N. Francez
M. Kaminski

Commutation-Augmented Pregroup Grammars and Mildly Context-Sensitive Languages

Abstract. The paper presents a generalization of pregroup, by which a freely-generated pregroup is augmented with a finite set of *commuting inequations*, allowing limited commutativity and cancelability. It is shown that grammars based on the commutation-augmented pregroups generate *mildly context-sensitive* languages. A version of Lambek's *switching lemma* is established for these pregroups. Polynomial parsability and semi-linearity are shown for languages generated by these grammars.

Keywords: Formal language theory, pregroup grammars, mildly context-sensitive languages.

1. Introduction

Since their introduction in [4], *pregroup grammars* have attracted a lot of attention, giving rise to a *radically lexicalized* theory of formal (and, of course, natural) languages. The theory of formal languages partly developed from an abstraction originating in the syntax of natural languages, namely *constituency* (known also as *phrase-structure*). By this abstraction, *rewrite-rules* formed the basis of formal grammar, culminating in their classification by the well-known *Chomsky hierarchy.* To their success in computer science contributed the realization of their suitability for specifying the syntax of programming languages, after they were abandoned as a tool for natural language syntax specification. The theory matured even more when the grammar classification was complemented by the classification of various classes of automata corresponding to the various classes of the Chomsky hierarchy of grammar formalisms. See [3], a standard reference to the area.

This standard approach to formal languages has certain characteristics, challenged by modern *computational linguistics*, summarized below.

- Terminals are *atomic* structureless entities, that can only be compared for equality.

Special Issue Categorial Grammars and Pregroups
Edited by **Wojciech Buszkowski and Anne Preller**

Studia Logica (2007) 87: 295–321
DOI: 10.1007/s11225-007-9088-z

- Similarly, *non-terminals* (better called *categories*) are also atomic, structureless entities, representing sets of strings (of terminals).
- Language variation (over some fixed set of terminals) is determined by *grammar variation*, which was taken to mean *variation in the rewrite rules.*
- String *concatenation* is the *only* admissible syntactic operation.

Modern computational linguistics is the source of a different abstraction, based on a different view of language theory known as *radical lexicalism.* There are several radically-lexicalized linguistic theories for natural language (we omit references, as the focus here is on *formal* languages), having the following characteristics.

- Terminals are *informative* entities, that have their own properties, determined by a *lexicon*, mapping terminals to "pieces of information" about them. The lexicon is the "heart" of a grammar. Most often, those pieces of information are taken as (finite) sets of complex categories.
- Similarly, *categories* are also structured entities, representing sets of strings (of terminals).
- Language variation (over some fixed set of terminals) is determined by *lexicon variation.* There is a *universal grammar* (common to all languages) that extends the lexicon by attributing categories to strings too, controlling the combinatorics of strings based on their categories.

There are variants that admit other syntactic operations besides concatenation. We will assume here that concatenation is maintained as the only operation.

Buszkowski [1] establishes the weak generative equivalence between pregroup grammars and context-free grammars. On the other hand, motivated by the syntactic structure of natural languages, computational linguists became interested in a family of languages that became to be known as *mildly context-sensitive* languages, that on the one hand transcend context-free languages in containing *multiple agreement* ($\{a^n b^n c^n : n \geq 1\}$), *crossed dependencies* ($\{a^m b^n c^m d^n : m, n \geq 1\}$), and *reduplication* ($\{ww : w \in \Sigma^+\}$), but on the other hand have *semi-linearity* [6] and their recognition problem is decidable in polynomial time (in the length of the input word). Several formalisms for grammar specification are known to converge to the same class [8], namely to mildly context-sensitive languages.

In this paper, we explore *commutation-augmented pregroup grammars*, a mild extension of pregroup grammars, obtained by adding to the underlying (free) pregroup a finite number of *commuting and canceling* inequations, whose effect is *limited commutativity and cancelability* of the pregroup operation. We prove a kind of switching lemma for the extended pregroups. We show that the above mentioned "characteristic languages" of mild context-sensitivity are expressible in our commutation-augmented pregroup grammars. We establish the main properties of this class of languages, namely polynomial parsability and semi-linearity.

After reviewing the standard definition of pregroups and grammars based on them, we give *detailed* proofs for the examples of the characteristic context-sensitive languages, under an *informal* presentation of the extension via commutation and cancellation inequations. Then, we formally introduce the extension, and prove the main properties of the extended grammars. In those proofs, we provide the construction and formulate induction hypotheses, but skip much of the detailed verification of the induction step.

2. Preliminaries and notation

2.1. Pregroup grammars

We first define pregroup grammars.

DEFINITION 2.1. A *pregroup* is a tuple $\mathcal{G} = \langle \boldsymbol{G}, \leq, \circ, \ell, r, 1\rangle$, such that $\langle \boldsymbol{G}, \leq, \circ, 1\rangle$ is a *partially-ordered monoid*,[1] i.e., satisfying

(mon) if $A \leq B$, then $CA \leq CB$ and $AC \leq BC$

and ℓ,r are unary operations (left/right inverses/adjoints) satisfying

(pre) $A^{\ell}A \leq 1 \leq AA^{\ell}$ and $AA^{r} \leq 1 \leq A^{r}A$

The following equalities can be shown to hold in any pregroup.

$$1^{\ell} = 1^{r} = 1,\ A^{\ell r} = A^{r\ell} = A,\ (AB)^{\ell} = B^{\ell}A^{\ell},\ (AB)^{r} = B^{r}A^{r}$$

Also, **(mon)** together with **(pre)** yield

$$A \leq B \text{ if and only if } B^{\ell} \leq A^{\ell} \text{ if and only if } B^{r} \leq A^{r} \tag{1}$$

Let $\langle \mathcal{B},\ \leq\rangle$ be a (finite) poset. *Terms* are of the form $A^{(n)}$, where $A \in \mathcal{B}$ and n is an integer. The set of all terms generated by $\mathcal{B}$ is denoted by $\tau(\mathcal{B})$, and for a set of terms T and an integer n we define the set of terms $T^{(n)}$ by

$$T^{(n)} = \{A^{(m+n)} : A^{m} \in T\}$$

[1]'$\circ$' is usually omitted.

Categories[2] are finite strings (products) of terms. The set of all categories generated by $\mathcal{B}$ is denoted by $\kappa(\mathcal{B})$.

Extend '$\leq$' to $\kappa(\mathcal{B})$ by setting it to the smallest quasi-partial-order[3] satisfying

(con) $\gamma A^{(n)} A^{(n+1)} \delta \leq \gamma\delta$ (*contraction*)

(exp) $\gamma\delta \leq \gamma A^{(n+1)} A^{(n)} \delta$ (*expansion*)

and

(ind) $\gamma A^{(n)} \delta \leq \gamma B^{(n)} \delta$ if $\begin{cases} A \leq B \text{ and } n \text{ is even, or} \\ B \leq A \text{ and } n \text{ is odd} \end{cases}$ (*induced steps*)

The following two inequalities can be easily derived from **(con)**, **(exp)**, and **(ind)**.

(gcon) $\gamma A^{(n)} B^{(n+1)} \delta \leq \gamma\delta$, if $\begin{cases} A \leq B \text{ and } n \text{ is even, or} \\ B \leq A \text{ and } n \text{ is odd} \end{cases}$ (*generalized contraction*)

and

(gexp) $\gamma\delta \leq \gamma A^{(n+1)} B^{(n)} \delta$, if $\begin{cases} A \leq B \text{ and } n \text{ is even, or} \\ B \leq A \text{ and } n \text{ is odd} \end{cases}$ (*generalized expansion*)

Obviously, **(con)** and **(exp)** are particular cases of **(gcon)** and **(gexp)**, respectively. Conversely, **(gcon)** can be obtained as **(ind)** followed by **(con)**, and **(gexp)** can be obtained as **(exp)** followed by **(ind)**. Consequently, if $\alpha' \leq \alpha''$, then there exists a *derivation*

$$\alpha' = \gamma_0 \leq \gamma_1 \leq \cdots \leq \gamma_m = \alpha'', \quad m \geq 0$$

such that for each $i = 1, 2, \ldots, m$, $\gamma_{i-1} \leq \gamma_i$ is **(gcon)**, **(gexp)**, or **(ind)**.

PROPOSITION 2.2. ([4, Proposition 2]) *If $\alpha' \leq \alpha''$ has a derivation of length m, then there exist types β and γ such that*

- *$\alpha' \leq \beta$ by* **(gcon)** *only;*
- *$\beta \leq \gamma$ by* **(ind)** *only;*
- *$\gamma \leq \alpha''$ by* **(gexp)** *only; and*
- *the sum of the lengths of the above three derivations is at most m.*

COROLLARY 2.3. *If $\alpha' \leq \alpha''$ where α'' is a term, then, effectively, this can be established without expansions.*

---

[2] They are also called *types*.

[3] That is, $\leq$ is not necessarily antisymmetrical.

