents* a monoid $\mathbf{M}$ if its structure monoid is isomorphic to $\mathbf{M}$. Two string data structures are said to be *Tietze equivalent* if they present isomorphic monoids. We will denote by $\mathcal{R}(D, \mathbb{S})$ the rewriting system on D, whose rules are

$$\gamma_{d,d'} : d \cdot d' \to d \star_I d' \tag{3.1.7}$$

for all d, d' in D. Every application of a rewriting rule is strictly decreasing in the number of generators, hence $\mathcal{R}(D,\mathbb{S})$ is terminating. Moreover, when $\mathbb{S}$ is associative, it is confluent. It is thus a convergent presentation of the monoid $\mathbf{M}(D,I)$. We will denote by $\mathrm{Nf}(D,\mathbb{S})$ the set of $\mathcal{R}(D,\mathbb{S})$-normal forms.

3.1.8. Proposition. *Let $\mathbb{S} = (D,\ell,I,R)$ be an associative string data structure. The rewriting system $\mathcal{R}(D,\mathbb{S})$ is Tietze equivalent to the rewriting system on D whose rules are*

$$\gamma_{d,\iota_D(x)} : d\cdot\iota_D(x) \to d \star_I \iota_D(x) \tag{3.1.9}$$

for all d in D and x in A.

Proof. Any rule (3.1.9) is a rule of $\mathcal{R}(D,\mathbb{S})$. Conversely, if $\ell(R(d)) = x_1 \dots x_k$, we have $d \star_I d' = d \star_I x_1 \star_I \dots \star_I x_k$, and $d\cdot d' = d\cdot(x_1 \star_I \dots \star_I x_k)$. Moreover, there exist the following rewriting paths with respect to the rules of (3.1.9):

$$d\cdot x_1\cdot x_2\cdot\ldots\cdot x_k \xrightarrow{\gamma_{x_1,x_2}} d\cdot(x_1\star_I x_2)\cdot\ldots\cdot x_k \xrightarrow{\gamma_{x_1\star_I x_2,x_3}} \cdots \xrightarrow{\gamma_{x_1\star_I\dots\star_I x_{k-1},x_k}} d.d'$$

$$d\cdot x_1\cdot x_2\cdot\ldots\cdot x_k \xrightarrow{\gamma_{d,x_1}} (d\star x_1)\cdot x_2\cdot\ldots\cdot x_k \xrightarrow{\gamma_{d\star_I x_1,x_2}} \cdots \xrightarrow{\gamma_{d\star_I x_1\star_I\dots\star_I x_{k-1},x_k}} d\star_I d'$$

Hence, for any rule $\gamma_{d,d'}$ in $\mathcal{R}(D,\mathbb{S})$, with d, d' in D, the source $d\cdot d'$ and the target $d\star_I d'$ are related by a zigzag sequence of rewriting paths with respect to rules (3.1.9). □

3.1.10. Cross-section property. We say that an associative string data structure $\mathbb{S}$ over A satisfies the *cross-section property* for a congruence relation $\approx$ on A^*, if $u \approx v$ holds if and only if $C_{\mathbb{S}}(u) = C_{\mathbb{S}}(v)$ holds for all u, v in A^*. That is, to each equivalence class with respect to $\approx$ it corresponds exactly one element in $\mathrm{Im}(C_{\mathbb{S}})$.

3.1.11. Compatibility with an equivalence relation. A string data structure $\mathbb{S} = (D,\ell,I,R)$ over A is said to be *compatible* with a congruence relation $\approx$ on A^*, if it satisfies the two following conditions:

i) for all $d \in D$ and $u, v \in A^*$, $u \approx v$ implies $I_\ell(d,u) = I_\ell(d,v)$,

ii) $RC_{\mathbb{S}}$ is equivalent to the identity with respect to the congruence $\approx$, that is,

$$RC_{\mathbb{S}}(u) \approx u \qquad \text{for all } u \text{ in } A^*. \tag{3.1.12}$$

Denote by $\mathbf{M}$ the quotient of the free monoid A^* by the congruence $\approx$, and by $\overline{u}$ the image of a word u in A^* by the quotient morphism $\pi : A^* \to \mathbf{M}$. If $\mathbb{S}$ is compatible with the relation $\approx$, then the insertion map I_ℓ induces a unique map $\widetilde{I}_\ell : D \times \mathbf{M} \to D$ such that the following diagram commutes:

$$\begin{array}{ccc} D \times A^* & \xrightarrow{I_\ell} & D \\ {\scriptstyle \mathrm{Id}\times\pi}\downarrow & \nearrow_{\widetilde{I}_\ell} & \\ D \times \mathbf{M} & & \end{array}$$

3.1.13. Theorem. *Let $\mathbb{S}$ be an associative right (resp. left) string data structure over A and let $\approx$ be a congruence relation on A^*. The following conditions are equivalent*

i) $\mathbb{S}$ *satisfies the cross-section property for the congruence relation* $\approx$,

ii) $\mathbb{S}$ *is compatible with the congruence relation* $\approx$,

iii) $\mathbb{S}$ *presents the quotient monoid* $A^*/\approx$ *(resp. the opposite of the quotient monoid* $A^*/\approx$*).*

Proof. We prove the result for a right string data structure $\mathbb{S} = (D, \ell, I, R)$, the proof is similar for a left one. Prove **i)** $\Rightarrow$ **ii)**. For all $u, v \in A^*$, $u \approx v$ if and only if $C_{\mathbb{S}}(u) = C_{\mathbb{S}}(v)$. We have

$$I_\ell(d, u) = d \star_I \iota_D(x_1) \star_I \ldots \star_I \iota_D(x_k) = d \star_I C_{\mathbb{S}}(u) \tag{3.1.14}$$

holds in D, for all d in D and $u = x_1 \ldots x_k$ in A^*. Then, for all $d \in D$ and $u, v \in A^*$, $u \approx v$ implies $I_\ell(d, u) = I_\ell(d, v)$. Moreover, for all $u \in A^*$, we have $C_{\mathbb{S}} R C_{\mathbb{S}}(u) = C_{\mathbb{S}}(u)$. Then $R(C_{\mathbb{S}}(u)) \approx u$, showing (3.1.12). That proves that $\mathbb{S}$ is compatible with the congruence relation $\approx$.

Prove **ii)** $\Rightarrow$ **iii)**. The map $C_{\mathbb{S}} : A^* \to D$ induces a map $\overline{C}_{\mathbb{S}} : A^*/\approx \to D$ defined by $\overline{C}_{\mathbb{S}}(\overline{w}) = \widetilde{I}_\ell(\emptyset, \overline{w})$ for all w in A^*. Let us prove that this map is bijective, whose inverse is the map $\overline{R} := \pi \circ R$. We have $C_{\mathbb{S}}(w) = \overline{C}_{\mathbb{S}}(\overline{w})$ for all w in A^*. Hence $\overline{C}_{\mathbb{S}}\overline{R}(d) = \overline{C}_{\mathbb{S}}(\overline{R(d)}) = C_{\mathbb{S}}(R(d)) = d$ for all d in D. On the other hand, following (3.1.12), we have $\overline{R}\,\overline{C}_{\mathbb{S}}(\overline{w}) = \overline{R}(C_{\mathbb{S}}(w)) = \overline{RC_{\mathbb{S}}(w)} = \overline{w}$ for every $w \in A^*$. This proves that the map $\overline{C}_{\mathbb{S}}$ is bijective. By definition $\overline{C}_{\mathbb{S}}(\lambda) = \emptyset$, let us prove that we have

$$\overline{C}_{\mathbb{S}}(\overline{u}\,\overline{v}) = \overline{C}_{\mathbb{S}}(\overline{u}) \star_I \overline{C}_{\mathbb{S}}(\overline{v}) \tag{3.1.15}$$

for all $\overline{u}, \overline{v}$ in $A^*/\approx$. We have

$$\begin{aligned} \overline{C}_{\mathbb{S}}(\overline{u}) \star_I \overline{C}_{\mathbb{S}}(\overline{v}) &= C_{\mathbb{S}}(u) \star_I C_{\mathbb{S}}(v), \\ &= I_\ell(C_{\mathbb{S}}(u), RC_{\mathbb{S}}(v)), \\ &= \widetilde{I}_\ell(C_{\mathbb{S}}(u), \overline{RC_{\mathbb{S}}(v)}). \end{aligned}$$

From (3.1.12) it follows that $\overline{C}_{\mathbb{S}}(\overline{u}) \star_I \overline{C}_{\mathbb{S}}(\overline{v}) = \widetilde{I}_\ell(C_{\mathbb{S}}(u), \overline{v})$. Moreover, the reading map ℓ being left-to-right, we have $C_{\mathbb{S}}(uv) = (C_{\mathbb{S}}(u) \leftsquigarrow_{I_\ell} v)$. This proves relation (3.1.15).

Prove **iii)** $\Rightarrow$ **i)**. The structure monoids $\mathbf{M}(D, I)$ and the quotient monoid $A^*/\approx$ are isomorphic. That is, $u \approx v$ if and only if $C_{\mathbb{S}}(u) =_I C_{\mathbb{S}}(v)$ for all u, v in A^*. This is our claim. □

3.1.16. Congruence generated by a string data structure. Let $\mathbb{S} = (D, \ell, I, R)$ be an associative string data structure over A. We denote by $\mathcal{R}(R)$ the rewriting system on A, whose rules are defined by

$$\gamma_{d,d'} : R(d)R(d') \to R(d \star_I d') \tag{3.1.17}$$

for all d, d' in D such that $R(d \star_I d') \neq R(d)R(d')$. We will denote by $\approx_{\mathbb{S}}$ the congruence relation on the monoid A^* generated by the rules (3.1.17). The map $\overline{R}$ defined in the proof of Theorem 3.1.13 is a monoid morphism from the structure monoid $\mathbf{M}(D, I)$ to $A^*/\approx_{\mathbb{S}}$. Indeed, for all d, d' in D the equalities $\overline{R}(d \star_I d') = \overline{R}(d)\overline{R}(d')$ and $\overline{R}(\emptyset) = \lambda$ holds in $A^*/\approx_{\mathbb{S}}$. However, note that the map $\overline{R}$ is not in general a morphism of monoids for an arbitrary congruence $\approx$.

3.1.18. Proposition. *For a right (resp. left) associative string data structure* $\mathbb{S} = (D, \ell, I, R)$*, the rewriting system* $\mathcal{R}(R)$ *is a convergent presentation of the monoid* $\mathbf{M}(D, I)$ *(resp. the opposite of the monoid* $\mathbf{M}(D, I)$*).*

Proof. Suppose $\mathbb{S}$ is right, the proof is similar for a left one. The termination of the rewriting system $\mathcal{R}(R)$ is a consequence of the termination of $\mathcal{R}(D,\mathbb{S})$. Indeed, any rewriting sequence with respect to $\mathcal{R}(R)$ gives rise to a rewriting sequence with respect to $\mathcal{R}(D,\mathbb{S})$. Hence if $\mathcal{R}(R)$ has an infinite rewriting path, so does for $\mathcal{R}(D,\mathbb{S})$. As $\mathcal{R}(D,\mathbb{S})$ is terminating this proves that $\mathcal{R}(R)$ is terminating. According to Newman's lemma, [39], we prove confluence from local confluence. It follows from the confluence of critical branchings of $\mathcal{R}(R)$. They have the form:

$$\begin{array}{ccccc} & \xrightarrow{\gamma_{d,d'}R(d'')} & R(d\star_I d')R(d'') & \xrightarrow{\gamma_{d\star_I d',d''}} & R((d\star_I d')\star_I d'')) \\ R(d)R(d')R(d'') & & & & \\ & \xrightarrow[R(d)\gamma_{d',d''}]{} & R(d)R(d'\star_I d'') & \xrightarrow[\gamma_{d,d'\star_I d''}]{} & R(d\star_I (d'\star_I d'')) \end{array}$$

for all d, d', d'' in D such that $R(d)R(d') \neq R(d\star_I d')$ and $R(d')R(d") \neq R(d'\star_I d'')$. These critical branching are confluent by associativity of $\star_I$.

Let us show that $\mathcal{R}(R)$ is a presentation of the monoid $\mathbf{M}(D,I)$. Following Theorem 3.1.13, it suffices to prove that $\mathbb{S}$ is compatible with the congruence relation $\approx_{\mathbb{S}}$. Suppose that $u \approx_{\mathbb{S}} v$, for u, v in A^* and prove that

$$(d \leftsquigarrow_{I_{\ell_1}} u) = (d \leftsquigarrow_{I_{\ell_1}} v)$$

holds for all d in D. The string data structure $\mathbb{S}$ being right, following (3.1.2), we have

$$C_{\mathbb{S}}(R(d)u) = (d \leftsquigarrow_{I_{\ell_1}} u)$$

for all u in A^* and d in D. Every word $w = x_1 \dots x_p$ in A^* can be written $w = R\iota_D(x_1)\dots R\iota_D(x_p)$. The rewriting system $\mathcal{R}(R)$ being convergent, any $\mathcal{R}(R)$-reduction on w ends at the normal form $R(\iota_D(x_1)\star \dots \star_{\mathbb{S}} \iota_D(x_p))$, that is equal to $RC_{\mathbb{S}}(w)$ by associativity of $\mathbb{S}$. Since $u \approx_{\mathbb{S}} v$, we have $R(d)u \approx_{\mathbb{S}} R(d)v$, and by the unique normal form property of $\mathcal{R}(R)$, we obtain $RC_{\mathbb{S}}(R(d)u) = RC_{\mathbb{S}}(R(d)v)$ for all d in D. The map R being injective, we deduce that $C_{\mathbb{S}}(R(d)u) = C_{\mathbb{S}}(R(d)v)$. That proves condition **i)** of 3.1.11. Moreover, since for every word w in A^* its $\mathcal{R}(R)$-normal from is $RC_{\mathbb{S}}(w)$, it follows that $RC_{\mathbb{S}}(w) \approx_{\mathbb{S}} w$, which proves condition **ii)** of 3.1.11. □

3.1.19. Remark. As a consequence of this result, one can prove that an associative string data structure $\mathbb{S}$ satisfies the cross-section property for a congruence relation $\approx$ on A^* by showing that $\mathcal{R}(R)$ is a presentation of the quotient monoid $A^*/\approx$. Indeed, in that case we have $u \approx v$ if and only if $u \approx_{\mathbb{S}} v$ if and only if $C_{\mathbb{S}}(u) = C_{\mathbb{S}}(v)$ for all u, v in A^*.

3.2. Commutation of insertions

3.2.1. Commutation of insertions. A *string data bistructure* over A is a quadruple (D, I, J, R) such that (D, ℓ_l, I, R) (resp. (D, ℓ_r, J, R)) is a right (resp. left) string data structure over A and such that the one-element insertion maps I and J *commute*, that is the following condition

$$y \rightsquigarrow_J (d \leftsquigarrow_I x) \;=\; (y \rightsquigarrow_J d) \leftsquigarrow_I x, \tag{3.2.2}$$

holds for all d in D and x, y in A.

3.2.3. Theorem. *If* (D, I, J, R) *is a string data bistructure over* A*, then the compositions* $\star_I$ *and* $\star_J$ *are associative and the following relation*

$$d \star_I d' = d' \star_J d \tag{3.2.4}$$

holds for all d, d' *in* D*. Moreover, the structure monoids* $\mathbf{M}(D, I)$ *and* $\mathbf{M}(D, J)$ *are anti-isomorphic.*

Proof. Let $\mathbb{S} = (D, \ell_l, I, R)$ and $\mathbb{T} = (D, \ell_r, J, R)$ be the right and left string data structures associated to (D, I, J, R). Let us first show by induction on the length of w that

$$C_{\mathbb{S}}(w) = C_{\mathbb{T}}(w) \tag{3.2.5}$$

holds for all w in A^*. By definition, $C_{\mathbb{S}}(x) = C_{\mathbb{T}}(x)$ holds for all x in A. Suppose that (3.2.5) holds for words of length $n \geqslant 1$ and consider wy a word in A^*, where $w = xv$ with x in A and $|v| = n - 1$. By induction hypothesis, we have $C_{\mathbb{S}}(wy) = I(C_{\mathbb{S}}(w), y) = I(C_{\mathbb{T}}(w), y)$, and by commutation of I and J, we have $I(J(C_{\mathbb{T}}(v), x), y) = J(I(C_{\mathbb{T}}(v), y), x)$. As a consequence, we have

$$I(C_{\mathbb{T}}(w), y) = I(J(C_{\mathbb{T}}(v), x), y) = J(I(C_{\mathbb{S}}(v), y), x) = J(C_{\mathbb{S}}(vy), x).$$

By induction, we deduce that $I(C_{\mathbb{T}}(w), y) = J(C_{\mathbb{T}}(vy), x)$. This proves the equality $C_{\mathbb{S}}(wy) = C_{\mathbb{T}}(wy)$. Having $d \star_I d' = C_{\mathbb{S}}(R(d)R(d')) = C_{\mathbb{T}}(R(d)R(d')) = d' \star_J d$ for all d, d' in D, the commutation relation (3.2.4) is an immediate consequence of relation (3.2.5).

The associativity of $\star_I$ and $\star_J$ is also immediate. Indeed, from (3.2.5) we have $C_{\mathbb{S}}(R(d)R(d')R(d'')) = C_{\mathbb{T}}(R(d)R(d')R(d''))$ for all $d, d', d'' \in D$. But

$$C_{\mathbb{S}}(R(d)R(d')R(d'')) = (d \star_I d') \star_I d'' \quad \text{and} \quad C_{\mathbb{T}}(R(d)R(d')R(d'')) = (d'' \star_J d') \star_J d$$

by definition, hence we have $(d \star_I d') \star_I d'' = (d'' \star_J d') \star_J d$ for all d, d', d'' in D. By commutation of I and J we obtain $(d'' \star_J d') \star_J d = d \star_I (d' \star_I d'')$. Hence, this proves the associativity of the product $\star_I$. The proof of the associativity of $\star_J$ is similar.

