New Frontiers beyond Context-freeness: DI-Grammars and DI-Automata.

Peter Staudacher
Institut für Allgemeine und Indogermanische
Sprachwissenschaft
Universität Regensburg
Postfach 397
8400 Regensburg 1
Germany

Abstract

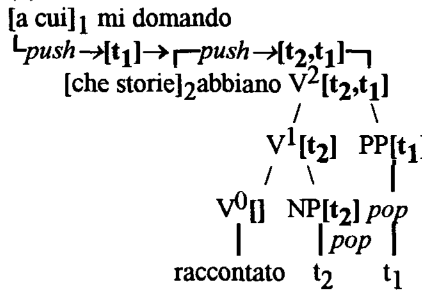
A new class of formal languages will be defined - the Distributed Index Languages (DI-languages). The grammar-formalism generating the new class - the DI-grammars - cover unbound dependencies in a rather natural way. The place of DI-languages in the Chomsky-hierarchy will be determined: Like Aho's indexed Languages, DI-languages represent a proper subclass of Type 1 (contextsensitive languages) and properly include Type 2 (context-free languages), but the DI-class is neither a subclass nor a superclass of Aho's indexed class. It will be shown that, apart from DI-grammars, DI-languages can equivalently be characterized by a special type of automata - DI-automata. Finally, the time complexity of the recognition-problem for an interesting subclass of DI-Grammars will approximately be determined.

1 Introduction

It is common practice to parse nested Wh-dependencies, like the classical example of Rizzi (1982) in (1),

(1) Tuo fratello, [a cui]$_1$ mi domando [che storie]$_2$ abbiano raccontato t_2 t_1, era molto preoccupato
(Your Brother, [to whom]$_1$ I wonder [which stories]$_2$ they told t_2 t_1 was very troubled)

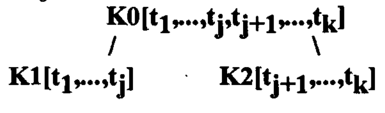
using a stack mechanism. Under the binary branching hypothesis the relevant structure of (1) augmented by wh-stacks is as follows:

(2)


Up to now it is unclear, how far beyond context-freeness the generative power of a Type 2 grammar formalism is being extended if such a stack mechanism is grafted on it (assuming, of course, that an upper bound for the size of the stack can not be motivated).

Fernando Pereira's concept of Extraposition Grammar (XG), introduced in his influential paper (Pereira, 1981; 1983; cf. Stabler, 1987) in order to delimit the new territory, can be shown to be inadequate for this purpose, since it is provable that the class of languages generable by XGs coincides with Type 0 (i.e. XGs have the power of Turing machines), whereas the increase of power by the stack mechanism is not even enough to generate all Type 1 languages (see below).

In (2) an additional point is illustrated: the stack [t_2,t_1] belonging to V^2 has to be divided into the substacks [t_2] and [t_1], which are then inherited by the daughters V^1 and PP. For the PP-index t_1 is not discharged from the top of the V^2-stack [t_2,t_1]. Generalizing to stacks of unlimited size, the partition of a stack among the inheriting subconstituents K_1 and K_2 of a constituent K_0 is as in (3)

(3)


If the generalization in (3) is tenable, the extension of context-free grammars (Vijay-Shanker and Weir, 1991, call the resulting formalism "linear indexed grammar" (LIG)) discussed by Gazdar in (Gazdar, 1988), in which stacks are exclusively passed over to a single daughter (as in (3.1)), is too weak.

(3.1) a) $$\begin{tikzcd} & K_0[t_1,\ldots,t_k] \arrow[dl, dash] \arrow[dr, dash] & \\ K_1[t_1,\ldots,t_k] & & K_2 \end{tikzcd}$$ b) $$\begin{tikzcd} & K_0[t_1,\ldots,t_k] \arrow[dl, dash] \arrow[dr, dash] & \\ K_1 & & K_2[t_1,\ldots,t_k] \end{tikzcd}$$

Stack-transmission by *distribution*, however, as in (3) suggests the definition of a new class of grammars properly containing the context-free class.

2 DI-Grammars and DI-languages

A DI-grammar is a 5-tupel G= (N,T,F,P,S), where N,T,S are as usual, F is a alphabet of indices, P is a set of rules of the following form

1) (a) $A \rightarrow \alpha$ (b) $A \rightarrow \alpha B f \beta$ (c) $Af \rightarrow \alpha$,
(A, B∈N; α, ß∈(N∪T)*;f∈F)

The relation "= >" or "directly derives" is defined as follows:

2) $\alpha => \beta$
if either i)
$\alpha = \delta A\,index\,\gamma$, $\delta,\gamma \in (NF^*\cup T)^*$, $index \in F^*$, $A \in N$,
$A \rightarrow B_1B_2\ldots B_n$ is a rule of form 1)(a)
$\beta = \delta B_1 index_1 B_2 index_2 \ldots B_n index_n \gamma$

or ii)
$\alpha = \delta A\,index\,\gamma$, $\delta,\gamma \in (NF^*\cup T)^*$, $index \in F^*$, $A \in N$,
$A \rightarrow B_1 .. B_k f \ldots B_n$ is a rule of form 1)(b), $f \in F$
$\beta = \delta B_1 index_1 \ldots B_k f index_k \ldots B_n index_n \gamma$

or iii)
$\alpha = \delta A f index\,\gamma$,δ ,$\gamma \in (NF^*\cup T)^*$, $index \in F^*$, $A \in N$,
$Af \rightarrow B_1B_2\ldots B_n$ is a rule of form 1)(c), $f \in F$
$\beta = \delta B_1 index_1 B_2 index_2 \ldots B_n index_n \gamma$

(*) and $index = index_1 index_2 \ldots index_n$,
and for $B_i \in T$: $index_i = \varepsilon$ (i.e. the empty word) $(0 \leq i \leq n)$

The reflexive and transitive closure *=> of => is defined as usual.

Replacing (*) by"$index_i = index$ for $B_i \in N$, $index_i = \varepsilon$ for $B_i \in T$", changes the above definition into a definition of Aho's well known indexed grammars. How index-percolation differs in indexed and Di-grammars is illustrated in (4).