$G_a = \langle \Sigma, \mathcal{B}, =, I, S \rangle$, where

$$\Sigma = \{a, b\}, \quad \mathcal{B} = \{S, A\}$$

$$I(a) = \{SA^\ell S^\ell, SA^\ell\}, \quad I(b) = \{A\}$$

Figure 1. A pregroup grammar for $\{a^n b^n : n \geq 1\}$

Let $\alpha' \equiv \alpha''$ if and only if $\alpha' \leq \alpha''$ and $\alpha'' \leq \alpha'$. The equivalence-classes of '$\equiv$' form the *free pregroup generated by* $\langle \mathcal{B}, \leq \rangle$, where $1 = [\epsilon]_\equiv$, $[\alpha']_\equiv \circ [\alpha'']_\equiv = [\alpha'\alpha'']_\equiv$; also, $[\alpha']_\equiv \leq [\alpha'']_\equiv$ if and only if $\alpha' \leq \alpha''$. The adjoints are defined as follows.

$$[A_1^{(n_1)} \cdots A_l^{(n_l)}]^\ell = [A_l^{(n_l-1)} \cdots A_1^{(n_1-1)}]$$

and

$$[A_1^{(n_1)} \cdots A_l^{(n_l)}]^r = [A_l^{(n_l+1)} \cdots A_1^{(n_1+1)}]$$

Definition 2.4. *A pregroup grammar* (PGG) is a tuple $G = \langle \Sigma, \mathcal{B}, \leq, I, S \rangle$, where Σ is a finite set of terminals (the *alphabet*), $\langle \mathcal{B}, \leq \rangle$ is a finite poset of *atoms*, I is a finite-range mapping $I : \Sigma \to 2^{\kappa(\mathcal{B})}$,[4] and $S \in \tau(\mathcal{B})$ is a *distinguished term*.

The language generated by G is defined by

$$L(G) = \{c_1 \cdots c_l : \text{there exist } \alpha_1, \ldots, \alpha_l \text{ such that } \alpha_i \in I(c_i), i \leq l, \text{ and } \alpha_1 \cdots \alpha_l \leq S\}$$

In Figure 1 is an example of a PGG for $L = \{a^n b^n : n \geq 1\}$. Below is a derivation for $a^3 b^3 \in L(G_a)$. The lexical category assignment chosen is

$$\overbrace{SA^\ell S^\ell}^{a}\,\overbrace{SA^\ell S^\ell}^{a}\,\overbrace{SA^\ell}^{a}\,\overbrace{A}^{b}\,\overbrace{A}^{b}\,\overbrace{A}^{b}$$

and cancellation is indicated by underline.

$$SA^\ell \underline{S^\ell S} A^\ell \underline{S^\ell S} A^\ell AAA \leq SA^\ell A^\ell \underline{A^\ell A} AA \leq SA^\ell \underline{A^\ell A} A \leq S\underline{A^\ell A} \leq S$$

Theorem 2.5. ([1]) *An ϵ-free language L is $L(G)$ for some pregroup grammar G if and only if L is context-free.*

Clearly it suffices to use types only of the form A, A^ℓ, for $A \in \mathcal{B}$.

[4] That is, $I(c)$ is finite for all $c \in \Sigma$.

$G_{ma} = \langle \Sigma, \mathcal{B}, =, I, S, \iota \rangle$, where

$$\Sigma = \{a, b, c\}, \quad \mathcal{B} = \{S, T, U, A, B\}$$

$$I(a) = \{SA^\ell S^\ell, SA^\ell T^\ell\},\ I(b) = \{TB^\ell T^\ell, TB^\ell U^\ell\},\ I(c) = \{UABU^\ell, UAB\}$$

$$\iota = \{B^\ell A \le AB^\ell\}$$

Figure 2. A commutation-augmented pregroup grammar for multiple agreement

3. Expressing Characteristic Mildly Context-Sensitive Languages by Commutation-Augmented Pregroup Grammars

In this section, we present examples, even before the formal definition, of grammars recognizing the characteristic mildly context-sensitive languages. The central idea is to add to the *free* pregroup underlying a grammar a finite set ι of *commuting and canceling inequations* between types. These inequations allow "movement" within a string of terms, so that "remote" types can still mutually cancel each other, by one of them "moving" left until it reaches its counterpart, and then a cancellation takes place. The examples below are based on different commuting and canceling inequations between some of the pregroup elements. Later, we present more rigorously the exact details of the extension, using commuting inequations of some specific form. However, to show the gist of the approach, we chose to use the simplest inequations fitting a given example, leaving uniformity to the formal presentation. Commuting is indicated by overline.

3.1. Multiple agreement

Let $\Sigma = \{a, b, c\}$, and $L_{ma} = \{a^n b^n c^n : n \ge 1\}$. Consider the grammar G_{ma} presented in Figure 2. Before proving the correctness of G_{ma}, below is a derivation for $aabbcc \in L_{ma}$. The lexical type-assignment is

$$\overbrace{SA^\ell S^\ell}^{a}\, \overbrace{SA^\ell T^\ell}^{a}\, \overbrace{TB^\ell T^\ell}^{b}\, \overbrace{TB^\ell U^\ell}^{b}\, \overbrace{UABU^\ell}^{c}\, \overbrace{UAB}^{c}$$

and the derivation is

$$SA^\ell \underline{S^\ell S} A^\ell \underline{T^\ell T} B^\ell \underline{T^\ell T} B^\ell \underline{U^\ell U} AB \underline{U^\ell U} AB \le SA^\ell A^\ell \overline{B^\ell B^\ell A} BAB$$

$$\le^{\iota \times 2} SA^\ell \underline{A^\ell A} B^\ell \underline{B^\ell B} AB \le SA^\ell \overline{B^\ell A} B \le^{\iota} S\underline{A^\ell AB^\ell B} \le S$$

Claim 3.1. $L(G_{ma}) = L_{ma}$.

3.1.1. Proof of the inclusion $L_{ma} \subseteq L(G_{ma})$

We start with a number of auxiliary inequations.

$$(SA^\ell S^\ell)^n \leq S(A^\ell)^n S^\ell \tag{2}$$
$$(TB^\ell T^\ell)^n \leq T(B^\ell)^n T^\ell \tag{3}$$
$$(UABU^\ell)^n \leq U(AB)^n U^\ell \tag{4}$$
$$(B^\ell)^n A \leq A(B^\ell)^n \tag{5}$$
$$\xi_n = (A^\ell)^n (B^\ell)^n (AB)^n \leq 1 \tag{6}$$

We show a detailed proof only for the last two inequations, the other being trivial.

Proof of (5). The proof is by induction on n.

Basis. $B^\ell A \leq AB^\ell$ by the inequation in ι.

Induction step.

$$(B^\ell)^{n+1} A = (B^\ell)^n \overline{B^\ell A} \leq^\iota (B^\ell)^n AB^\ell \leq^{\text{ind. hyp.}} A(B^\ell)^n B^\ell = A(B^\ell)^{n+1}$$

■

Proof of (6). The proof is by induction on n.

Basis. $\xi_1 = A^\ell \overline{B^\ell A} B \leq^\iota \underline{A^\ell A B^\ell B} \leq 1$.

Induction step.

$$\xi_{n+1} = (A^\ell)^{n+1} (B^\ell)^{n+1} (AB)^{n+1} = (A^\ell)^n A^\ell (B^\ell)^n \overline{B^\ell A} B (AB)^n$$

$$\leq^\iota (A^\ell)^n A^\ell (B^\ell)^n AB^\ell B (AB)^n \leq^{(5)} (A^\ell)^n \underline{A^\ell A} (B^\ell)^n \underline{B^\ell B} (AB)^n$$

$$\leq \xi_n \leq^{\text{ind. hyp.}} 1$$

■

Now for the type assignment

$$(SA^{\ell}S^{\ell})^{n-1}SA^{\ell}T^{\ell}(TB^{\ell}T^{\ell})^{n-1}TB^{\ell}U^{\ell}(UABU^{\ell})^{n-1}UAB \in I(a^n b^n c^n)$$

we have

$$\begin{gathered}(SA^{\ell}S^{\ell})^{n-1}SA^{\ell}T^{\ell}(TB^{\ell}T^{\ell})^{n-1}TB^{\ell}U^{\ell}(UABU^{\ell})^{n-1}UAB \\ \leq^{(2),(3),(4)} S\xi_n \leq^{(6)} S\end{gathered}$$

3.1.2. Proof of the inclusion $L(G_{ma}) \subseteq L_{ma}$

Consider any $w \in \Sigma^+$ such that for some $\xi \in I(w)$ there is a derivation establishing $\xi \leq S$. It follows from Proposition 2.2 that after replacing all As and Bs in ξ with 1 (which, obviously, preserves the inequation in ι),[5] we obtain

$$S(S^{\ell}S)\cdots(S^{\ell}S)(T^{\ell}T)\cdots(T^{\ell}T)(U^{\ell}U)\cdots(U^{\ell}U)$$

Thus, $w \in a^+b^+c^+$. The argument for the equal number of occurrences is simple too. The only way to cancel A^{ℓ} in the type of a is to have a matching A to its right. Note that the inequation does not change the multiplicity of basic types. Hence, the number of A^{ℓ}'s must be equal to the number of A's. This implies that the number of a's equals the number of b's in w. A similar argument regarding B (with B^{ℓ} in the type of b and B in the type of c) implies that the number of b's in w equals the number of c's. Altogether, $w \in L_{ma}$.

3.2. Crossed dependencies

Let $\Sigma = \{a, b, c, d\}$, and $L_{cd} = \{a^m b^n c^m d^n : m, n \geq 1\}$. Consider the grammar G_{cd} in Figure 3.

Again, we present a derivation for $aabccd \in L_{cd}$ before the general correctness proof. The lexical type-assignment is

$$\overbrace{SA^{\ell}S^{\ell}}^{a}\,\overbrace{SA^{\ell}T^{\ell}}^{a}\,\overbrace{TB^{\ell}U^{\ell}}^{b}\,\overbrace{UAU^{\ell}}^{c}\,\overbrace{UAV^{\ell}}^{c}\,\overbrace{VB}^{d}$$

and the first cancellations are

$$SA^{\ell}\underline{S^{\ell}S}A^{\ell}\underline{T^{\ell}T}B^{\ell}\underline{U^{\ell}U}A\underline{U^{\ell}U}A\underline{V^{\ell}V}B$$

yielding $SA^{\ell}A^{\ell}B^{\ell}AAB$. Next, two applications of ι yield $SA^{\ell}A^{\ell}AAB^{\ell}B$ which cancels to S.