Finally, the anti-isomorphism between monoids $\mathbf{M}(D, I)$ and $\mathbf{M}(D, J)$ is a consequence of the fact that the rewriting systems $\mathcal{R}(D, \mathbb{S})$ and $\mathcal{R}(D, \mathbb{T})$ are presentations of these monoids and the commutation of $\star_I$ with $\star_J$. □

As consequence of this result, when (D, I, J, R) is a string data bistructure over A, for all d in D and x in A, we can relate the insertion algorithms from each other using the following relations:

$$(d \leftsquigarrow_I x) = (R(d) \rightsquigarrow_J \iota_D(x)), \tag{3.2.6}$$
$$(x \rightsquigarrow_J d) = (\iota_D(x) \leftsquigarrow_I R(d)). \tag{3.2.7}$$

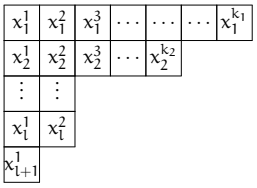
3.3. Example: plactic monoids of classical types

3.3.1. Plactic monoids of type A. Recall that the *plactic monoid* of type A of rank n introduced in [33], denoted by $\mathbf{P}_n$, is presented by the rewriting system on $[n]$ whose rules are the *Knuth relations*, [29]:

$$\begin{aligned} &\xi_{x,y,z} : zxy \to xzy \ \text{ for } \ 1 \leqslant x \leqslant y < z \leqslant n, \\ &\zeta_{x,y,z} : yzx \to yxz \ \text{ for } \ 1 \leqslant x < y \leqslant z \leqslant n. \end{aligned} \tag{3.3.2}$$

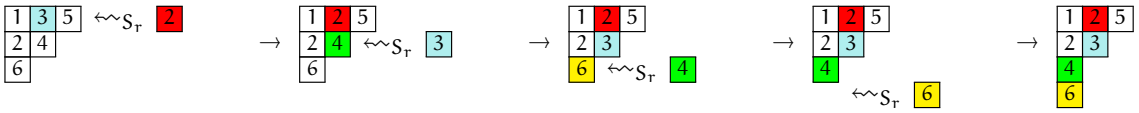
We will denote by $\approx_{\mathbf{P}_n}$ the congruence relation of $[n]^*$ generated by this presentation.

3.3.3. String data bistructures on Young tableaux. Knuth in [29] described the congruence $\approx_{\mathbf{P}_n}$ using the notion of Young tableau. Recall from [45] that a *(Young) tableau* over $[n]$ is a collection of boxes in left-justified rows



filled with elements of $[n]$, where the entries weakly increase along each row, *i.e.* $x_k^i \leqslant x_k^{i+1}$ for all $k, i \geqslant 1$, and strictly increase down each column, *i.e.* $x_k^i < x_{k+1}^i$ for all $k, i \geqslant 1$. A *column* (resp. *row*) over $[n]$ is a tableau such that every row (resp. column) contains exactly one box. Denote by Yt_n (resp. $\mathrm{Col}(n)$) the set of tableaux (resp. columns) over $[n]$.

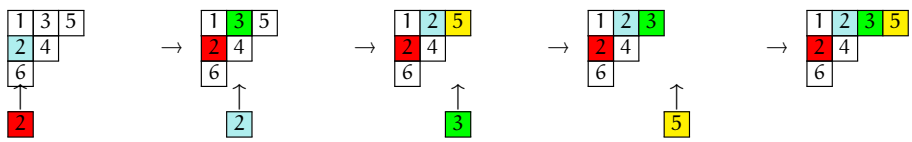
Schensted defined two algorithms to insert an element x of $[n]$ into a tableau t of Yt_n, [42]. The *right (or row) insertion algorithm* S_r computes a tableau $(t \leftsquigarrow_{S_r} x)$ as follows. If x is at least as large as the last element of the top row of t, then put x to the right of this row. Otherwise, let y be the smallest element of the top row of t such that $y > x$. Then x replaces y in this row and y is bumped into the next row where the process is repeated. The algorithm terminates when the element which is bumped is at least as large as the last element of the next row. Then it is placed at the right of that row. For example, the four steps to compute $\left(\begin{smallmatrix}1&3&5\\2&4\\6\end{smallmatrix} \leftsquigarrow_{S_r} 2\right)$ are:



The *left (or column) insertion algorithm* S_l computes a tableau $(x \rightsquigarrow_{S_l} t)$ as follows. If x is larger than the first element of the leftmost column of t, then put x to the bottom of this column. Otherwise, let y be the smallest element of the leftmost column of t such that $y \geqslant x$. Then x replaces y in this column and y is bumped into the next column where the process is repeated. The algorithm terminates when the element which is bumped is greater than all the elements of the next column. Then it is placed at the bottom of that column. Note that the left insertion algorithm can be deduce from the right one by the relation (3.2.7). Indeed, we have:

$$(x \rightsquigarrow_{S_l} t) = (x \leftsquigarrow_{S_r} R(t)).$$

For example, the four steps to compute $\left(2 \rightsquigarrow_{S_l} \begin{smallmatrix}1&3&5\\2&4\\6\end{smallmatrix}\right)$ are:



Denote by $R_{col} : \mathrm{Yt}_n \to [n]^*$ the map that reads tableaux column by column, from left to right and from bottom to top. Schensted's algorithms induce two string data structures on Yt_n: a right

one $\mathbb{Y}^{r}_{n} := (\mathrm{Yt}_n, \ell_l, \mathrm{S}_r, \mathrm{R}_{col})$ and a left one $\mathbb{Y}^{c}_{n} := (\mathrm{Yt}_n, \ell_r, \mathrm{S}_l, \mathrm{R}_{col})$. Note that the insertion S_r with the readings ℓ_r and R_{col} does not induce an associative structure on Yt_n as shown by the following example:

$$\left(\begin{array}{|c|}\hline 1\\\hline 4\\\hline 6\\\hline\end{array} \star_{\mathrm{S}_r} \begin{array}{|c|}\hline 2\\\hline 3\\\hline\end{array}\right) \star_{\mathrm{S}_r} \begin{array}{|c|}\hline 1\\\hline\end{array} = \begin{array}{|c|c|c|}\hline 1&2&3\\\hline 4\\\cline{1-1} 6\\\cline{1-1}\end{array} \star_{\mathrm{S}_r} \begin{array}{|c|}\hline 1\\\hline\end{array} = \begin{array}{|c|c|c|}\hline 1&1&3\\\hline 2\\\cline{1-1} 4\\\cline{1-1} 6\\\cline{1-1}\end{array} \neq \begin{array}{|c|c|c|c|}\hline 1&1&2&3\\\hline 4\\\cline{1-1} 6\\\cline{1-1}\end{array} = \begin{array}{|c|}\hline 1\\\hline 4\\\hline 6\\\hline\end{array} \star_{\mathrm{S}_r} \begin{array}{|c|}\hline 1\\\hline 2\\\hline 3\\\hline\end{array} = \begin{array}{|c|}\hline 1\\\hline 4\\\hline 6\\\hline\end{array} \star_{\mathrm{S}_r} \left(\begin{array}{|c|}\hline 2\\\hline 3\\\hline\end{array} \star_{\mathrm{S}_r} \begin{array}{|c|}\hline 1\\\hline\end{array}\right)$$

Finally, note that we can show that the following equalities hold, see [29, Theorem 5],

$$\begin{aligned} \mathrm{C}_{\mathbb{Y}^{r}_{n}}(zxy) &= \mathrm{C}_{\mathbb{Y}^{r}_{n}}(xzy) \text{ for } 1 \leqslant x \leqslant y < z \leqslant n,\\ \mathrm{C}_{\mathbb{Y}^{r}_{n}}(yzx) &= \mathrm{C}_{\mathbb{Y}^{r}_{n}}(yxz) \text{ for } 1 \leqslant x < y \leqslant z \leqslant n. \end{aligned}$$

More generally, Knuth showed that for any word w, w' in $[n]^*$, $\mathrm{C}_{\mathbb{Y}^{r}_{n}}(w) = \mathrm{C}_{\mathbb{Y}^{r}_{n}}(w')$ holds if and only if $w \approx_{\mathbf{P}_n} w'$ holds, [29, Theorem 6], that is the string data structure $\mathbb{Y}^{r}_{n}$ satisfies the cross-section property for $\approx_{\mathbf{P}_n}$.

3.3.4. Commutation of Schensted's insertions. Schensted showed that S_r and S_l commute, [42, Lemma 6]. Hence we have a string data bistructure $(\mathrm{Yt}_n, \mathrm{S}_r, \mathrm{S}_l, \mathrm{R}_{col})$ over $[n]$. From Theorem 3.2.3, we deduce that the string data structures $\mathbb{Y}^{r}_{n}$ and $\mathbb{Y}^{c}_{n}$ are associative and the structure monoids $\mathbf{M}(\mathbb{Y}^{r}_{n}, \mathrm{S}_r)$ and $\mathbf{M}(\mathbb{Y}^{c}_{n}, \mathrm{S}_l)$ are anti-isomorphic. Note that $\mathbb{Y}^{r}_{n}$ being compatible with $\approx_{\mathbf{P}_n}$, by Theorem 3.1.13 the monoid $\mathbf{M}(\mathbb{Y}^{r}_{n}, \mathrm{S}_r)$ is isomorphic to $\mathbf{P}_n$. Let R^{o}_{col} be the reading map on Yt_n obtained by reading the columns from right to left and from top to bottom. The string data structure $(\mathrm{Yt}_n, \ell_l, \mathrm{S}_l, \mathrm{R}^{o}_{col})$ is compatible with the congruence generated by the following rules

$$\begin{aligned} zxy &\to xzy \text{ for } 1 \leqslant x < y \leqslant z \leqslant n\\ yzx &\to yxz \text{ for } 1 \leqslant x \leqslant y < z \leqslant n, \end{aligned} \tag{3.3.5}$$

as pointed out by Knuth in [29, Section 6]. Note that these relations are used to present the plactic monoid of type A in the theory of crystal graphs, [12, 32, 37].

3.3.6. The plactic monoids of classical types. In Subsection 3.3, we have given a string data bistructure that presents plactic monoids of type A. Using Kashiwara's theory of crystal bases, the plactic congruence of plactic monoids of type A generated by the relations 3.3.5 characterizes the representations of the general Lie algebra $\mathfrak{gl}_n$ of n by n matrices, [12, 32]. We refer the reader to [27] for details on crystal bases theory and to [35–37] for characterizations of representations of Lie algebras by plactic congruences. More generally, since Kashiwara's theory of crystal bases also exists for all classical semisimple Lie algebras, a plactic monoid was introduced for each of these algebras using a case-by-case analysis, [35–37]. To each semisimple Lie algebra it is associated a finite alphabet A indexing a basis of the vector representation of the algebra and a congruence $\approx_{\mathbf{P}_n(A)}$ on the free monoid A^* is defined using the crystal graph of the standard representation. In this way, to each semisimple Lie algebra it corresponds a plactic monoid defined as the quotient of A^* by the congruence $\approx_{\mathbf{P}_n(A)}$. In particular, the plactic monoid of type C, B and D corresponds respectively to the representations of the symplectic Lie algebra, the odd-dimensional orthogonal Lie algebra and the even-dimensional orthogonal Lie algebra. Lecouvey in [35, 36] introduced the notion of admissible columns generalizing the notion of columns in type A , and the notions of symplectic tableaux for type C and orthogonal tableaux for type B and D generalizing the notion of tableaux for type A. He also introduced a Schensted-like left insertion on symplectic tableaux, see [35,

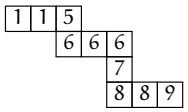
Section 4] and orthogonal tableaux, see [36, Section 3.3]. These insertion algorithms define a left string data structure on the set of symplectic and orthogonal tableaux for type C, B, D. However, the existence of a right insertion algorithm on symplectic and orthogonal tableaux that commutes with Lecouvey's left insertion, and thus a string data bistructure on these tableaux is still an open problem.

3.4. Other examples

3.4.1. The hypoplactic monoid. Recall that the *hypoplactic monoid* of rank n introduced in [30, 40], is the monoid presented by the rewriting system on $[n]$ and whose rules are the Knuth relations (3.3.2), together with the following rules

$$zxty \to xzyt \quad \text{for } 1 \leqslant x \leqslant y < z \leqslant t \leqslant n \quad \text{and} \quad tyzx \to ytxz \quad \text{for } 1 \leqslant x < y \leqslant z < t \leqslant n.$$

The congruence generated by this presentation can be described by using *quasi-ribbon tableaux*, [40], and in terms of Kashiwara's theory of crystal bases, [7]. Recall that a quasi-ribbon tableau over $[n]$ 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 will denote by Qr_n the set of quasi-ribbon tableaux over $[n]$. Let denote by $\mathrm{R_c}$ the reading map on Qr_n obtained by reading the columns from left to right and from bottom to top. For instance, the following diagram



is a quasi-ribbon tableau over $[9]$ and its reading is 1165687689. Novelli proved in [40, Theorem 4.7] that the set Qr_n satisfies the cross-section property for the hypoplactic monoid.

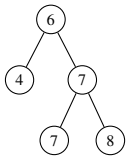
A right insertion algorithm $\mathrm{H^r} : \mathrm{Qr}_n \times [n] \to \mathrm{Qr}_n$ that inserts an element x in $[n]$ into a quasi-ribon tableau t is introduced in [40, Algorithm 4.4] as follows. If x is smaller than each element of t, create a new box filled by x and attach t to the bottom of this box by its topmost and leftmost box. Otherwise, let y be the rightmost and the bottommost element of t that is smaller or equal to x. Create a new box filled by x to the right of the box containing y and attach the other boxes of t situated to the right and below of $\boxed{y}$ onto the bottom of $\boxed{x}$. This algorithm defines a right string data structure $\mathbb{Q}^{\mathrm{r}}_n = (\mathrm{Qr}_n, \ell_{\mathrm{l}}, \mathrm{H^r}, \mathrm{R_c})$ over $[n]$.

A left insertion algorithm $\mathrm{H^l} : \mathrm{Qr}_n \times [n] \to \mathrm{Qr}_n$ that inserts an element x in $[n]$ into a quasi-ribon tableau t is also introduced in [5, Algorithm 4.4] as follows. If x is bigger than each element of t, create a new box filled by x and attach t to the top of this box by its bottommost and rightmost box. Otherwise, let y be the leftmost and the topmost element of t that is bigger or equal to x. Create a new box filled by x to the left of the box containing y and attach the other boxes of t situated to the left and above of $\boxed{y}$ onto the top of $\boxed{x}$. This algorithm defines a left string data structure $\mathbb{Q}^{\mathrm{l}}_n = (\mathrm{Qr}_n, \ell_{\mathrm{r}}, \mathrm{H_l}, \mathrm{R_c})$ over $[n]$. However, the commutation of the right insertion algorithm $\mathrm{H^r}$ and the left insertion algorithm $\mathrm{H^l}$, and thus the existence of a string data bistructure on quasi-ribbon tableaux is still an open problem.

3.4.2. The sylvester monoid. The structure of *sylvester monoid* appeared in the combinatorial study of Loday-Ronco's algebra of planar binary trees related to non-commutative symmetric functions and free symmetric functions, [25]. It is presented by the rewriting system on $[n]$ whose rules are

$$zxwy \to xzwy \qquad \text{for all } 1 \leqslant x \leqslant y < z \leqslant n \text{ and } w \in [n]^*.$$

The sylvester monoid can be constructed using the notion of binary search trees and a Schensted-like left insertion on such trees, [25, Definition 7], and also by using the theory of crystal bases, [8]. Recall that a *(right strict) binary search tree* is a labelled rooted binary tree where the label of each node is greater than or equal to the label of every node in its left sub-tree, and strictly less than every node in its right sub-tree. We will denote by Bt_n the set of binary search trees on $[n]$. Denote by L_l the reading map on Bt_n by recursively performing the right to left postfix reading of the right sub-tree of a tree, then recursively performing the right to left postfix reading of its left sub-tree and finally add the root of the tree. For instance, the following tree



is a binary search tree on $[8]$ and its reading is 78746. Note that the set Bt_n satisfies the cross-section property for the sylvester monoid, [25]. The left insertion algorithm $\mathrm{I}_{\mathrm{Bt}_n}$ introduced in [25, Subsection 3.3] inserts an element x in $[n]$ into a binary search tree t as follows. If t is empty, create a node and label it by x. If t is non-empty, then if x is strictly greater than the label of the root node, then recursively insert x into the right sub-tree of t. Otherwise recursively insert x into its right sub-tree. This algorithm defines a left string data structure $\mathbb{B}_n^l = (\mathrm{Bt}_n, \ell_r, \mathrm{I}_{\mathrm{Bt}_n}, \mathrm{L}_l)$ over $[n]$. The existence of a string data bistructure on Bt_n is still an open problem.

3.4.3. The patience sorting monoids. Recall from [9, Section 3] that the *left (resp. right) patience sorting monoid*, or lPS (resp. rPS) monoid for short, of rank n is the monoid presented by the rewriting system on $[n]$ and whose rules are

$$yx_p \dots x_1 x \to yxx_p \dots x_1 \quad \text{for } x < y \leqslant x_1 < \dots < x_p \quad (\text{resp. } x \leqslant y < x_1 \leqslant \dots \leqslant x_p).$$

Recall that an *lPS (resp. rPS) tableau* over $[n]$ is a collection of boxes in bottom-justified columns, filled with elements of $[n]$, where the entries weakly (resp. strictly) increase along each row from left to right and strictly (resp. weakly) decrease along each column from top to bottom. Denote by Pl_n (resp. Pr_n) the set of lPS (resp. rPS) tableaux over $[n]$, and by R^c the reading map on Pl_n (resp. Pr_n) obtained by reading the columns of an lPS (resp. rPS) tableau from left to right and from top to bottom. For instance, the following tableaux

are respectively an lPS and an rPS tableaux over $[5]$ and their readings are respectively 1421324 and 1422445.

A right insertion algorithm $\mathrm{P}_l^r : \mathrm{Pl}_n \times [n] \to \mathrm{Pl}_n$ (resp. $\mathrm{P}_r^r : \mathrm{Pr}_n \times [n] \to \mathrm{Pr}_n$) that inserts an element x in $[n]$ into an lPS (resp. rPS) tableau t is introduced in [44, Subsection 3.2] as follows. If x is greater or equal (resp. greater) to every element of the bottom row of t, create a box filled by x to the right of this row. Otherwise, let y be the leftmost element of the bottom row of t that is greater than (resp. greater or equal to) x, replace y by x and attach the column containing y to to the top of the box filled by x. This algorithm defines a right string data structure $\mathbb{PL}_n^r = (\mathrm{Pl}_n, \ell_l, \mathrm{P}_l^r, \mathrm{R}^c)$ (resp. $\mathbb{PR}_n^r = (\mathrm{Pr}_n, \ell_l, \mathrm{P}_r^r, \mathrm{R}^c)$) over $[n]$. A left insertion algorithm that inserts an element of $[n]$ into an lPS (resp. rPS) tableau is also

introduced in [9, Algorithm 3.14], yielding a left string data structure over $[n]$. The commutation of this algorithm with the right insertion algorithm P_l^r (resp. P_r^r), and thus the existence of a string data bistructure on these tableaux is still an open problem.

