Unidirectional Inheritance of Indices
A Weakly Context Free Facet of LIGs*

Jens Michaelis Christian Wartena

Universität Potsdam
Institut für Linguistik/Innovationskolleg
Postfach 60 15 53
14415 Potsdam
Germany

michaelis@ling.uni-potsdam.de
wartena@ling.uni-potsdam.de

Abstract. In this paper a restricted type of linear indexed grammar (LIG) is defined, called *unidirectional indexed grammar (UIG)*. This is done essentially by demanding that the inheritance of an index stack is either strictly rightmost or strictly leftmost as long as the stack is not empty. Although they turn out to be weakly equivalent to context free grammars (CFGs), UIGs give rise to structural descriptions that are not generatable by any CFG, but seem to be suitable for linguistic purposes. Moreover, the parsing problem for UIGs is shown to be solvable in $O(n^3)$–time. Thus, also in this respect, UIGs constitute an interesting restriction of LIGs, the latter known to be of parsing complexity $O(n^6)$–time in general.

1 Motivation

Structures of natural language sentences are commonly described in terms of trees. Corresponding (formal) grammars usually provide a set of admissible local subtrees. But such grammars should additionally include mechanisms to describe non–local dependencies, as these apparently show up in human languages. The potentials of indexing mechanisms to account for such dependencies were first exploited in Gazdar 1988, which brought up the formalism of linear indexed grammar (LIG). The class of languages generated by LIGs is known to be a proper superclass of the class of context free

*This paper has been worked out at the University Potsdam within the *Innovationskolleg 'Formale Modelle kognitiver Komplexität' (INK 12)* funded by the Deutsche Forschungsgemeinschaft (DFG).

languages (CFLs). However, in Michaelis & Wartena 1997 we have argued that almost all of the expressive power of LIGs useful for the description of natural languages is already provided by a proper subclass of LIGs. Relying on the observation that extraction in almost all languages is only possible out of objects and assuming some kind of common X–bar scheme we concluded that the path from a moved element to its base position can always be recovered by successively descending the rightmost branch at each node. We defined a corresponding restricted form of LIGs, right LIGs (RLIGs), in which a stack of indices always has to be passed on from a mother to her rightmost daughter. For X–bar schemes in which a complement is allowed to precede its head we defined a slightly weakened, but still restricted form of LIGs, extended RLIGs (ERLIGs). We have shown that the weak generative capacity of ERLIGs lies properly in between that of context free grammars (CFGs) and LIGs, whereas RLIGs generate exactly the class of CFLs. In this paper we will address the issue concerning restrictions on inheritance of indices from a more technical point of view and try to gain a deeper understanding of the kind of restrictions that, if added to the LIG–formalism, make grammars of the resulting grammar class generate CFLs only. For this purpose we define the formalism of *unidirectional indexed grammar* (UIG). In a UIG index symbols can be inherited either to the left or to the right. But once that a stack of index symbols is passed on to a rightmost (leftmost) daughter this stack can only be hosted by rightmost (leftmost) descendants until it is empty. Thus the UIG–formalism can be considered as a generalization of each, RLIGs as well as their primarily technical counterpart, left LIGs (LLIGs) (Michaelis & Wartena 1997), resulting from combining these two formalisms in a limited way. UIGs allow for more structural descriptions than plain RLIGs (LLIGs), while the property of RLIGs (LLIGs) to be weakly equivalent to context free grammars (CFGs) is preserved. From a linguistic point of view some rare examples of non–local dependencies along left branches (cf. Gazdar 1988, p. 77), which could not be accounted for within the RLIG–formalism, might get an appropriate formalisation now. Moreover, we show that the parsing problem for UIGs, that has not yet been examined for RLIGs in particular, can be solved in $O(n^3)$–time. Thus, also w.r.t. parsing, UIGs constitute an interesting restriction of LIGs, the latter known to be of parsing complexity $O(n^6)$–time in general. Further we briefly mention some connections to general LIG–parsing and to other research in constraining mildly context sensitive grammar formalisms so as to identify weakly context free subclasses.

2 Constraining LIGs

Definition 2.1 A *unidirectional indexed grammar (UIG) G* is a five–tuple (N, Σ, I, P, S), where N, Σ and I are finite sets, pairwise disjoint, which respectively denote the sets of *nonterminal*, *terminal* and *index symbols*. S denotes a distinguished nonterminal (the *start symbol*). P is a finite set of pairs, called *productions*, of the form: 1. $A[f...] \rightarrow \beta_l B[g...]\beta_r$ or 2. $A[\,] \rightarrow w$, for some $A, B \in N$, $f, g \in I \cup \{\epsilon\}$ with $|fg| \leq 1$,[1] $\beta_l, \beta_r \in (\Sigma \cup N)^*$ with $\beta_l = \epsilon$ or $\beta_r = \epsilon$, $w \in \Sigma^*$, and $[\,,...,\,]$ extra symbols not in $N \cup \Sigma \cup I$.