[5] Actually, this is a pregroup homomorphism.

$G_{cd} = \langle \Sigma, \mathcal{B}, =, I, S, \iota \rangle$, where

$$\Sigma = \{a, b, c, d\}, \quad \mathcal{B} = \{S, T, U, V, A, B\}$$

$$I(a) = \{SA^\ell S^\ell, SA^\ell T^\ell\}, \quad I(b) = \{TB^\ell T^\ell, TB^\ell U^\ell\},$$
$$I(c) = \{UAU^\ell, UAV^\ell\}, \quad I(d) = \{VBV^\ell, VB\},$$

$$\iota = \{B^\ell A \leq AB^\ell\}$$

Figure 3. A commutation-augmented pregroup grammar for crossed dependencies

CLAIM 3.2. $L(G_{cd}) = L_{cd}$.

PROOF. The proof is similar to that of Claim 3.1.
$L_{cd} \subseteq L(G_{cd})$: Like in the corresponding case of the proof of Claim 3.1, we start with a number of auxiliary inequations.

$$(UAU^\ell)^n \leq UA^nU^\ell \tag{7}$$
$$(VBV^\ell)^n \leq VB^nV^\ell \tag{8}$$
$$\xi_{m,n} = (A^\ell)^m(B^\ell)^nA^mB^n \leq 1 \tag{9}$$

We show a detailed proof only for the last inequation, the first two being trivial. The proof of (9) is similar to that of (5) and is by induction on m.

Basis. $\xi_{1,n} = A^\ell(B^\ell)^nAB^n \leq^{(5)} A^\ell A(B^\ell)^nB^n \leq 1.$[6]

Induction step.

$$\xi_{m+1,n} = (A^\ell)^{m+1}(B^\ell)^nA^{m+1}B^n = (A^\ell)^mA^\ell(B^\ell)^nAA^mB^n$$
$$\leq^{(5)} (A^\ell)^m\underline{A^\ell A}(B^\ell)^nA^mB^n \leq \xi_{m,n} \leq^{\text{ind. hyp.}} 1$$

We now turn to the claim itself. Let $w = a^mb^nc^md^n$. We show in detail the case $n, m > 1$, the other cases being similar. For the type assignment

$$(SA^\ell S^\ell)^{n-1}SA^\ell T^\ell(TB^\ell T^\ell)^{m-1}TB^\ell U^\ell(UAU^\ell)^{n-1}UAV^\ell(VBV^l)^{m-1}VB \in I(w)$$

[6] We may use (5), because both G_{ma} and G_{rd} are based on the same inequation.

$G_{rd} = \langle \Sigma, \mathcal{B}, =, I, Z, \iota \rangle$, where

$$\Sigma = \{a, b\}, \quad \mathcal{B} = \{S, T, U, A', A'', B', B''\}$$

$$I(a) = \{ZA'U^\ell, ZA'V^\ell, UA'U^\ell, UA'V^\ell, VA''V^\ell, VA'',\}$$

$$I(b) = \{ZB'U^\ell, ZB'V^\ell, UB'U^\ell, UB'V^\ell, VB''V^\ell, VB''\}$$

For the inequations, let X and Y range over $\{A, B\}$.

$$\iota = \{(1)\ X'Y'' \leq Y''X', \quad (2)\ ZX'X'' \leq Z\}$$

Figure 4. A commutation-augmented pregroup grammar for reduplication

we have

$$(SA^\ell S^\ell)^{n-1} SA^\ell T^\ell (TB^\ell T^\ell)^{m-1} TB^\ell U^\ell (UAU^\ell)^{n-1} UAV^\ell (VBV^l)^{m-1} VB$$
$$\leq^{(2),(3),(7),(8)}$$
$$S(A^\ell)^{n-1} \underline{S^\ell S} A^\ell \underline{T^\ell T} (B^\ell)^{m-1} \underline{T^\ell T} B^\ell \underline{U^\ell U} A^{n-1} \underline{U^\ell U} A \underline{V^\ell V} B^{m-1} \underline{V^l V} B$$
$$\leq S\xi_{m,n} \leq^{(9)} S$$

Therefore, $w \in L(G_{cd})$.

$L(G_{cd}) \subseteq L_{cd}$: Suppose that for some $\xi \in I(w)$ we have $\xi \leq S$. Similarly to the proof of the corresponding inclusion of Claim 3.1, one can show that $(G_{cd}) \subseteq a^+b^+c^+d^+$. Also, a similar argument shows that $\#_{A^\ell}(\xi) = \#_A(\xi)$ and $\#_{B^\ell}(\xi) = \#_B(\xi)$, establishing $\#_a(w) = \#_c(w)$ and $\#_b(w) = \#_d(w)$, i.e., $w \in L_{cd}$. ■

3.3. Reduplication

Let $\Sigma = \{a, b\}$ and let $L_{rd} = \{ww : w \in \Sigma^+\}$. A commutation-augmented pregroup grammar G_{rd} for L_{rd} is presented in Figure 4. In some sense, this example is generic, in that its grammar is the one obtained by the general construction (cf. Definition 4.8 in the next section).

Claim 3.3. $L_{rd} = L(G_{rd})$.

3.3.1. Proof of the inclusion $L_{rd} \subseteq L(G_{rd})$

Let $x = ww \in L_{rd}$. The proof is by induction on $n = |w|$. To formulate the induction hypotheses, some notation is needed. For a letter $c \in \Sigma$, we denote by $\widehat{c}$ the capital case of c.[7] For $c \in \Sigma$ we define $\gamma^1(c) = \widehat{c}\,'$ and $\gamma^2(c) = \widehat{c}\,''$. Then we naturally extend γ^1, γ^2 to words, by

$$\gamma^i(cu) = \gamma^i(c)\gamma^i(u),\ i = 1, 2,\ c \in \Sigma$$

For $|x| > 2$, the following lexical type-assignment is used. We associate $\xi^1_w \in I(w)$ with the first w in x, and $\xi^2_w \in I(w)$ with the second w in x. The type assignments in ξ^1_w and ξ^2_w are defined as follows.

Let $w = c_1 c_2 \cdots c_l$. The type assignment in ξ^1_w is defined by

- c_1 is assigned $Z\widehat{c_1}'U^\ell$;
- c_i, $i = 2, 3, \ldots, l-1$, is assigned $U\widehat{c_i}'U^\ell$; and
- c_l is assigned $U\widehat{c_l}'V^\ell$.

Similarly, the type assignment in ξ^2_w is defined by

- c_i, $i = 1, 2, \ldots, l-1$, is assigned $V\widehat{c_i}''V^\ell$; and
- c_l is assigned $V\widehat{c_l}''$.

In addition, for $x = cc$, i.e., $w = c$ and $|x| = 2$, ξ^1_c is $Z\widehat{c}'V^\ell$, and ξ^2_c is $V\widehat{c}''$.

Finally, we define the category $\widehat{\xi^1_w}$ by $\xi^1_w = Z\widehat{\xi^1_w}$.

We shall prove the following inequations.

$$\gamma^1(w)X'' \leq X''\gamma^1(w),\ X \in \{A, B\} \tag{10}$$

$$\widehat{\xi^1_w} \leq \gamma^1(w)V^\ell \text{ and } \xi^2_w \leq V\gamma^2(w) \tag{11}$$

$$Z\gamma^1(w)\gamma^2(w) \leq Z \tag{12}$$

Proof of (10). The proof is by induction on $|w|$.

Basis. Let $c \in \{a, b\}$ and let $X \in \{A, B\}$. Then

$$\gamma^1(c)X'' = \widehat{c}\,'X'' \leq^{\iota(1)} X''\widehat{c}\,' = X''\gamma^1(c)$$

[7] That is, $\widehat{a}$ is A and $\widehat{b}$ is B.

Induction step.

$$\gamma^1(cw)X'' = \gamma^1(c)\gamma^1(w)X'' \leq^{\text{ind .hyp.}} \gamma^1(c)X''\gamma^1(w)$$

$$\leq^{\iota(1)} X''\gamma^1(c)\gamma^1(w) = X''\gamma^1(cw)$$

■

Proof of (11). Again, the proof is by induction on $|w|$.

Basis. Follows from the definition of $\widehat{\xi^1_c}$, ξ^2_c, γ^1, and γ^2.

Induction step.

$$\widehat{\xi^1_{cw}} = \gamma^1(c)\underline{U^\ell U}\widehat{\xi^1_w} \leq^{\text{ind. hyp}} \gamma^1(c)\gamma^1(w)V^\ell = \gamma^1(cw)V^\ell$$

The case for ξ^2 is similar. ■

Proof of (12). As above, we proceed by induction on $|w|$.

Basis. $Z\widehat{c}'\widehat{c}'' \leq^{\iota(1)} Z$.

Induction step.

$$Z\gamma^1(cw)\gamma^2(cw) = Z\widehat{c}\,'\gamma^1(w)\widehat{c}''\gamma^2(w)$$

$$\leq^{(10)} Z\widehat{c}'\widehat{c}''\gamma^1(w)\gamma^2(w) \leq^{\iota(2)} Z\gamma^1(w)\gamma^2(w) \leq^{\text{ind. hyp.}} Z$$

■

We now show that $\xi^1_w\xi^2_w \leq Z$, implying $ww \in L(R_{rd})$.

$$\xi^1_w\xi^2_w = Z\widehat{\xi^1_w}\xi^2_w \leq^{(11)} Z\gamma^1(w)\underline{V^\ell V}\gamma^2(w) \leq Z\gamma^1(w)\gamma^2(w) \leq^{(12)} Z$$

3.3.2. Proof of the inclusion $L(G_{rd}) \subseteq L_{rd}$

Assume that $u \in L(G_{rd})$ and fix an appropriate type assignment for the symbols in u. Like in the proof of the corresponding inclusion of Claim 3.1, replacing in this type assignment all As and Bs with 1 we obtain

$$ZU^\ell U \cdots U^\ell UV^\ell V \cdots V^\ell V$$

For this to cancel out to Z, we have that $u = vw$, where the type assignment to the symbols in v and w are, respectively, ξ_v^1 and ξ_w^2 from the proof of the inclusion $L_{rd} \subseteq L(G_{rd})$.