4. Coherent presentations by insertion

In this section we show how to generate a string data structure (D, ℓ, I, R) by a subset Q of D. This allows us to consider an associated rewriting system $\mathcal{R}(Q, \mathbb{S})$ that presents the monoid $\mathbf{M}(D, I)$ with a more economic set of rules than $\mathcal{R}(D, \mathbb{S})$. We explain also how to extend the rewriting system $\mathcal{R}(Q, \mathbb{S})$ into a coherent presentation of $\mathbf{M}(D, I)$, whose generating 3-cells are interpreted in terms of strategy among insertions.

4.1. Generating set of a string data structure

In this subsection $\mathbb{S} = (D, \ell, I, R)$ denotes a right associative string data structure over A. Note that all definitions and results remain valid when $\mathbb{S}$ is a left associative string data structure.

4.1.1. Generating set of a string data structure. A *generating set* for $\mathbb{S}$ is a subset Q of D such that the three following conditions hold:

i) $\iota_D(x) \in Q$ for all x in A,

ii) any element d in D can be decomposed as $d = c_1 \star_I c_2 \star_I \dots \star_I c_k$, where $c_1, \dots, c_k \in Q$,

iii) there exists a unique decomposition $d = c_1 \star_I \dots \star_I c_l$, with $c_1, \dots, c_l$ in Q satisfying the two following conditions:

- $c_i \star_I c_{i+1} \notin Q$ for all $1 \leqslant i \leqslant l-1$,
- $R(c_1 \star_I \dots \star_I c_l) = R(c_1) \dots R(c_l)$ holds in A^*.

We suppose that the empty element $\emptyset$ in D is decomposed into an empty product. The decomposition $c_1 \star_I \dots \star_I c_l$ in **iii)** will be denoted by $[d]_Q$. For example, the set D is a generating set for $\mathbb{S}$ by considering trivial decomposition in conditions **ii)** and **iii)**, with $[d]_D = d$ for all d in D. As an other trivial example, the set $\iota_D(A)$ is a generating set for $\mathbb{S}$. Indeed, following condition **iii)** of 3.1.1, any d in D can be decomposed into a product for $\star_I$ of elements $\iota_D(x)$ with x in A. Moreover, we have $[d]_{\iota_D(A)} = \iota_D(x_1) \star_I \dots \star_I \iota_D(x_l)$ for all d in D with $R(d) = x_1 \dots x_l$. The unicity of the decomposition follows from the injectivity of the reading map R.

4.1.2. Given a generating set Q of $\mathbb{S}$ one defines a string data structure $\mathbb{S}_Q := (D, \ell_Q, I_Q, R_Q)$ over Q by setting

i) ℓ_Q is the left-to-right reading of words on Q,

ii) $I_Q : D \times Q \to D$ is defined by $I_Q(d, c) = d \star_I c$ for all c in Q and d in D, and it induces an insertion map $I_{\ell_Q} : D \times Q^* \to D$ defined by

$$I_{\ell_Q}(d, c_1 \cdot \dots \cdot c_k) = I_{\ell_Q}(d \star_I c_1, c_2 \cdot \dots \cdot c_k)$$

for all d in D and $c_1, \dots, c_k$ in Q, where $\cdot$ denotes the product in Q^*, and $I_{\ell_Q}(d, \lambda) = d$,

iii) $R_Q : D \to Q^*$ is defined by $R_Q(d) = c_1 \cdot c_2 \cdot \ldots \cdot c_k$, for any d in D, with $[d]_Q = c_1 \star_I c_2 \star_I \ldots \star_I c_k$, and $c_1, c_2, \ldots, c_k$ in Q.

4.1.3. Proposition. *The string data structures $\mathbb{S}$ and $\mathbb{S}_Q$ are Tietze equivalent.*

Proof. By definition, the equality $c \star_{I_Q} c' = c \star_I c'$ holds in D for all c, c' in Q. Hence for any word $w = c_1 \cdot c_2 \cdot \ldots \cdot c_k$ in Q^* we have $C_{\mathbb{S}_Q}(w) = c_1 \star_I c_2 \star_I \ldots \star_I c_k$. Now consider d, d' in D. We have $d \star_{I_Q} d' = I_{\ell_Q}(d, R_Q(d'))$. Moreover there exists a unique decomposition $[d']_Q = c'_1 \star_I \ldots \star_I c'_l$ such that $R(d') = R(c'_1) \ldots R(c'_l)$ and $c'_i \star_I c'_{i+1} \notin Q$ for all $1 \leqslant i \leqslant l-1$. Hence

$$I_{\ell_Q}(d, R_Q(d')) = I_{\ell_Q}(d, c'_1 \cdot \ldots \cdot c'_l) = I_\ell(d, R(d')) = d \star_I d'.$$

Hence the compositions $\star_I$ and $\star_{I_Q}$ coincide on D, that proves the Tietze equivalence of $\mathbb{S}$ and $\mathbb{S}_Q$. □

4.1.4. Given a generating set Q of $\mathbb{S}$, we denote by $\mathcal{R}(Q, \mathbb{S})$ the rewriting system on Q whose rules are

$$\gamma_{c,c'} : c \cdot c' \to R_Q(c \star_I c') \tag{4.1.5}$$

for all c, c' in Q, whenever $c \cdot c' \neq R_Q(c \star_I c')$, and where $\cdot$ denotes the product in the free monoid on Q. We will denote by $\mathrm{Nf}(Q, \mathbb{S})$ the set of $\mathcal{R}(Q, \mathbb{S})$-normal forms. Note that when $Q = D$, we recover the rewriting system $\mathcal{R}(D, \mathbb{S})$ defined in (3.1.7) and that presents the structure monoid $\mathbf{M}(D, I)$.

4.1.6. Well-founded generating set. A generating set Q of $\mathbb{S}$ is called *well-founded* (resp. *quadratic*) if the rewriting system $\mathcal{R}(Q, \mathbb{S})$ is terminating (resp. quadratic). When Q is well-founded, we denote by $\sigma^{\top,Q}$ (resp. $\sigma^{\bot,Q}$) the leftmost (resp. rightmost) reduction strategy on $\mathcal{R}(Q, \mathbb{S})$. Given d in D and c in Q, by associativity of $\star_I$, the rewriting path $\sigma^{\top,Q}_{R_Q(d)\cdot c}$ reduces $R_Q(d) \cdot c$ to $R_Q(d \star_I c)$. More generally, the strategy $\sigma^{\top,Q}$ reduces any word w in Q^* to $R_Q C_{\mathbb{S}_Q}(w)$, that is, it defines a rewriting path

$$\sigma^{\top,Q}_w : w \to^* R_Q C_{\mathbb{S}_Q}(w)$$

for all w in Q^*. Note that, the rewriting system $\mathcal{R}(D, \mathbb{S})$ being convergent, any normalization strategy σ on $\mathcal{R}(D, \mathbb{S})$ reduces any word w in Q^* to $R_Q C_{\mathbb{S}_Q}(w)$.

4.1.7. Termination of $\mathcal{R}(Q, \mathbb{S})$. In most applications, the termination of $\mathcal{R}(Q, \mathbb{S})$ can be showed by introducing a well-founded order on the free monoid Q^* defined as follows. Given two well-founded ordered sets $(X_1, \leqslant)$ and $(X_2, \preceq)$, and two maps $g : Q \to X_1$ and $f : Q^* \to X_2$, one defines a lexicographic order $\prec_{f,g}$ on Q^* by setting

$$u \prec_{f,g} v \quad \text{if and only if} \quad \big(f(u) < f(v)\big) \ \text{ or } \ \big(f(u) = f(v) \text{ and } g(c_1) \prec g(c'_1)\big)$$

for all $u = c_1 \cdot \ldots \cdot c_k$ and $v = c'_1 \cdot \ldots \cdot c'_l$ in Q^*. The order $\prec_{f,g}$ is well-founded, and we can prove the termination of the rewriting system $\mathcal{R}(Q, \mathbb{S})$ by using such an order compatible with rules (4.1.5), that is the inequalities $f(R_Q(c \star_I c')) \leqslant f(c \cdot c')$ and $g(c_1) \prec g(c)$ hold, where c_1 is the first element in the decomposition of $R_Q(c \star_I c')$ in Q^*. Then a reduction with respect to $\mathcal{R}(Q, \mathbb{S})$ must decrease a word in Q^* either with respect to f or with respect to g. In particular, this method is used to prove the termination of the column presentation for the plactic monoids of type C in [6, 23], and for other classical types A, B and D in [6], by introducing a well-founded order on the set of column generators corresponding to each type

and where the map f counts the number of columns and g is the length of each column. Note that for the plactic monoid of type G_2, the termination of the column presentation cannot be proved by using the lexicographic order of the form $\prec_{f,g}$ since the Lecouvey insertion of one column into another one can produce a tableau with three columns as shown in [6].

4.1.8. Proposition. *Let $\mathbb{S} = (D, \ell, I, R)$ be a right associative string data structure, and let Q be a well-founded generating set of $\mathbb{S}$. Then $\mathcal{R}(Q, \mathbb{S})$ is a convergent presentation of the structure monoid $\mathbf{M}(D, I)$, and the set $\mathrm{Nf}(Q, \mathbb{S})$ satisfies the cross-section property for the structure monoid $\mathbf{M}(D, I)$.*

Proof. Prove that $\mathcal{R}(Q, \mathbb{S})$ is confluent. Any critical pair of $\mathcal{R}(Q, \mathbb{S})$ has the form $(\gamma_{c,c'} \cdot c'', c \cdot \gamma_{c',c''})$, for c, c', c'' in Q. By 4.1.6, the target of the rewriting path $\sigma^{\top,Q}_{R_Q(c \star_I c') \cdot c''}$ is $R_Q C_{\mathbb{S}_Q}(R_Q(c \star_I c') \cdot c'')$. Suppose $R_Q(c \star_I c') = c_1 \cdot \ldots \cdot c_k$, with $c_1, \ldots, c_k$ in Q. The map ℓ_Q being a left-to-right reading, we have

$$C_{\mathbb{S}_Q}(R_Q(c \star_I c') \cdot c'') = I_{\ell_Q}(c_1 \star_I c_2 \star_I \ldots \star_I c_k, c'').$$

Moreover, the equality $c_1 \star_I c_2 \star_I \ldots \star_I c_k = c \star_I c'$ holds in D. Hence $C_{\mathbb{S}_Q}(R_Q(c \star_I c') \cdot c'') = (c \star_I c') \star_I c''$. Similarly, one shows that the target of $\sigma^{\top,Q}_{c \cdot R_Q(c' \star_I c'')}$ is $R_Q\big(c \star_I (c' \star_I c'')\big)$. Then any critical pair of $\mathcal{R}(Q, \mathbb{S})$ has the following reduction diagram:

$$\begin{array}{ccccc} & \xrightarrow{\gamma_{c,c'} \cdot c''} & R_Q(c \star_I c') \cdot c'' & \xrightarrow{\sigma^{\top,Q}_{R_Q(c \star_I c') \cdot c''}} & R_Q\big((c \star_I c') \star_I c''\big) \\ c \cdot c' \cdot c'' & & & & \\ & \xrightarrow{c \cdot \gamma_{c',c''}} & c \cdot R_Q(c' \star_I c'') & \xrightarrow[\sigma^{\top,Q}_{c \cdot R_Q(c' \star_I c'')}]{} & R_Q\big(c \star_I (c' \star_I c'')\big) \end{array}$$

which is confluent by the associativity of the product $\star_I$. This proves that the rewriting system $\mathcal{R}(Q, \mathbb{S})$ is locally confluent and thus confluent by termination hypothesis.

Prove that $\mathbb{S}_Q$ is compatible with the congruence relation $\approx_{\mathcal{R}(Q,\mathbb{S})}$. Consider a word w in Q^*. The rewriting system $\mathcal{R}(Q, \mathbb{S})$ being terminating, the reduction strategy $\sigma^{\top,Q}$ reduces w to $R_Q C_{\mathbb{S}_Q}(w)$ which proves that $R_Q C_{\mathbb{S}_Q}(w) \approx_{\mathcal{R}(Q,\mathbb{S})} w$, showing condition **ii)** of 3.1.11. Suppose now that $u \approx_{\mathcal{R}(Q,\mathbb{S})} v$, for u, v in Q^* and prove that $I_{\ell_Q}(d, u) = I_{\ell_Q}(d, v)$ holds for all d in D. The string data structure $\mathbb{S}_Q$ being right associative, we have $I_{\ell_Q}(d, u) = d \star_{I_Q} C_{\mathbb{S}_Q}(u)$, for all $u \in A^*$ and $d \in D$. Since $u \approx_{\mathcal{R}(Q,\mathbb{S})} v$, by the unique normal form property of $\mathcal{R}(Q, \mathbb{S})$, the equality $R_Q C_{\mathbb{S}_Q}(u) = R_Q C_{\mathbb{S}_Q}(v)$ holds. The map R_Q being injective, we obtain that $C_{\mathbb{S}_Q}(u) = C_{\mathbb{S}_Q}(v)$. We deduce that $d \star_{I_Q} C_{\mathbb{S}_Q}(u) = d \star_{I_Q} C_{\mathbb{S}_Q}(v)$, for all d in D. That proves condition **i)** of 3.1.11. Then by Theorem 3.1.13 $\mathbb{S}_Q$ presents the quotient monoid $Q^*/\approx_{\mathcal{R}(Q,\mathbb{S})}$. Hence, by Proposition 4.1.3, the rewriting system $\mathcal{R}(Q, \mathbb{S})$ is a presentation of the structure monoid $\mathbf{M}(D, I)$.

The fact that $\mathrm{Nf}(Q, \mathbb{S})$ satisfies the cross-section property for the monoid $\mathbf{M}(D, I)$ is an immediate consequence of the confluence of $\mathcal{R}(Q, \mathbb{S})$ as explained in 3.1.10. □

As a consequence of Proposition 4.1.8, when the generating set Q is well-founded, the rewriting systems $\mathcal{R}(R)$ and $\mathcal{R}(Q, \mathbb{S})$ are Tietze-equivalent. Indeed, by this result the rewriting systems $\mathcal{R}(Q, \mathbb{S})$ is Tietze-equivalent to $\mathcal{R}(D, \mathbb{S})$, that is Tietze-equivalent to $\mathcal{R}(R)$ by Proposition 3.1.18.

4.1.9. Corollary. *Let $\mathbb{S} = (D, \ell, I, R)$ be a right associative string data structure.*

i) *If $\mathbb{S}$ has a finite well-founded generating set Q, then the structure monoid $\mathbf{M}(D, I)$ has finite derivation type and thus finite homological type.*

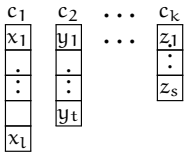
ii) *If $\mathbb{S}$ has a quadratic well-founded generating set Q, then the structure monoid $\mathbf{M}(D, I)$ is Koszul.*

Proof. In [43] the authors showed that if a monoid admits a finite convergent presentation, then it is of finite derivation type. Moreover, the property finite derivation type implies the property finite homological type. If $\mathbb{S}$ has a finite well-founded generating set Q, then by Proposition 4.1.8 the monoid $\mathbf{M}(D, I)$ admits $\mathcal{R}(Q, \mathbb{S})$ as a finite convergent presentation, and thus it has finite derivation type.

The assertion **ii)** is a consequence of the fact that a monoid having a quadratic convergent presentation is Koszul, see [1, 18]. If $\mathbb{S}$ has a quadratic well-founded generating set Q, by Proposition 4.1.8 the rewriting system $\mathcal{R}(Q, \mathbb{S})$ is a quadratic convergent presentation of the structure monoid $\mathbf{M}(D, I)$, and thus $\mathbf{M}(D, I)$ is Koszul. □

For instance, the plactic monoid of type A has finite derivation type, but it is not Koszul, [13]. As a consequence, there is no quadratic well-founded generating set for Young structures of type A.

4.1.10. Example: column presentation of plactic monoids of type A. As an illustration, we prove that the set of columns $\mathrm{Col}(n)$ defined in Example 3.3 is a generating set for the associative string data structure $\mathbb{Y}^r_n$. For all x in $[n]$, $C_{\mathbb{S}}(x)$ is a Young tableau with only one box filled by x, hence a column in $\mathrm{Col}(n)$. Every d in Yt_n can be uniquely decomposed into a sequence $(c_1, \ldots, c_k)$ of columns in $\mathrm{Col}(n)$:



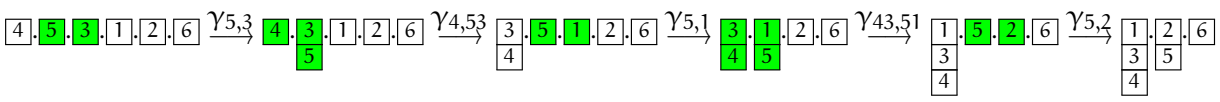
where $c_1, \ldots, c_k$ are the columns of d from left to right. By definition of the tableau, we have $d = c_1 \star_{S_r} \ldots \star_{S_r} c_k$, and $c_i \star_{S_r} c_{i+1} \notin \mathrm{Col}(n)$ for all $1 \leqslant i \leqslant k-1$. Moreover, by definition of the reading map, the equality $R_{col}(d) = R_{col}(c_1) \ldots R_{col}(c_k)$ holds in $[n]^*$.

The rules of the rewriting system $\mathcal{R}(\mathrm{Col}(n), \mathbb{Y}^r_n)$ are of the form

$$\gamma_{c,c'} : c \cdot c' \to R_{\mathrm{Col}(n)}(c \star_{S_r} c')$$

for all c, c' in $\mathrm{Col}(n)$ such that $c \cdot c' \neq R_{\mathrm{Col}(n)}(c \star_{S_r} c')$, where the reading map $R_{\mathrm{Col}(n)} : \mathrm{Yt}_n \to \mathrm{Col}(n)^*$ sends a tableau to the product of its columns from left to right. By using a lexicographic order as defined in 4.1.7, one shows that the rewriting system $\mathcal{R}(\mathrm{Col}(n), \mathbb{Y}^r_n)$ is terminating. Following Proposition 4.1.8 the rewriting system $\mathcal{R}(\mathrm{Col}(n), \mathbb{Y}^r_n)$ is convergent and Tietze-equivalent to $\mathcal{R}(\mathrm{Yt}_n, \mathbb{Y}^r_n)$.