(4) **Index-Percolation**

(i) *in Aho's Indexed-Grammars*

$$\begin{tikzcd} & Mf_1f_2f_3f_4 \arrow[dl, dash] \arrow[dr, dash] & \\ Lf_1f_2f_3f_4 & & Rf_1f_2f_3f_4 \end{tikzcd}$$

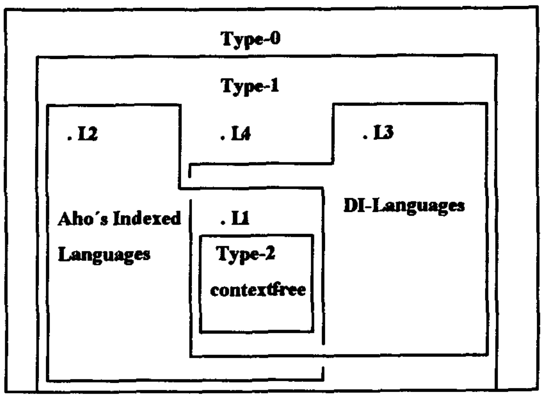
(ii) *in DI-Grammars*

$$\begin{tikzcd} & Mf_1f_2f_3f_4 \arrow[dl, dash] \arrow[dr, dash] & \\ Lf_1f_2 & & Rf_3f_4 \end{tikzcd}$$

i.e:index multiplication vs. index distribution

The region in the Chomsky hierarchy occupied by the class of DI-languages is indicated in (5)

(5)



where
(5.1) $L_1 = \{a^nb^nc^n;\ n\geq 1\}$
(5.2) $L_2 = \{a^k;\ k = 2^n,\ 0 \leq n\}$
(5.3) $L_3 = \{w_1w_2\ldots w_nz_1w_n\ldots z_nw_1z_{n+1}m(w_n)m(w_{n-1})\ldots m(w_2)m(w_1);\ n\geq 1\ \&\ w_i\in\{a,b\}^+\ (1\leq i\leq n)\ \&\ z_1z_2\ldots z_nz_{n+1} \in D_1\}$
$m(y)$ is the mirror image of y and D_1 is the Dyck language generated by the following CFG G_k (D_1=L(G_k)), $G_k = (\{S\},\{[,]\},R_k, S)$, where $R_k = \{S \rightarrow [S],\ S \rightarrow SS,\ S \rightarrow \varepsilon\}$

(5.4) $L_4 = \{a^k;\ k = n^n,\ n\geq 1\}$; ($L_4$ is not an indexed language, s. Takeshi Hayashi (1973)).

By definition (see above), the intersection of the class of indexed languages and the class of DI-languages includes the context-free (cfr) languages. The inclusion is proper, since the (non-cfr) language L_1 is generated by $G_1 = (\{S,A,B\}, \{a,b,c\}, \{f,g\}, R_1, S)$, where $R_1 = \{S \rightarrow aAfc,\ A \rightarrow aAgc,\ A \rightarrow B,\ Bg \rightarrow bB,\ Bf \rightarrow b\}$, and G_1 obviously is both a DI-grammar and and an indexed grammar.-

Like cfr. languages and unlike indexed languages, DI-languages have the constant growth property (i.e. for every DI-grammar G there exists a k∈N, s.th. for every w∈L(G), s.th. |w|>k, there exists a sequence w_1

(=w), $w_2,w_3,...(w_i \in L(G))$, such that $|w_n| < |w_{n+1}| < (n+1)\times|w|$ for every member w_n of the sequence). Hence L2, and a fortiori L4, is not a DI-language. But L2 is an indexed language, since it is generated by the indexed grammar $G_2 =(\{S,A,D\}, \{a\}, \{f,g\}, R_2, S)$, where $R_2 = \{S \to Ag, A \to Af, A \to D, Df \to DD, Dg \to a\}$.

L_3 is a DI-language, since it is generated by the DI-grammar $G3 = (\{S,M,Z\},\{a,b,[,]\},\{f,g\},R_3,S)$ where
$R_3 = \{S \to aS/a, M \to [M], Z/ \to Za, Zg \to Zb,$
$S \to bSgb, M \to MM, Z/ \to a, Zg \to b$
$S \to M, \quad M \to Z\}$
e.g. abb[b[ab]]bba ($\in L_3$) is derived as follows:

S ⇒ aS/a ⇒ abSg/ba ⇒ abbSgg/bba ⇒ abbMgg/bba ⇒ abb[Mgg/]bba ⇒ abb[MgMg/]bba (*here the index "gg/" has been distributed*) ⇒ abb[ZgMg/]bba ⇒ abb[bMg/]bba ⇒ abb[b[Mg/]]bba ⇒ abb[b[Zg/]]bba ⇒ abb[b[Z/b]]bba ⇒ abb[b[ab]]bba.

2.1 DI-Grammars and Indexed Grammars

Considering the well known generative strength of indexed grammars, it is by no means obvious that L_3 is not an indexed language. In view of the complexity of the proof that L_3 is not indexed, only some important points can be indicated - referring to the 3 main parts of every word $x \in L_3$ by x_l, $[x_m]$,x_r,as illustrated in the example (6):

(6)
```
 ab  abb abbb abbbb[[abbbb[[abbb]abb]]ab]bbbbabbbabbaba
└w1┘└w2┘└w3┘└w4─┘ └─w4─┘└w3─┘└w2┘└w1┘
└──────x1──────┘└──────[xm]──────┘└─────xr─────┘
                  = x
```

Assume that there is a indexed grammar G_I= (N,T,F,P,S) such that L_3=L(G_I):

1. Since G_I can not be contextfree, it follows from the intercalation (or "pumping") lemma for indexed grammars proved by Takeshi Hayashi in (Hayashi, 1973) that there exists for G_I an integer k such that for any $x \in L_3$ such that $|x|>k$ a derivation can be found with the following properties:

$$\begin{aligned} S &=*=> zAf\eta z' =*=> zs_1Af\mu' f\eta s_1'z' \\ &=*=> zs_1r_1Af\mu' f\eta r_1's_1'z' =*=> zs_1r_1Bf\mu' f\eta r_1's_1'z' \\ &=*=> zs_1rt_1Bf\eta t_1'r's_1'z' =*=> x, \\ &(zz', r_1r_1' \in T^*, s_1t_1t_1's_1' \in T^+, f \in F, \mu, \eta \in F^*) \end{aligned}$$

By intercalating subderivations which can effectively be constructed this derivation can be extended as follows

$$\begin{aligned} S &=*=> zAf\eta z' =*=> zs_1Af\mu' f\eta s_1'z' \\ &=*=> zs_1...s_nA(f\mu')^nf\eta s_n...s_1'z' \\ &=*=> zs_1...s_nr_nB(f\mu')^nf\eta r_n's_n...s_1'z' \\ &=*=> zs_1...s_nr_nt_n...t_1Bf\eta t_1'...t_n'r_n's_n'...s_1'z' \\ &=*=> zs_1...s_nr_nt_n...t_1wt_1'...t_n'r_n's_n'...s_1'z' \end{aligned}$$

The interdependently extendible parts of x $s_1...s_n$, $t_n...t_1$, $t_1'...t_n'$, r_nr_n', and $s_n'...s_1'$, can not all be subwords of the central component $[x_m]$ of x (or all be subwords of the peripheral components x_lx_r), else, $[x_m]$ (or x_lx_r) could be increased and decreased independently of the peripheral components x_l and x_r (or of $[x_m]$, respectively) of x , contradicting the assumption that $x \in L_3$. Rather, the structure of x necessitates that $s_1...s_n$ and $s_n'...s_1'$ be subwords of x_lx_r and that the "pumped" index $(f\mu')^n$ be discharged deriving the central component $[x_m]$. Thus, we know that for every $l>0$ there exists an index $\mu \in F^+$, a $x \in L_3$, and a subword $[x_m']$ of the central part $[x_m]$ of x such that $[x_m']>l$ and $M\mu$=*=>$[x_m']$ (M=B or the nonterminal of a descendant of $A(f\mu')^nf\eta$). To simplify our exposition we write $[x_m']$ instead of $[x_m]$ and have

(7) $M\mu$ =*=> $[x_m]$

with the structure of x_l and x_r being encoded and stored in the index μ.