Let $v = c_1 c_2 \cdots c_l \in \Sigma^*$ and $w = d_1 d_2 \cdots d_m \in \Sigma^*$, $l + m \geq 1$.

Replacing (in the original type-assignment) all Us and Vs with 1, we obtain

$$Z \widehat{c_1}' \widehat{c_2}' \cdots \widehat{c_l}' \, \widehat{d_1}'' \widehat{d_2}'' \cdots \widehat{d_m}'' \leq Z$$

Thus, the desired inclusion will follow when we show that $l = m$ and $c_i = d_i$, $i = 1, 2, \ldots, m$. The latter is a particular case of Corollary 4.7 in the next section.

4. Augmenting Pregroup Grammars with Commutations

This section deals with properties of pregroup grammars based on the extension of the underlying free pregroup with commuting and canceling *inequations* of the forms **(m)** and **(c)** below.

Definition 4.1. Let $\langle \mathcal{B}_1, \leq_1 \rangle$ and $\mathcal{B}_2$ be a finite poset and a finite set, respectively, such that $\mathcal{B}_1 \cap \mathcal{B}_2 = \emptyset$, and let Z, U, and V be three new *distinguished* elements. Let $C' = \langle A_1', A_2', \ldots, A_k' \rangle \in (\tau(\mathcal{B}_2))^k$ and $C'' = \langle A_1'', A_2'', \ldots, A_k'' \rangle \in (\tau(\mathcal{B}_2))^k$ be k-tuples of $\mathcal{B}_2$-terms. We denote by $\mathcal{G}(\mathcal{B}_1, \leq_1, \mathcal{B}_2, C', C'', Z, U, V)$ the pregroup that is obtained from the poset $\langle \mathcal{B}_1 \cup \mathcal{B}_2 \cup \{Z, U, V\}, \leq_1 \cup = \rangle$ by imposing, in addition to **(con)**, **(ind)**, and **(exp)**, the following relations **(m)** (for *move*) and **(c)** (for *cancel*), cf. the set of inequations ι in Figure 4.

(m) $BA \leq AB$, $B \in \tau(\mathcal{B}_1) \cup C'$ and $A \in C''$ (recall that $\tau(\mathcal{B}_1)$ is the set of all terms generated by $\mathcal{B}_1$)

and

(c) $ZA_i'A_i'' \leq Z$, $i = 1, 2, \ldots, k$

and extending **(ind)** with the following *induced* relations.

$(\mathbf{ind_m^n})$ $\gamma B^{(n)} A^{(n)} \delta \leq \gamma A^{(n)} B^{(n)} \delta$, $B \in \tau(\mathcal{B}_1) \cup C'$ and $A \in C''$

and

$$(\mathbf{ind_c^n}) \quad \begin{cases} \gamma Z^{(n)} A_i'^{(n)} A_i''^{(n)} \delta \leq \gamma Z^{(n)} \delta, & \text{if } n \text{ is even, or} \\ \gamma Z^{(n)} \delta \leq \gamma A_i''^{(n)} A_i'^{(n)} Z \delta, & \text{if } n \text{ is odd} \end{cases}, \quad i = 1, 2, \ldots, k$$

REMARK 4.2. Note that it follows from **(m)** that all elements of C'' can be moved (in a number of steps) through *any* category in $\kappa(\mathcal{B}_1)$. Consequently, by $(\mathbf{ind}^n_{\mathbf{m}})$s, all elements of

$$\{A_i''^{(n)} : i = 1, 2, \ldots, k,\ n = 0, \pm 1, \pm 2, \ldots\}$$

can be moved through *any* category in $\kappa(\mathcal{B}_1)$ as well.

The above pregroup $\mathcal{G}(\mathcal{B}_1, \leq_1, \mathcal{B}_2, C', C'', Z, U, V)$ is called a *restricted commutation-augmented pregroup* (RCAPG).[8]

Next we show that **(m)** is equivalent to

$(\mathbf{m}')$ $\quad A^\ell B \leq BA^\ell, \quad B \in \tau(\mathcal{B}_1) \cup C'$ and $A \in C''$

That is, **(m)** implies $(\mathbf{m}')$ and vice versa. Indeed, we have

$$A^\ell B \leq^{(\mathbf{exp})} A^\ell BAA^\ell \leq^{(\mathbf{ind}^{\mathbf{0}}_{\mathbf{m}})} A^\ell ABA^\ell \leq^{(\mathbf{con})} BA^\ell$$

and

$$BA \leq^{(\mathbf{exp})} AA^\ell BA \leq^{(\mathbf{ind}^{\mathbf{0}}_{\mathbf{m}'})} ABA^\ell A \leq^{(\mathbf{con})} AB$$

where

$(\mathbf{ind}^n_{\mathbf{m}'})$ $\quad \gamma A^{(n-1)} B^{(n)} \delta \leq \gamma B^{(n)} A^{(n-1)} \delta, \quad B \in \tau(\mathcal{B}_1) \cup C'$ and $A \in C''$

Thus, we shall use$(\mathbf{ind}^n_{\mathbf{m}'})$ in addition to $(\mathbf{ind}^n_{\mathbf{m}})$s.

Proposition 4.3 below, whose long and technical proof is presented in the next section, is the RCAPG counterpart of Proposition 2.2.

PROPOSITION 4.3. (Cf. Proposition 2.2.) *If $\alpha' \leq \alpha''$ has a derivation of length m, then it has a derivation of length at most m in which no term introduced by* **(exp)** *is later canceled by* **(con)**.[9]

The following example shows that not always can we precede *all* **(exp)**s by **(con)**s. However, it easily follows from the proof of the proposition, that we can do that, if α'' contains no Z.

EXAMPLE 4.4. Let $C' = \langle A' \rangle$ and $C'' = \langle A'' \rangle$. Then

$$Z^\ell ZA' \leq^{(\mathbf{exp})} Z^\ell ZA'A''A''^\ell \leq^{(\mathbf{ind}^{\mathbf{0}}_{\mathbf{c}})} Z^\ell ZA''^\ell \leq^{(\mathbf{con})} A''^\ell \tag{13}$$

However, there is no derivation of (13) without **(con)**s after the **(exp)**.

---

[8]The reason for 'restricted' is the correspondence, reported in [2], with a certain *restricted canceling pushdown automaton*. More general notions of commutation-augmented pregroup grammars, as well as canceling PDAs, will be reported elsewhere.

[9]Recall that $(\mathbf{ind}^n_{\mathbf{m}'})$s are allowed as well.

COROLLARY 4.5. (Cf. Corollary 2.3.) *Let*

$$\alpha \leq 1 \tag{14}$$

If $\alpha \in \kappa(\mathcal{B}_2)$, then, effectively, (14) *can be established without expansions.*

PROOF. Substituting 1 for all elements of $\mathcal{B}_1 \cup \mathcal{B}_2$ (which, obviously, preserves all **(m)**, **(m′)**, and **(c)**) in a derivation of (14), we see no derivation of (14) contains Z. Therefore, each term introduced by **(exp)** must be later canceled by **(con)**, and the proof follows from Proposition 4.3. ∎

For what follows we shall need one more corollary to Proposition 4.3. This corollary involves the following definition.

DEFINITION 4.6. The tuples $C' = \langle A'_1, A'_2, \ldots, A'_k\rangle \in (\tau(\mathcal{B}_2))^k$ and $C'' = \langle A''_1, A''_2, \ldots, A''_k\rangle \in (\tau(\mathcal{B}_2))^k$ are called *unrelated* if

- for all $A, B \in C' \cup C''$ such that $A \neq B$, $A \notin \{B^\ell, B, B^r\}$.[10]

COROLLARY 4.7. *Let $\mathcal{G}(\mathcal{B}_1, \leq_1, \mathcal{B}_2, C', C'', Z, U, V)$ be an RCAPG such that C' and C'' are unrelated and let $A'_{i_1}, A'_{i_2}, \ldots, A'_{i_l} \in C'$ and $A''_{j_1}, A''_{j_2}, \ldots, A''_{j_m} \in C''$ be such that*

$$ZA'_{i_1}A'_{i_2}\cdots A'_{i_l}A''_{j_1}A''_{j_2}\cdots A''_{j_m} \leq Z \tag{15}$$

Then $l = m$ and $i_h = j_h$, $h = 1, 2, \ldots, m$.

PROOF. It follows from (15) that for each application of **(exp)** of the form $\gamma Z^{(n)}Z^{(n+1)}\delta$ at least one of $Z^{(n+1)}$ or $Z^{(n)}$ is canceled later. Since it can be canceled by **(con)** only, by Proposition 4.3, we may assume that (15) is derived without **(exp)**s of the form $\gamma Z^{(n+1)}Z^{(n)}\delta$. Therefore, we may assume that in (15) all applications of **($\mathbf{ind}^0_\mathbf{c}$)** are delayed to the end. That is,

$$ZA'_{i_1}A'_{i_2}\cdots A'_{i_l}A''_{j_1}A''_{j_2}\cdots A''_{j_m} \leq ZA'_{g_1}A''_{g_1}A'_{g_2}A''_{g_2}\cdots A'_{g_n}A''_{g_n} \leq Z$$

Substituting 1 for Z we obtain

$$A'_{i_1}A'_{i_2}\cdots A'_{i_l}A''_{j_1}A''_{j_2}\cdots A''_{j_m} \leq A'_{g_1}A''_{g_1}A'_{g_2}A''_{g_2}\cdots A'_{g_n}A''_{g_n}$$

Therefore,

$$A''^{\ell}_{g_n}A'^{\ell}_{g_n}A''^{\ell}_{g_{n-1}}A'^{\ell}_{g_{n-1}}\cdots A''^{\ell}_{g_1}A'^{\ell}_{g_1}A'_{i_1}A'_{i_2}\cdots A'_{i_l}A''_{j_1}A''_{j_2}\cdots A''_{j_m} \leq 1 \tag{16}$$

[10] Of course, by $A \in C'$ (respectively, $A \in C''$) we mean that for some $i = 1, 2, \ldots, k$, $A = A'_i$ (respectively, $A = A''_i$).