Note that Schensted's insertion S_r corresponds to the application of the leftmost normalisation strategy $\sigma^{\top, \mathrm{Col}(n)}$. For instance, consider the word 453126 in $[6]^*$. To compute the tableau $C_{\mathbb{Y}^r_6}(453126)$, one applies the following successive rules of $\mathcal{R}(\mathrm{Col}_6, \mathbb{Y}^r_6)$:



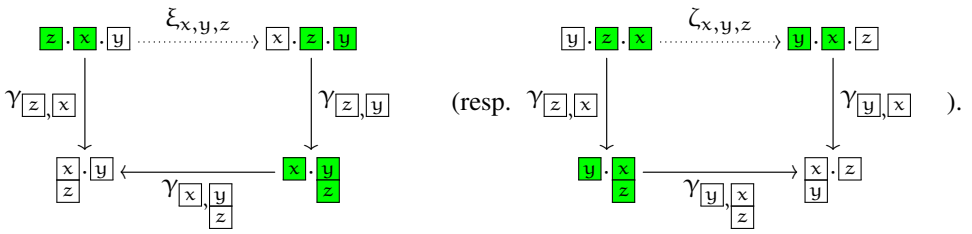
producing $C_{\mathbb{Y}^c_6}(453126) = \begin{array}{|c|c|c|}\hline 1 & 2 & 6 \\ \hline 3 & 5 \\ \cline{1-2} 4 \\ \cline{1-1}\end{array}$.

Moreover, the readings of the source and the target of any rule of $\mathcal{R}(\mathrm{Col}(n), \mathbb{Y}^r_n)$ are related by Knuth's relations (3.3.2), that is $\mathrm{R_{col}}(c)\mathrm{R_{col}}(c') \approx_{\mathbf{P}_n} \mathrm{R_{col}}(c \star_{S_r} c')$, for all c, c' in $\mathrm{Col}(n)$ such that $c \cdot c' \neq \mathrm{R}_{\mathrm{Col}(n)}(c \star_{S_r} c')$. Indeed, it is sufficient to show that

$$\mathrm{R_{col}}(d \star_{S_r} \iota_{\mathbb{Y}^r_n}(x)) \approx_{\mathbf{P}_n} \mathrm{R_{col}}(d)x \tag{4.1.11}$$

for all d in Yt_n and x in $[n]$. By definition of Schensted's insertion, the process that occurs on the first row of a tableau, is repeated on the next rows of the same tableau. Then, it is sufficient to show the equivalence (4.1.11) in the case where d is a row on $[n]$. Since $\mathrm{R_{col}}(u)$ and $\mathrm{R_{col}}(v)$ are strictly decreasing words, the rows of the tableau $u \star_{S_r} v$ are of length at most 2. Then it is also sufficient to show the equivalence (4.1.11) in the case where d is a row of length at most 2. Suppose that $\mathrm{R_{col}}(d) = x_1x_2$ with $x_1 \leqslant x_2$ and let x be in $[n]$ such that $x_2 > x$. There are two cases: $x_1 \leqslant x < x_2$ or $x < x_1 \leqslant x_2$. In the first case, $\mathrm{R_{col}}(d)x = x_1x_2x \approx_{\mathbf{P}_n} x_2x_1x$ by applying ξ_{x_1,x,x_2}, and $\mathrm{R_{col}}(d \star_{S_r} \iota_{\mathbb{Y}^r_n}(x)) = x_2x_1x$. In the second case, $\mathrm{R_{col}}(d)x = x_1x_2x \approx_{\mathbf{P}_n} x_1xx_2$ by applying ζ_{x,x_1,x_2}, and $\mathrm{R_{col}}(d \star_{S_r} \iota_{\mathbb{Y}^r_n}(x)) = x_1xx_2$. Then, in the two cases, we obtain $\mathrm{R_{col}}(d)x \approx_{\mathbf{P}_n} \mathrm{R_{col}}(d \star_{S_r} \iota_{\mathbb{Y}^r_n}(x))$.

Finally, let us show that the string data structure $\mathbb{Y}^r_n$ presents the plactic monoid $\mathbf{P}_n$ by using the properties of the rewriting system $\mathcal{R}(\mathrm{Col}(n), \mathbb{Y}^r_n)$, recovering then Knuth's Theorem, [29, Theorem 6]. Following Theorem 3.1.13, it suffices to prove that $\mathbb{Y}^r_n$ is compatible with the plactic congruence $\approx_{\mathbf{P}_n}$. Suppose that $u \approx_{\mathbf{P}_n} v$, for u, v in A^* and prove that $(d \leftsquigarrow_{S_r} u) = (d \leftsquigarrow_{S_r} v)$ holds for all d in D. The string data structure $\mathbb{Y}^r_n$ being right, following (3.1.14) we have $(d \leftsquigarrow_{I_{S_r}} u) = d \star_{S_r} \mathrm{C}_{\mathbb{Y}^r_n}(u)$, for all $u \in A^*$ and $d \in D$. Moreover, for any $1 \leqslant x \leqslant y < z \leqslant n$ (resp. $1 \leqslant x < y \leqslant z \leqslant n$), the rules $\xi_{x,y,z}$ (resp. $\zeta_{x,y,z}$) can be decomposed by rules in $\mathcal{R}(\mathrm{Col}(n), \mathbb{Y}^r_n)$ as follows:



Then, since $u \approx_{\mathbf{P}_n} v$, the words u and v are also related by the rules of $\mathcal{R}(\mathrm{Col}(n), \mathbb{Y}^r_n)$, and by the unique normal form property of $\mathcal{R}(\mathrm{Col}(n), \mathbb{Y}^r_n)$, we obtain $\mathrm{C}_{\mathbb{Y}^r_n}(u) = \mathrm{C}_{\mathbb{Y}^r_n}(v)$. We deduce that $d \star_{\mathbb{Y}^r_n} \mathrm{C}_{\mathbb{Y}^r_n}(u) = d \star_{\mathbb{Y}^r_n} \mathrm{C}_{\mathbb{Y}^r_n}(v)$, for all d in D. That proves condition **i)** of 3.1.11. Now consider a word $w = x_1 \dots x_p$ in A^*. To compute the tableau $\mathrm{C}_{\mathbb{Y}^r_n}(w)$, one applies the leftmost normalisation strategy $\sigma^{\top,\mathrm{Col}(n)}$ on w. Then $\mathrm{C}_{\mathbb{Y}^r_n}(w)$ and w are related by the rules of $\mathcal{R}(\mathrm{Col}(n), \mathbb{Y}^r_n)$. The readings of the source and the target of any rule of $\mathcal{R}(\mathrm{Col}(n), \mathbb{Y}^r_n)$ being related by Knuth's relations (3.3.2), it follows that $\mathrm{R_{col}}\mathrm{C}_{\mathbb{Y}^r_n}(w) \approx_{\mathbf{P}_n} w$, which proves condition **ii)** of 3.1.11.

As a consequence, we obtain that the rewriting system $\mathcal{R}(\mathrm{Col}(n), \mathbb{Y}^r_n)$ is a finite convergent presentation of the monoid $\mathbf{P}_n$. By this way, we recover the results of [4, Theorem 3.4] and [2, Theorem 4.5].

4.1.12. Example: row presentation of plactic monoids of type A. Let consider the string data structure $\mathbb{Y}^{\mathrm{Row}}_n = (\mathrm{Yt}_n, \ell_1, S_r, \mathrm{R}_{row})$, where S_r is Schensted's insertion recalled in 3.3.3, and R_{row} is the reading map on Yt_n that reads a tableau row by row, from left to right and from bottom to top. We denote by $\mathrm{Row}(n)$ the set of rows on $[n]$. The set $\mathrm{Row}(n)$ forms a generating set for $\mathbb{Y}^{\mathrm{Row}}_n$. Indeed, for all x in $[n]$ the tableau $\mathrm{C}_{\mathbb{Y}^{\mathrm{Row}}_n}(x)$ belongs to $\mathrm{Row}(n)$. Every tableau d in Yt_n can be uniquely

decomposed as $d = r_1 \star_{S_r} \dots \star_{S_r} r_k$, where $r_1, \dots, r_k$ are the rows of d from bottom to top. By definition of the tableau, $r_i \star_{S_r} r_{i+1} \notin \mathrm{Row}(n)$ for all $1 \leqslant i \leqslant k-1$, and by definition of the reading map, the equality $R_{row}(d) = R_{row}(r_1) \dots R_{row}(r_k)$ holds in $[n]^*$.

The rules of the rewriting system $\mathcal{R}(\mathrm{Row}(n), \mathbb{Y}_n^{Row})$ are of the form

$$\gamma_{r,r'} : r \cdot r' \to R_{\mathrm{Row}(n)}(r \star_{S_r} r')$$

for all r, r' in $\mathrm{Row}(n)$ such that $r \cdot r' \neq R_{\mathrm{Row}(n)}(r \star_{S_r} r')$, and where $R_{\mathrm{Row}(n)} : \mathrm{Yt}_n \to \mathrm{Row}(n)^*$ is the reading map sending a tableau to the product of its rows from bottom to top. Using the arguments of 4.1.10, one proves that the string data structure $\mathbb{Y}_n^{Row}$ presents the plactic monoid $\mathbf{P}_n$. Using a lexicographic order as defined in 4.1.7 one proves that $\mathcal{R}(\mathrm{Row}(n), \mathbb{Y}_n^{Row})$ is terminating. Then by Proposition 4.1.8 the rewriting system $\mathcal{R}(\mathrm{Row}(n), \mathbb{Y}_n^{Row})$ is a convergent presentation of the monoid $\mathbf{P}_n$, that is infinite contrary to the column presentation that is finite. By this way, we recover the result of [2, Theorem 3.2].

4.2. Coherent presentations and string data structures

4.2.1. Theorem. *Let $\mathbb{S}$ be a right associative string data structure, and let Q be a well-founded generating set of $\mathbb{S}$. Then the rewriting system $\mathcal{R}(Q, \mathbb{S})$ extends into a coherent convergent presentation of the structure monoid $\mathbf{M}(D, I)$ by adjunction of a generating 3-cell*

$$\begin{array}{ccc} & \overset{\sigma^{\top,Q}_{c\cdot c'\cdot c''}}{\Rightarrow} & \\ c\cdot c'\cdot c'' & \Downarrow A_{c,c',c''} & R_Q(c \star_I c' \star_I c'') \\ c\cdot\gamma_{c',c''}\Rightarrow & c\cdot R_Q(c' \star_I c'') & \overset{\sigma^{\top,Q}_{c\cdot R_Q(c'\star_I c'')}}{\Rightarrow} \end{array} \tag{4.2.2}$$

for any c, c', c'' in Q such that $c\cdot c' \neq R_Q(c \star_I c')$ and $c'\cdot c'' \neq R_Q(c' \star_I c'')$.

Proof. Any critical branching of $\mathcal{R}(Q, \mathbb{S})$ has the form

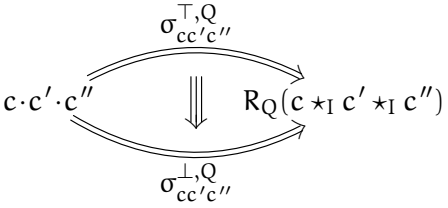
$$\begin{array}{ccc} & \overset{\gamma_{c,c'}\cdot c''}{\Rightarrow} & R_Q(c \star_I c')\cdot c'' \\ c\cdot c'\cdot c'' & & \\ & c\cdot\gamma_{c',c''}\Rightarrow & c\cdot R_Q(c' \star_I c'') \end{array}$$

with c, c', c'' in Q such that $c\cdot c' \neq R_Q(c \star_I c')$ and $c'\cdot c'' \neq R_Q(c' \star_I c'')$. By Proposition 4.1.8, $\mathcal{R}(Q, \mathbb{S})$ is confluent, hence such a critical branching is confluent with a confluence diagram as in (4.2.2). We conclude with coherent Squier's theorem recalled in 2.2.3. □

4.2.3. Coherent presentations and insertion. Let (D, I, J, R) be a string data bistructure over A and let $\mathbb{S}$ (resp. $\mathbb{T}$) be the corresponding right (resp. left) string data structure. Given a well-founded generating set Q of $\mathbb{S}$, we consider the rewriting system on Q, whose rules are

$$c\cdot c' \to R_Q(c' \star_J c)$$

for any c, c' in Q such that $c \cdot c' \neq R_Q(c' \star_J c)$. By definition, we have $R_Q(c' \star_J c) = R_Q(c \star_I c')$ for all c, c' in Q, thus it coincides with the rewriting systems $\mathcal{R}(Q, \mathbb{S})$. When $\mathcal{R}(Q, \mathbb{S})$ is convergent, by Theorem 4.2.1 it can be extended into a coherent convergent presentation of the monoid $\mathbf{M}(D, I)$ by adjunction of a generating 3-cell



for any c, c', c'' in Q such that $c \cdot c' \neq R_Q(c \star_I c')$ and $c' \cdot c'' \neq R_Q(c' \star_I c'')$. In this way, the application of the leftmost (resp. rightmost) normalisation strategy $\sigma^{\top,Q}$ (resp. $\sigma^{\bot,Q}$) on the word $c \cdot c' \cdot c''$ corresponds to the application of the right (resp. left) insertion

$$\emptyset \leftsquigarrow_I R(c)R(c')R(c'') \qquad \big(\text{resp. } R(c)R(c')R(c'') \rightsquigarrow_J \emptyset\big).$$

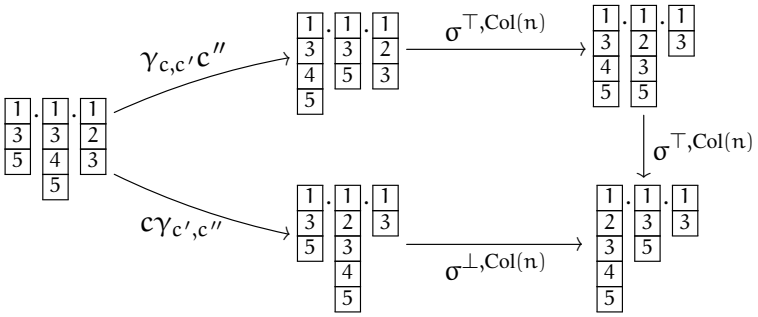
4.2.4. Example. As an illustration, consider the string data bistructure $(\mathrm{Yt}_n, S_r, S_l, R_{col})$ and the convergent presentation $\mathcal{R}(\mathrm{Col}(n), \mathbb{Y}^r_n)$ of the plactic monoid $\mathbf{P}_n$ from Example 4.1.10. Let

$$c = \begin{array}{|c|}\hline 1 \\ \hline 3 \\ \hline 5 \\ \hline \end{array}, \qquad c' = \begin{array}{|c|}\hline 1 \\ \hline 3 \\ \hline 4 \\ \hline 5 \\ \hline \end{array}, \qquad c'' = \begin{array}{|c|}\hline 1 \\ \hline 2 \\ \hline 3 \\ \hline \end{array}$$

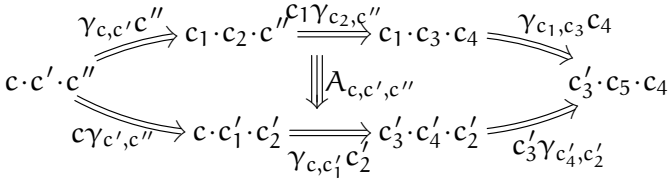
be columns in $\mathrm{Col}(n)$. We have

$$(\,\emptyset \leftsquigarrow_{S_r} R_{Col}(c)R_{Col}(c')R_{Col}(c'')\,) = \begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 2 & 3 & 3 \\ \hline 3 & 5 \\ \cline{1-2} 4 \\ \cline{1-1} 5 \\ \cline{1-1} \end{array} = (\,R_{Col}(c)R_{Col}(c')R_{Col}(c'') \rightsquigarrow_{S_l} \emptyset\,).$$

Moreover, the normalisation strategy $\sigma^{\top,\mathrm{Col}(n)}$ reduces $c \cdot c' \cdot c''$ into $R_{\mathrm{Col}(n)}(C_{\mathbb{Y}^r_n}(R_{Col}(c)R_{Col}(c)R_{Col}(c)))$ and the normalisation strategy $\sigma^{\bot,\mathrm{Col}(n)}$ reduces $c \cdot c' \cdot c''$ into $R_{\mathrm{Col}(n)}(C_{\mathbb{Y}^c_n}(R_{Col}(c)R_{Col}(c)R_{Col}(c)))$, as shown in the following diagram:



4.2.5. Example: coherent column presentations of plactic monoids of type A. By definition of Schensted's algorithms, the leftmost and the rightmost normalization strategy with respect to $\mathcal{R}(\mathrm{Col}_n, \mathbb{Y}^r_n)$ on the sources of its critical branchings, lead to the normal form, after applying three steps of reductions rules. Then, by Theorem 4.2.1 the rewriting system $\mathcal{R}(\mathrm{Col}_n, \mathbb{Y}^r_n)$ can be extended into a coherent convergent presentation by adjunction of the following generating 3-cells:



for all c, c', c'' in $\mathrm{Col}(n)$ such that $c\cdot c' \neq R_{\mathrm{Col}(n)}(c \star_{S_r} c')$, and $c'\cdot c'' \neq R_{\mathrm{Col}(n)}(c' \star_{S_r} c'')$, and where $R_{\mathrm{Col}_n}(c \star_{S_r} c') = c_1\cdot c_2$, $R_{\mathrm{Col}_n}(c_2 \star_{S_r} c'') = c_3\cdot c_4$, $R_{\mathrm{Col}_n}(c_1 \star_{S_r} c_3) = c'_3\cdot c_5$, $R_{\mathrm{Col}_n}(c'' \star_{S_l} c') = c'_1\cdot c'_2$, $R_{\mathrm{Col}_n}(c'_1 \star_{S_l} c) = c'_3\cdot c'_4$, and $R_{\mathrm{Col}_n}(c'_2 \star_{S_l} c'_4) = c_5\cdot c_4$. By this way, we recover the result in [24, Theorem 1].