2. The balanced parentheses of $[x_m]$ can not be encoded in the index μ in (7) in such a manner that $[x_m]$ is a homomorphic image of μ. For the set $I=\{\mu'; S=*=>x_lM\mu'x_r =*=>x_l[x_m]x_r \in L_3\}$ of all indices which satisfy (7) is regular (or of Type 3), but because of the Dyck-structure of $[x_m]$, $L_M=\{[x_m];x_l[x_m]x_r \in L_3\}$ is not regular but essentially context-free or of Type 2.

3. In the derivation underlying (7) essential use of branching rules of the type $A \to B_1B_2...B_k$ $(k\geq2)$ has to be made in the sense that the effect of the rules can not be simulated by linear rules. Else the central part $[x_m]$ could only have linear and not trans-linear Dyck-structure. Without branching rules the required trans-linear parenthetical structure could only be generated by the use of additional index-introducing rules in (7), in order to "store" and coordinate parentheses, which, however, would destroy the dependence of $[x_m]$ from x_l and x_r.

4.For every $n\geq1$, L_3 contains words w of the form

(8)

$$w_1..w_k\underbrace{[[[}_{n+1}..[[[[w_k][w_{k-1}]][[w_{k-2}][w_{k-3}]]]..\underbrace{]]]]}_{n+1}..]]]m(w_k)..m(w_1)$$

where $k=2^n$, $w_i \in \{a,b\}^+$ for $1 \leq i \leq 2^n$; $m(w_i)$ is the mirror image of w_i.
i.e the central part $[x_m]$ of such a word contains $2^{n+1}-1$ pairs of parentheses, as shown in (9) for n=3:

(9) $[[[[w_8][w_7]][[w_6][w_5]]][[[w_4][w_3]][[w_2][w_1]]]]$

According to our assumption, G_I generates all words having the form (8). Referring to the derivation in (7), consider a path from $M\mu$ to any of the parenthesized parts w_i of $[x_m]$ in (8). (Ignoring for expositional purposes the possibility of "storing" (a constant amount of) parentheses in nonterminal nodes,) because of 2. and 3. an injective mapping can be defined from the set of pairs of parentheses containing at least two other (and because of the structure of (8) disjunct) pairs of parentheses into the set of branching nodes with (at least) two nonterminal daughters . Call a node in the range of the mapping a *P-Node*. Assuming without loss of generality that each node has at most two nonterminal daughters, there are 2^n-1 such P-nodes in the subtree rooted in $M\mu$ and yielding the parenthesized part $[x_m]$ of (8). Furthermore, every path from $M\mu$ to the root W_i of the subtree yielding $[w_i]$ contains exactly n P-nodes (where 2^n=k in (8)).

Call an index-symbol f inside the index-stack μ a *w_i-index* if f is discharged into a terminal constituting a parenthesized w_i in (8) (or equivalently, if f encodes a symbol of the peripheral $x_1..x_r$).

Let f_t be the first (or leftmost) w_i-index from above in the index-stack μ, and let w_t be the subword of $[x_m]$ containing the terminal into which f_t is discharged, i.e all other w_i-indices in μ are only accessible after f_t has been consumed. Thus, for $\mu=\pi f_t\sigma$ we get from (7)

$$M\pi f_t\sigma =+=> uB_t[\tau_t f_t\sigma]v =+=> u_t W_t[\tau f_t\sigma]v_t \text{ and}$$
$$W_t[\tau f_t\sigma] =+=> w_t$$

The path P_t from $M\mu$ to w_t contains n B-nodes, for $k=2^n$ in (8). For every B-node B_j ($0 \leq j < n$) of P_t we obtain because of the index-multiplication effected by noterminal branching:

$$B_j[\tau_j f_t\sigma] => L_j[\tau_j f_t\sigma]R_j[\tau_j f_t\sigma] \text{ and}$$
$$L_j[\tau_j f_t\sigma] =*=> u_{j+1}B_{j+1}[\tau_{j+1} f_t\sigma]v_{j+1}$$
$(B_j, B_{j+1}, L_j, R_j \in N, \tau_j, \tau_{j+1}, \sigma \in F^*, f_t \in F, u_{j+1}, v_{j+1} \in \{a, b, [,]\}^*)$

Every path P_j branching off from P_t at $B_j[\tau_j f_t\sigma]$ leads to a word w_j derived exclusively by discharging w_i-indices situated in μ below (or on the right side of) f_t. Consequently, f_t has to be deleted on every such path P_j, before the appropriate indices become accessible, i.e. we get for every j with $0 \leq j < n$:

$$B_j[\tau_j f_t\sigma] => u_j R_j[\tau_j f_t\sigma]y_j =*=> y_j C_j[f_t\sigma]z_j,$$
$(B_j, R_j, C_j \in N, \tau_j, \sigma \in F^*, f_t \in F)$

Thus, for $n>|N|$ in (8) ($|N|$ the cardinality of the nonterminal alphabet N of G_I, ignoring, as before the constant amount of parenthesis-storing in nonterminals) because of $|\{C_j; 0 \leq j < n\}|=n$ the node-label $C_j[f_t\sigma]$ occurs twice on two different paths branching off from P_t , i.e. there exist p, q ($0 \leq p < q < n$) such that:

$$M\pi f_t\sigma =+=> u_p R_p[\tau_p f_t\sigma]v_q R_q[\tau_q f_t\sigma]y \quad \text{and}$$
$$R_p[\tau_p f_t\sigma] =*=> y_p C[f_t\sigma]z_p =+=> y_p z z_p$$
$$R_q[\tau_q f_t\sigma] =*=> y_q C[f_t\sigma]z_q =+=> y_q z z_q,$$
$(\mu=\pi f_t\sigma,\ \sigma_p, \tau_q, \sigma \in F^*, f_t \in F;\ M, R_p, R_q, C \in N;\ u_p, v_q, y, y_q, y_p, z_p, z_q, z \in T^*)$
where $z \in \{z_1 w_1 \ldots z_r w_r z_{r+1}; w_i \in \{a,b\}^+ \& z_1 \ldots z_{r+1} \in D_1$ (= the Dyck-language from (5.3)$\}$.
I.e. G_I generates words w" =x_l"[x_m"]x_r", the central part of which contain a duplication (of "z" in [x_m"]=$y_1 z y_2 z y_3$) without correspondence in x_l" or x_r", thus contradicting the general form of words of L_3. Hence L_3 is not indexed.

