

**FG-MOL 2005:**  
**The 10th conference on**  
**Formal Grammar**  
**and**  
**The 9th Meeting on**  
**Mathematics of Language**  
**Edinburgh**  
**5–7 August 2005**

**Organizing Committee:**  
**Gerhard Jaeger      Paola Monachesi**  
**Gerald Penn        James Rogers**  
**Shuly Wintner**

**CENTER FOR THE STUDY  
OF LANGUAGE  
AND INFORMATION**



---

# Finite Presentations of Pregroups and the Identity Problem

ALEXA H. MATER AND JAMES D. FIX

## Abstract

We consider finitely generated pregroups, and describe how an appropriately defined rewrite relation over words from a generating alphabet yields a natural partial order for a pregroup structure. We investigate the *identity problem* for pregroups; that is, the algorithmic determination of whether a word rewrites to the identity element. This problem is undecidable in general, however, we give a dynamic programming algorithm and an algorithm of Oerhle (2004) for free pregroups, and extend them to handle more general pregroup structures suggested in Lambek (1999). Finally, we show that the identity problem for a certain class of non-free pregroups is NP-complete.

**Keywords** PREGROUPS, FREE PREGROUPS, WORD PROBLEM, DYNAMIC PROGRAMMING, NP-COMPLETE

Lambek (1999) introduced Compact Bilinear Logic (CBL) to provide a computational method for deciding whether an utterance in a natural language is grammatical. Using CBL, utterances are modeled as elements of mathematical structures called *pregroups*. The structure of a pregroup is based on a partial order over its elements and a set of rules for how pregroup element multiplication relates to the partial order.

*FG-MOL 2005:*  
*The 10th conference on Formal Grammar*  
*and*  
*The 9th Meeting on Mathematics of Language*  
*Edinburgh*  
*5-7 August 2005.*  
Organizing Committee:, Gerhard Jaeger, Paola Monachesi, Gerald Penn, James Rogers,  
Shuly Wintner.  
Copyright © 2005, CSLI Publications.

## 1.1 Basic definitions

**Definition 1** [Protogroup] A *protogroup*  $\mathcal{P}$  is a quintuple  $(\mathcal{A}, \cdot, \leq, \ell, r)$  consisting of a set of elements, a binary operation, and a binary relation, and two unary operations respectively, which satisfy the following:

1.  $\cdot$  is associative.
2.  $\leq$  is a partial order; i.e. it is reflexive, transitive and anti-symmetric.
3. There is a  $1 \in \mathcal{A}$  where  $1 \cdot a = a = a \cdot 1$  for every  $a \in \mathcal{A}$ .
4. For every  $a \in \mathcal{A}$ ,  $a^\ell \cdot a \leq 1$  and  $a \cdot a^r \leq 1$ .
5. For any  $a, b, x, y \in \mathcal{A}$ , if  $a \leq b$ , then  $x \cdot a \cdot y \leq x \cdot b \cdot y$ .

The elements  $a^\ell$  and  $a^r$  are the left and right inverses of  $a$ , respectively. With these, it is natural to assume that

$$1^r = 1 = 1^\ell, a^{\ell r} = a = a^{r\ell}, (a \cdot b)^r = b^r \cdot a^r, (a \cdot b)^\ell = b^\ell \cdot a^\ell$$

for any  $a, b \in \mathcal{A}$ . An  $a \in \mathcal{A}$  may have an infinite sequence of inverses:

$$\dots, a^{\ell\ell}, a^\ell, a, a^r, a^{rr}, \dots$$

We will follow Lambek in denoting  $a$  by  $a^{(0)}$ , the  $n$ -th left inverse of  $a$  by  $a^{(-n)}$  and the  $n$ -th right inverse of  $a$  by  $a^{(n)}$ , so that the above sequence can be written

$$\dots, a^{(-2)}, a^{(-1)}, a^{(0)}, a^{(1)}, a^{(2)} \dots$$

**Definition 2** [Pregroup] A *pregroup*  $\mathcal{P} = (\mathcal{A}, \cdot, \leq, \ell, r)$  is a protogroup satisfying the additional property:

6. For every  $a, b \in \mathcal{A}$ , if  $a \leq b$  then  $a^{(2i)} \leq b^{(2i)}$  and  $b^{(2i+1)} \leq a^{(2i+1)}$ .