4.2.6. Remark. In previous example, the shape of the generating 3-cell $A_{c,c',c''}$ can be deduced from the Schützenberger involution, as shown in [24, Remark 3.2.7]. More generally, for a well-founded generating set Q of $\mathbb{S}$, such an involution transforms the leftmost reduction strategy $\sigma^{\top,Q}$ of $\mathcal{R}(Q,\mathbb{S})$ into the rightmost reduction strategy $\sigma^{\perp,Q}$, and conversely. We call *involution on* $\mathbb{S}$ *with respect to* Q a map $^\star : Q \to Q$, extended into a map $^\star : Q^* \to Q^*$ by setting $(c_1\cdot\ldots\cdot c_k)^\star = c_k^\star\cdot\ldots\cdot c_1^\star$ for all $c_1, \ldots, c_k \in Q$, and satisfying the two following conditions:

i) for $u, v \in Q^*$, if $u \approx_{\mathcal{R}(Q,\mathbb{S})} v$ then $u^\star \approx_{\mathcal{R}(Q,\mathbb{S})} v^\star$,

ii) if u is a $\mathcal{R}(Q,\mathbb{S})$-normal form in Q^*, then $u^\star$ is a $\mathcal{R}(Q,\mathbb{S})$-normal form.

As a consequence, the equality $R_Q C_{\mathbb{S}_Q}(u^\star) = (R_Q C_{\mathbb{S}_Q}(u))^\star$ holds, for all $u \in Q^*$. Indeed, $\mathcal{R}(Q,\mathbb{S})$ being terminating, we have $R_Q C_{\mathbb{S}_Q}(u^\star) \approx_{\mathcal{R}(Q,\mathbb{S})} u^\star$. Condition **i)** implies that $(R_Q C_{\mathbb{S}_Q}(u^\star))^\star \approx_{\mathcal{R}(Q,\mathbb{S})} u$, and by condition **ii)**, the word $(R_Q C_{\mathbb{S}_Q}(u^\star))^\star$ is a $\mathcal{R}(Q,\mathbb{S})$-normal form. Finally, by the unique normal form property of $\mathcal{R}(Q,\mathbb{S})$, we have $(R_Q C_{\mathbb{S}_Q}(u^\star))^\star = R_Q C_{\mathbb{S}_Q}(u)$, showing that $R_Q C_{\mathbb{S}_Q}(u^\star) = (R_Q C_{\mathbb{S}_Q}(u))^\star$.

By applying the involution on the sources and the targets of the rules (4.1.5) of $\mathcal{R}(Q,\mathbb{S})$, these rules turn into

$$\gamma_{c'^\star,c^\star} : c'^\star\cdot c^\star \to (R_Q C_{\mathbb{S}_Q}(c\cdot c'))^\star = R_Q C_{\mathbb{S}_Q}(c'^\star\cdot c^\star)$$

for all c, c' in Q, whenever $c\cdot c' \neq R_Q(c \star_I c')$. In this way, the involution transforms the rightmost normalisation strategy $\sigma^{\perp,Q}$ into the leftmost normalisation strategy $\sigma^{\top,Q}$, and conversely.

In particular for the string data structure $\mathbb{Y}^r_n$, the Schützenberger involution * is defined on Col_n by sending each column to its complement in Col_n. That is, for a column u in Col_n containing p boxes, $u^\star$ is the column containing $n-p$ boxes filled by the complements of the elements of u. One shows that the Schützenberger involution satisfies the conditions **i)** and **ii)**.

5. String data structures for Chinese monoids

In this section we construct a string data bistructure that presents the Chinese congruence. The *Chinese monoid of rank* $n > 0$, introduced in [14], and denoted by $\mathbf{C}_n$, is presented by the rewriting system on $[n]$,

whose rules are the *Chinese relations*:

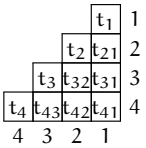
$$\begin{array}{llll} zyx \to yzx & \text{and} & zxy \to yzx & \text{for all} \quad 1 \leqslant x < y < z \leqslant n, \\ yyx \to yxy & \text{and} & yxx \to xyx & \text{for all} \quad 1 \leqslant x < y \leqslant n. \end{array} \tag{5.0.1}$$

In next subsection, we recall the structure of Chinese staircase and the right and left insertion algorithms in Chinese staircases. The main result of this section, Theorem 5.1.4, states that these two algorithms commute. In Subsection 5.2 we give a construction of a semi-quadratic convergent presentation of the Chinese monoid, that we extend in Subsection 5.3 into a coherent one.

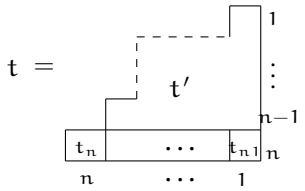
5.1. Presentation of Chinese monoids by string data structures

We recall from [10] the notion of Chinese monoid and the representation of this monoid by Chinese staircases. The set of Chinese staircases satisfies the cross-section property for the Chinese monoid.

5.1.1. Chinese staircases. A *Ferrers diagram* of shape $(1, 2, \dots, n)$ is a collection of boxes in right-justified rows, whose rows (resp. columns) are indexed with $[n]$ from top to bottom (resp. from right to left) and where every i-th row contains i boxes for $1 \leqslant i \leqslant n$. A *(Chinese) staircase* over $[n]$ is a Ferrers diagram of shape $(1, 2, \dots, n)$ filled with non-negative integers. Denote by t_{ij} (resp. t_i) the contents of the box in row i and column j for $i > j$ (resp. $i = j$). A box filled by 0 is called *empty*. Denote by Ch_n the set of staircases over $[n]$ and by $R_r : \mathrm{Ch}_n \to [n]^*$ the map that reads a staircase row by row, from right to left and from top to bottom, and where the i-th row is read as follows $(i1)^{t_{i1}}(i2)^{t_{i2}} \dots (i(i-1))^{t_{i(i-1)}}(i)^{t_i}$, for $1 \leqslant i \leqslant n$. For instance, for the following staircase t over $[4]$:



we have $R_r(t) = 1^{t_1}(21)^{t_{21}}(2)^{t_2}(31)^{t_{31}}(32)^{t_{32}}(3)^{t_3}(41)^{t_{41}}(42)^{t_{42}}(43)^{t_{43}}(4)^{t_4}$. Given a staircase over $[n]$



by removing the bottom row, we obtain a staircase over $[n-1]$, denoted by t' on the picture. According to this, a staircase t over $[n]$ can be denoted by (t', R_1), where R_1 is the bottom row of t, and t' is the staircase over $[n-1]$ obtained by removing the row R_1.

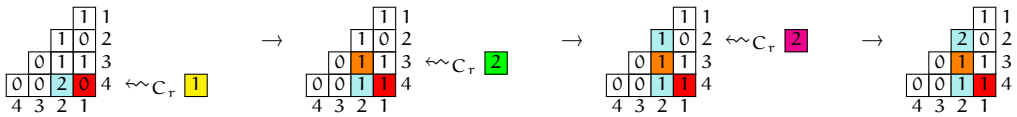
5.1.2. The right insertion algorithm. Recall the right insertion map $C_r : \mathrm{Ch}_n \times [n] \to \mathrm{Ch}_n$ introduced in [10, Subsection 2.2]. Let t be a staircase and x an element in $[n]$. If $x = n$, then $C_r(t, x) = (t', R_1')$, where R_1' is obtained from R_1 by adding 1 to t_n. If $x < n$, let y_1 be maximal such that the entry in column y_1 of R_1 is non-zero or if such a y_1 does not exist, set $y_1 = x$. Three cases appear:

i) If $x \geqslant y_1$, then $C_r(t, x) = (C_r(t', x), R_1)$,

ii) If $x < y_1 < n$, then $C_r(t, x) = (C_r(t', y_1), R_1')$, where R_1' is obtained from R_1 by subtracting 1 from t_{ny_1} and adding 1 to t_{nx},

iii) If $x < y_1 = n$, then $C_r(t, x) = (t', R_1')$, where R_1' is obtained from R_1 by substracting 1 from t_n and adding 1 to t_{nx}.

For example, we compute $\left(\begin{array}{cccc} & & & 1 \\ & & 1 & 2 \\ & 0 & 1 & 3 \\ 0 & 0 & 2 & 0 & 4 \\ 4 & 3 & 2 & 1 \end{array} \leftsquigarrow_{C_r} 1 \right)$ in three steps:



5.1.3. The left insertion algorithm. A left insertion map $C_l : \mathrm{Ch}_n \times [n] \to \mathrm{Ch}_n$ that inserts an element x in $[n]$ into a staircase t, is defined in [5, Algorithm 3.5] in two steps as follows. Let y be an element in $[n] \cup \{\lambda\}$, initially set to λ.

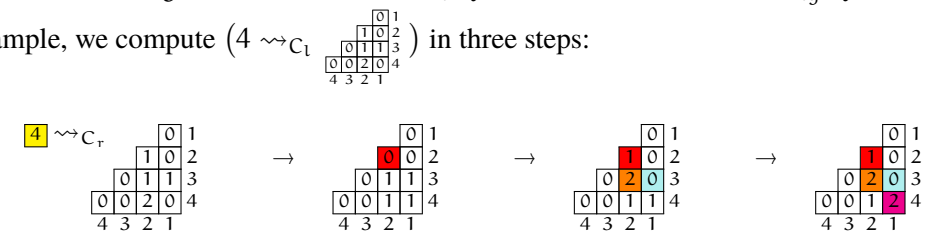
Step 1. For $i = 1, \ldots, x-1$, iterate the following. If every entry in the i-th row is empty, do nothing. Otherwise, let z be minimal such that t_{iz} is non-zero. There are two cases according to the values of y:

i) Suppose $y = \lambda$. If $z < i$, decrement t_{iz} by 1, increment t_i by 1, and set $y = z$. If $z = i$, decrement t_i by 1, and set $y = z$.

ii) Suppose $y \neq \lambda$. If $z < y$, decrement t_{iz} by 1, increment t_{iy} by 1, and set $y = z$. If $z \geqslant y$, do nothing.

Step 2. For $i = x$, if $y = \lambda$, then increment t_i by 1. Otherwise, decrement t_{iy} by 1.

For example, we compute $\left(4 \rightsquigarrow_{C_l} \begin{array}{cccc} & & & 1 \\ & & 1 & 2 \\ & 0 & 1 & 3 \\ 0 & 0 & 2 & 0 & 4 \\ 4 & 3 & 2 & 1 \end{array}\right)$ in three steps:



5.1.4. Theorem. *For all staircase* t *in* Ch_n *and* x, y *in* $[n]$*, the following equality*

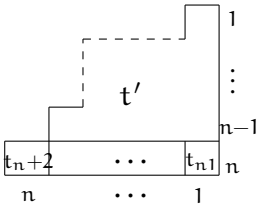
$$y \rightsquigarrow_{C_l} (t \leftsquigarrow_{C_r} x) = (y \rightsquigarrow_{C_l} t) \leftsquigarrow_{C_r} x \tag{5.1.5}$$

holds in Ch_n.

By this result we deduce a string data bistructure $(\mathrm{Ch}_n, C_r, C_l, R_r)$ on staircases over $[n]$, and following Theorem 3.2.3, the compositions $\star_{C_r}$ and $\star_{C_l}$ are associative. Moreover, the insertion algorithms C_r and C_l can be deduced to each other by formulas (3.2.6) and (3.2.7). The rest of this section is devoted to the proof of Theorem 5.1.4. We consider a staircase $t = (t', R_1)$ and x, y in $[n]$. We prove relation (5.1.5) by considering four cases according to the values of x and y.

5.1.6. Case 1: $x = y = n$**.** The staircase $t \leftsquigarrow_{C_r} n$ is obtained from t by adding 1 to t_n.
Case 1. A. Suppose that any box in t' is empty. The staircase $n \rightsquigarrow_{C_l} t$ is obtained from t by adding 1 to t_n. Similarly, the staircase $n \rightsquigarrow_{C_l} (t \leftsquigarrow_{C_r} n)$ is obtained from $t \leftsquigarrow_{C_r} n$ by adding 1 to t_n. Then $n \rightsquigarrow_{C_l} (t \leftsquigarrow_{C_r} n)$ is obtained from t by adding 2 to t_n. Moreover, the staircase $(n \rightsquigarrow_{C_l} t) \leftsquigarrow_{C_r} n$ is obtained from $n \rightsquigarrow_{C_l} t$ by adding 1 to t_n, and thus it is obtained from t by adding 2 to t_n. Hence

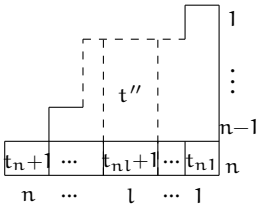
$n \rightsquigarrow_{C_l} (t \leftsquigarrow_{C_r} n)$ =



= $(n \rightsquigarrow_{C_l} t) \leftsquigarrow_{C_r} n$

Case 1. B. Suppose that t' contains at least one non-empty box. The bottom row of the staircase $n \rightsquigarrow_{C_l} (t \leftsquigarrow_{C_r} n)$ is obtained from the bottom one of $t \leftsquigarrow_{C_r} n$ by adding 1 to t_{nl} where the l-th column is the last one in which we have eliminating 1 after applying Step 1 of 5.1.3 on the remaining rows of $t \leftsquigarrow_{C_r} n$. Then the staircase $n \rightsquigarrow_{C_l} (t \leftsquigarrow_{C_r} n)$ is obtained from t by adding 1 to t_{nl} and t_n after applying Step 1 of 5.1.3 on the remaining rows of t. Similarly, the bottom row of $n \rightsquigarrow_{C_l} t$ is obtained from the bottom one of t by adding 1 to t_{nl}. Then $(n \rightsquigarrow_{C_l} t) \leftsquigarrow_{C_r} n$ is obtained from t by adding 1 to t_{nl} and t_n after applying Step 1 of 5.1.3 on t'. Hence

$n \rightsquigarrow_{C_l} (t \leftsquigarrow_{C_r} n)$ =

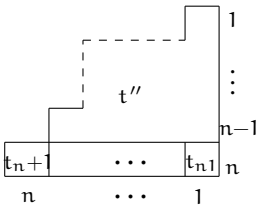


= $(n \rightsquigarrow_{C_l} t) \leftsquigarrow_{C_r} n$

where t'' is the staircase obtained from t' by applying Step 1 of 5.1.3 on t' when computing $n \rightsquigarrow_{C_l} t$.

5.1.7. Case 2: $y < n$ **and** $x = n$**.** The staircase $t \leftsquigarrow_{C_r} n$ is obtained from t by adding 1 to t_n. Since $y < n$, by definition of C_l, when computing $y \rightsquigarrow_{C_l} (t \leftsquigarrow_{C_r} n)$ we only change the contents of the boxes in t' and no operations are performed in the bottom row of $t \leftsquigarrow_{C_r} n$. Similarly, when computing $y \rightsquigarrow_{C_l} t$, we only change the contents of the boxes in t' and no operations are done in R_1. Moreover, the staircase $(y \rightsquigarrow_{C_l} t) \leftsquigarrow_{C_r} n$ is obtained from $y \rightsquigarrow_{C_l} t$ by adding 1 to t_n. Hence

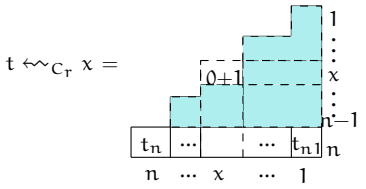
$y \rightsquigarrow_{C_l} (t \leftsquigarrow_{C_r} n)$ =



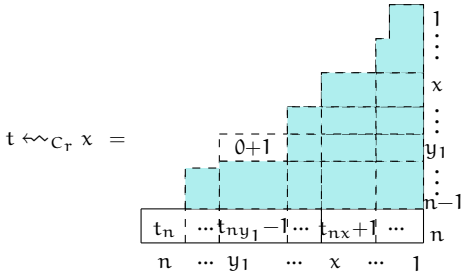
= $(y \rightsquigarrow_{C_l} t) \leftsquigarrow_{C_r} n$

where t'' is the staircase obtained from t' by applying Step 1 of 5.1.3 on t' when computing $y \rightsquigarrow_{C_l} t$.

5.1.8. Case 3: $y = n$ and $x < n$. There are two subcases.
Case 3. A. Suppose that all the contents of the boxes in t' are zero. In this case, the staircase $n \rightsquigarrow_{C_l} t$ is obtained from t by adding 1 to t_n. Then $(n \rightsquigarrow_{C_l} t) \leftsquigarrow_{C_r} x$ is obtained from $n \rightsquigarrow_{C_l} t$ by eliminating 1 from t_n and by adding 1 to t_{nx} in its bottom row. Hence $(n \rightsquigarrow_{C_l} t) \leftsquigarrow_{C_r} x$ is obtained from t by adding 1 to t_{nx}. Let us compute the staircase $n \rightsquigarrow_{C_l} (t \leftsquigarrow_{C_r} x)$. Let y_1 be maximal such that the entry in column y_1 of R_1 is non-zero. Three new subcases appear.
Case 3. A. 1. $x \geqslant y_1$. We have $t \leftsquigarrow_{C_r} x = \left(t' \leftsquigarrow_{C_r} x, R_1\right)$. Since all the boxes of t' are empty, the staircase $t' \leftsquigarrow_{C_r} x$ is obtained from t' by adding 1 to t_x. We obtain

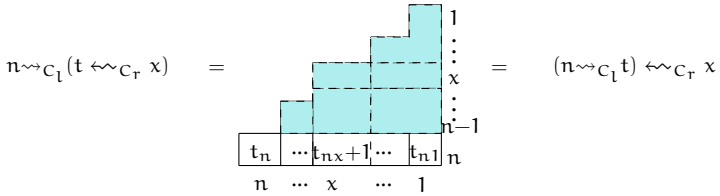


where the shaded area denotes empty boxes. Then the staircase $n \rightsquigarrow_{C_l} (t \leftsquigarrow_{C_r} x)$ is obtained from $t \leftsquigarrow_{C_r} x$ by eliminating 1 from t_x and by adding 1 to t_{nx}. Hence the staircase $n \rightsquigarrow_{C_l} (t \leftsquigarrow_{C_r} x)$ is obtained from t by adding 1 to t_{nx} in R_1.
Case 3. A. 2. $x < y_1 = n$. We have $t \leftsquigarrow_{C_r} x = \left(t', R_1'\right)$, where R_1' is obtained from R_1 by eliminating 1 from t_n and by adding 1 to t_{nx}. Moreover, the staircase $n \rightsquigarrow_{C_l} (t \leftsquigarrow_{C_r} x)$ is obtained from $t \leftsquigarrow_{C_r} x$ by adding 1 to t_n. Hence the staircase $n \rightsquigarrow_{C_l} (t \leftsquigarrow_{C_r} x)$ is obtained from t by adding 1 to t_{nx} in R_1.
Case 3. A. 3. $x < y_1 < n$. We have $t \leftsquigarrow_{C_r} x = \left(t' \leftsquigarrow_{C_r} y_1, R_1'\right)$, where R_1' is obtained from R_1 by eliminating 1 from t_{ny_1} and by adding 1 to t_{nx}, and $t' \leftsquigarrow_{C_r} y_1$ is obtained from t' by adding 1 to t_{y_1}. Then, we obtain



Moreover, the staircase $n \rightsquigarrow_{C_l} (t \leftsquigarrow_{C_r} x)$ is obtained from $t \leftsquigarrow_{C_r} x$ by eliminating 1 from t_{y_1} and t_{ny_1}. Hence it is obtained from t by adding 1 to t_{nx} in R_1.