2.2 DI-Grammars and Linear Indexed Grammars[1]

As already mentioned above, Gazdar in (Gazdar, 1988) introduced and discussed a grammar formalism, afterwards (e.g. in (Weir and Joshi, 1988)) called linear indexed grammars (LIG's), using index stacks in which only one nonterminal on the right-hand-side of a rule can inherit the stack from the left-hand-side, i.e. the rules of a LIG G=(N,T, F, P, S) with N,F,T,S as above, are of the Form

i. $A[..] \rightarrow A_1[]..A_i[..]..A_n$
ii. $A[..] \rightarrow A_1[]..A_i[f..]..A_n$
iii. $A[f..] \rightarrow A_1[]..A_i[..]..A_n$
iv. $A[] \quad \rightarrow a$

where $A_1,\ldots,A_n \in N$, $f \in F$, and $a \in T \cup \{\varepsilon\}$. The "derives"-relation => is defined as follows

$$\alpha A[f_1..f_n]\beta => \alpha A_1[]..A[f_1..f_n]..A_n[]\beta$$
$$\text{if } A[..] \rightarrow A_1[]..A_i[..]..A_n \in P$$

[1]Thanks to the anonymous referees for suggestions for this section and the next one.

$\alpha A[f_1..f_n]\beta \Rightarrow \alpha A_1[]..A[ff_1..f_n]..A_n[]\beta$
    if $A[..] \to A_1[]..A_i[f..]..A_n \in P$
$\alpha A[ff_1..f_n]\beta \Rightarrow \alpha A_1[]..A[f_1..f_n]..A_n[]\beta$
    if $A[f..] \to A_1[]..A_i[..]..A_n \in P$
$\alpha A[]\beta \Rightarrow \alpha a\beta$
    if $A[] \to a \in P$
=*=> is the reflexive and transitive closure of =>, and $L(G)=\{w;\ w \in T^* \ \&\ S[] =^*\Rightarrow w\}$.
Gazdar has shown that LIGs are a (proper) subclass of indexed grammars. Joshi, Vijay-Shanker, and Weir (Joshi, Vijay-Shanker, and Weir, 1989; Weir and Joshi, 1988) have shown that LIGs, Combinatory Categorial Grammars (CCG), Tree Adjoinig Grammars (TAGs), and Head Grammars (HGs) are weakly equivalent. Thus, if an inclusion relation can be shown to hold between DI-languages (DIL) and LILs, it simultaneously holds between the DIL-class and all members of the family.

To simulate the restriction on stack transmission in a LIG $G_l=(N_l, T, F_l, P_l, S_l)$ the following construction of a DI-grammar G_d suggests itself:
Let $G_d=(N, T, F, P, S)$ where $N-\{S\}=\{X';\ X \in N_l\}$, $F=\{f';\ f \in F_l\} \cup \{\#\}$, and $P=\{S \to S_l'\#\}$
$\cup\{A' \to A_1'\#...A_i'...A_n'\#;\ A[..] \to A_1[]..A_i[..]..A_n \in P_l\}$
$\cup\{A' \to A_1'\#...A_i'f'...A_n'\#; A[..] \to A_1[].A_i[f..].A_n \in P_l\}$
$\cup\{A'f' \to A_1'\#...A_i'...A_n'\#; A[f..] \to A_1[].A_i[..].A_n \in P_l\}$
$\cup\{A'\# \to a;\ A[] \to a \in P_l\}$
It follows by induction on the number of derivation steps that for $X' \in N$, $X \in N_I$, $\mu' \in F^*$, $\mu \in F_I^*$, and $w \in T^*$
(10) $X'\mu'\# =^*{}_G\Rightarrow w$ if and only if $X[\mu] =^*{}_{G_l}\Rightarrow w$
where $X'=h(X)$ and $\mu'=h(\mu)$ (h is the homomorphism from $(N_l \cup F_l)^*$ into $(N \cup F)^*$ with $h(Z)=Z'$). For the nontrivial part of the induction, note that $A'\#\mu'$ can not be terminated in G.
Together with $S \Rightarrow S_1'\#$ (10) yields $L(G_I)=L(G)$.

The inclusion of the LIG-class in the DI-class is proper, since L_3 above is not a LIG-language, or to give a more simple example:
$L_w=\{a^n a_1^{n1} a_2^{n1} b_1^{n2} b_2^{n2} b^n \mid n=n1+n2\}$ is according to (Vijay-Shanker, Weir and Joshi, 1987) not in TAL, hence not in LIL. But (the indexed langauge) L_w is generated by the DI-Grammar
$G_w=(\{S,A,B\},\{a,b,a_1,a_2,b_1,b_2\},\{S \to aSfb, S \to AB, Af \to a_1Aa_2, Bf \to b_1Bb_2, Af \to a_1a_2, Bf \to b_1b_2, A \to \varepsilon, B \to \varepsilon\},S)$.

2.3 Generalized Composition and Combinatory Categorial Grammars

The relation of DI-grammars to Steedman's Combinatory Categorial Grammars with Generalized Composition (GC-CCG for short) in the sense of (Weir and Joshi, 1988) is not so easy to determine. If for each $n \geq 1$ composition rules of the form

$(x/y)\ (...(y|_1z_1|)|_2...|_nz_n) \to (...(x|_1z_1|)|_2...|_nz_n)$ and
$(...(y|_1z_1|)|_2...|_nz_n)\ (x\backslash y) \to (...(x|_1z_1|)|_2...|_nz_n)$

are permitted, the generative power of the resulting grammars is known to be stronger than TAGs (Weir and Joshi, 1988).

Now, the GC-CCG given by

$f(\varepsilon)=\{\#\}$ $f(a_1)=\{S/X/\#,\ S/X\backslash\#,\ \#/X/\#, \#/X\backslash\#\}$
$f(a)=\{A, X\backslash A\}$ $f(b_1)=\{S/Y/\#,\ S/Y\backslash\#,\ \#/Y/\#, \#/Y\backslash\#\}$
$f(b)=\{B, Y\backslash B\}$ $f([)=\{K\}$ $f(])=\{\#/\#\backslash K,\ \#\backslash\#\backslash K\}$

generates a language L_c, which when intersected with the regular set

$$\{a,b\}^+\{[,],a_1,b_1\}^+\{a,b\}^+$$

yields a language L_p which is for similar reasons as L_3 not even an indexed language. But L_p does not seem to be a DI-language either. Hence, since indexed languages and DI-languages are closed under intersection with regular sets., L_c is neither an indexed nor (so it appears) a DI-language.