A production $p \in P$ of the 1st form is *right indexed* or *left indexed* in case that $\beta_r = \epsilon$ or $\beta_l = \epsilon$, respectively. Furthermore, any such p is either *index introducing* (if $g \in I$), *index absorbing* (if $f \in I$) or *constraint free* (if $fg = \epsilon$). The sets of right indexed, left indexed, index introducing, index absorbing and constraint free productions are denoted by P_{ri}, P_{li}, P_{in}, P_{ab} and P_{cf}, respectively.

Let $G = (N, \Sigma, I, P, S)$ be an UIG. The set NI^* is called the set of *nonterminal objects*. A string $\sigma \in (NI^* \cup \Sigma)^*$ is called a *sentential form*. A sentential form σ *directly derives right indexed (left indexed)* a sentential form τ, written $\sigma \overset{\mathrm{ri}}{\underset{G}{\Rightarrow}} \tau$ ($\sigma \overset{\mathrm{li}}{\underset{G}{\Rightarrow}} \tau$), if either (1) or (2):

$$
\begin{array}{lll}
(1) \quad \sigma = \gamma_l A f \zeta \gamma_r & \qquad & (2) \quad \sigma = \gamma_l A \gamma_r \\
A[f...] \rightarrow \beta_l B[g...]\beta_r \in P_{\mathrm{ri}} \ (\in P_{\mathrm{li}}) & & A[\,] \rightarrow w \in P \\
\tau = \gamma_l \beta_l B g \zeta \beta_r \gamma_r & & \tau = \gamma_l w \gamma_r
\end{array}
$$

for some $A, B \in N$, $f, g \in I \cup \{\epsilon\}$, $\beta_l, \beta_r \in (\Sigma \cup N)^*$, $w \in \Sigma^*$, $\zeta \in I^*$, and $\gamma_l, \gamma_r \in (NI^* \cup \Sigma)^*$.[2] It is helpful to think of a (maximal) nonterminal object occurring in a sentential form as a nonterminal associated with a stack of indices. Thus, we see that in case (1) such a stack of indices is passed on from the mother to exactly one daughter,[3] while possibly the topmost index symbol f is popped from, or a new index g is pushed onto the stack. In this case, with respect to the corresponding occurrences, $Bg\zeta$ is called the *distinguished child* of its mother $Af\zeta$.

We say, a sentential form σ *directly derives (unidirectional indexed in G)* a sentential form τ, written $\sigma \underset{G}{\Rightarrow} \tau$, if $\sigma \overset{\mathrm{ri}}{\underset{G}{\Rightarrow}} \tau$ or $\sigma \overset{\mathrm{li}}{\underset{G}{\Rightarrow}} \tau$. If $\sigma \underset{G}{\Rightarrow} \tau$ by applying $p \in P$, we also write $\sigma \overset{p}{\underset{G}{\Rightarrow}} \tau$. Now, generalizing $\underset{G}{\Rightarrow}$ we will inductively define a binary relation $\overset{\mathrm{ui}}{\underset{G}{\Rightarrow}}$ on the set of sentential forms. But contrasting with the general LIG–case, $\overset{\mathrm{ui}}{\underset{G}{\Rightarrow}}$ will characterize only a subset of the reflexive and transitive closure of $\underset{G}{\Rightarrow}$:[4] For each sentential form σ set $\sigma \overset{0}{\underset{G}{\Rightarrow}} \sigma$. For $n \in \mathbb{N}$ let $\sigma_0 \ldots \sigma_{n+1}$ be a sequence of sentential forms such that $\sigma_i \underset{G}{\Rightarrow} \sigma_{i+1}$ for $0 \leq i \leq n$

[1] For any set M and $\sigma \in M^*$, M^* the set of finite strings in M including the empty string ϵ, $|\sigma|$ denotes the length of σ.

[2] Note that γ_r cannot start with an index. So, the terminal production $A[\,] \rightarrow w$ can only be applied, if no index appears immediately to the right of A.

[3] Namely, the nonterminal that immediately precedes $[g...]$ in the production.

[4] Up to here, the UIG–definition differs from the general LIG–definition (as presented e.g. in Vijay-Shanker & Weir 1994) only by establishing restrictions on the type of permissible productions of the following two kinds: At each derivation step index inheritance has

and such that $\sigma_0 \overset{n}{\underset{G}{\Rightarrow}} \sigma_n$.[5] Thus, a particular occurrence of some $A_n\eta_n \in NI^*$ is rewritten in $\sigma_n \underset{G}{\Rightarrow} \sigma_{n+1}$ by applying some $p_n \in P$. Consider the case $\eta_n \neq \epsilon$ such that for some $0 \leq i \leq n$ the instance of $A_n\eta_n$, rewritten in $\sigma_n \overset{p_n}{\underset{G}{\Rightarrow}} \sigma_{n+1}$, has been introduced as a distinguished child in $\sigma_i \underset{G}{\Rightarrow} \sigma_{i+1}$ by applying some $p_i \in P_{\mathrm{ri}}$ ($p_i \in P_{\mathrm{li}}$). Then $\sigma_0 \overset{n+1}{\underset{G}{\Rightarrow}} \sigma_{n+1}$, only if $p_n \in P_{\mathrm{ri}}$ ($p_n \in P_{\mathrm{li}}$), as well. In all other cases, $\sigma_0 \overset{n+1}{\underset{G}{\Rightarrow}} \sigma_{n+1}$ without restriction. We say, a sentential form σ *derives (unidirectional indexed in G)* a sentential form τ, written $\sigma \overset{\mathrm{ui}}{\underset{G}{\Rightarrow}} \tau$, if $\sigma \overset{n}{\underset{G}{\Rightarrow}} \tau$ for some $n \in \mathbb{N}$. The set $L(G) = \{w \in \Sigma^* \mid S \overset{\mathrm{ui}}{\underset{G}{\Rightarrow}} w\}$ is the *unidirectional indexed language (UIL)* generated by G.