As a consequence, in the three subcases above we obtain:

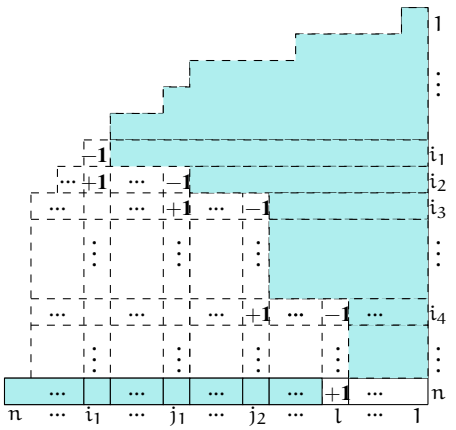


Case 3. B. Suppose that t' contains at least one non-empty box. Let y_1 be maximal such that the entry in column y_1 of R_1 in non-zero. There are two subcases.

Case 3. B. 1. $x < y_1 = n$. We have $t \leftsquigarrow_{C_r} x = \left(t', R'_1\right)$, where R'_1 is obtained from R_1 by eliminating 1 from t_n and by adding 1 to t_{nx}. The bottom row of $n \rightsquigarrow_{C_l} (t \leftsquigarrow_{C_r} x)$ is obtained from the bottom one of $t \leftsquigarrow_{C_r} x$ by adding 1 to t_{nl}, where the l-th column is the last one in which we have eliminating 1 after applying Step 1 of 5.1.3 on the remaining rows of $t \leftsquigarrow_{C_r} x$. Then the staircase $n \rightsquigarrow_{C_l} (t \leftsquigarrow_{C_r} x)$ is obtained from t by adding 1 to t_{nl} and t_{nx} and by eliminating 1 from t_n after applying Step 1 of 5.1.3 on the remaining rows of t. On the other hand, the staircase $n \rightsquigarrow_{C_l} t$ is obtained from t by applying Step 1 of 5.1.3 on t' and by adding 1 to t_{nl}. Moreover, the staircase $(n \rightsquigarrow_{C_l} t) \leftsquigarrow_{C_r} x$ is obtained from $n \rightsquigarrow_{C_l} t$ by eliminating 1 from t_n and by adding 1 to t_{nx}. That proves (5.1.5) in this case.

Case 3. B. 2. $x \geqslant y_1$ or $x < y_1 < n$. The other cases being similar, we study the case $x < y_1 < l < n$, where the l-th column is the last one in which we have eliminating 1 after applying Step 1 of 5.1.3, when computing $n \rightsquigarrow_{C_l} t$. We have

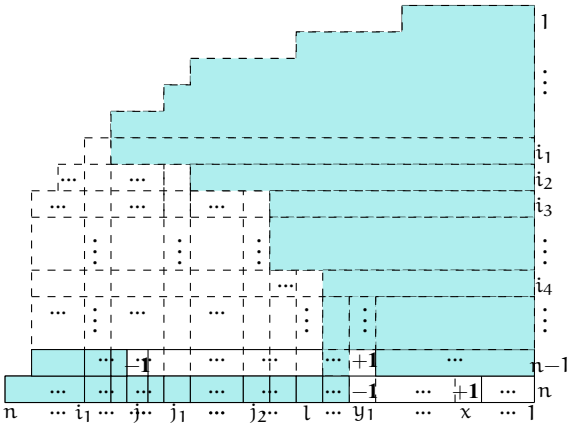
$n \rightsquigarrow_{C_l} t \;=$



where the symbol $+\mathbf{k}$ (resp. $-\mathbf{k}$) indicates that k is added (resp. eliminated) in the corresponding box. Then, the staircase $(n \rightsquigarrow_{C_l} t) \leftsquigarrow_{C_r} x$ is obtained from $n \rightsquigarrow_{C_l} t$ by eliminating 1 from t_{nl}, by adding 1 to t_{nx}, by eliminating 1 from $t_{(n-1)j}$ where j is maximal such that $t_{(n-1)j}$ is non-zero, by adding 1 to $t_{(n-1)l}$ and by applying steps **i)**, **ii)** and **iii)** of 5.1.2 on the remaining rows of $n \rightsquigarrow_{C_l} t$ in the non-shaded area.

On the other hand, the staircase $t \leftsquigarrow_{C_r} x$ is obtained from t by eliminating 1 from t_{ny_1}, by adding 1 to t_{nx}, by eliminating 1 from $t_{(n-1)j}$, by adding 1 to $t_{(n-1)y_1}$ and by applying steps **i)**, **ii)** and **iii)** of 5.1.2 on the remaining rows of t in the non-shaded area, as shown in the following diagram:

$t \leftsquigarrow_{C_r} x \;=$



Then, the staircase $n \rightsquigarrow_{C_l} (t \leftsquigarrow_{C_r} x)$ is obtained from $t \leftsquigarrow_{C_r} x$ by applying Step 1 of 5.1.3 in the non-shaded area, by eliminating 1 from $t_{(n-1)y_1}$ and by adding 1 to t_{ny_1}. That proves (5.1.5) in this case.

5.1.9. Case 4: $x < n$ and $y < n$. Let y_1 be maximal such that the entry in column y_1 of R_1 in non-zero.
Case 4. A. Suppose $x < y_1 = n$. In this case, $t \leftsquigarrow_{C_r} x = (t', R_1')$, where R_1' is obtained from R_1 by substracting 1 from t_n and by adding 1 to t_{nx}. Since $y < n$, when computing $y \rightsquigarrow_{C_l} (t \leftsquigarrow_{C_r} x)$ we only modify the contents of the boxes in t'. Then we obtain

$$y \rightsquigarrow_{C_l} (t \leftsquigarrow_{C_r} x) = y \rightsquigarrow_{C_l} (t', R_1') = (y \rightsquigarrow_{C_l} t', R_1').$$

Moreover, we have $y \rightsquigarrow_{C_l} t = (y \rightsquigarrow_{C_l} t', R_1)$, hence $(y \rightsquigarrow_{C_l} t) \leftsquigarrow_{C_r} x = (y \rightsquigarrow_{C_l} t', R_1')$. That proves (5.1.5) in this case.
Case 4. B. Suppose $x \geqslant y_1$ or $x < y_1 < n$. In this case, we have $t \leftsquigarrow_{C_r} x = (t' \leftsquigarrow_{C_r} s, K_1)$, where

- $s = x$ and $K_1 = R_1$ for $x \geqslant y_1$,
- $s = y_1$ and K_1 is obtained from R_1 by subtracting 1 from t_{ny_1} and by adding 1 to t_{nx}, for $x < y_1 < n$.

Let us show (5.1.5) by induction on n. One proves (5.1.5) for a staircase $t = \begin{array}{|c|c|}\hline t_1 & 1 \\ \hline t_2 & t_{21} \\ \hline \end{array}\;\begin{array}{c}1\\2\end{array}$ over $[2]$ by considering four cases according to the values of $t_1, t_2, t_{21} \in [n] \cup \{0\}$:

	$t \leftsquigarrow_{C_r} 1$	$1 \rightsquigarrow_{C_l} t$	$1 \rightsquigarrow_{C_l} (t \leftsquigarrow_{C_r} 1) = (1 \rightsquigarrow_{C_l} t) \leftsquigarrow_{C_r} 1$
$t_1 = t_2 = 0$ and $t_{21} \in [n] \cup \{0\}$	$\begin{array}{\|c\|c\|}\hline 1 & 1 \\ \hline 0 & t_{21} \\ \hline \end{array}\;\begin{array}{c}1\\2\end{array}$ 2 1	$\begin{array}{\|c\|c\|}\hline 0 & 1 \\ \hline 1 & t_{21} \\ \hline \end{array}\;\begin{array}{c}1\\2\end{array}$ 2 1	$\begin{array}{\|c\|c\|}\hline 0 & 1 \\ \hline 0 & +1 \\ \hline \end{array}\;\begin{array}{c}1\\2\end{array}$ 2 1
$t_1, t_2 \neq 0$ and $t_{21} \in [n] \cup \{0\}$	$\begin{array}{\|c\|c\|}\hline t_1 & 1 \\ \hline -1 & +1 \\ \hline \end{array}\;\begin{array}{c}1\\2\end{array}$ 2 1	$\begin{array}{\|c\|c\|}\hline -1 & 1 \\ \hline t_2 & +1 \\ \hline \end{array}\;\begin{array}{c}1\\2\end{array}$ 2 1	$\begin{array}{\|c\|c\|}\hline -1 & 1 \\ \hline -1 & +2 \\ \hline \end{array}\;\begin{array}{c}1\\2\end{array}$ 2 1
$t_1 \neq 0$, $t_2 = 0$ and $t_{21} \in [n] \cup \{0\}$	$\begin{array}{\|c\|c\|}\hline +1 & 1 \\ \hline 0 & t_{21} \\ \hline \end{array}\;\begin{array}{c}1\\2\end{array}$ 2 1	$\begin{array}{\|c\|c\|}\hline -1 & 1 \\ \hline 0 & +1 \\ \hline \end{array}\;\begin{array}{c}1\\2\end{array}$ 2 1	$\begin{array}{\|c\|c\|}\hline t_1 & 1 \\ \hline 0 & +1 \\ \hline \end{array}\;\begin{array}{c}1\\2\end{array}$ 2 1
$t_2 \neq 0$, $t_1 = 0$ and $t_{21} \in [n] \cup \{0\}$	$\begin{array}{\|c\|c\|}\hline 0 & 1 \\ \hline -1 & +1 \\ \hline \end{array}\;\begin{array}{c}1\\2\end{array}$ 2 1	$\begin{array}{\|c\|c\|}\hline 0 & 1 \\ \hline +1 & t_{21} \\ \hline \end{array}\;\begin{array}{c}1\\2\end{array}$ 2 1	$\begin{array}{\|c\|c\|}\hline 0 & 1 \\ \hline t_2 & +1 \\ \hline \end{array}\;\begin{array}{c}1\\2\end{array}$ 2 1

Suppose now that (5.1.5) is verified for staircases over $[n-1]$, and prove it for a staircase t over $[n]$. By hypothesis, the equality $y \rightsquigarrow_{C_l} (t \leftsquigarrow_{C_r} x) = y \rightsquigarrow_{C_l} (t' \leftsquigarrow_{C_r} s, K_1)$ holds. Since $y < n$, by definition of C_l, when computing $y \rightsquigarrow_{C_l} (t' \leftsquigarrow_{C_r} s, K_1)$ we do not change the contents of the boxes in K_1 and all the modifications are performed in $t' \leftsquigarrow_{C_r} s$. Then

$$y \rightsquigarrow_{C_l} (t' \leftsquigarrow_{C_r} s, K_1) = (y \rightsquigarrow_{C_l} (t' \leftsquigarrow_{C_r} s), K_1).$$

The staircase t' being a staircase over $[n-1]$, the following equality holds by induction hypothesis

$$(y \rightsquigarrow_{C_l} (t' \leftsquigarrow_{C_r} s), K_1) = ((y \rightsquigarrow_{C_l} t') \leftsquigarrow_{C_r} s, K_1).$$

On the other hand, since $y < n$, the equality $y \rightsquigarrow_{C_l} t = (y \rightsquigarrow_{C_l} t', R_1)$ holds. Hence we have

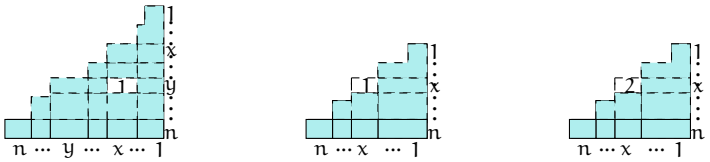
$$(y \rightsquigarrow_{C_l} t) \leftsquigarrow_{C_r} x = ((y \rightsquigarrow_{C_l} t') \leftsquigarrow_{C_r} s, K_1).$$

That proves (5.1.5) in this case.

5.2. Semi-quadratic convergent presentations for Chinese monoids

In this subsection we construct a finite semi-quadratic convergent presentation of the Chinese monoid $\mathbf{C}_n$ by adding the columns in $[n]^*$ of length at most 2 and square generators to the presentation (5.0.1). We will denote by $\mathbb{C}_n$ the right string data structure $(\mathrm{Ch}_n, C_r, \ell_l, R_r)$.

5.2.1. Reduced column presentation. We consider one *column generator* c_{yx} *of length* 2 for all $1 \leqslant x < y \leqslant n$, one *column generator* c_x *of length* 1 for all $1 \leqslant x \leqslant n$, and one *square generator* c_{xx} for all $1 < x < n$, corresponding to the following three staircases:



where the shaded areas represent empty boxes. We will denote by Q_n the set defined by

$$Q_n := \{c_{yx} \mid 1 \leqslant x < y \leqslant n\} \cup \{c_{xx} \mid 1 < x < n\} \cup \{c_1, \ldots, c_n\}.$$

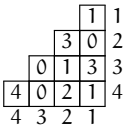
Let us define a map $R_{Q_n} : \mathrm{Ch}_n \to Q_n^*$ that reads a staircase row by row, from right to left and from top to bottom, and where the reading of the i-th row, for $1 \leqslant i \leqslant n$, is the word

$$\underbrace{c_{i1} \cdot \ldots \cdot c_{i1}}_{t_{i1} \text{ times}} \cdot \underbrace{c_{i2} \cdot \ldots \cdot c_{i2}}_{t_{i2} \text{ times}} \cdot \ldots \cdot c_i \cdot \underbrace{c_{ii} \cdot \ldots \cdot c_{ii}}_{\frac{1}{2}(t_i - 1) \text{ times}}$$

in Q^*, when t_i is an odd number, or the word

$$\underbrace{c_{i1} \cdot \ldots \cdot c_{i1}}_{t_{i1} \text{ times}} \cdot \underbrace{c_{i2} \cdot \ldots \cdot c_{i2}}_{t_{i2} \text{ times}} \cdot \ldots \underbrace{c_{ii} \cdot \ldots \cdot c_{ii}}_{\frac{1}{2}t_i \text{ times}}$$

in Q^*, when t_i is an even number. For instance, consider the following staircase t over $[4]$:



we have $R_{Q_n}(t) = c_1 \cdot c_2 \cdot c_{22} \cdot c_{31} \cdot c_{31} \cdot c_{31} \cdot c_{32} \cdot c_{41} \cdot c_{42} \cdot c_{42} \cdot c_{44} \cdot c_{44}$.

5.2.2. Lemma. *The set Q_n is a generating set of the string data structure $\mathbb{C}_n$.*

Proof. By definition $\iota_{Ch_n}(x) = c_x$ belongs to Q_n for all x in $[n]$. For c in $Q_n \setminus \{c_1, \dots, c_n\}$, then $c \star_{C_r} c$ is the staircase whose all boxes are empty except the box corresponding to c that is filled by 2 (resp. 4) if c is a column generator of length 2 (resp. a square generator). For any c, c' in Q_n such that the non-empty box of c is located above or to the right of the non-empty one of c', then $c \star_{C_r} c'$ is the staircase whose all boxes are empty expect the two boxes corresponding to those of c and c'. As a consequence, for any c in $Q_n \setminus \{c_1, \dots, c_n\}$ (resp. c, c' in Q_n), the staircase $c \star_{C_r} c$ (resp. $c \star_{C_r} c'$) does not belong to Q_n. Moreover, following the reading R_{Q_n} any staircase t in Ch_n can be uniquely decomposed as $t = c_{u_1} \star_{C_r} \dots \star_{C_r} c_{u_l}$, where $c_{u_1}, \dots, c_{u_l}$ belong to Q_n, and the non-empty box of c_{u_i} is located above or to the right of the non-empty one of $c_{u_{i+1}}$ for all $1 \leqslant i \leqslant l-1$. By remark above and property of the decomposition of t with respect the reading R_{Q_n}, we have $c_{u_i} \star_{C_r} c_{u_{i+1}} \notin Q_n$. Finally, by definition of R_r, we have $R_r(t) = R_r(c_{u_1}) \dots R_r(c_{u_l})$ in $[n]^*$. This proves that Q_n is a generating set of $\mathbb{C}_n$. □

By this lemma, one can consider the rewriting system $\mathcal{R}(Q_n, \mathbb{C}_n)$ defined in 4.1.4. Its rules are

$$\gamma_{u,v} : c_u \cdot c_v \to R_{Q_n}(c_u \star_{C_r} c_v)$$

such that $c_u \cdot c_v \neq R_{Q_n}(c_u \star_{C_r} c_v)$. Note that the leftmost and rightmost reductions are the only reductions on a word $c_u \cdot c_v \cdot c_t$ in Q_n^* with respect to $\mathcal{R}(Q_n, \mathbb{C}_n)$. There will be denoted respectively by

$$\gamma_{\widehat{u,v},t} := \gamma_{u,v} \cdot c_t \qquad \text{and} \qquad \gamma_{u,\widehat{v,t}} := c_u \cdot \gamma_{v,t}. \tag{5.2.3}$$

5.2.4. Theorem. *The rewriting system $\mathcal{R}(Q_n, \mathbb{C}_n)$ is a finite semi-quadratic convergent presentation of the Chinese monoid $\mathbf{C}_n$.*

As a consequence of this result, the set of $\mathcal{R}(Q_n, \mathbb{C}_n)$-normal forms, that we call *Chinese normal forms*, satisfies the cross-section property for the monoid $\mathbf{C}_n$. Note that such a cross-section property is showed in [10, Theorem 2.1], with an other approach based on combinatorial properties of the right insertion algorithm C_r.