The problem of a comparison of DI-grammars and GC-CCGs is that, inspite of all appearances, the combination of generalized forward and backward composition can not directly simulate nor be simulated by index-distribution, at least so it seems.

3 DI-Automata

An alternative method of characterizing DI-languages is by means of DI-automata defined below.

DI-automata (dia) have a remote resemblance to Aho's *nested stack automata (nsa)*. They can best be viewed as push down automata (pda) with additional power: they can not only read and write on top of their push down store, but also travel down the stack and (recursively) create new embedded substacks (which can be left only after deletion). *dia's* and *nsa's* differ in the following respects:

1. a *dia* is only allowed to begin to travel down the stack or enter the stack reading mode, if a tape-symbol A on top of the stack has been deleted and stored in a special stack-reading-state q_A, and the stack-reading mode has to be terminated as soon as the first index-symbol *f* from above is being scanned, in which case the index-symbol concerned is deleted and an embedded stack is created, provided the transition-function gives permission. Thus, every occurrence of an index-symbol on the stack can only be "consumed" once, and

only in combination with a "matching" non-index-symbol.

A *nsa*, on the other hand, embeds new stacks behind tape symbols which are preserved and can, thus, be used for further stack-embeddings. This provides for part of the stack multiplication effect.

2. Moving through the stack in the stack reading mode, a *dia* is not allowed to pass or skip an index symbol. Moreover, no scanning of the input or change of state is permitted in this mode.

A *nsa*, however, is allowed both to scan its input and change its state in the stack reading mode, which, together with the license to pass tape symbols repeatedly, provides for another part of the stack multiplication effect.

3. Unlike a *nsa*, a *dia* needs two tape alphabets, since only "index symbols" can be replaced by new stacks, moreover it requires two sets of states in order to distinguish the pushdown mode from the stack reading mode.

Formally, a di-automaton is a 10-tuple $D = \{q, Q_\Gamma, T, \Gamma, I, \delta, Z_0, \$, ¢, \#)$,
where q is the *control state for the pushdown mode*,
$Q_\Gamma = \{q_A;\ A \varepsilon \Gamma\}$ a finite set of *stack reading states*,
T a finite set of *input symbols*,
Γ a finite set of *storage symbols*,
I a finite set of *index symbols* where $I \cap \Gamma = \emptyset$,
$Z_0 \in \Gamma$ is the *initial storage symbol*,
$\$$ is the *top-of-stack marker* on the storage tape,
$¢$ is the *bottom-of embedded stack marker* on the storage tape,
$\#$ marks the *bottom of the storage tape*, where $\$, ¢, \# \notin \Gamma \cup T \cup I$,
$Dir = \{-1,0,1\}$ (for "1 step upwards","stay","1 step downwards", respectivly,
$E = \{0,1\}$ ("halt input tape", "shift input tape", respectively),
$T' = T \cup \{\#\}$, $\Gamma' = \Gamma \cup \{¢\}$,
δ is a mapping
1) in the push down mode:
from $\{q\} \times T' \times \$\Gamma$ into finite subsets of
$\{q\} \times D \times \$\Gamma((\Gamma \cup I)^*)$
2) in the stack reading mode: for every $A \in \Gamma$
(a)from $\{q_A\} \times T' \times \Gamma'$ into subsets of $\{q_A\} \times \{0\} \times \{1\}$
(for walking down the stack)
(b)from $\{q\} \times T' \times \$\{A\}$ into subsets of $\{q_A\} \times \{0\} \times \{1\}$
(for initiating the stack reading mode)
(c) from $\{q\} \times T' \times \{A\}$ into subsets of $\{q\} \times \{0\} \times \{-1\}$
(for climbing up the stack)
3) in the stack creation mode:
from $Q_\Gamma \times T' \times I$ into finite subsets of
$\{q\} \times \{0\} \times \$\Gamma((\Gamma \cup I)^*)¢$, and from $Q_\Gamma \times T' \times \$I$ into finite subsets of $\{q\} \times \{0\} \times \$\$\Gamma((\Gamma \cup I)^*)¢$ (for replacing index symbols by new stacks, preserving the top-of-stack marker $\$$)
4) in the stack destruction mode:
from $\{q\} \times T' \times \{\$¢\}$ into subsets of $\{q\} \times \{0\}$.

As in the case of Aho's *nested stack* automaton a *configuration* of a DI-automaton D is a quadruple $(p, a_1....a_n\#, i, X_1...\hat{}X_j...X_m)$,
where
1. $p \in \{q\} \cup Q_\Gamma$ is the current state of D;
2. $a_1...a_n$ is the input string, # the input endmarker;
3. i $(1 \leq i \leq n+1)$ the position of the symbol on the input tape currently being scanned by the input head $(=a_i)$;
4. $X_1...\hat{}X_j...X_m$ the content of the storage tape where for $m>1$ $X_1=\$A$, $A \varepsilon \Gamma$, $X_m=\#$, $X_2...X_{m-1} \in (\Gamma \cup I \cup \{\$, ¢\})^*$; X_j is the stack symbol currrently being read by the storage tape head. If m=1, then $X_m=\$\#$.
As usual, a relation $\vdash_D$ representing a move by the automaton is defined over the set of configurations:

(i)$(q, a_1...a_n\#, i, \alpha\$\hat{}AY\beta)$
$\vdash_D (q, a_1...a_n\#, i+d, \alpha\$\hat{}Z_1...Z_kY\beta)$,
if $(q, d, \$Z_1...Z_k) \in \delta(q, a_i, \$A)$.

(ii)$(p, a_1...a_n\#, i, X_1...\hat{}X_j...X_m)$
$\vdash_D (q_A, a_1...a_n\#, i, X_1..X_i\hat{}X_{j+1}...X_m)$,
if, $(q_A, 0, 1) \in \delta(p, a_i, X_j)$, where either $X_j=\$A$ and $p=q$, or $X_j \neq \$A$ $(A \in \Gamma)$ and $p=q_A$;

(iii)$(q, a_1...a_n\#, i, X_1...\hat{}X_j...X_m) \vdash_D$
$(q, a_1...a_n\#, i, X_1...X_{j-1}O\$\hat{}A_1...A_k¢X_{j+1}...X_m)$,
if $(q, 0, \$A_1...A_k¢) \in \delta(q, a_i, X_j)$, where $X_j \in I$ and $O=\varepsilon$, or $X_j=\$F$ $(F \in I)$ and $O=\$$;

(iv)$(q, a_1...a_n\#, i, X_1...X_{j-1}\$\hat{}¢X_{j+1}...X_m) \vdash_D$
$(q, a_1...a_n\#, i, X_1...\hat{}X_{j-1}X_{j+1}...X_m)$,
if $(q, 0) \in \delta(q, a_i, \$\hat{}¢)$.