For $n \in \mathbb{N}$ let $d = \sigma_0 \dots \sigma_n$ be a sequence of sentential forms with $\sigma_0 \overset{\mathrm{ui}}{\underset{G}{\Rightarrow}} \sigma_i$ for $0 \leq i \leq n$. In particular, $\sigma_i \overset{p_i}{\underset{G}{\Rightarrow}} \sigma_{i+1}$ for some $p_i \in P$. We sometimes write $\sigma_0 \overset{\pi}{\underset{G}{\Rightarrow}} \sigma_n$ instead of $\sigma_0 \overset{\mathrm{ui}}{\underset{G}{\Rightarrow}} \sigma_n$, where $\pi = p_0 \dots p_{n-1} \in P^*$. d is called a *derivation (from σ_0 to σ_n in G)*, and π a *parse (from σ_0 to σ_n in G)*. In case that $\sigma_0 = S$ we also say, d is a *derivation of σ_n* and π a *parse of σ_n (in G)*.

To make the definition of $\overset{\mathrm{ui}}{\underset{G}{\Rightarrow}}$ somewhat more transparent, we (informally) define a *spine* as a sequence of objects $A_0\eta_0 \dots A_k\eta_k$ in NI^* such that $A_{i+1}\eta_{i+1}$ is the distinguished child of $A_i\eta_i$ with respect to some derivation d in G. In this case, $A_k\eta_k$ is called a *distinguished descendant* of $A_0\eta_0$. Then, in terms of derivation trees, a spine necessarily coincides with the path from a node ν to the rightmost (leftmost) descendant of ν as long as the stack which is passed on along the spine is not empty.

Take $\overset{\mathrm{ri}}{\underset{G}{\Rightarrow}}{}^*$, $\overset{\mathrm{li}}{\underset{G}{\Rightarrow}}{}^*$ and $\underset{G}{\Rightarrow}^*$ to denote the reflexive and transitive closure of $\overset{\mathrm{ri}}{\underset{G}{\Rightarrow}}$, $\overset{\mathrm{li}}{\underset{G}{\Rightarrow}}$ and $\underset{G}{\Rightarrow}$, respectively. Note that $\overset{\mathrm{ri}}{\underset{G}{\Rightarrow}}{}^* \cup \overset{\mathrm{li}}{\underset{G}{\Rightarrow}}{}^* \subseteq \overset{\mathrm{ui}}{\underset{G}{\Rightarrow}} \subseteq \underset{G}{\Rightarrow}^*$. Equality for both inclusions holds, if $I = \emptyset$, but also if all indexed productions, i.e. all productions of the 1st form, belong to P_{ri} (to P_{li}). The latter case corresponds to a *right LIG* (*left LIG*) as defined in Michaelis & Wartena 1997, and we have $\overset{\mathrm{ri}}{\underset{G}{\Rightarrow}}{}^* = \overset{\mathrm{ui}}{\underset{G}{\Rightarrow}}$ ($\overset{\mathrm{li}}{\underset{G}{\Rightarrow}}{}^* = \overset{\mathrm{ui}}{\underset{G}{\Rightarrow}}$). In case $I = \emptyset$ the UIG can simply be taken as a CFG.[6]

Example 2.2 The UIG $G = (\{S,T\},\{a,b\},\{f\},P,S)$ with production set $P = \{S[...] \to S[f...]b, S[...] \to T[...], T[f...] \to T[...]a, T[] \to \epsilon\}$, that is $\overset{\mathrm{li}}{\underset{G}{\Rightarrow}}{}^* = \overset{\mathrm{ui}}{\underset{G}{\Rightarrow}}$, generates $L(G) = \{a^nb^n | n \in \mathbb{N}\}$ in a "non–center–embedded" way, hence in a non–context free manner, exemplified by Figure 1 for $n = 3$.