By Corollary 4.5, (16) can be derived by means of **(con)**s, $(\mathbf{ind}^0_{\mathbf{m}})$s, and $(\mathbf{ind}^0_{\mathbf{m}'})$s, only.

Since C' and C'' are unrelated, the elements of C' (respectively, C'') can be canceled only by the corresponding elements of ${C'}^{\ell}$ (respectively, ${C''}^{\ell}$), and neither of $(\mathbf{ind_m})$ or $(\mathbf{ind'_m})$ can change the order of the terms from C', ${C'}^{\ell}$, C'' or ${C''}^{\ell}$ in the left-hand side of (16). Therefore, $l = m = n$ and $i_h = j_h = g_h$, $h = 1, 2, \ldots, n$. ■

At last we have arrived at the definition of (restricted) commutation-augmented pregroup grammars.

DEFINITION 4.8. (Cf. Section 3.3.) A *restricted commutation-augmented pregroup grammar* (RCAPGG) based on an RCAPG $\mathcal{G}(\mathcal{B}_1, \leq_1, \mathcal{B}_2, C', C'', Z, U, V)$ with unrelated C' and C'' is a tuple $G = \langle \Sigma, \mathcal{B}_1, \leq_1, \mathcal{B}_2, C', C'', Z, U, V, I', I, I'', S\rangle$,[11] where Σ is a finite set of terminals (the *alphabet*), $S \in \tau(\mathcal{B}_1)$ is a *distinguished term*, and the *lexical type assignments* I', I, and I'' are defined as follows.

For each $c \in \Sigma$

- $I'(c)$ is a (finite) subset of the free monoid generated by C',
- I is an ordinary type assignment over $\kappa(\mathcal{B}_1)$,[12] and
- $I''(c)$ is a (finite) subset of the free monoid generated by C''.

The language generated by G is defined by

$$L(G) = \{c_1 \cdots c_l\colon \text{ there exist } \alpha_1, \ldots, \alpha_l \text{ such that}$$
$$\alpha_i \in \{Z, U\} I'(c_i) \{U^{\ell}, V^{\ell}\} \cup \{Z, V\} I(c_i) I''(c_i) \{V^{\ell}, 1\},$$
$$i \leq l, \text{ and } \alpha_1 \cdots \alpha_l \leq ZS\}$$

REMARK 4.9. Note that PGGs can be embedded into RCAPGGs in the following sense. Let $G_1 = \langle \Sigma, \mathcal{B}_1, \leq_1, I, S\rangle$ be a PGG. Then for the RCAPGG $G = \langle \Sigma, \mathcal{B}_1, \leq_1, \emptyset, \langle\rangle, \langle\rangle, Z, U, V, \emptyset, I, 1, S\rangle$,[13] $L(G_1) = L(G)$. In particular, since $\mathcal{B}_2 = \emptyset$, the underlying pregroup is free, i.e., no inequations are added.

For Theorem 4.10 below we recall the notion of *substitution*.

Let Σ and Θ be alphabets. A substitution S is a mapping of Σ into subsets of Θ^*. It is of a *finite range*, if $S(c)$ is finite for all $c \in \Sigma$, cf. footnote 4.

---

[11] The sets of inequations ι in the examples in Section 3 are determined by the tuples C' and C''.

[12] Recall that $\kappa(\mathcal{B}_1)$ is the set of all categories generated by $\mathcal{B}_1$.

[13] That is, I'' is the constant function 1.

Finally, S extends onto the whole Σ^* by the following induction: $S(\epsilon) = \{\epsilon\}$, and $S(wc) = S(w)S(c)$, i.e., $S(wc)$ is the *concatenation* of $S(w)$ and $S(c)$.

THEOREM 4.10. *A language $L \subseteq \Sigma^+$ is generated by an RCAPGG if and only if there exists a context-free language $L' \subseteq \Sigma^+$, an finite alphabet Θ, and finite range substitutions $S', S'' : \Sigma \to 2^{\Theta^*}$ such that*

$$L = \{vw \in \Sigma^+ : w \in L' \text{ and } S'(v) \cap S''(w) \neq \emptyset\}$$

PROOF. We start with the proof of the “if” part of the theorem. Let $L' \subseteq \Sigma^+$ be a context-free language, $\Theta = \{A_1, A_2, \ldots, A_k\}$, $S', S'' : \Sigma \to 2^{\Theta^*}$ be finite range substitutions, and let $G' = \langle \Sigma, \mathcal{B}_1, \leq_1, I, S\rangle$ be a pregroup grammar that generates L'. Let homomorphisms $h' : \Theta \to C'$ and $h'' : \Theta \to C''$ be defined by $h'(A_i) = A'_i$ and $h''(A_i) = A''_i$, $i = 1, 2, \ldots, k$.

Consider an RCAPGG $G(G', S', S'') = \langle \mathcal{B}_1, \leq_1, \mathcal{B}_2, C', C'', Z, U, V, I', I, I'', S\rangle$, where

- $\mathcal{B}_2 = \{A'_i : i = 1, 2, \ldots, k\} \cup \{A''_i : i = 1, 2, \ldots, k\}$,[14]
- Z, U, and V are fresh atoms,
- $C' = \langle A'_1, A'_2, \ldots, A'_k\rangle$ and $C'' = \langle A''_1, A''_2, \ldots, A''_k\rangle$,
- $I' = h' \circ S'$, and
- $I'' = h'' \circ S''$.

Let $v = c_1c_2 \cdots c_l \in \Sigma^*$ and $w = d_1d_2 \cdots d_m \in \Sigma^+$ be such that $S'(v) \cap S''(w) \neq \emptyset$ and let $\theta \in S'(v) \cap S''(w)$. Let $\theta'_i \in S'(c_i)$, $i = 1, 2, \ldots, l$, and $\theta''_j \in S''(d_j)$, $j = 1, 2, \ldots, m$, be such that

$$\theta'_1\theta'_2 \cdots \theta'_l = \theta''_1\theta''_2 \cdots \theta''_m$$

Let $\alpha_j \in I(d_j)$, $j = 1, 2, \ldots, m$ be such that $\alpha_1\alpha_2 \cdots \alpha_m \leq S$. Then

$$Zh'(\theta'_1)U^\ell Uh'(\theta'_2)U^\ell \cdots Uh'(\theta'_l)V^\ell V\alpha_1 h''(\theta''_1)V^\ell V\alpha_2 h''(\theta''_2)V^\ell \cdots V\alpha_m h''(\theta''_m) \leq ZS$$

using **(con)** to contract $U^\ell U$ and $V^\ell V$, $(\mathbf{ind^0_m})$ to move the θ''_js (atom by atom) all the way to Z, $(\mathbf{ind^0_c})$ to cancel each atom from $h'(\theta'_1\theta'_2 \cdots \theta'_l)$ against its counterpart in $h''(\theta''_1\theta''_2 \cdots \theta''_m)$ at Z, cf. Section 3.3, and, finally, using G' to contract $\alpha_1\alpha_2 \cdots \alpha_m$ to S. This, in turn, implies that $vw \in L(G(G', S', S''))$.

[14] That is, $\mathcal{B}_2$ consists of two copies of Θ.

Conversely, assume that $u \in L(G(G', S', S''))$ and fix an appropriate type assignment for the symbols in u. The argument is similar to the proof of $L(G_{rd}) \subset L_{rd}$ in Section 3.3.2. Substituting 1 for all elements of $\mathcal{B}_1 \cup \mathcal{B}_2$ (which, obviously, preserves both **(m)** and **(c)**) in this type assignment, we obtain that $u = vw$, where the type assignment to the symbols in v is given by I' and the type assignment to the symbols in w is given by I and I''.

Let $v = c_1c_2 \cdots c_l \in \Sigma^*$, $w = d_1d_2 \cdots d_m \in \Sigma^+$ and let

- $Z\alpha'_1U^\ell$ be the type assignment to c_1,[15]
- $U\alpha'_iU^\ell$ be the type assignment to c_i, $i = 2, 3, \ldots, l-1$,
- $U\alpha'_lV^\ell$ be the type assignment to c_l,
- $V\alpha_j\alpha''_jV^\ell$, $\alpha_j \in I(d_j)$ and $\alpha''_j \in I''(d_j)$, be the type assignment to d_j, $j = 1, 2, \ldots, m-1$, and
- $V\alpha_m\alpha''_m$, $\alpha_j \in I(d_j)$ and $\alpha''_j \in I''(d_j)$, be the type assignment to d_m

such that

$$Z\alpha'_1U^\ell U\alpha'_2U^\ell \cdots U\alpha'_lV^\ell V\alpha_1\alpha''_1V^\ell V\alpha_2\alpha''_2V^\ell \cdots V\alpha_m\alpha''_m \leq ZS$$

Since the set of atomic generators $\mathcal{B}_1$ and $\mathcal{B}_2$ are disjoint, substituting 1 for Z, U, V, and all elements of $\mathcal{B}_2$, we obtain $\alpha_1\alpha_2 \cdots \alpha_m \leq S$, implying $w \in L'$.