$\vdash_D^*$ is the reflexive and transitive closure of $\vdash_D$. N(D) or the language accepted by empty stack by D is defined as follows

$$N(D) = \{w;\ w \in T^* \ \&\ (q, w\#, 1, \$\hat{}Z_0\#) \vdash_D^* (q, w\#, |w|+1, \$\hat{}\#)$$

To illustrate, the DI-automaton DI_3 accepting L_3 by empty stack is specified:
$DI_3 = (q$ (state for pda-mode), $(Q_\Gamma =)$ $\{q_S, q_M, q_Z, q_\$\}$ (states for stack reading mode), $(T=)$ $\{a,b,[,]\}$ (=input alphabet), $(G=)\{S,M,Z,a,b,[,],\}$ (=tape symbols for pda-mode), $(I=)\{f,g\}$ (=tape symbols representing indices), $\delta, S, \$, ¢, \#)$
where for every $x \in T$:
$\delta(q, x, \$S) = \{(q,0,\$aSfa), (q,0,\$bSgb), (q,0,¢M),\}$,
(for the G_3-rules: $S \rightarrow aSfa$, $S \rightarrow bSgb$, $S \rightarrow M$)
$\delta(q, x, \$M) = \{(q,0,\$[M]), (q,0,\$MM), (q,0,\$Z),\}$,

(for: $M \rightarrow [M]$, $M \rightarrow MM$, $M \rightarrow Z$)

$\delta(q,x,\$x) = \{(q,1,\$)\}$

(i.e.: if input symbol x = "predicted" terminal symbol *x*, then shift input-tape one step ("*1*") and delete successful prediction" (replace *\$x* by *\$*))

$\delta(q,x,\$Z)$ contains $\{(q_Z,0,\$)\}$,

(i.e.: change into stack reading mode in order to find indices belonging to the nonterminal *Z*)

$\delta(q_Z,x,\$Y) = \delta(q_Z,x,Y)$ contain $\{(q_Z,0,1)\}$ (for every $x \in T$, $Y \in \Gamma$)

(i.e.seek first index-symbol belonging to *Z* inside the stack)

$\delta(q_Z,x,\$f) = \{(q_Z,0,\$\$Za\text{¢}),(q_Z,0,\$\$a\text{¢})\}$,

$\delta(q_Z,x,\$g) = \{(q_Z,0,\$\$Zb\text{¢}),(q_Z,0,\$\$b\text{¢})\}$,

$\delta(q_Z,x,f) = \{(q,x,\$Za\text{¢}),(q,x,\$a\text{¢})\}$,

$\delta(q_Z,x,g) = \{(q,x,\$Zb\text{¢}),(q,x,\$b\text{¢})\}$,

(i.e. simulate the index-rules $Zf \rightarrow Za$, $Zf \rightarrow a$ by creation of embedded stacks)

$\delta(q,x,\$\text{¢}) = \{(q,0)\}$,

(i.e. delete empty sub-stack)

$\delta(q,x,Y) = \{(q,0,-1)\}$ (for $x \in T$, $Y \in G-\{f,g\}$)

(i.e. move to top of (sub-)stack).

The following theorem expresses the equivalence of DI-grammars and DI-automata

> (11) DI-THEOREM: *L is a DI-language (i.e. L is generated by a DI-grammar) if and only if L is accepted by a DI-automaton.*

Proof sketch:

I. "only if":(to facilitate a comparison this part follows closely Aho's corresponding proof for indexed grammars and *nsa's* (theorem 5.1) in (Aho, 1969))

If L is a DI-language, then there exists a DI-grammar G=(N,T,F,P,S) with L(G)=L. For every DI-grammar an equivalent DI-grammar in a normal form can be constructed in which each rule has the form $A \rightarrow BC$, $A \rightarrow a$, $A \rightarrow Bf$ or $Af \rightarrow B$, with $A \in N$; $B,C \in (N-\{S\})$, $a \in T$, $f \in F$; and $\varepsilon \in L(G)$, only if $S \rightarrow \varepsilon$ is in P. (The proof is completely analogous to the corresponding one for indexed grammars in (Aho, 1968) and is therefore omitted). Thus, we can assume without loss of generality that G is in normal form.

A DI-automaton *D* such that N(*D*)=L(G) is constructed as follows:

Let $D=(q,Q_\Gamma,T,\Gamma,I,\delta,Z_0,\$,\text{¢},\#)$, with $\Gamma = N \cup T \cup \{\$,\text{¢},\#\}$, $Q_\Gamma=\{q_A;A \varepsilon \Gamma\}$, $I=F$, $Z_0=S$ where δ is constructed in the following manner for all $a \in T$:

1. $(q,0,\$BC) \in \delta(q,a,\$A)$, if $A \rightarrow BC \in P$,
   $(q,0,\$b) \in \delta(q,a,\$A)$, if $A \rightarrow b \in P$,
   $(q,0,\$Bf) \in \delta(q,a,\$A)$, if $A \rightarrow Bf \in P$
2. $(q,1,\$) \in \delta(q,a,\$a)$
3. $(q_A,0,\$) \in \delta(q,a,\$A)$ for all $A \in \Gamma$,
4. $(q_A,0,1) \in \delta(q_A,a,B)$ for all $A \in \Gamma$ and all $B \in \Gamma$,
5. i.$(q,0,\$B\text{¢}) \in \delta(q_A,a,f)$ and
   ii.$(q,0,\$\$B\text{¢}) \in \delta(q_A,a,\$f)$ for all $A \in \Gamma$ with $Af \rightarrow B \in P$,
6. $(q,0) \in \delta(q,a,\$\text{¢})$
7. $(q,0,-1) \in \delta(q,a,B)$ for all $B \in \Gamma \cup \{\text{¢}\}$
8. $(q,0,\$) \in \delta(q,\#,\$S)$ if and only if $S \rightarrow \varepsilon$ is in P.

LEMMA 1.1

If

(i) $Af_1 \ldots f_k = ^n \Rightarrow a_1 \ldots a_m$

is a valid leftmost derivation in G with $k \geq 0$, $m \geq 1$ and $A \in N$, then for $n \geq 1$, $Z\beta_1 \ldots \beta_k \in (N \cup \{\text{¢}\})^*$, $\alpha \in (N \cup \{\$,\text{¢}\})^*$, $\mu \in (N \cup F \cup \{\text{¢}\})^*$:

(ii) $(q,a_1 \ldots a_m\#,1,\alpha\$\text{^}AZ\beta_1 f_1 \ldots \beta_k f_k \mu\#)$
$\vdash_D^* (q,a_1 \ldots a_m\#,m+1,\alpha\$\text{^}Z\beta_1 \ldots \beta_k \mu\#)$.