It will be useful for our purposes to consider an equivalent pregroup property, as noted by Buszkowski (2003):

- 6'. For all  $a \in \mathcal{A}$ ,  $1 \leq a^{(i+1)} a^{(i)}$ .

## 1.2 Applying CBL to Natural Languages

Lambek's idea was to encode a natural language as a pregroup  $\mathcal{P}$  by associating with each word in the language an element of  $\mathcal{P}$ . An utterance is encoded as a product  $b = a_1 a_2 \dots a_n$  of pregroup elements  $a_i$ , one for each word in the utterance. The goal is to have  $b \leq S$ , for some distinguished element  $S$ , whenever the sentence encoded as  $b$  is grammatical. For example, from Lambek (1999) we might assign elements of a pregroup to the sentence "I have been seen" as follows:

$$\begin{array}{cccc} \text{I} & \text{have} & \text{been} & \text{seen} \\ \pi & \pi^r S p^\ell & p o^{\ell\ell} p^\ell & p o^\ell \end{array}$$

where the symbols  $\pi, S, p, o$  are elements of a pregroup. This sequence reduces to  $S$  as demonstrated by the following:

$$\begin{aligned} \pi(\pi^r S p^\ell)(p o^{\ell\ell} p^\ell)(p o^\ell) &= (\pi\pi^r)S(p^\ell p)(o^{\ell\ell}(p^\ell p)o^\ell) \\ &\leq (\pi\pi^r)S(p^\ell p)(o^{\ell\ell} 1 o^\ell) \\ &= (\pi\pi^r)S(p^\ell p)(o^{\ell\ell} o^\ell) \\ &\leq S \end{aligned}$$

Thus, “I have been seen” is a grammatical utterance with this pregroup assignment. Other examples of the use of pregroups to model natural languages can be found in Casadio and Lambek (2001) for Italian, and Bargelli and Lambek (2001) for French.

As a result, we are interested in the algorithmic question of determining whether  $x \leq y$  in a pregroup, where  $x$  and  $y$  are given as a product of pregroup elements. This is the natural *word problem* for pregroups. For *free* pregroups (defined formally below) where cancellations alone yield the pregroup ordering, Oerhle (2004) presented an efficient algorithm for the important case when  $y$  is given as a single element, rather than as a product. We will review a version of this algorithm shortly.

The treatment of pregroups in Lambek (1999) suggests that algorithms for the word problem need to consider additional order relations among elements. For example, Lambek suggests first, second, and third person singular pronouns could be assigned the types  $\pi_1, \pi_2,$  and  $\pi_3,$  respectively, and the additional order relations  $\pi_i \leq \pi$  for a type  $\pi$  of all singular pronouns could be assumed part of the pregroup structure. In Buszkowski (2001) it is shown that type grammars based on pregroups where there are such additional order relations— what we call generator promotions in this paper— are equivalent to context-free grammars.

Considering this further, the pregroup order could include a relation like  $bc \leq a$ . Such additional constraints could give a pregroup a more explicit context-free rewrite structure, the reverse of Chomsky Normal Form productions  $A \rightarrow BC$ . In general, adding order relations between words can make algorithmic solution of the word problem more difficult.

To handle the additional structure that may come with a pregroup, care must be taken in specifying the form in which it is presented as input to an algorithm. To do this, we adapt the notion from combinatorial group theory (see Cohen (1989), Magnus et al. (1966)) of a finite group presentation to pregroups and protogroups. This is a syntactic treatment of pregroups, one where strings of symbols from an alphabet form the pregroup elements, and where the ordering structure comes from rewrite steps, possibly augmented with additional order relations

between words.

### 1.3 Pregroup Presentations

For any set  $\mathcal{G}$ , let  $\mathcal{G}' = \{a^{(i)} \mid a \in \mathcal{G}, i \in \mathcal{Z}\}$  be the set of pregroup letters over  $\mathcal{G}$ . A pregroup word  $W$  over  $\mathcal{G}$  is a sequence of letters, that is,  $W = x_1x_2\dots x_n$  where each  $x_k \in \mathcal{G}'$ . We include the empty letter sequence, denoted by  $\varepsilon$ , in the set of pregroup words. Let  $\mathcal{G}^*$  denote the set of all pregroup words over  $\mathcal{G}$ . The elements of  $\mathcal{G}$  are the generators of  $\mathcal{G}^*$ .