Similarly, substituting 1 for U, V, and all elements of $\mathcal{B}_1$ (in the original type assignment) we obtain

$$Z\alpha'_1\alpha'_2 \cdots \alpha'_l\alpha''_1\alpha''_2 \cdots \alpha''_m \leq Z$$

Since $\alpha'_1\alpha'_2 \cdots \alpha'_l \in S'(v)$, $\alpha''_1\alpha''_2 \cdots \alpha''_m \in S''(w)$, and, by Corollary 4.7,

$$h'^{-1}(\alpha'_1\alpha'_2 \cdots \alpha'_l) = h''^{-1}(\alpha''_1\alpha''_2 \cdots \alpha''_m)$$

we have $S'(v) \cap S''(w) \neq \emptyset$, which completes the proof of the "if part" of the theorem.

For the proof of the "only if" part, let $G = \langle \mathcal{B}_1, \leq_1, \mathcal{B}_2, C', C'', Z, U, V, I', I, I'', S \rangle$, $C' = \langle A'_1, A'_2, \ldots, A'_k \rangle$ and $C'' = \langle A''_1, A''_2, \ldots, A''_k \rangle$, be an RCAPGG and let

- $L' = L(G')$, where $G' = \langle \Sigma, \mathcal{B}_1, \leq_1, I, S \rangle$;
- $\Theta = \{A_1, A_2, \cdots, A_k\}$;

[15]If $m = 1$, then, naturally, the type assignment is $Z\alpha'_1V^\ell$.

- $S' = h'^{-1} \circ I'$; and
- $S'' = h''^{-1} \circ I''$, where the homomorphisms h' and h'' are defined in the proof of the "only if" part of the theorem.

Then $G = G(G', S', S'')$, and the result follows from the "if" part. ■

COROLLARY 4.11. *The following languages are generated by RCAPGG.*

1. *All (ϵ-free) context-free languages.*
2. $L_{ma} = \{a^n b^n c^n : n = 1, 2, \ldots\}$.
3. $L_{cd} = \{a^m b^n c^m d^n : m, n = 1, 2, \ldots\}$.
4. $L_{rd} = \{ww : w \in \Sigma^+\}$.

PROOF. 1. For an ϵ-free context-free language L, let L' be L itself, substitution S' be any one element (or any finite) subset of Θ^+, and substitution S'' be $\{\epsilon\}$. Hence, if vw is such that $w \in L'$ and $S'(v) \cap S''(w) \neq \emptyset$, then $v = \epsilon$, because $S''(w)$ is not empty, except for $w = \epsilon$.

2. Let L' be $\{b^n c^n : n = 1, 2, \ldots\}$, Θ be $\{A, B\}$, and define S' and S'' as follows.

   - $S'(a) = \{A\}$ and $S'(b) = S'(c) = \{B\}$; and
   - $S''(b) = \{A\}$ and $S''(a) = S''(c) = \{\epsilon\}$.

   If vw, $w = b^n c^n$, satisfies the condition, then $v = a^n$. Otherwise $S'(v)$ contains B, which is not present in the image of S''.

3. Let L' be $\{c\}^+\{d\}^+$, Θ be $\{A, B, C\}$, and define S' and S'' as follows.

   - $S'(a) = \{A\}$, $S'(b) = \{B\}$, and $S'(c) = S'(d) = \{C\}$; and
   - $S''(a) = S''(b) = \{\epsilon\}$, $S''(c) = \{A\}$ and $S''(d) = \{B\}$.

   Then, $S'(a^{m_1} b^{n_1}) = \{A^{m_1} B^{n_1}\}$, $S''(c^{m_2} d^{n_2}) = \{A^{m_2} B^{n_2}\}$. The intersection is not empty if and only if $m_1 = m_2, n_1 = n_2$. The converse part is equally easy. If vw, $w = c^m d^n$, satisfies the condition, then $v = a^m b^n$. Otherwise, $S'(v)$ contains C, which is not present in the image of S''.

4. Let L' be Σ^+, Θ be Σ, and S' and S'' be the "identity" substitution.[16]

■

REMARK 4.12. Note that the substitutions in the proof of Corollary 4.11 are, actually, homomorphisms.

[16]That is, for $c \in \Sigma$, $S'(c) = S''(c) = \{c\}$.

Corollary 4.13. *RCAPGG languages are in* $\boldsymbol{P}$.

For the proof of Corollary 4.13 and Theorem 4.16 below, in addition to the "standard" pushdown automata which accept by *empty stack*, see [3], we shall use the following extended, but still equivalent, model of computation called *extended* pushdown automata, cf. [5].

Definition 4.14. An *extended* pushdown automaton is a tuple $A = \langle \Sigma, Q, q_0, \Gamma, \gamma_0, \mu, F \rangle$, where

- Σ is the input alphabet,
- Q is a finite set of states,
- $q_0 \in Q$ is the initial state,
- Γ is the stack alphabet,
- $\gamma_0 \in \Gamma^*$ is the *initial content* of the stack,
- μ is a finite subset of $(Q \times \Sigma \times \Gamma^*) \times (Q \times \Gamma^*)$ called the *transition relation*, and
- $F \subseteq Q$ is the set of *final* states.

A *configuration* of A is a triple $(q, w, \gamma) \in Q \times \Sigma^* \times \Gamma^*$. As usual, q is the automaton current state, w is the input suffix to be read, and γ is the content of the stack, read bottom up. The triple (q_0, w, γ_0) is called the *initial* configuration (on input w).

We say that configuration (q', w', γ') *yields in one step* configuration (q'', w'', γ''), denoted $(q', w', \gamma') \vdash (q'', w'', \gamma'')$, if for some $c \in \Sigma$, $w' = cw''$ and the following condition is satisfied.

For some $\gamma, \delta', \delta'' \in \Gamma^*$, $\gamma' = \gamma\delta'$, $\gamma'' = \gamma\delta''$, and $((q', c, \delta'), (q'', \delta'')) \in \mu$.

As usual, $\vdash^*$ denotes the reflexive-transitive closure of $\vdash$, and we define the language accepted by A as

$$L(A) = \{w \in \Sigma^* : (q_0, w, \gamma_0) \vdash^* (q, \epsilon) \text{ for some } q \in F\}$$

Proof of Corollary 4.13. Let L' be a context-free language and let $S' : \Sigma \to 2^{\Theta^*}$ and $S'' : \Sigma \to 2^{\Theta^*}$ be finite range substitutions. The test for a membership of a word $c_1 c_2 \cdots c_l$ in $\{vw \in \Sigma^+ : w \in L' \text{ and } S'(v) \cap S''(w) \neq \emptyset\}$ is as follows.

For each $i = 0, 1, \ldots, l-1$, first check whether

1. $S'(c_1 c_2 \cdots c_i) \cap S''(c_{i+1} c_{i+2} \cdots c_l) \neq \emptyset$ and, if so,
2. check whether $c_{i+1} c_{i+2} \cdots c_l$ is in L'.

Whereas the second part of the above scheme can obviously be performed in polynomial (cubic) time, a polynomial time algorithm for the first part is as described below.

Let $A = \langle \Sigma \cup \{\$\}, \{s, f\}, s, \Theta, \epsilon, \mu, \{f\} \rangle$, $\$ \notin \Sigma$, be an extended pushdown automaton, where μ is the union of

- $\{((s, c, \epsilon), (s, \theta) : \theta \in S'(c)\}$,
- $\{(s, \$, \epsilon)(f, \epsilon)\}$, and
- $\{((f, c, \theta)(f, \epsilon) : \theta \in S''(c)\}$.

It can be easily seen that

$$L(A) = \{v\$w : v, w \in \Sigma^+ \text{ and } S'(v) \cap S''(\overleftarrow{w}) \neq \emptyset\}$$

where $\overleftarrow{w}$ is the *reversal* of w. Therefore, in order to check whether the intersection $S'(c_1c_2 \cdots c_i) \cap S''(c_{i+1}c_{i+2} \cdots c_l)$ is non-empty, it suffices to check whether $c_1c_2 \cdots c_i\$c_lc_{l-1} \cdots c_{i+1}$ is accepted by A, which can be done in polynomial (cubic) time. Thus, the time complexity of the above algorithm is $O(n^4)$. ∎

REMARK 4.15. Note that the construction of A extends to *regular* substitutions in a straightforward manner.

THEOREM 4.16. *For each context-free language $L' \subseteq \Sigma^+$ and finite range substitutions $S', S'' : \Sigma \to 2^{\Theta^*}$ there exists a context-free language L'' such that the following holds.*

1. *Each word in $\{vw \in \Sigma^+ : w \in L' \text{ and } S'(v) \cap S''(w) \neq \emptyset\}$ can be written in the form $v_1v_2 \cdots v_md_1d_2 \cdots d_m$, $v_i \in \Sigma^*$, $d_i \in \Sigma$, $i = 1, 2, \ldots, m$, such that $v_1d_1v_2d_2 \cdots v_md_m \in L''$.*
2. *Conversely, each word in L'' can be written in the form $v_1d_1v_2d_2 \cdots v_md_m$, $v_i \in \Sigma^*$, $d_i \in \Sigma$, $i = 1, 2, \ldots, m$, such that $v_1v_2 \cdots v_md_1d_2 \cdots d_m \in \{vw \in \Sigma^+ : w \in L' \text{ and } S'(v) \cap S''(w) \neq \emptyset\}$.*

PROOF. Let $A' = \langle \Sigma, \Gamma, Z_0, \mu \rangle$ be a standard one-state pushdown automaton without ϵ-transitions that accepts L' (by empty stack). Consider an extended pushdown automaton $A'' = \langle Q, q_0, \Sigma, \Gamma, Z_0, \mu'', F \rangle$, where the set of states Q, the initial state q_0, the set of final states F, and the transition relation μ'' are defined as follows:

- Q consists of all words in Θ^* of length not exceeding $\max\{|\theta| : \theta \in \bigcup_{c \in \Sigma} S'(c)\} + \max\{|\theta| : \theta \in \bigcup_{c \in \Sigma} S''(c)\}$;

- $q_0 = \epsilon$;
- $F = \{\epsilon\}$; and
- $\mu'' = \mu_1 \cup \mu_2$, where
  - μ_1 consists of all transitions $((q, c, \epsilon)(q\theta, \epsilon))$ such that $q \in Q$ and $\theta \in S'(c)$,[17] and
  - μ_2 consists of all transitions $((\theta q, c, X)(q, \gamma))$ such that $((c, X), \gamma) \in \mu$ and $\theta \in S''(c)$.