Proof by induction on n (i.e. the number of derivation steps):

If n=1, then (i) is of the form $A \Rightarrow a$ where $a \in T$ and k=0, since only a rule of the form $A \rightarrow a$ can be applied because of the normal form of G and since in DI-grammars (unlike in indexed grammars) unconsumed indices can not be swallowed up by terminals. Because of the construction of δ, (ii) is of the form

$$\begin{aligned}(q,a\#,1,\alpha\$\text{^}AZ\beta_1 \ldots \beta_k \mu\#) &= (q,a\#,1,\alpha\$\text{^}AZ\mu\#)\\ &\vdash_D (q,a\#,1,\alpha\$\text{^}aZ\mu\#)\\ &\vdash_D (q,a\#,2,\alpha\$\text{^}Z\mu\#)\\ &=(q,a\#,2,\alpha\$\text{^}Z\beta_1 \ldots \beta_k \mu\#)\end{aligned}$$

Suppose Lemma 1.1 is true for all n<n' with n'>1.

A leftmost derivation $Af_1 \ldots f_k = ^{n'} \Rightarrow a_1 \ldots a_m$ can have the following three forms according as A is expanded in the first step:

1)$Af_1 \ldots f_j f_{j+1} \ldots f_k \Rightarrow Bf_1 \ldots f_j Cf_{j+1} \ldots f_k$
$= ^{n1} \Rightarrow a_1 \ldots a_i Cf_{j+1} \ldots f_k$
$= ^{n2} \Rightarrow a_1 \ldots a_i a_{i+1} \ldots a_m$

with $n_1 < n'$ and $n_2 < n'$

2)$Af_1 \ldots f_k \rightarrow Bff_1 \ldots f_k = ^{n1} \Rightarrow a_1 \ldots a_m$ with $n_1 < n'$.

3)$Af_1 \ldots f_k \rightarrow Bf_2 \ldots f_k = ^{n1} \Rightarrow a_1 \ldots a_m$
with $n_1 < n'$ and $(Af_1 \rightarrow B) \in P$.

From the inductive hypothesis and from 1.-8. above, it follows

1')

$(q,a_1 \ldots a_m\#,1,\alpha\$\text{^}AZ\beta_1 f_1 \ldots \beta_j f_j \beta_{j+1} f_{j+1} \ldots \beta_k f_k \mu\#)$
$\vdash_D (q,a_1 \ldots a_m\#,1,\alpha\$\text{^}BCZ\beta_1 f_1 \ldots \beta_j f_j \beta_{j+1} f_{j+1} \ldots \beta_k f_k \mu\#)$
$\vdash_D^* (q,a_1 .. a_m\#,i+1,\alpha\$\text{^}CZ\beta_1 \beta_2 \ldots \beta_j \beta_{j+1} f_{j+1} \ldots \beta_k f_k \mu\#)$
$\vdash_D^* (q,a_1 \ldots a_m\#,m+1,\alpha\$\text{^}Z\beta_1 \ldots \beta_j \beta_{j+1} \ldots \beta_k \mu\#)$

2')$(q,a_1...a_m\#,1,\alpha\$\wedge AZ\beta_1 f_1...\beta_k f_k \mu\#)$
$\vdash_D (q,a_1...a_m\#,1,\alpha\$\wedge BfZ\beta_1 f_1...\beta_k f_k \mu\#)$
$\vdash_D^* (q,a_1...a_m\#,m+1,\alpha\$\wedge Z\beta_1...\beta_k \mu\#)$

3')$(q,a_1...a_m\#,1,\alpha\$\wedge AZ\beta_1 f_1...\beta_k f_k \mu\#)$
$\vdash_D (q_A,a_1...a_m\#,1,\alpha\$\wedge Z\beta_1 f_1...\beta_k f_k \mu\#)$
$\vdash_D^* (q_A,a_1...a_m\#,1,\alpha\$Z\beta_1\wedge f_1...\beta_k f_k \mu\#)$
$\vdash_D (q,a_1...a_m\#,1,\alpha\$Z\beta_1\$\wedge B\text{¢}\beta_2 f_2...\beta_k f_k \mu\#)$
$\vdash_D^* (q,a_1...a_m\#,m+1,\alpha\$Z\beta_1\$\wedge\text{¢}\beta_2...\beta_k \mu\#)$
$\vdash_D (q,a_1...a_m\#,m+1,\alpha\$Z\sigma\wedge X\beta_2...\beta_k \mu\#)$
$\vdash_D^* (q,a_1...a_m\#,m+1,\alpha\$\wedge Z\sigma X\beta_2...\beta_k \mu\#)$
where $\sigma X=\beta_1$.

LEMMA 1.2

If for $Z\beta_1...\beta_k\in(N\cup\{\text{¢}\})^*$, $\alpha\in(N\cup\{\$,\text{¢}\})^*$, and $m\geq 1$, $\mu\in(N\cup F\cup\{\text{¢}\})^*$
$(q,a_1...a_m\#,1,\alpha\$\wedge AZ\beta_1 f_1...\beta_k f_k \mu\#)$
$\vdash_D^* (q,a_1...a_m\#,m+1,\alpha\$\wedge Z\beta_1...\beta_k \mu\#)$
then for all $n\geq 1$
$Af_1...f_k \overset{*}{\Rightarrow} a_1...a_m$.

The proof (by induction on n) is similar to the proof of Lemma 1.1 and is, therefore, omitted.

II.("if")

If L is accepted by a DI-automaton $D=(q,Q_\Gamma,T,\Gamma,\delta,I,Z_0,\$,\text{¢},\#)$, then we can assume without loss of generality

a) that D writes at most two symbols on a stack in either the push down mode or the stack creation mode (it follows from the Di-automaton definition that the first one of the two symbols cannot be a index symbol from I),

b) that T and Γ are disjunct.