Define the left inverse function  $L : \mathcal{G}^* \rightarrow \mathcal{G}^*$  as follows:

$$L(W) = \begin{cases} \varepsilon & \text{if } W = \varepsilon \\ a^{(i-1)} & \text{if } W = a^{(i)} \text{ for } a \in \mathcal{G} \\ L(x_n) \dots L(x_2)L(x_1) & \text{if } W = x_1 \dots x_n \text{ and each } x_k \in \mathcal{G}'. \end{cases}$$

The right inverse function  $R : \mathcal{G}^* \rightarrow \mathcal{G}^*$  is defined similarly.

Any binary relation  $\mathcal{R}$  on  $\mathcal{G}^*$  induces a rewrite relation on  $\mathcal{G}^*$  defined by:

1.  $W \xrightarrow{\mathcal{R}} W$ ,
2.  $W_1 \xrightarrow{\mathcal{R}} W_3$  whenever  $W_1 \xrightarrow{\mathcal{R}} W_2$  and  $W_2 \xrightarrow{\mathcal{R}} W_3$ ,
3.  $W_1 W W_2 \xrightarrow{\mathcal{R}} W_1 W' W_2$  whenever  $W \xrightarrow{\mathcal{R}} W'$ ,
4.  $\lambda \xrightarrow{\mathcal{R}} \rho$  for  $(\lambda, \rho) \in \mathcal{R}$ , and
5.  $a^{(i)} a^{(i+1)} \xrightarrow{\mathcal{R}} \varepsilon$  for  $a \in \mathcal{G}$ .

A rewrite step employing rule (5) (in conjunction with rule (3)) is called a *contraction*.

If we identify two words  $W_1$  and  $W_2$ , that is, say that  $W_1 = W_2$ , whenever both  $W_1 \xrightarrow{\mathcal{R}} W_2$  and  $W_2 \xrightarrow{\mathcal{R}} W_1$  hold, we have the following proposition:

**Proposition 1** *The set of all words over  $\mathcal{G}$  forms a protogroup under the binary operation of juxtaposition with identity element  $\varepsilon$ , the unary operations  $L$  and  $R$ , and the binary predicate  $\xrightarrow{\mathcal{R}}$ .*

Thus, it is natural to define the following.

**Definition 3** [Protogroup Presentation] If  $\mathcal{P}$  is a protogroup given by  $\mathcal{G}$  and  $\mathcal{R}$  according to the previous proposition then  $\mathcal{G}$  and  $\mathcal{R}$  form a *protogroup presentation* of  $\mathcal{P}$ .

The relations  $\mathcal{R}$  can encode the additional structure among words. For example, a presentation of Lambek's natural language pregroup in the discussion above would include generators  $\pi, \pi_1, \pi_2, \pi_3$  and the relations  $(\pi_1, \pi), (\pi_2, \pi), (\pi_3, \pi)$ .

**Definition 4** [Pregroup Presentation] Extend  $\xrightarrow{\mathcal{R}}$  to include the property

$$6. \varepsilon \xrightarrow{\mathcal{R}} a^{(i+1)}a^{(i)} \text{ for } a \in \mathcal{G}.$$

A rewrite step employing this rule is called an *expansion*. Using this, we have a proposition for pregroups analogous to Proposition 1. We also obtain an analogous notion of a *pregroup presentation* which we will denote by  $\mathcal{P} = \langle \mathcal{G}, \mathcal{R} \rangle_{\mathcal{P}}$ .

In what follows, we assume that an algorithm's input includes a finite-sized presentation of the structure being considered. We will be less careful about describing presentations syntactically, and return to using  $\leq$  instead of  $\xrightarrow{\mathcal{R}}$ ,  $\ell$  and  $r$  instead of  $L$  and  $R$ , and 1 instead of  $\varepsilon$ .