The rest of this section is devoted to the proof of Theorem 5.2.4. Let us prove that $\mathcal{R}(Q_n, \mathbb{C}_n)$ is a semi-quadratic presentation of the monoid $\mathbf{C}_n$. We first add in 5.2.5 the columns generators of length 2 and the square generators with their defining rules. This forms a non-confluent rewriting system that we complete into a presentation of $\mathbf{C}_n$, that we call the *precolumn presentation*. Then we show in 5.2.8 that the rules of $\mathcal{R}(Q_n, \mathbb{C}_n)$ are obtained from the precolumn presentation by applying one step of Knuth-Bendix's completion, [28], on the precolumn presentation. Hence $\mathcal{R}(Q_n, \mathbb{C}_n)$ is a presentation of the monoid $\mathbf{C}_n$. In Lemma 5.2.9, we show that the generating set Q_n is well-founded, that is, the rewriting system $\mathcal{R}(Q_n, \mathbb{C}_n)$ is terminating. The confluence of $\mathcal{R}(Q_n, \mathbb{C}_n)$ follows from Proposition 4.1.8 and Theorem 5.1.4.

5.2.5. Precolumn presentation. Consider the rewriting system $\mathrm{Ch}_2(n)$ on $\{c_1, \dots, c_n\}$ and whose rules are given by the following four families

$$\begin{aligned} &\varepsilon_{x,y,z} : c_z \cdot c_y \cdot c_x \to c_y \cdot c_z \cdot c_x \ \text{ and } \ \eta_{x,y,z} : c_z \cdot c_x \cdot c_y \to c_y \cdot c_z \cdot c_x \quad \text{for all} \quad 1 \leqslant x < y < z \leqslant n, \\ &\varepsilon_{x,y} : c_y \cdot c_y \cdot c_x \to c_y \cdot c_x \cdot c_y \ \text{ and } \ \eta_{x,y} : c_y \cdot c_x \cdot c_x \to c_x \cdot c_y \cdot c_x \quad \text{for all} \quad 1 \leqslant x < y \leqslant n, \end{aligned} \tag{5.2.6}$$

corresponding to the Chinese relations (5.0.1), hence is a presentation of the monoid $\mathbf{C}_n$. We add to the set of rules (5.2.6) the following set of rules

$$\Gamma_2(n) = \{\, \gamma_{y,x} : c_y \cdot c_x \to c_{yx} \mid 1 \leqslant x < y \leqslant n \,\} \cup \{\, \gamma_{x,x} : c_x \cdot c_x \to c_{xx} \mid 1 < x < n \,\},$$

making a rewriting system $\mathrm{Ch}_2^c(n) = \Gamma_2(n) \cup \mathrm{Ch}_2(n)$ on Q_n that presents the monoid $\mathbf{C}_n$.

5.2.7. Lemma. *For $n > 0$, the rewriting system $\mathrm{PreCol}_2(n)$ on Q_n, whose set of rules is $\Gamma_2(n) \cup \Delta_2(n)$, where*

$$\begin{aligned} \Delta_2(n) = &\{\, \gamma_{y,yx} : c_y \cdot c_{yx} \to c_{yx} \cdot c_y \ \textit{for}\ 1 \leqslant x < y \leqslant n \ \textit{and}\ \gamma_{yy,x} : c_{yy} \cdot c_x \to c_{yx} \cdot c_y \ \textit{for}\ 1 \leqslant x < y < n \} \\ &\cup \{\, \gamma_{zy,x} : c_{zy} \cdot c_x \to c_y \cdot c_{zx} \ \textit{and}\ \gamma_{z,yx} : c_z \cdot c_{yx} \to c_y \cdot c_{zx} \ \textit{for}\ 1 \leqslant x \leqslant y < z \leqslant n \,\} \\ &\cup \{\, \gamma_{zx,y} : c_{zx} \cdot c_y \to c_y \cdot c_{zx} \ \textit{for}\ 1 \leqslant x < y < z \leqslant n \,\}. \end{aligned}$$

is a finite semi-quadratic presentation of the Chinese monoid $\mathbf{C}_n$.

Proof. We make explicit a Tietze equivalence between the rewriting systems $\mathrm{Ch}_2^c(n)$ and $\mathrm{PreCol}_2(n)$. For $1 \leqslant x < y \leqslant n$, consider the following critical branching

$$\begin{array}{ccccc} & & \varepsilon_{x,y} & & \gamma_{\widehat{y,x},y} \\ & \nearrow & & \searrow & \\ c_y \cdot c_y \cdot c_x & & & & c_{yx} \cdot c_y \\ & \searrow & & \nearrow & \\ & & \gamma_{y,\widehat{y,x}} & & \end{array}$$

of the rewriting system $\mathrm{Ch}_2^c(n)$. We consider the Tietze transformation that substitutes the rule $\gamma_{y,yx} : c_y \cdot c_{yx} \to c_{yx} \cdot c_y$ to the rule $\varepsilon_{x,y}$, for every $1 \leqslant x < y \leqslant n$. Similarly, we substitute the rules $\gamma_{yx,x}$, $\gamma_{yy,x}$, $\gamma_{y,xx}$, $\gamma_{zy,x}$, $\gamma_{zx,y}$ and $\gamma_{z,yx}$ respectively to the rules $\eta_{x,y}$, $\varepsilon_{x,y}$, $\eta_{x,y}$, $\varepsilon_{x,y,z}$, $\eta_{x,y,z}$ and $\varepsilon_{x,y,z}$ using the following critical branchings of the rewriting system $\mathrm{Ch}_2^c(n)$:

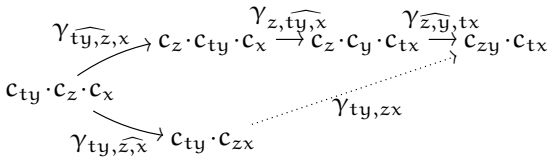
$$\begin{array}{ccccc} & & \eta_{x,y} & & \gamma_{x,\widehat{y,x}} \\ & \nearrow & & \searrow & \\ c_y \cdot c_x \cdot c_x & & & & c_x \cdot c_{yx} \\ & \searrow & & \nearrow & \\ & & \gamma_{\widehat{y,x},x} & & \end{array} \qquad \begin{array}{ccccc} & & \varepsilon_{x,y} & & \gamma_{\widehat{y,x},y} \\ & \nearrow & & \searrow & \\ c_y \cdot c_y \cdot c_x & & & & c_{yx} \cdot c_y \\ & \searrow & & \nearrow & \\ & & \gamma_{\widehat{y,y},x} & & \end{array} \qquad \begin{array}{ccccc} & & \eta_{x,y} & & \gamma_{x,\widehat{y,x}} \\ & \nearrow & & \searrow & \\ c_y \cdot c_x \cdot c_x & & & & c_x \cdot c_{yx} \\ & \searrow & & \nearrow & \\ & & \gamma_{y,\widehat{x,x}} & & \end{array}$$

$$\begin{array}{ccccc} & & \varepsilon_{x,y,z} & & \gamma_{y,\widehat{z,x}} \\ & \nearrow & & \searrow & \\ c_z \cdot c_y \cdot c_x & & & & c_y \cdot c_{zx} \\ & \searrow & & \nearrow & \\ & & \gamma_{\widehat{z,y},x} & & \end{array} \qquad \begin{array}{ccccc} & & \eta_{x,y,z} & & \gamma_{y,\widehat{z,x}} \\ & \nearrow & & \searrow & \\ c_z \cdot c_x \cdot c_y & & & & c_y \cdot c_{zx} \\ & \searrow & & \nearrow & \\ & & \gamma_{\widehat{z,x},y} & & \end{array} \qquad \begin{array}{ccccc} & & \varepsilon_{x,y,z} & & \gamma_{y,\widehat{z,x}} \\ & \nearrow & & \searrow & \\ c_z \cdot c_y \cdot c_x & & & & c_y \cdot c_{zx} \\ & \searrow & & \nearrow & \\ & & \gamma_{z,\widehat{y,x}} & & \end{array}$$

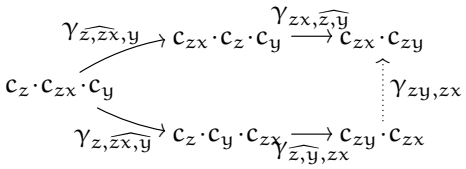
The set of rule $\gamma_{-,-}$ obtained in this way is equal to $\Delta_2(n)$. This proves that the rewriting systems $\mathrm{Ch}_2^c(n)$ and $\mathrm{PreCol}_2(n)$ are Tietze equivalent. □

5.2.8. Completion of the precolumn presentation. The rewriting system $\mathrm{PreCol}_2(n)$ is not confluent, it has the following non-confluent critical branchings, that can be completed by Knuth-Bendix completion, [28], by the dotted arrows as follows:

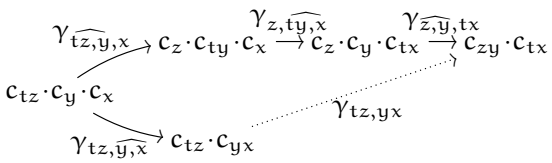
i) for every $1 \leqslant x \leqslant y < z < t \leqslant n$:

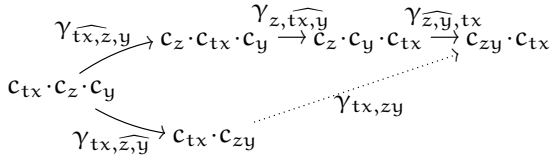


ii) for every $1 \leqslant x < y < z \leqslant n$:

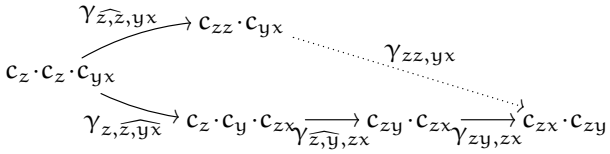


iii) for every $1 \leqslant x < y \leqslant z < t \leqslant n$:

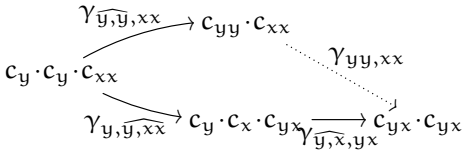




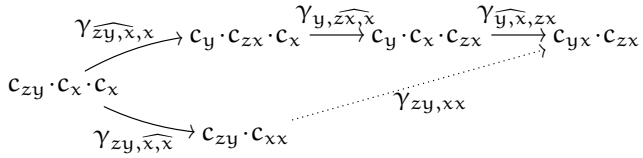
iv) for every $1 \leqslant x < y \leqslant z \leqslant n$:



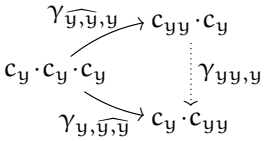
v) for every $1 < x < y < n$:



vi) for every $1 \leqslant x \leqslant y < z \leqslant n$:



vii) for every $1 < y < n$:



The rules of $\mathrm{PreCol}_2(n)$ together with the family of the dotted rules defined by **i)**-**vii)** form the set

$$\left\{ \gamma_{u,v} : c_u \cdot c_v \to \mathrm{R}_{Q_n}(c_u \star_{C_r} c_v) \mid c_u, c_v \in Q_n \right\}.$$

That is, the set of rules of $\mathcal{R}(Q_n, \mathbb{C}_n)$. Finally, by this construction, we prove that $C_{\mathbb{C}_n}(c_u c_v)$ is at most of length 2 in Q_n^*, showing the semi-quadraticity of the presentation.

5.2.9. Lemma. *The generating set Q_n is well-founded.*

Proof. Following 4.1.7, the termination of $\mathcal{R}(Q_n, \mathbb{C}_n)$ can be proved using a lexicographic order induced by the total order $\preccurlyeq_{\mathrm{Ch}}$ defined on Q_n by $c_u \preccurlyeq_{\mathrm{Ch}} c_v$ if

$$\big(u = yx \text{ and } v = y \quad \text{for } 1 \leqslant x < y \leqslant n \big) \quad \text{or} \quad |u| < |v| \quad \text{or} \quad \big(|u| = |v| \text{ and } u <_{\mathrm{lex}} v \big),$$

where $<_{\mathrm{lex}}$ denotes the lexicographic order on $[n]^*$ induced by the natural order on $[n]$. □

5.3. Coherent presentations for Chinese monoids

In this subsection we extend the rewriting system $\mathcal{R}(Q_n, \mathbb{C}_n)$ into a finite coherent convergent presentation of the Chinese monoid $\mathbf{C}_n$ with an explicit description of the generating 3-cells. By semi-quadraticity of $\mathcal{R}(Q_n, \mathbb{C}_n)$, any rewriting path with source $c_u \cdot c_v \cdot c_t$ is an alternated composition of reductions of the form (5.2.3). Moreover, any rewriting rule $\gamma_{-,-}$ of $\mathcal{R}(Q_n, \mathbb{C}_n)$ can be written

$$\gamma_{yx_1,x_2x_3} : c_{yx_1} \cdot c_{x_2x_3} \to c_{x_{\sigma(1)}x_{\sigma(2)}} \cdot c_{yx_{\sigma(3)}} \tag{5.3.1}$$

where $y \in [n]$, $x_1, x_2, x_3 \in [n] \cup \{0\}$, σ is a permutation on $[n] \cup \{0\}$, and c_{x0} denotes the column generator c_x for all $1 < x < n$.

5.3.2. Note that when c_{yx_1} is not a square generator, then $x_{\sigma(1)}$ takes value y only if rule (5.3.1) is one of the *commutation rules* of the form

$$c_y \cdot c_{yx} \to c_{yx} \cdot c_y, \quad c_{zy} \cdot c_{zx} \to c_{zx} \cdot c_{zy}, \quad c_{yy} \cdot c_y \to c_y \cdot c_{yy}, \quad c_{yy} \cdot c_{yx} \to c_{yx} \cdot c_{yy} \tag{5.3.3}$$

for $x < y < z$. When c_{yx_1} is a square generator, with $y > x_2$, then $x_{\sigma(1)}$ takes value y only if rule (5.3.1) is one of the form

$$c_{yy} \cdot c_x \to c_{yx} \cdot c_y, \quad c_{yy} \cdot c_{xx} \to c_{yx} \cdot c_{yx}, \quad c_{zz} \cdot c_{yx} \to c_{zx} \cdot c_{zy}. \tag{5.3.4}$$

We obtain the following bounds for the rewriting paths with source a critical branching of $\mathcal{R}(Q_n, \mathbb{C}_n)$.

5.3.5. Proposition. *For all c_u, c_v, c_t in Q_n such that $c_u \cdot c_v$ and $c_v \cdot c_t$ are not Chinese normal forms, the two following inequalities hold:*

$$\ell_l(c_u \cdot c_v \cdot c_t) \leqslant 5, \qquad \textit{and} \qquad \ell_r(c_u \cdot c_v \cdot c_t) \leqslant 5. \tag{5.3.6}$$

The proof of this result is based on the two following preliminaries lemmas.

5.3.7. Lemma. *Let c_u, c_v, c_t in Q_n. If $\rho_{l,3}(c_u \cdot c_v \cdot c_t)$ is not a Chinese normal form, then the reductions applied to obtain the words $\rho_{l,p}(c_u \cdot c_v \cdot c_t)$, for $3 < p \leqslant 5$, consist only on the commutation rules (5.3.3).*

Proof. Let $c_{yx_1}, c_{x_2x_3}, c_{x_4x_5}$ be in Q_n such that $c_{yx_1} \cdot c_{x_2x_3}$ and $c_{x_2x_3} \cdot c_{x_4x_5}$ are not Chinese normal forms. By definition of $\mathcal{R}(Q_n, \mathbb{C}_n)$, we have

$$\begin{aligned} c_{yx_1} \cdot c_{x_2x_3} \cdot c_{x_4x_5} \to c_{x_{\sigma(1)}x_{\sigma(2)}} \cdot c_{yx_{\sigma(3)}} \cdot c_{x_4x_5} \to c_{x_{\sigma(1)}x_{\sigma(2)}} \cdot c_{x_{\sigma'(\sigma(3))}x_{\sigma'(4)}} \cdot c_{yx_{\sigma'(5)}} \\ \to c_{z_1z_2} \cdot c_{x_{\sigma(1)}z_3} \cdot c_{yx_{\sigma'(5)}} \end{aligned} \tag{5.3.8}$$

with $z_1 = x_{\sigma''(\sigma(2))}$, $z_2 = x_{\sigma''(\sigma'(\sigma(3)))}$, $z_3 = x_{\sigma''(\sigma'(4))}$, and where σ, σ', σ'' are permutations on $[n] \cup \{0\}$, and $c_{x_{\sigma(1)}x_{\sigma(2)}} \cdot c_{yx_{\sigma(3)}}$, $c_{x_{\sigma'(\sigma(3))}x_{\sigma'(4)}} \cdot c_{yx_{\sigma'(5)}}$, $c_{z_1z_2} \cdot c_{x_{\sigma(1)}z_3}$ are Chinese normal forms.

Suppose that $c_{x_{\sigma(1)}z_3} \cdot c_{yx_{\sigma'(5)}}$ is not a Chinese normal form. Following 5.3.2, its only possible reductions are of form (5.3.3) or (5.3.4). Let us prove that the rules (5.3.4) cannot be applied. On the contrary, then $x_{\sigma(1)} = z_3 > y$. Since $c_{z_1z_2} \cdot c_{x_{\sigma(1)}z_3}$ is a Chinese normal form, we obtain that $z_1 = z_3$ and $c_{x_{\sigma(1)}x_{\sigma(2)}} \cdot c_{x_{\sigma'(\sigma(3))}x_{\sigma'(4)}} \cdot c_{yx_{\sigma'(5)}} = c_{z_3z_3} \cdot c_{z_3z_2} \cdot c_{yx_{\sigma'(5)}}$. Since $z_3 > y$, this proves that $c_{z_3z_2} \cdot c_{yx_{\sigma'(5)}} = c_{x_{\sigma'(\sigma(3))}x_{\sigma'(4)}} \cdot c_{yx_{\sigma'(5)}}$ is not a Chinese normal form, which yields a contradiction.