Let $<_{\mathrm{lex}}$ be the lexicographical order on $\mathbb{N}^*$. That is, $\omega <_{\mathrm{lex}} \omega i \varphi$ and $\omega j \chi <_{\mathrm{lex}} \omega k \psi$ for all $\omega, \varphi, \chi, \psi \in \mathbb{N}^*$ and $i, j, k \in \mathbb{N}$, where $j < k$. Then, w.r.t. $<_{\mathrm{lex}}$ a particular (linear) derivation strategy can be defined for $\overset{\mathrm{ui}}{\underset{G}{\Rightarrow}}$.[7]

---

to be either rightmost or leftmost, and in addition at most one index can be introduced or absorbed. From this perspective each UIG could just be considered as an LIG in a particular reduced form, while the class of grammars defined this way would be weakly equivalent to the proper superclass formed by all LIGs, as is well–known.

[5] $\mathbb{N}$ denotes the set of all non–negative integers.

[6] Conversely, e.g. each CFG in Chomsky normal form can be interpreted as a UIG with $I = \emptyset$. Thus, each CFL turns out as a UIL.

[7] Like e.g. Boullier (1996) did for $\underset{G}{\Rightarrow}^*$ considering a "(canonical) two–form" of full LIGs.

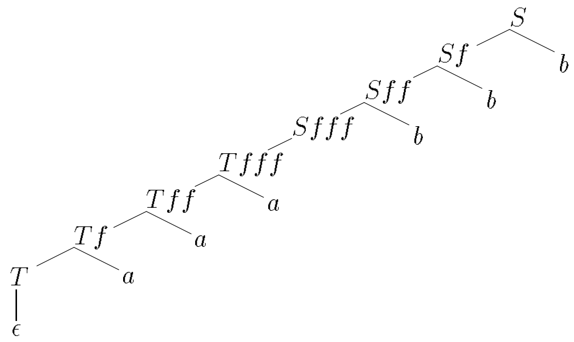

Figure 1: The (unique) derivation structure, as licensed by the grammar G of Example 2.2, belonging to the terminal string a^3b^3.

The strategy contrasts with the concept of a rightmost or a leftmost derivation, respectively, since it works "developing spines last". To achieve this, an address $\omega \in \mathbb{N}^*$ will be assigned to each instance of a nonterminal object $A\eta$ within a sentential form σ in a canonical way. We present this method in some more detail, since it will be of particular interest, when we prove G to be weakly equivalent to a CFG G' and to be parsable in $O(n^3)$–time. Within a derivation the corresponding addresses are defined inductively depending on the length of the derivation in the following way:

Consider $p \in P_{\text{ri}} \cup P_{\text{li}}$. Let $p = A[f...] \rightarrow \beta_l B[g...]\beta_r$ for some $A, B \in N$, $f, g \in I \cup \{\epsilon\}$ and $\beta_l, \beta_r \in (\Sigma \cup N)^*$. Set $m(p) = |\beta_l\beta_r|$. Then, there are $X_0, \ldots, X_{m(p)-1} \in N \cup \Sigma$ such that (ri) or (li) hold.

$$\text{(ri)} \quad \beta_r = \epsilon \text{ and } \beta_l = X_0 \ldots X_{m(p)-1} \quad \text{(li)} \quad \beta_l = \epsilon \text{ and } \beta_r = X_{m(p)-1} \ldots X_0$$

Suppose that, for some $\zeta \in I^*$, p is applied to an instance of $Af\zeta$ with address $\omega \in \mathbb{N}^*$. In both cases, (ri) and (li), for $0 \leq k < m(p)$ the address of the corresponding child X_k is ωk, and the address of the distinguished descendant $Bg\zeta$ is $\omega\, m(p)$.

Let σ be a sentential form such that for each instance of a nonterminal object in σ a unique address in $\mathbb{N}^*$ is defined. We also refer to such an address simply as an *address of* σ. Now, assume d to be a derivation in G starting with σ.[8] Then, according to the just mentioned method a (unique) address is assigned to each new instance of a nonterminal object introduced in a derivation step. Say, $d = \sigma_0 \ldots \sigma_n$ such that $\sigma_i \overset{p_i}{\underset{G}{\Rightarrow}} \sigma_{i+1}$ for some $p_i \in P$ applied to a particular occurrence of some $A_i\zeta_i \in NI^*$ with address $\omega_i \in \mathbb{N}^*$. Set $\pi = p_0 \ldots p_{n-1}$. W.r.t. $<_{\text{lex}}$ the addresses ω_i form a (totally) linearly ordered set. We call the derivation d and the parse π *linear (w.r.t.* $<_{\text{lex}}$*)*, if $\omega_{i-1} <_{\text{lex}} \omega_i$ for $1 \leq i < n$. In this case, if $\sigma_n \in \Sigma^*$, and if ω_i is the highest address of σ_i, then $\sigma_i = u_i A_i \zeta_i v_i$ for some $u_i, v_i \in \Sigma^*$. Now, suppose that d

[8] If not defined for some other reasons, in any derivation $\tilde{d}$ starting with a sentential form $\tilde{\sigma}$, $\tilde{\sigma}$ is tacitly assumed to be addressed in the same manner as σ here.

is linear such that $\sigma_i = \tau_i B_i \eta_i$ ($\sigma_i = B_i \eta_i \tau_i$) for $0 \leq i < n$, where $B_i \eta_i \in NI^*$ and $\tau_i \in (NI^* \cup \Sigma)^*$. If, in addition, for $0 \leq i < n$ the highest address of σ_i belongs to the rightmost (leftmost) instance of $B_i \eta_i$, then d, just as π, is also called *weakly right linear (weakly left linear)*. In this case, if $\sigma_n \in \Sigma^*$, the rightmost (leftmost) occurrence of $B_i \eta_i$ can be rewritten only if $\tau_i \in \Sigma^*$.