## 1.4 The Identity Problem

From a linguistic perspective, we are primarily interested in determining, whether  $W \leq S$  where  $S$  is some distinguished element in the generating set for  $\mathcal{P}$ . Note that  $W \leq S$  if  $W = W_1TW_2$ ,  $W_1 \leq 1$ ,  $W_2 \leq 1$ , and  $T \leq S$ . Using this fact for the pregroups we consider, leads us to consider the following problem:

### The Pregroup Identity Problem

**Given:**  $\mathcal{P} = \langle \mathcal{G}, \mathcal{R} \rangle_{\mathcal{P}}$ ,  $W \in \mathcal{G}^*$

**Determine:** whether  $W \leq 1$  in  $\mathcal{P}$ .

We say that  $W$  is *nullable* in  $\mathcal{P}$  whenever  $W \leq 1$ .

In general, the identity problem is undecidable for pregroups. This can be proven by showing that any finitely presented group can be encoded as a finitely presented pregroup. Since the identity problem is known to be undecidable for finite presentations of groups, this implies that the identity problem is undecidable for pregroups in general. The identity problem is decidable, however, for certain restricted classes of pregroups. Below we describe algorithms for these classes.

### 1.4.1 An Algorithm for Free Pregroups

We say that a pregroup is *free* if its presentation has  $\mathcal{R} = \emptyset$ . The following is a consequence of a corollary of Lambek (1999), page 21:

**Proposition 2** *Let  $W$  be a nullable word in pregroup  $\mathcal{P} = \langle \mathcal{G}, \emptyset \rangle_{\mathcal{P}}$ . Then  $W \leq 1$  by a series of contractions.*

As an immediate consequence, the identity problem for free protogroups and pregroups are equivalent. In addition, a simple dynamic programming algorithm, similar to the CYK algorithm (Younger, 1967, Kasami, 1965) for parsing strings in a context-free grammar, can be employed

/	1	2	3	4	5	6	7	8
1	×	1	×	0	×	1	×	1
2	×	×	1	×	1	×	0	×
3	×	×	×	0	×	0	×	0
4	×	×	×	×	1	×	0	×
5	×	×	×	×	×	0	×	0
6	×	×	×	×	×	×	0	×
7	×	×	×	×	×	×	×	1
8	×	×	×	×	×	×	×	×

FIGURE 1 The table  $T$  corresponding to the word  $a^\ell aa^r a^\ell aaaa^r$ .

to determine whether  $W$  is nullable. Let  $\nu$  be a boolean predicate over words  $\mathcal{G}^*$  that holds exactly when a word is nullable. Let  $W = x_1 \dots x_n$  for  $x_k \in \mathcal{G}'$ . The following recurrence holds:

$$\nu(x_i \dots x_j) = 1 \text{ whenever } \begin{cases} x_i \in \text{lefts}(x_j) \text{ and } j = i + 1 \\ x_i \in \text{lefts}(x_j) \text{ and } \nu(x_{i+1} \dots x_{j-1}) = 1 \\ \nu(x_i \dots x_k) = 1 \text{ and } \nu(x_{k+1} \dots x_j) = 1 \\ \text{for some } k, \end{cases}$$

$$\nu(x_i \dots x_j) = 0 \text{ otherwise.}$$

In the above,  $\text{lefts}(a^{(i)})$  is just the singleton set  $\{a^{(i-1)}\}$ .

Briefly, to determine  $\nu(x_1 \dots x_n)$  the dynamic programming algorithm simply constructs a table  $T$  with entries

$$T[i, j] = \nu(x_i \dots x_j)$$

Entries  $T[i, j]$  where  $j - i = 1$  are determined first, followed by those where  $j - i = 3$ , then  $j - i = 5$ , and so on, ending with the determination of  $T[1, n]$ . The table's entries can be determined in  $O(n^3)$  time assuming that membership in  $\text{lefts}(x_j)$  can be determined in constant time. Figure 1 gives the table for the word  $a^\ell aa^r a^\ell aaaa^r$ .

#### 1.4.2 Allowing ambiguity: Oehrle's algorithm

In linguistic applications of pregroups it is common to assume that a lexical element may have more than one word associated with it. Oehrle (2004) gives a clever extension of the above algorithm, as a graph rewrite algorithm, that efficiently handles word assignment ambiguity. We give a brief review and reformulation of it here.