Then we can only apply a commutation rule on $c_{x_{\sigma(1)}z_3}\cdot c_{yx_{\sigma'(5)}}$, with $x_{\sigma(1)} = y$, and we rewrite the word $c_{z_1z_2}\cdot c_{x_{\sigma(1)}z_3}\cdot c_{yx_{\sigma'(5)}}$ into $c_{z_1z_2}\cdot c_{yx_{\sigma'(5)}}\cdot c_{x_{\sigma(1)}z_3}$. Suppose that $c_{z_1z_2}\cdot c_{yx_{\sigma'(5)}}$ is not a Chinese normal form, then we can apply on it a rule of type (5.3.3) or (5.3.4). As in the previous step, let us prove that the rules (5.3.4) cannot be applied. On the contrary, then $z_1 = z_2 > y$. Since $c_{z_1z_2}\cdot c_{x_{\sigma(1)}z_3}$ is a Chinese normal form, we obtain that $z_1 = z_2 = x_{\sigma(1)} = y$, which yields a contradiction. Then we can only apply a commutation rule on $c_{z_1z_2}\cdot c_{yx_{\sigma'(5)}}$.

We have thus proved that the reductions applied to obtain the words $\rho_{l,4}(c_{yx_1}\cdot c_{x_2x_3}\cdot c_{x_4x_5})$ and $\rho_{l,5}(c_{yx_1}\cdot c_{x_2x_3}\cdot c_{x_4x_5})$ consist only on the commutation rules. □

5.3.9. Lemma. *For all* c_u, c_v, c_t *in* Q_n *such that* c_u *is a square generator and the words* $c_u\cdot c_v$ *and* $c_v\cdot c_t$ *are not Chinese normal forms, the inequality* $\ell_r(c_u\cdot c_v\cdot c_t) \leqslant 5$ *holds.*

Proof. By hypotheses, the word $c_u\cdot c_v\cdot c_t$ have the following forms: $c_{rr}\cdot c_{tz}\cdot c_{yx}$ and $c_{rr}\cdot c_{tx}\cdot c_{zy}$, for all $x < y < z < t \leqslant r$, $c_{tt}\cdot c_{zy}\cdot c_{zx}$, $c_{tt}\cdot c_{zx}\cdot c_y$, $c_{tt}\cdot c_{zy}\cdot c_x$ and $c_{tt}\cdot c_{zy}\cdot c_{yx}$, for all $x < y < z \leqslant t$, $c_{zz}\cdot c_{yx}\cdot c_x$ and $c_{zz}\cdot c_y\cdot c_x$, for all $x < y \leqslant z$, $c_{rr}\cdot c_{ty}\cdot c_{zx}$, for all $x \leqslant y < z < t \leqslant r$, and $c_{tt}\cdot c_z\cdot c_{yx}$ for all $x < y \leqslant z \leqslant t$. For all these forms, we show that $\ell_r(c_u\cdot c_v\cdot c_t) \leqslant 5$. □

5.3.10. Proof of Proposition 5.3.5. Let $c_{yx_1}, c_{x_2x_3}, c_{x_4x_5}$ be in Q_n such that $c_{yx_1}\cdot c_{x_2x_3}$ and $c_{x_2x_3}\cdot c_{x_4x_5}$ are not Chinese normal forms. Let us prove that $\ell_l(c_{yx_1}\cdot c_{x_2x_3}\cdot c_{x_4x_5}) \leqslant 5$. Suppose that $\rho_{l,2}(c_{yx_1}\cdot c_{x_2x_3}\cdot c_{x_4x_5})$ is not a Chinese normal form.

Consider a reduction as in (5.3.8), and suppose that $c_{x_{\sigma(1)}z_3}\cdot c_{yx_{\sigma'(5)}}$ is not a Chinese normal form. By Lemma 5.3.7 its only possible reductions are commutation rules, hence there is a reduction $c_{z_1z_2}\cdot c_{x_{\sigma(1)}z_3}\cdot c_{yx_{\sigma'(5)}} \to c_{z_1z_2}\cdot c_{yx_{\sigma'(5)}}\cdot c_{x_{\sigma(1)}z_3}$. Suppose that $c_{z_1z_2}\cdot c_{yx_{\sigma'(5)}}$ is not a Chinese normal form, then by the same argument there is a reduction $c_{z_1z_2}\cdot c_{yx_{\sigma'(5)}}\cdot c_{x_{\sigma(1)}z_3} \to c_{yx_{\sigma'(5)}}\cdot c_{z_1z_2}\cdot c_{x_{\sigma(1)}z_3}$, where $c_{yx_{\sigma'(5)}}\cdot c_{x_{\sigma(1)}z_3}$ and $c_{yx_{\sigma'(5)}}\cdot c_{z_1z_2}$ are Chinese normal forms. Since $c_{z_1z_2}c_{x_{\sigma(1)}z_3}$ is a Chinese normal form, we obtain that $c_{yx_{\sigma'(5)}}c_{x_{\sigma(1)}z_3}$ is a Chinese normal form. This proves the first inequality in 5.3.6.

Let us prove that $\ell_r(c_{yx_1}\cdot c_{x_2x_3}\cdot c_{x_4x_5}) \leqslant 5$. Suppose that the word $\sigma_{r,3}(c_{yx_1}\cdot c_{x_2x_3}\cdot c_{x_4x_5})$ is not a Chinese normal form. By definition of $\mathcal{R}(Q_n, \mathbb{C}_n)$, we have the following reductions

$$c_{yx_1}\cdot c_{x_2x_3}\cdot c_{x_4x_5} \to c_{yx_1}\cdot c_{x_{\sigma(3)}x_{\sigma(4)}}\cdot c_{x_2x_{\sigma(5)}} \to c_{x_{\sigma'(1)}y_1}\cdot c_{yy_2}\cdot c_{x_2x_{\sigma(5)}}$$

and (5.3.11)

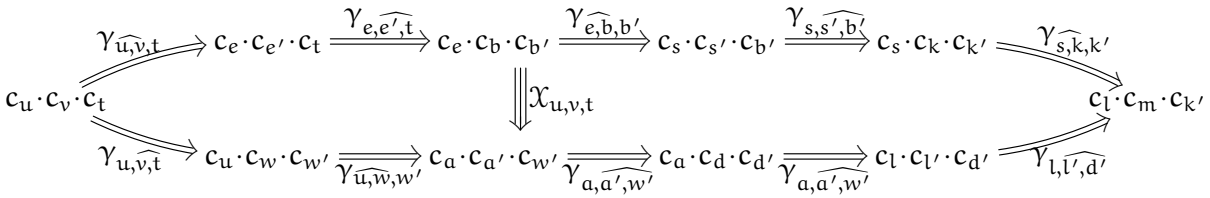
$$c_{x_{\sigma'(1)}y_1}\cdot c_{yy_2}\cdot c_{x_2x_{\sigma(5)}} \to c_{x_{\sigma'(1)}y_1}\cdot c_{x_{\sigma''(2)}z_1}\cdot c_{yz_2} \to c_{t_1t_2}\cdot c_{x_{\sigma'(1)}t_3}\cdot c_{yz_2}$$

with $y_1 = x_{\sigma'(\sigma(3))}$, $y_2 = x_{\sigma'(\sigma(4))}$, $z_1 = x_{\sigma''(\sigma'(\sigma(4)))}$, $z_2 = x_{\sigma''(\sigma(5))}$, $t_1 = x_{\sigma_1(\sigma'(1))}$, $t_2 = x_{\sigma_1(\sigma'(\sigma(3)))}$, $t_3 = x_{\sigma_1(\sigma''(\sigma'(\sigma(1))))}$, and where $\sigma, \sigma', \sigma'', \sigma_1$ are permutations on $[n]\cup\{0\}$, and $c_{x_{\sigma(3)}x_{\sigma(4)}}\cdot c_{x_2x_{\sigma(5)}}$, $c_{x_{\sigma'(1)}y_1}\cdot c_{yy_2}$, $c_{x_{\sigma''(2)}z_1}\cdot c_{yz_2}$ and $c_{t_1t_2}\cdot c_{x_{\sigma'(1)}t_3}$ are Chinese normal forms.

Suppose that $\sigma_{r,4}(c_{yx_1}\cdot c_{x_2x_3}\cdot c_{x_4x_5})$ is not a Chinese normal form. Then $x_{\sigma'(1)} = y$ and the second reduction of (5.3.11) is $c_{yx_1}\cdot c_{x_{\sigma(3)}x_{\sigma(4)}}\cdot c_{x_2x_{\sigma(5)}} \to c_{yy_1}\cdot c_{yy_2}\cdot c_{x_2x_{\sigma(5)}}$. Following 5.3.2, the rule $\gamma_{yx_1,x_{\sigma(3)}x_{\sigma(4)}}$ is of form (5.3.3) or (5.3.4). Let us prove that it cannot be of form (5.3.3). On the contrary, since $c_{x_{\sigma(3)}x_{\sigma(4)}}\cdot c_{x_2x_{\sigma(5)}}$ is a Chinese normal form, we obtain $x_{\sigma(3)} = y \geqslant x_2$. Moreover, since $c_{yx_1}\cdot c_{x_2x_3}$ is not a Chinese normal form, the inequality $y \leqslant x_2$ holds, hence $y = x_2$. In this way, the first reduction of (5.3.11) is $c_{yx_1}\cdot c_{yx_3}\cdot c_{yx_5} \to c_{yx_3}\cdot c_{yx_1}\cdot c_{yx_5}$, where $c_{yx_3}c_{yx_5}$ is a Chinese normal form, and its second reduction is $c_{yx_3}\cdot c_{yx_1}\cdot c_{yx_5} \to c_{yx_3}\cdot c_{yx_5}\cdot c_{yx_1}$. Since $\sigma_{r,3}(c_{yx_1}\cdot c_{yx_3}\cdot c_{yx_5})$ is not a Chinese normal form, the word $c_{yx_3}\cdot c_{yx_5}$ is not a Chinese normal form, which yields a contradiction.

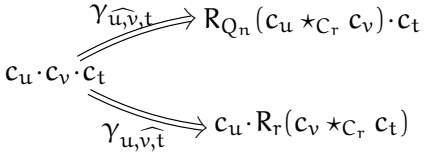
Thus, the rule $\gamma_{yx_1,x_{\sigma(3)}x_{\sigma(4)}}$ is of form (5.3.4) and c_{yx_1} is a square generator such that $c_{yx_1}\cdot c_{x_2x_3}$ and $c_{x_2x_3}\cdot c_{x_4x_5}$ are not Chinese normal forms. Hence by Lemma 5.3.9 we obtain $\ell_r(c_{yx_1}\cdot c_{x_2x_3}\cdot c_{x_4x_5}) \leqslant 5$. This proves the second inequality in (5.3.6).

5.3.12. Theorem. *The rewriting system $\mathcal{R}(Q_n, \mathbb{C}_n)$ extends into a finite coherent convergent presentation of the Chinese monoid $\mathbf{C}_n$ by adjunction of a generating 3-cell*



for any c_u, c_v, c_t in Q_n such that $c_u\cdot c_v$ and $c_v\cdot c_t$ are not Chinese normal forms, and where the 2-cells $\gamma_{-,-}$ denote either a rewriting rule of $\mathcal{R}(Q_n, \mathbb{C}_n)$ or an identity.

Proof. Any critical branching of $\mathcal{R}(Q_n, \mathbb{C}_n)$ has the form



for any c_u, c_v, c_t in Q_n such that $c_u\cdot c_v$ and $c_v\cdot c_t$ are not Chinese normal forms, that is confluent by Theorem 5.2.4. Moreover by Proposition 5.3.5, $\ell_l(c_u\cdot c_v\cdot c_t) \leqslant 5$ and $\ell_r(c_u\cdot c_v\cdot c_t) \leqslant 5$. We conclude with coherent Squier's theorem recalled in 2.2.3. □

5.3.13. Remark. Some 2-cells $\gamma_{-,-}$ in the boundary of the generating 3-cell $\mathcal{X}_{u,v,t}$ can be identity. However, following construction given in the proof of Proposition 5.3.5, if the source (resp. target) of $\mathcal{X}_{u,v,t}$ is of length 5, then its target (resp. source) is of length at most 4.

References

[1] Roland Berger. Confluence and Koszulity. *J. Algebra*, 201(1):243–283, 1998.

[2] Leonid Bokut, Yuqun Chen, Weiping Chen, and Jing Li. New approaches to plactic monoid via Gröbner–Shirshov bases. *J. Algebra*, 423:301–317, 2015.

[3] Ronald Book and Friedrich Otto. *String-rewriting systems*. Texts and Monographs in Computer Science. Springer-Verlag, 1993.

[4] Alan J. Cain, Robert D. Gray, and António Malheiro. Finite Gröbner–Shirshov bases for Plactic algebras and biautomatic structures for Plactic monoids. *J. Algebra*, 423:37–53, 2015.

[5] 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.

[6] Alan J. Cain, Robert D. Gray, and António Malheiro. Crystal monoids & crystal bases: rewriting systems and biautomatic structures for plactic monoids of types A_n, B_n, C_n, D_n, and G_2. *J. Combin. Theory Ser. A*, 162:406–466, 2019.

[7] 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. *Journal of Algebraic Combinatorics*, 45(2):475–524, 2017.

[8] Alan J. Cain and António Malheiro. Crystals and trees: quasi-Kashiwara operators, monoids of binary trees, and Robinson-Schensted-type correspondences. *J. Algebra*, 502:347–381, 2018.

[9] Alan J. Cain, António Malheiro, and Fabio M. Silva. The monoids of the patience sorting algorithm. arXiv:1706.06884, 2017.

[10] Julien Cassaigne, Marc Espie, Daniel Krob, Jean-Christophe Novelli, and Florent Hivert. The Chinese monoid. *Internat. J. Algebra Comput.*, 11(3):301–334, 2001.

[11] Yuqun Chen and Jianjun Qiu. Gröbner-Shirshov basis for the Chinese monoid. *J. Algebra Appl.*, 7(5):623–628, 2008.

[12] Etsurō Date, Michio Jimbo, and Tetsuji Miwa. Representations of $U_q(\mathfrak{gl}(n, C))$ at $q = 0$ and the Robinson-Schensted correspondence. In *Physics and mathematics of strings*, pages 185–211. World Sci. Publ., Teaneck, NJ, 1990.

[13] Michel Dubois-Violette and Todor Popov. Homogeneous algebras, statistics and combinatorics. *Letters in Mathematical Physics*, 61(2):159–170, 2002.

[14] Gérard Duchamp and Daniel Krob. Plactic-growth-like monoids. In *Words, languages and combinatorics, II (Kyoto, 1992)*, pages 124–142. World Sci. Publ., River Edge, NJ, 1994.

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

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

[17] Samuele Giraudo. Algebraic and combinatorial structures on pairs of twin binary trees. *J. Algebra*, 360:115–157, 2012.

[18] Yves Guiraud, Eric Hoffbeck, and Philippe Malbos. Convergent presentations and polygraphic resolutions of associative algebras. arXiv:1406.0815v2, December 2017.

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

[20] Yves Guiraud and Philippe Malbos. Identities among relations for higher-dimensional rewriting systems. In *OPERADS 2009*, volume 26 of *Sémin. Congr.*, pages 145–161. Soc. Math. France, Paris, 2013.

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

[22] Eylem Güzel Karpuz. Complete rewriting system for the Chinese monoid. *Appl. Math. Sci. (Ruse)*, 4(21-24):1081–1087, 2010.

[23] Nohra Hage. Finite convergent presentation of plactic monoid for type C. *Internat. J. Algebra Comput.*, 25(8):1239–1263, 2015.

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

[25] F. Hivert, J.-C. Novelli, and J.-Y. Thibon. The algebra of binary search trees. *Theoret. Comput. Sci.*, 339(1):129–165, 2005.

[26] Florent Hivert, Jean-Christophe Novelli, and Jean-Yves Thibon. Commutative combinatorial hopf algebras. *Journal of Algebraic Combinatorics*, 28(1):65, Jun 2007.

[27] Jin Hong and Seok-Jin Kang. *Introduction to quantum groups and crystal bases*, volume 42 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, RI, 2002.

[28] Donald Knuth and Peter Bendix. Simple word problems in universal algebras. In *Computational Problems in Abstract Algebra (Proc. Conf., Oxford, 1967)*, pages 263–297. Pergamon, Oxford, 1970.

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

[30] 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.

[31] Łukasz Kubat and Jan Okniński. Gröbner-Shirshov bases for plactic algebras. *Algebra Colloq.*, 21(4):591–596, 2014.

[32] 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.

[33] Alain Lascoux and Marcel-Paul 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.

[34] Victoria Lebed. Plactic monoids: a braided approach. Preprint arXiv:1612.05768, 2016.

[35] Cédric Lecouvey. Schensted-type correspondence, plactic monoid, and jeu de taquin for type C_n. *J. Algebra*, 247(2):295–331, 2002.

[36] Cédric Lecouvey. Schensted-type correspondences and plactic monoids for types B_n and D_n. *J. Algebraic Combin.*, 18(2):99–133, 2003.

[37] Peter Littelmann. A plactic algebra for semisimple Lie algebras. *Adv. Math.*, 124(2):312–331, 1996.

[38] M. Lothaire. *Algebraic combinatorics on words*, volume 90 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, Cambridge, 2002.

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

[40] Jean-Christophe Novelli. On the hypoplactic monoid. *Discrete Math.*, 217(1-3):315–336, 2000.

[41] Jean-Baptiste Priez. Lattice of combinatorial hopf algebras: binary trees with multiplicities. *Discrete Mathematics & Theoretical Computer Science*, 2013.

REFERENCES

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

[43] Craig C. Squier, Friedrich Otto, and Yuji Kobayashi. A finiteness condition for rewriting systems. *Theoret. Comput. Sci.*, 131(2):271–294, 1994.

[44] Hugh Thomas and Alexander Yong. Longest increasing subsequences, plancherel-type measure and the hecke insertion algorithm. *Advances in Applied Mathematics*, 46(1):610 – 642, 2011.

[45] Alfred Young. On Quantitative Substitutional Analysis. *Proc. London Math. Soc.*, S2-28(1):255.

Nohra Hage
nohra.hage@usj.edu.lb
Ecole 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