Lemma 2.3 *Consider $\sigma, \tau \in (NI^* \cup \Sigma)^*$. If $\sigma \overset{\text{ui}}{\underset{G}{\Rightarrow}} \tau$, there is also a linear derivation from σ to τ in G.* □

3 Weak Generative Capacity

Let $G = (N, \Sigma, I, P, S)$ be a UIG. We give a method to construct a CFG $G' = (N', \Sigma, P', S')$ which is weakly equivalent to G: First, the nonterminal set N' is defined as the set of all pairs (A, x) with $A \in N$ and $x \in P_{\text{ab}} \cup \{\epsilon\}$, (S, ϵ) the start symbol S'. Then, for all $A, B, C, D \in N$, $f \in I$, and $\gamma, \beta \in (N \cup \Sigma)^*$, $w \in \Sigma^*$ the production set P' is defined as[9]

(R1) $(A, x) \to (\gamma, \epsilon)(C, q)(\beta, \epsilon)(B, x) \in P'$ for all $x \in (P_{\text{ri}} \cap P_{\text{ab}}) \cup \{\epsilon\}$,
if $p, q \in P_{\text{ri}}$, where $p = A[...] \to \gamma C[f...]$ and $q = D[f...] \to \beta B[...]$

(L1) $(A, x) \to (B, x)(\beta, \epsilon)(C, q)(\gamma, \epsilon) \in P'$ for all $x \in (P_{\text{li}} \cap P_{\text{ab}}) \cup \{\epsilon\}$,
if $p, q \in P_{\text{li}}$, where $p = A[...] \to C[f...]\gamma$ and $q = D[f...] \to B[...]\beta$

(R2) $(A, x) \to (\gamma, \epsilon)(C, x) \in P'$ for all $x \in (P_{\text{ri}} \cap P_{\text{cf}}) \cup \{\epsilon\}$,
if $p = A[...] \to \gamma C[...] \in P_{\text{ri}}$

(L2) $(A, x) \to (C, x)(\gamma, \epsilon) \in P'$ for all $x \in (P_{\text{li}} \cap P_{\text{cf}}) \cup \{\epsilon\}$,
if $p = A[...] \to C[...]\gamma \in P_{\text{li}}$

(3) $(A, \epsilon) \to w \in P'$, if $p = A[\,] \to w \in P$

(R4) $(D, q) \to \epsilon \in P'$, if $q = D[f...] \to \beta B[...] \in P_{\text{ri}}$

(L4) $(D, q) \to \epsilon \in P'$, if $q = D[f...] \to B[...]\beta \in P_{\text{li}}$

The productions of the form (R1) and (R2) (the form (L1) and (L2)) build the set of *passing on rightmost (passing on leftmost)* productions, which is denoted by P'_{r} (by P'_{l}).[10]

In terms of trees, what G' does to a given complete UIG-tree in G is flatten the structure by recursively splitting up spines in a particular way without

[9] For $\alpha \in (N \cup \Sigma)^*$ the string $(\alpha, \epsilon) \in (N' \cup \Sigma)^*$ is inductively defined as follows: First, let $(\epsilon, \epsilon) := \epsilon$. Then, for $\beta \in (N \cup \Sigma)^*$, $a \in \Sigma$ and $A \in N$ set $(a\beta, \epsilon) := a(\beta, \epsilon)$ and $(A\beta, \epsilon) := (A, \epsilon)(\beta, \epsilon)$, respectively.

[10] The expression *passing on rightmost (passing on leftmost)* has been chosen, because the second component of the mother can be considered as being passed onto the rightmost (leftmost) child while applying such a production.

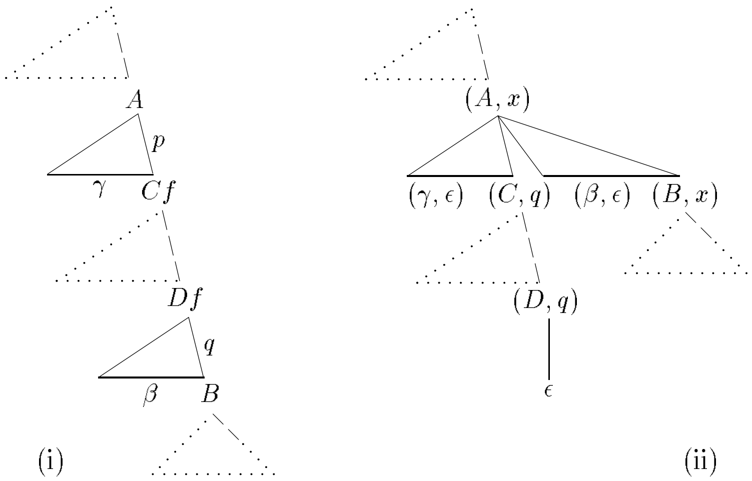

Figure 2: (i) A partial weakly right linear tree in G and (ii) the corresponding CFG-tree in G'.