In Oehrle's algorithm, each lexical element  $e_i$  of a candidate sentence  $e_1 \dots e_m$  is assigned a directed acyclic graph fragment  $F_i$ . If  $e_i$  can be assigned the pregroup word  $W = x_1 \dots x_n$ , then the graph fragment  $F_i$  has a subfragment with  $n$  vertices  $v_1, \dots, v_n$  each labelled with letters

$x_1$  through  $x_n$ . A directed edge connects  $v_j$  with  $v_{j+1}$  in this subfragment.  $F_i$  is just a union of the subfragments of all words  $W$  that can be assigned to  $e_i$  (these are assumed to form a finite set).

Let  $s(F_i)$  be the set of vertices in  $F_i$  with no predecessors (incoming edges) and let  $t(F_i)$  be those with no successors (outgoing edges). A graph  $G$  is constructed from the candidate sentence by adding edges from all vertices of  $t(F_i)$  to all vertices of  $s(F_{i+1})$ , for all  $i$ . In addition, a distinguished vertex  $s$  is added with edges to  $s(F_1)$  and a distinguished vertex  $t$  is added with edges from  $t(F_m)$ .

The following graph modification algorithm is then applied to  $G$

```

Q := the edges of G
while Q is not empty do
  remove an edge (u, v) from Q
  if label(u) ∈ lefts(label(v)) then
    for each predecessor u' of u, successor v' of v do
      add edge (u', v') to G and Q
  
```

The candidate sentence is nullable if  $s$  and  $t$  are connected by an edge in the resulting graph.

### 1.4.3 Non-free Pregroups with Promotions

As we noted above, allowing arbitrary relations in  $\mathcal{R}$  makes the identity problem undecidable. We consider two restrictions on  $\mathcal{R}$ :

**Generator Promotions:** All relations in  $\mathcal{R}$  are of the form  $a \leq b$  where  $a, b \in \mathcal{G}$ , that is,  $\mathcal{R} \subset \mathcal{G} \times \mathcal{G}$ .<sup>1</sup> In this case, we can extend our rewrite system to include

$$\begin{array}{ccc} a^{(2k)} & \xrightarrow{\mathcal{R}} & b^{(2k)} \\ b^{(2k+1)} & \xrightarrow{\mathcal{R}} & a^{(2k+1)} \end{array}$$

for all  $(a, b) \in \mathcal{R}$  and  $k \in \mathcal{Z}$ .

**Letter Promotions:** All relations in  $\mathcal{R}$  are of the form  $a^{(i)} \leq b^{(j)}$  where  $a, b \in \mathcal{G}$ , that is,  $\mathcal{R} \subset \mathcal{G}' \times \mathcal{G}'$ . In this case, we can extend our rewrite system to include

$$\begin{array}{ccc} a^{(i+2k)} & \xrightarrow{\mathcal{R}} & b^{(j+2k)} \\ b^{(j+2k+1)} & \xrightarrow{\mathcal{R}} & a^{(i+2k+1)} \end{array}$$

for all  $(a^{(i)}, b^{(j)}) \in \mathcal{R}$  and  $k \in \mathcal{Z}$ .

Employing one of these rules in a rewrite step is called a *promotion* (note that Lambek calls Generator Promotions *induced steps*). The fol-

---

<sup>1</sup>Note that such presentations are called free pregroups by Lambek (1999).

lowing generalization of Corollary 1 of Lambek (1999), page 21 holds for these presentations:

**Theorem 3** *Let  $W$  be a nullable word in a pregroup  $\mathcal{P} = \langle \mathcal{G}, \mathcal{R} \rangle$  where  $\mathcal{R} \subset \mathcal{G}' \times \mathcal{G}'$ . Then  $W \leq 1$  by a series of contractions and promotions.*

*Proof.* Let  $W \leq UV \leq Ua^{(i+1)}a^{(i)}V \leq 1$  where the last inequality occurs only through contractions and promotions. We have two cases to consider:

**Case 1:**  $U \leq 1$  without expansions,  $a^{(i+1)} \leq b^{(j)}$  by promotions,  $V \leq 1$  without expansions, and  $a^{(i)} \leq b^{(j+1)}$  by promotions. In other words, the expansion introduces two letters that cancel after a series of promotions. In this case,  $UV \leq 1$  without expansions.