Let $L'' = L(A'')$ and we start with the proof of the first part of the theorem. Let $c_1 c_2 \cdots c_l \in \Sigma^*$ and $d_1 d_2 \cdots d_m \in L'$ be such that $S'(c_1 c_2 \cdots c_l) \cap S''(d_1 d_2 \cdots d_m) \neq \emptyset$. Let

$$(d_1 d_2 \cdots d_m, \gamma_1) \vdash_{A'} (d_2 \cdots d_m, \gamma_2) \vdash_{A'} \cdots \vdash_{A'} (\epsilon, \gamma_{n+1})$$

$\gamma_1 = Z_0$ and $\gamma_{n+1} = \epsilon$, be an accepting run of A' on $d_1 d_2 \cdots d_m$ and let $\theta'_i \in S'(c_i)$, $i = 1, 2, \ldots, m$ and $\theta''_j \in S''(d_j)$, $j = 1, 2, \ldots, m$, be such that

$$\theta'_1 \theta'_2 \cdots \theta'_l = \theta''_1 \theta''_2 \cdots \theta''_m$$

Let i_j, $j = 1, 2, \ldots, m$, be the minimal index such that

$$|\theta'_1 \theta'_2 \cdots \theta'_{i_j}| \geq |\theta''_1 \theta''_2 \cdots \theta''_j|$$

and let θ_j, $j = 1, 2, \ldots, m$ be such that

$$\theta''_1 \theta''_2 \cdots \theta''_j \theta_j = \theta'_1 \theta'_2 \cdots \theta'_{i_j}$$

Let $v_j = c_{i_{j-1}+1} c_{i_{j-1}+2} \cdots c_{i_j}$, $j = 1, 2, \ldots, m$.[18]

A straightforward induction shows that for $j = 1, 2, \ldots, m$,

$$(\epsilon, v_1 d_1 v_2 d_2 \cdots v_m d_m, Z) \vdash^*_{A''} (\theta_j, v_{j+1} d_{j+1} v_{i+2} d_{j+2} \cdots v_m d_m, \gamma_{j+1})$$

Therefore, $v_1 d_1 v_2 d_2 \cdots v_m d_m \in L(A'')$.

Conversely, let $u \in L(A'')$ and fix an accepting run of A'' on u. Write u in the form $v_1 d_1 v_2 d_2 \cdots v_m d_m$, $v_j = c_{i_{j-1}+1} c_{i_{j-1}+2} \cdots c_{i_j}$, $j = 1, 2, \ldots, m$, where in the accepting run of A'' on u

- the transition on c_i is $((q, c_i, \epsilon)(q\theta'_i, \epsilon)) \in \mu_1$, and
- the transition on d_j is $((\theta''_j q, d_j, X)(q, \gamma)) \in \mu_2$.

---

[17] We assume that $q\theta \in Q$, as well.

[18] Of course, if $i_j < i_{j-1} + 1$, then $v_j = \epsilon$.

Then, obviously, $d_1 d_2 \cdots d_m \in L'$.

Let θ_j and γ_j, $j = 1, 2, \ldots, m$, be such that in the accepting run of A'' on u,

$$(\epsilon, v_1 d_1 v_2 d_2 \cdots v_m d_m, Z) \vdash_{A''}^{*} (\theta_j, v_{j+1} d_{j+1} v_{j+2} d_{j+2} \cdots v_m d_m, \gamma_j)$$

A straightforward induction shows that for $j = 1, 2, \ldots, m$,

$$\theta''_1 \theta''_2 \cdots \theta''_j \theta_j = \theta'_1 \theta'_2 \cdots \theta'_{i_j}$$

Since $\theta_m = \epsilon$, we have

$$\theta'_1 \theta'_2 \cdots \theta'_m = \theta''_1 \theta''_2 \cdots \theta''_m$$

Therefore,

$$v_1 v_2 \cdots v_m d_1 d_2 \cdots d_m \in \{vw \in \Sigma^+ : w \in L' \ \text{ and } \ S'(v) \cap S''(w) \neq \emptyset\}$$

■

For Corollary 4.17 below we recall the definition of *semi-linear* languages.

A language L is called semi-linear if $\{|w| : w \in L\}$ is a finite union of sets of integers of the form $\{l + im : i = 0, 1, \ldots\}$, $l, m \geq 0$.

COROLLARY 4.17. *RCAPGG languages are semi-linear.*

PROOF. Let L be an RCAPGG language. By Theorem 4.10, there exists a context-free language $L' \subseteq \Sigma^+$, a finite alphabet Θ, and finite range substitutions $S', S'' : \Sigma \to 2^{\Theta^*}$ such that

$$L = \{vw \in \Sigma^+ : w \in L' \ \text{ and } \ S'(v) \cap S''(w) \neq \emptyset\}$$

Let L'' be the context-free language provided by Theorem 4.16 for L', S', and S''. Then $\{|w| : w \in L\} = \{|w| : w \in L'\}$, and the corollary follows from the fact that context-free languages are semi-linear, see [7]. ■

5. Proof of Proposition 4.3

The proof is by induction on the number of applications of **(exp)**s such that one of the terms they introduce is later canceled by **(con)**.

5.1. The case of (exp)s of the form $\gamma\delta \leq \gamma Z^{(n+1)} Z^{(n)} \delta$

Consider the first such cancellation. Since two consecutive steps in a derivation can be interchanged, if the relevant terms do not overlap, we may assume that either

$$\begin{aligned} \alpha' \leq \beta Z^{(n)}\gamma\delta \leq^{\mathbf{(exp)}} \beta Z^{(n)}\gamma Z^{(n+1)} Z^{(n)}\delta \\ \leq \beta Z^{(n)} Z^{(n+1)} Z^{(n)}\delta \leq^{\mathbf{(con)}} \beta Z^{(n)}\delta \leq \alpha'' \end{aligned} \tag{17}$$

i.e., $Z^{(n+1)}$ is canceled from the left, or

$$\begin{aligned} \alpha' \leq \beta\gamma Z^{(n+1)}\delta \leq^{\mathbf{(exp)}} \beta Z^{(n+1)} Z^{(n)}\gamma Z^{(n+1)}\delta \\ \leq \beta Z^{(n+1)} Z^{(n)} Z^{(n+1)}\delta \leq^{\mathbf{(con)}} \beta Z^{(n+1)}\delta \leq \alpha'' \end{aligned} \tag{18}$$

i.e., $Z^{(n)}$ is canceled from the right.

First consider (17), where we have $Z^{(n)}\gamma Z^{(n+1)} \leq Z^{(n)} Z^{(n+1)}$. Assume that n is even. Then, moving all applications of $(\mathbf{ind}_{\mathbf{c}}^{n+1})$ to the beginning of the derivation we obtain that for some $i_1, i_2, \ldots, i+l = 1, 2, \ldots, k$,

$$Z^{(n)}\gamma A''^{(n+1)}_{i_1} A'^{(n+1)}_{i_1} A''^{(n+1)}_{i_2} A'^{(n+1)}_{i_2} \cdots A''^{(n+1)}_{i_l} A'^{(n+1)}_{i_l} \leq Z^{(n)} \tag{19}$$

Therefore, we can replace (17) with

$$\begin{array}{rll} \alpha' & \leq \ \beta Z^{(n)}\gamma\delta & \text{by (17)} \\ & \leq \ \beta Z^{(n)}\gamma A''^{(n+1)}_{i_1} A'^{(n+1)}_{i_1} A''^{(n+1)}_{i_2} A'^{(n+1)}_{i_2} \cdots A''^{(n+1)}_{i_l} A'^{(n+1)}_{i_l} & \\ & \quad A'^{(n)}_{i_l} A''^{(n)}_{i_l} A'^{(n)}_{i_{l-1}} A''^{(n)}_{i_{l-1}} \cdots A'^{(n)}_{i_1} A''^{(n)}_{i_1} \delta & \text{by } l \ (\mathbf{exp})\text{s} \\ & \leq \ \beta Z^{(n)} A'^{(n)}_{i_l} A''^{(n)}_{i_l} A'^{(n)}_{i_{l-1}} A''^{(n)}_{i_{l-1}} \cdots A'^{(n)}_{i_1} A''^{(n)}_{i_1} \delta & \text{by (19)} \\ & \leq \ \beta Z^{(n)}\delta & \text{by } l \ (\mathbf{ind}_{\mathbf{c}}^{\mathbf{n}})\text{s} \\ & \leq \ \alpha'' & \text{by (17)} \end{array}$$

The case of an odd n is is immediate, because in this case $\gamma \leq 1$.