changing the yield of the original tree. I.e. productions of the form (R2), (L2) and (3) simply imitate the constraint free and terminal productions from the original UIG G. But, a $p' \in P'$ of the form (R1) or (L1) crucially imitates the introduction of an index and its absorption in one step (the "structure–flattening" step). The second component of the corresponding (C, q) stores the information that the index f introduced with C applying $p \in P_{\text{in}}$ has been popped applying $q \in P_{\text{ab}}$. Since $q \neq \epsilon$, no $(Z, q) \in N'$ can subsequently be rewritten as a terminal string, except by using a production of the form (R4) or (L4) at some derivation step. This kind of constraint ensures the "instant popping" of f to be a legitimate operation (see also Figure 2 indicating the case with regard to (R1) and (R4)).[11]

The linear derivation strategy for UIGs can be transferred to G' by means of the following: For any $p' = A' \to \alpha' \in P'_{\text{r}} \cup P'_{\text{l}}$ we define $m(p') = |\alpha'| - 1$. Then we fix existing $X_0, \ldots, X_{m(p')} \in (N' \cup \Sigma)^*$ such that $\alpha' = X_0 \ldots X_{m(p')}$ if $p' \in P'_{\text{r}}$ or such that $\alpha' = X_{m(p')} \ldots X_0$ if $p' \in P'_{\text{l}}$. If $\omega \in \mathbb{N}^*$ is the address of A', for $0 \leq k \leq m(p')$ the address of X_k is ωk. It should be clear what is meant when a derivation d' or a parse π' in G' is called *linear, weakly right linear* or *weakly left linear (w.r.t. $<_{\text{lex}}$)*.

Lemma 3.1 *Let σ', τ' be sentential forms in G'. If $\sigma' \overset{*}{\underset{G'}{\Rightarrow}} \tau'$, there is also a linear derivation from σ' to τ' in G'.* $\square$

[11] Note that a less complex second component would actually be sufficient to prove $L(G) = L(G')$. All that has to be stored is the nonterminal of the left hand side of q and the information whether $q \in P_{\text{ri}}$ or $q \in P_{\text{li}}$. Storing the whole production is useful in order to solve the parsing problem for UIGs.

Proposition 3.2 *It can be shown that*

(a) $A \overset{\text{ui}}{\underset{G}{\Rightarrow}} wX$ *weakly right linear iff* $(A, x) \overset{*}{\underset{G'}{\Rightarrow}} w(X, x)$ *weakly right linear,*

(b) $A \overset{\text{ui}}{\underset{G}{\Rightarrow}} Yw$ *weakly left linear iff* $(A, y) \overset{*}{\underset{G'}{\Rightarrow}} (Y, y)w$ *weakly left linear,*

(c) $\sigma \overset{\text{ui}}{\underset{G}{\Rightarrow}} w$ *linear iff* $(\sigma, \epsilon) \overset{*}{\underset{G'}{\Rightarrow}} w$ *linear,*

f.a. $A, X, Y \in N$, $\sigma \in (N \cup \Sigma)^*$, $w \in \Sigma^*$, *and f.a.* $x \in P_{\text{ri}} \cap P_{\text{ab}}$, $y \in P_{\text{li}} \cap P_{\text{ab}}$.

Thus $L(G) = L(G')$ by Lemma 2.3 and 3.1, since Proposition 3.2 (c) especially holds for $\sigma = S$. Note that a proof of Proposition 3.2 can be done simultaneously for (a),(b), and (c) by an induction on the length of a derivation in G for "only if"– and in G' for the "if"–part of the respective clauses. Loosely speaking, the crucial fact is the connection between partial weakly right (left) linear derivations in G and G' as indicated in Figure 2 displaying the weakly right linear case: applying $q \in P_{\text{ri}}$ to $Df \in NI$ absorbs the instance of $f \in I$ introduced by applying $p \in P_{\text{ri}}$ to $A \in N$.

4 Parsing Complexity

Parsing with a UIG G can be done rather straightforwardly in two steps: First, a linear parse of a string $w \in L(G)$ is found w.r.t. the weakly equivalent CFG G' obtained by the method given in Section 3. Then, each production of this parse is mapped to a production of G yielding a parse of w in the original UIG. We will argue that all this can be done in $O(|w|^3)$–time.