**Case 2:**  $U = U_1xU_2$ ,  $x \leq b^{(j)}$  by promotions,  $a^{(i+1)} \leq b^{(j+1)}$  by promotions,  $U_2 \leq 1$  without expansions,  $V = V_1yV_2$ ,  $y \leq c^{(k+1)}$  by promotions,  $a^{(i)} \leq c^{(k)}$  by promotions,  $V_1 \leq 1$  without expansions, and  $U_1V_2 \leq 1$  without expansions. In other words, the first inserted letter cancels with the a letter in the prefix  $U$  and the second inserted letter cancels a letter in the suffix  $V$ . In this case, note that,  $b^{(j)} \leq a^{(i)}$  and  $c^{(k+1)} \leq a^{(i+1)}$  by promotions, and so

$$xy \leq b^{(j)}y \leq a^{(i)}y \leq a^{(i)}c^{(k+1)} \leq c^{(k)}c^{(k+1)} \leq 1$$

by promotions and one contraction. Thus

$$UV = U_1xU_2V_1yV_2 \leq U_1xV_1yV_2 \leq U_1xyV_2 \leq U_1V_2 \leq 1$$

without expansions.

Since, in either case,  $UV \leq 1$ , we can always remove the last expansion to demonstrate the nullability of  $W$ , and the theorem follows.  $\square$

Given a presentation where  $\mathcal{R}$  has only generator promotions, we can compute the transitive closure  $\mathcal{R}^*$  of  $\mathcal{R}$ . Using  $\mathcal{R}^*$  we can employ the algorithms for free pregroups given above using

$$\text{lefts}(a^{(i)}) = \{b^{(i)} \mid (a, b) \in \mathcal{R}^*, b \in \mathcal{G}\}$$

for even  $i$  and

$$\text{lefts}(a^{(i)}) = \{b^{(i)} \mid (b, a) \in \mathcal{R}^*, b \in \mathcal{G}\}$$

for odd  $i$ .

A presentation with letter promotions, however, makes the identity problem NP-hard. Consider the following problem:

**The Pregroup Letter Promotion Problem**

**Given:** pregroup  $\mathcal{P} = \langle \mathcal{G}, \mathcal{R} \rangle_p$  where  $\mathcal{R} \subseteq \mathcal{G}' \times \mathcal{G}'$ ,  $x, y \in \mathcal{G}'$

**Determine:** whether  $x \leq y$  by promotions.

Suppose  $x = a^{(i)}$  and  $y = b^{(j)}$  for  $a, b \in \mathcal{G}$ . Note that  $x \leq y$  by promotions exactly when there exists a generator sequence  $a_0, a_1, \dots, a_n \in \mathcal{G}$  with  $a_0 = a$  and  $a_n = b$ , an integer sequences  $i_1, \dots, i_n$  and  $k_1, \dots, k_n$  where either

$$\left( a_{m-1}^{(2k_m)}, a_m^{(i_m+2k_m)} \right) \in \mathcal{R},$$

or

$$\left( a_m^{(i_m+2k_m+1)}, a_{m-1}^{(2k_m+1)} \right) \in \mathcal{R}$$

for each  $m$  with  $0 < m \leq n$ , and

$$j - i = \sum_{m=1}^n i_m$$

as this allows the series of promotions

$$a^{(i)} = a_0^{(i)} \leq a_1^{(i+i_1)} \leq a_2^{(i+i_1+i_2)} \leq \dots \leq a_n^{(i+i_1+i_2+\dots+i_n)} = b^{(j)}$$

that demonstrate that  $x \leq y$ . The summation requirement is the key to our NP-hardness proof.