Cancellation from the right, where we have $Z^{(n)}\gamma Z^{(n+1)} \leq Z^{(n)} Z^{(n+1)}$, is treated in a similar (dual) manner. Assume that n is even. Then, delaying all applications of $(\mathbf{ind}_{\mathbf{c}}^{n})$ to the end of the derivation we obtain

$$\gamma Z^{(n+1)} \leq A'^{(n)}_{i_1} A''^{(n)}_{i_1} A'^{(n)}_{i_2} A''^{(n)}_{i_2} \cdots A'^{(n)}_{i_l} A''^{(n)}_{i_l} Z^{(n+1)} \tag{20}$$

Therefore, we can replace (18) with

$$
\begin{array}{rll}
\alpha' & \leq \beta\gamma Z^{(n+1)}\delta & \text{by (18)} \\
& \leq \beta A'^{(n)}_{i_1} A''^{(n)}_{i_1} A'^{(n)}_{i_2} A''^{(n)}_{i_2} \cdots A'^{(n)}_{i_l} A''^{(n)}_{i_l} Z^{(n+1)}\delta & \text{by (20)} \\
& \leq \beta A'^{(n)}_{i_1} A''^{(n)}_{i_1} A'^{(n)}_{i_2} A''^{(n)}_{i_2} \cdots A'^{(n)}_{i_l} A''^{(n)}_{i_l} & \\
& \quad A''^{(n+1)}_{i_l} A'^{(n+1)}_{i_l} A''^{(n+1)}_{i_{l-1}} A'^{(n+1)}_{i_{l-1}} \cdots A''^{(n+1)}_{i_1} A'^{(n+1)}_{i_1} & \text{by } l \ (\mathbf{ind_c^{n+1}})\text{s} \\
& \leq \beta Z^{(n+1)}\delta & \text{by } l \ (\mathbf{con})\text{s} \\
& \leq \alpha'' & \text{by (18)}
\end{array}
$$

Again, if n is odd, then $\gamma \leq 1$.

5.2. The case of (exp)s of the form $\gamma\delta \leq \gamma A^{(n+1)} A^{(n)} \delta$, $A \neq Z$

Consider the last application of such **(exp)**. Like in the case of Z, we shall distinguish between the following two cases.

a) $A^{(n+1)}$ is canceled from the left, and

b) $A^{(n)}$ is canceled from the right.

For case *a*), since $A^{(n+1)}$ has been introduced by the last **(exp)**, the term $A^{(n)}$ that cancels $A^{(n+1)}$ already appears in the category before the application of the last **(exp)**. Therefore, we have

$$
\begin{aligned}
\alpha' \leq \beta A^{(n)}\gamma\delta \leq^{(\mathbf{exp})} \beta A^{(n)}\gamma A^{(n+1)} A^{(n)}\delta \\
\leq \beta' A^{(n)} A^{(n+1)} \gamma' A^{(n)} \delta' \leq^{(\mathbf{con})} \beta\gamma' A^{(n)}\delta' \leq \alpha''
\end{aligned}
\tag{21}
$$

We may assume that no term that appears in δ can be moved to the left of $A^{(n+1)}$. Indeed, instead of moving a term to the left of $A^{(n+1)}$ we can just move it to the beginning of δ and then "insert" $A^{(n+1)}A^{(n)}$ after that term. Also, since two consecutive steps in a derivation can be interchanged, if the relevant terms do not overlap, we may assume that

- $\beta' = \beta$;
- $\gamma' = \gamma = B_1 B_2 \cdots B_l$ and one of the conditions below is satisfied:[19]
  - if $A^{(n+1)} \in C''^{(m)}$, then $B_i \in \tau(\mathcal{B}_1) \cup C'^{(m)}$, $i = 1, 2, \ldots, l$,
  - if $A^{(n+1)} \in C'^{(m)}$, then $B_i \in C''^{(m-1)}$, $i = 1, 2, \ldots, l$, and
  - if $A^{(n+1)} \in \tau(\mathcal{B}_1)$, then $B_i \in \tau(C'')$, $i = 1, 2, \ldots, l$;
- $\delta' = \delta$.

[19] Of course, l may be zero.

That is, (21) is of the form

$$\begin{array}{rlll} \alpha' & \leq & \beta A^{(n)} B_1 B_2 \cdots B_l \delta & \text{by (21)} \\ & \leq & \beta A^{(n)} B_1 B_2 \cdots B_l A^{(n+1)} A^{(n)} \delta & \text{by } (\mathbf{exp}) \\ & \leq & \beta A^{(n)} A^{(n+1)} B_1 B_2 \cdots B_l A^{(n)} \delta & \text{by } l\ (\mathbf{ind}^{m}_{\mathbf{m}})\text{s} \\ & \leq & \beta B_1 B_2 \cdots B_l A^{(n)} \delta & \text{by } (\mathbf{con}) \\ & \leq & \alpha'' & \text{by (21)} \end{array}$$

which can be replaced just with

$$\begin{array}{rlll} \alpha' & \leq & \beta A^{(n)} B_1 B_2 \cdots B_l \delta & \text{by (21)} \\ & \leq & \beta B_1 B_2 \cdots B_l A^{(n)} \delta & \text{by } l\ (\mathbf{ind}^{m}_{\mathbf{m}'})\text{s} \\ & \leq & \alpha'' & \text{by (21)} \end{array}$$

Case b) is treated in a similar manner. After interchanging appropriate consecutive steps in a derivation in which the relevant terms do not overlap, we obtain

$$\begin{array}{rlll} \alpha' & \leq & \beta B_1 B_2 \cdots B_l A^{(n+1)} \delta & \\ & \leq & \beta A^{(n+1)} A^{(n)} B_1 B_2 \cdots B_l A^{(n+1)} \delta & \text{by } (\mathbf{exp}) \\ & \leq & \beta A^{(n+1)} B_1 B_2 \cdots B_l A^{(n)} A^{(n+1)} \delta & \text{by } l\ (\mathbf{ind}^{m}_{\mathbf{m}'})\text{s} \\ & \leq & \beta A^{(n+1)} B_1 B_2 \cdots B_l \delta & \text{by } (\mathbf{con}) \\ & \leq & \alpha'' & \end{array}$$

which we replace with

$$\begin{array}{rlll} \alpha' & \leq & \beta B_1 B_2 \cdots B_l A^{(n+1)} \delta & \\ & \leq & \beta A^{(n+1)} B_1 B_2 \cdots B_l \delta & \text{by } l\ (\mathbf{ind}^{m}_{\mathbf{m}})\text{s} \\ & \leq & \alpha'' & \end{array}$$

6. Conclusions

The paper presents an extension of pregroup grammars, by which a certain limited amount of commutativity and cancelability is introduced into a free pregroup, allowing the grammars based on them to recognize mildly context-sensitive languages. Polynomial parsing and semi-linearity are established for languages generated by the commutation augmented pregroup grammars.

In a sequel paper [2], we present an automata-theoretic counterpart of RCAPGGs, the (*restricted*) *canceling pushdown automaton*, which has the same weak generative power. We also aim at another automaton, which is more *tightly related* to the grammars studied here, in the sense that there is a stronger relationship between automaton computation and grammar derivation. In a future paper, we will also present closure properties of the class of languages studied here. Currently, one cancellation move of the automaton is simulated by a linear number of commutations, to the "left end" of the string of terms followed by cancellation.

In the longer range, we aim at a lexicalized, type driven, presentation of the whole Chomsky hierarchy, including context-sensitive and RE languages.

Acknowledgments. The authors are grateful to Wojciech Buszkowski for his comments on an earlier version of this paper.

This research was supported by the Loewengart research fund, by the fund for the promotion of research at the Technion, and by a grant from the Software Technology Laboratory (STL) in the Department of Computer Science, the Technion.

References

[1] Buszkowski, W., 'Lambek grammars based on pregroups', in P. De Groote, G. Morrill, and C. Retorè (eds.), *Logical Aspects of Computational Linguistics*, vol. 2099 of *Lecture Notes in Computer Science*, Springer Verlag, Berlin Heidelberg, 2001, pp. 95–109.

[2] Francez, N., and M. Kaminski, *Pushdown automata with cancellation and commutation-augmented pregroup grammars*, in R. Loos, S.Z. Fazekas, and C. Martín-Vide (eds.), *Pre-proceedings of the 1st International Conference on Language and Automata Theory and Applications (LATA07)*, Tarragona, Spain, March 29 – April 4, 2007, pp. 7–25.

[3] Hopcroft, J. E., and J. D. Ulmann, *Introduction to Automata Theory, Languages and Computation*, Addison Wesley, New York, 1979.

[4] Lambek, J., 'Type grammars revisited', in A. Lecomte, F. Lamarche, and G. Perrier, (eds.), *Logical Aspects of Computational Linguistics*, vol. 1582 of *Lecture Notes in Computer Science*, Springer Verlag, Berlin Heidelberg, 1999, pp. 1–27.

[5] Lewis, H. R, and Ch. H. Papadimitriou, *Elements of the Theory of Computation*, Prentice Hall, Englewood Cliffs, NJ, 1981.

[6] Michaelis, J., and M. Kracht, 'Semilinearity as a syntactic invariant', in Ch. Retoré (ed.), *Logical Aspects of Computational Linguistics*, vol. 1328 of *Lecture Notes in Computer Science*, Springer Verlag, Berlin Heidelberg, 1997, pp. 329–345.

[7] Parikh, R. J., 'On context-free languages', *Journal of the ACM* 13, 4 (1966), 570–581.

[8] Vijay-Shanker, K., and David J. Weir, The equivalence of four extensions of context-free grammars', *Mathematical System Theory* 27 (1994), 511–546.

Nissim Francez
Department of Computer Science
Technion-Israel Institute of Technology
Haifa 32000, Israel
francez@cs.technion.ac.il

Michael Kaminski
Department of Computer Science
Technion-Israel Institute of Technology
Haifa 32000, Israel
kaminski@cs.technion.ac.il