Let $G = (\Sigma, N, I, P, S)$ be a UIG and let $G' = (\Sigma, N', P', S')$ be the weakly equivalent CFG constructed as in Section 3. The function $h : P' \to P$ with $p' \mapsto p$ for $p' \in P'$ is defined by:

(h–R1) $p = A[...] \to \gamma C[f...]$, if $p' = (A, x) \to (\gamma, \epsilon)(C, q)(\beta, \epsilon)(B, x)$

(h–L1) $p = A[...] \to C[f...]\gamma$, if $p' = (A, y) \to (B, y)(\beta, \epsilon)(C, q)(\gamma, \epsilon)$

(h–R2) $p = A[...] \to \gamma C[...]$, if $p' = (A, x) \to (\gamma, \epsilon)(C, x)$

(h–L2) $p = A[...] \to C[f...]\gamma$, if $p' = (A, y) \to (C, y)(\gamma, \epsilon)$

(h–3) $p = A[\,] \to w$, if $p' = (A, \epsilon) \to w$

(h–R4), (h–L4) $p = q$, if $p' = (D, q) \to \epsilon$ and $q \neq \epsilon$

where the interpretation of all symbols is as in (R1)–(L4) from the previous section. h can be extended to a homomorphism $\bar{h} : P'^* \to P^*$ by setting $\bar{h}(\epsilon) = \epsilon$ and $\bar{h}(p'\pi') = h(p')\bar{h}(\pi')$ for all $p' \in P'$, $\pi' \in P'^*$. For $\bar{h}$ a somewhat strengthened version of the "if"–part of Proposition 3.2 (c) holds, stated here for the special case $\sigma = S$ as

Corollary 4.1 *If $\pi' \in P'^*$ is a linear parse of $w \in \Sigma^*$ in G', then $\bar{h}(\pi') \in P^*$ is a linear parse of w in G.*

Lemma 4.2 *For all $w \in \Sigma^*$ the parsing problem w.r.t. G' can be solved in $O(|w|^3)$–time such that: if $w \in L(G')$, a linear parse of w in G' is found.*

Proof. Earley's parsing algorithm, which is known to be executable in cubic time, can easily be modified without changing the complexity in order to give a linear parse as output instead of a right or left parse. The original algorithm, solving the recognition problem for some $w \in \Sigma^*$ of length $n \in \mathbb{N}$, can be adopted without modification. This algorithm constructs lists of items $I_0, \ldots, I_n$ such that for $0 \leq i \leq j \leq n$ an item $[A' \to \alpha' \cdot \beta', i]$ is on list I_j iff $A' \to \alpha'\beta' \in P'$, $S' \underset{G'}{\overset{*}{\Rightarrow}} \gamma' A' \delta'$, $\gamma' \underset{G'}{\overset{*}{\Rightarrow}} a_1 \ldots a_i$ and $\alpha' \underset{G'}{\overset{*}{\Rightarrow}} a_{i+1} \ldots a_j$, where $w = a_1 \ldots a_n$. If $w \in L(G')$, a linear parse π' of w can be found from the parse lists, as exemplified below for cycle–free CFGs (Algorithm 4.3). □

Algorithm 4.3

Input (1) The cycle free CFG $G' = (N', \Sigma, P', S')$;
(2) An input string $w = a_1 a_2 \ldots a_n \in \Sigma^*$;
(3) The parse lists $I_0, I_1, \ldots I_n$ for w.

Output A linear(!) parse π' of w, or an error message.

Method IF no item of the form $[S' \to \alpha' \cdot, 0]$ is on I_n, THEN write "error"; halt. ELSE write (the result of) $\mathbf{F}([S' \to \alpha' \cdot, 0], n, \epsilon)$; halt.

$\mathbf{F}([A' \to \beta' \cdot, i], j, \omega)$:

```
p' := A' → β' ; π' := ε ; β' = X_0 X_1 ... X_{m-1}; k := m − 1 ; l := j;
IF p' ∈ P'_r THEN ι := m − 1 ELSE ι := 0
REPEAT until k = −1:
  IF X_k ∈ Σ THEN
    k := k − 1 ; l := l − 1;
    IF p' ∈ P'_r THEN ι := ι − 1 ELSE ι := ι + 1
  ELSE IF X_k ∈ N' THEN
    find [X_k → γ'·, r] ∈ I_l for some r such that
    [A' → X_0 X_1 ... X_{k-1}·X_k ... X_{m-1}, i] ∈ I_r;
    IF p' ∈ P'_r THEN
      π' := F([X_k → γ'·, r], l, ωι) π';
    ELSE
      π' := π' F([X_k → γ'·, r], l, ωι);
    k := k − 1 ; l := r;
    IF p' ∈ P'_r THEN ι := ι − 1 ELSE ι := ι + 1
RETURN p'π'
```

Proposition 4.4 *Let $G = (\Sigma, N, I, P, S)$ be a UIG, and let $w \in \Sigma^*$. Then, the parsing problem for w w.r.t. G is solvable in $O(|w|^3)$–time.*

Proof. First, we construct the weakly equivalent CFG $G' = (\Sigma', N, P', S')$ as given in (R1)–(L4) and compute $h : P' \to P$ as given in (h–R1)–(h–L4). This of course takes $O(|w|^0)$–time depending on the cardinality of N and P. According to Lemma 4.2 the parsing problem for w w.r.t. G' is solvable in $O(|w|^3)$–time such that, if $w \in L(G') = L(G)$, a linear parse π' for w is found. Since we may assume that $|\pi'|$ is in $O(|w|)$, we conclude that the computation of $\bar{h}(\pi')$ lasts $O(|w|)$–time. From Corollary 4.1 we conclude that $\bar{h}(\pi')$ is a parse of w in G. □