**Theorem 4** *The Pregroup Letter Promotion Problem is NP-complete*

*Proof.* The following problem is NP-complete (see Garey and Johnson (1979)):

**The Subset Sum Problem**

**Given:** integer subset  $S$ , integer  $s$

**Determine:** the existence of an  $X \subseteq S$  with  $s = \sum_{x \in X} x$ .

An instance of the Subset Sum Problem with set  $S = \{t_1, t_2, \dots, t_n\}$  and target sum  $s$  can be reduced to the Letter Problem instance

$$\mathcal{G} = \{a_0, a_1, \dots, a_n\}$$

$$\mathcal{R} = \{(a_{i-1}, a_i) \mid 0 \leq i \leq n\} \cup \{(a_{i-1}, a_i^{(2t_i)}) \mid 0 \leq i \leq n\}$$

$$\begin{aligned} x &= a_0 \\ y &= a_n^{(2s)} \end{aligned}$$

It should be clear that  $x \leq y$  by promotions exactly when there exists a subset of  $S$  that sums to  $s$ . □

### 1.5 Further Questions

Though allowing general letter promotions make the word problem for pregroups NP-hard, there may be algorithms that work well in practice. For example, it is likely that any application of pregroups would make use of letter promotions of the form  $a^{(i)} \leq b^{(j)}$ , where  $|j - i| \leq M$ , where  $M$  is sufficiently small. In addition, it seems interesting to investigate what pregroup structures result from other natural classes of

presentation relations  $\mathcal{R}$  beyond those that allow just generator and letter promotions, and from an algorithmic perspective, to determine the complexity of the word problems for these classes.

## References

- Bargelli, D. and J. Lambek. 2001. An algebraic approach to French sentence structure. In P. de Groote et al., ed., *Logical Aspects of Computational Linguistics*, vol. 2099 of *Lecture Notes in Artificial Intelligence*, pages 62–78. Berlin: Springer-Verlag.
- Buszkowski, W. 2001. Lambek grammars based on pregroups. In P. de Groote et al., ed., *Logical Aspects of Computational Linguistics*, vol. 2099 of *Lecture Notes in Artificial Intelligence*, pages 95–109. Berlin: Springer-Verlag.
- Buszkowski, W. 2002. Pregroups: Models and grammars. In H. de Swart, ed., *Relational Methods in Computer Science*, vol. 2561 of *Lecture Notes in Computer Science*, pages 35–49. Berlin: Springer-Verlag.
- Buszkowski, W. 2003. Sequent systems for compact bilinear logic. *Mathematical Logic Quarterly* 49(5):467–474.
- Casadio, C. and J. Lambek. 2001. An algebraic analysis of clitic pronouns in Italian. In P. de Groote et al., ed., *Logical Aspects of Computational Linguistics*, vol. 2099 of *Lecture Notes in Artificial Intelligence*, pages 110–124. Berlin: Springer-Verlag.
- Cohen, D. E. 1989. *Combinatorial Group Theory: a topological approach*, vol. 14 of *London Mathematical Society Student Texts*. Cambridge: Cambridge University Press.
- Garey, M. R. and D.S. Johnson. 1979. *Computers and Intractability*. New York, NY: W.H. Freeman.
- Kasami, T. 1965. An efficient recognition and syntax-analysis algorithm for context-free languages. *Scientific Report AFCRL-65-758* .
- Lambek, J. 1995. Bilinear logic in algebra and linguistics. In J.-Y. Girard, Y. Lafont, and L. Regnier, eds., *Advances in Linear Logic*, London Mathematical Society Lecture Note Series, pages 43–60. Cambridge, UK: Cambridge University Press.
- Lambek, J. 1999. Type grammar revisited. In A. Lecomte, F. Lamarche, and G. Perrier, eds., *Logical Aspects of Computational Linguistics*, vol. 1582 of *Lecture Notes in Computer Science*, pages 1–27. Berlin: Springer-Verlag.

- Magnus, W., A. Karrass, and D. Solitar. 1966. *Combinatorial Group Theory: Presentations of Groups in Terms of Generators and Relations*, vol. XIII of *Pure and Applied Mathematics: A Series of Texts and Monographs*. New York: Interscience Publishers.
- Oerhle, R. 2004. A parsing algorithm for pregroup grammars. In *Categorical Grammars: An Efficient Tool for Natural Language Processing*.
- Younger, D. H. 1967. Recognition and parsing of context-free languages in time  $O(n^3)$ . *Information and Control* 10(2):609–617.