Comparing our parsing procedure with a strategy to parse full LIGs one might ask what is common to them. Looking at e.g. the LIG–parser of Vijay-Shanker and Weir (1991) we observe that, roughly speaking, a nonterminal object $Af\zeta$ and a distinguished descendant $B\zeta$ of $Af\zeta$ are stored as a pair, if the lower part ζ of the stack is unaffected while deriving $B\zeta$. The efficiency of the parser crucially depends on storing at most the topmost stack symbol together with the associated nonterminal instead of a complete nonterminal object. This is essentially the same as what is done by our transformation algorithm. The property, however, that takes responsibility for the smaller complexity of the UIG–parser is, informally speaking, that the correct position of the string which rewrites the distinguished descendant while popping the index f is already known at the moment f is introduced. This is obviously not the case for full LIGs.

5 Final Remarks

We have identified a weakly context free subclass of LIGs, namely UIGs, which still permits to characterize sets of structures that cannot be generated by any CFG (see e.g. Example 2.2). Further we have presented a parsing strategy for UIGs working in cubic time. Because the class of LIGs is weakly equivalent to the class of *tree adjoining grammars (TAGs)*,[12] our work is directly comparable to similar work of Rogers (1994) and of Schabes and Waters (1995). Both approaches constrain the original TAG–formalism yielding a cubic time parsable formalism which is weakly equivalent to CFGs.

Rogers' work contrasts with our result more generally. In particular he defines a subclass of TAGs such that each grammar (in terms of trees) generates a *recognizable set*, and thus is strongly equivalent *(modulo projection)* to some CFG (Thatcher 1967). Interesting parallels to the work of Schabes and Waters (1995) arise from their way of restricting the general relation of tree adjunction defined for TAGs obtaining the class of *tree insertion grammars (TIGs)*. There is arguably a strong connection between their constraints on left and right adjunction and our restrictions on index inheritance. But the

[12] TAGs as defined e.g. in Vijay-Shanker & Weir 1994, where also a proof of the mentioned weak equivalence can be found.

classes of tree sets that can be characterized by TIGs and UIGs, respectively, are different. On the one hand, unlike in the case of TIGs, there exist e.g. ϵ– and cycle–free CFGs such that no UIG generates the same set of structures. This is due to the restricted form of the non–terminal productions in a UIG. Recall that, apart from index symbols, the rightmost (leftmost) element of the righthand side of a right (left) indexed production has to be a nonterminal. On the other hand, although it gives rise to various non–recognizable sets, the TIG-formalism as proposed in Schabes & Waters 1995 does not give rise to each such set of structures licensed by some UIG (e.g. the set of trees corresponding to Example 2.2). Up to nonterminal labeling this could be achieved adding TAG–like adjunction constraints to the TIG–formalism, allowing the adjunction of a tree from some distinguished set of trees at a particular node labeled A rather than the adjunction of an arbitrary auxiliary tree with root and foot labeled A. In the UIG–formalism, roughly speaking, this kind of constraints is already available by the index mechanism as defined.

References

Boullier, Pierre. 1996. Another facet of LIG parsing. In: *34th Annual Meeting of the Association for Computational Linguistics (ACL '96)*, Santa Cruz, CA, pp. 87–94. ACL.

Gazdar, Gerald. 1988. Applicability of indexed grammars to natural languages. In: U. Reyle and C. Rohrer (editors), *Natural Language Parsing and Linguistic Theories*, pp. 69–94. D. Reidel Publishing Company, Dordrecht.

Michaelis, Jens and Christian Wartena. 1997. How linguistic constraints on movement conspire to yield languages analyzable with a restricted form of LIGs. In: *Proceedings of the Conference on Formal Grammar (FG '97)*, Aix en Provence, pp. 158–168.

Rogers, James. 1994. Capturing CFLs with tree adjoining grammars. In: *32th Annual Meeting of the Association for Computational Linguistics (ACL '94)*, Las Cruces, NM, pp. 155–162. ACL.

Schabes, Yves and Richard C. Waters. 1995. Tree insertion grammar: A cubic–time parsable formalism that lexicalizes context–free grammar without changing the trees produced. *Computational Linguistics*, 21:479–513.

Thatcher, James W. 1967. Characterizing derivation trees of context–free grammars through a generalization of finite automata theory. *Journal of Computer and System Sciences*, 1:317–322.

Vijay-Shanker, K. and David J. Weir. 1991. Polynomial parsing of extensions of context-free grammars. In: M. Tomita (editor), *Current Issues in Parsing Technology*, pp. 191–206. Kluwer Acadamic Publishers, Norwell, MA.

Vijay-Shanker, K. and David J. Weir. 1994. The equivalence of four extensions of context-free grammars. *Mathematical Systems Theory*, 27:511–546.