A DI-grammar G with L(G)=N(D)=L can be constructed as follows:

Let G=(N,F,T,P,S) with N=Γ, F=I, S=Z_0. P contains for all $a\in T$, $A,B,C\in N$, and $f\in F$ the productions
($d_a=a$, if d=1, else $d_a=\varepsilon$)

1.	$A\to d_aBC$,	if	$(q,d,\$BC)\in\delta(q,a,\$A)$,
2.	$A\to d_aBf$,	if	$(q,d,\$Bf)\in\delta(q,a,\$A)$,
3.	$A\to d_aB$,	if	$(q,d,\$B)\in\delta(q,a,\$A)$,
4.	$A\to d_a$,	if	$(q,d,\$)\in\delta(q,a,\$A)$,
5.	$Af\to BC$,	if	$(q,0,\$BC\text{¢})\in\delta(q_A,a,f)$ or $(q,0,\$\$BC\text{¢})\in\delta(q_A,a,\$f)$
6.	$Af\to B$,	if	$(q,0,\$B\text{¢})\in\delta(q_A,a,f)$ or $(q,0,\$\$B\text{¢})\in\delta(q_A,a,\$f)$

For all $n\geq 1$, $m\geq 1$, $\beta_1...\beta_k\in(N\cup\{\text{¢}\})^*$, $\alpha\in(N\cup\{\$,\text{¢}\})^*$, $f_1,f_2,....,f_k\in F$, $A\in N$, $\mu\in(N\cup F\cup\{\text{¢}\})^*$, and $a_1...a_m\in T^*$ II.1 and II.2 is true:

II.1: If $(q,a_1...a_m\#,1,\alpha\$\wedge A\beta_1 f_1...\beta_k f_k \mu\#)$
$\vdash_D^n (q,a_1...a_m\#,m+1,\alpha\$\wedge\beta_1...\beta_k \mu\#)$
then in G the derivation is valid
$Af_1...f_k \overset{*}{\Rightarrow} a_1...a_m$.

II.2: If
$Af_1...f_k \overset{n}{\Rightarrow} a_1...a_m$.
is a leftmost derivation in G, then the following transition of D is valid
$(q,a_1...a_m\#,1,\alpha\$\wedge A\beta_1 f_1...\beta_k f_k \mu\#)$
$\vdash_D^* (q,a_1...a_m\#,m+1,\alpha\$\wedge\beta_1...\beta_k \mu\#)$

The proofs by induction of I.1 and II.2 (unlike the proofs of the corresponding lemmata for *nsa*'s and indexed grammars (s.Aho, (1969)) are as elementary as the one given above for I.1 and are omitted.

The DI-automaton concept can be used to show the inclusion of the class of DI-languages in the class of context-sensitive languages. The proof is strucurally very similar to the one given by Aho (Aho, 1968) for the inclusion of the indexed class in the context-sensitive class: For every DI-automaton A, an equivalent DI-automaton A' can be constructed which accepts its input w if and only if A accepts w and which in addition uses a stack the length of which is bounded by a linear function of the length of the input w. For A' a linear bounded automaton M (i.e the type of automaton characteristic of the context-sensitive class) can be constructed which simulates A'. For reasons of space the extensive proof can not be given here.

4 Some Remarks on the Complexity of DI-Recognition

The time complexity of the recognition problem for DI-grammars will only be considered for a subclass of DI-grammars. As the restriction on the form of the rules is reminiscent of the Chomsky normal form for context-free grammars (CFG), the grammars in the subclass will be called DI-Chomsky normal form (DI-CNF) grammars

A DI-grammar G=(N,T,F,P,S) is a DI-CNF grammar if and only if each rule in P is of one of the following forms where $A,B,C\in N-\{S\}$, $f\in F$, $a\in T$, $S\to\varepsilon$, if $\varepsilon\in L(G)$,

(a) $A\to BC$, (b)A-BfC, (c) $A\to BCf$,
(d) $Af\to BC$, (e)$Af\to a$, (f) $A\to a$

The question whether the class of languages generated by DI-CNF grammars is a proper or improper subclass of the DI-languages will be left open.

In considering the recognition of DI-CNF grammars an extension of the CKY algorithm for CFGs (Kasami, 1965; Younger, 1967) will be used which is essentially inspired by an idea of Vijay-Shanker and Weir in (Vijay-Shanker and Weir, 1991).

Let the $n(n+1)/2$ cells of a CKY-table for an input of length n be indexed by i and j ($1 \leq i \leq j \leq n$) in such a manner that cell $Z_{i,j}$ builds the top of a pyramid the base of which consists of the input $a_i \ldots a_j$.

As in the case of CFGs a label E of a node of a derivation tree (or a code of E) should be placed into cell $Z_{i,j}$ only if in G the derivation $E =*=> a_i \ldots a_j$ is valid. Since nonterminal nodes of DI-derivation trees are labeled by pairs (A,μ) consisting of a nonterminal A and an index stack μ and since the number of such pairs with $(A,\mu) =*=> w$ can grow exponentially with the length of w, intractability can only be avoided if index stacks can be encoded in such a way that substacks shared by several nodes are represented only once.

Vijay-Shanker and Weir solved the problem for linear indexed grammars (LIGs) by storing for each node K not its complete label $Af_1f_2 \ldots f_n$, but the nonterminal part A together with only the top f_1 of its index stack and an indication of the cell where the label of a descendant of K can be found with its top index f_2 continuing the stack of its ancestor K. In the following this idea will be adopted for DI-grammars, which, however, require a supplementation.

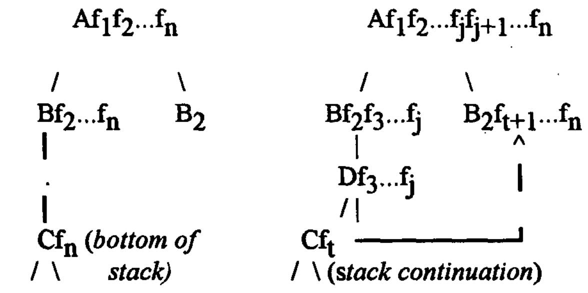
Thus, if the cell $Z_{i,j}$ of the CKY-table contains an entry beginning with "$<A,f_1, (B,f_2,q,p),..>$", then we know that

$$A\mu =*=> a_i \ldots a_j \text{ with } \mu = f_1\mu_1 \in F^*$$

is valid, and further that the top index symbol f_2 on μ_1(i.e. the continuation of f_1) is in an entry of cell $Z_{p,q}$ beginning with the noterminal B. If, descending in such a manner and guided by pointer quadruples like "$<B,f_2,p,q>$", an entry of the form $<C,f_n,-,..>$ is met, then, in the case of a LIG-table, the bottom of stack has been reached. So, entries of the form $<A,f_1,(B,f_2,p,q)>$ are sufficient for LIGs.

But, of course, in the case of DI-derivations the bottom of stack of a node, because of index distribution, does not coincide with the bottom of stack of an arbitrary index inheriting descendant, cf.

(13)



Rather, the bottom of stack of a DI-node coincides with the bottom of stack of its rightmost index inheriting descendant. Therefore, the pointer mechanism for DI-entries has to be more complicated. In particular, it must be possible to add an "intersemital" pointer to a sister path. However, since the continuation of the unary stack (like of Cf_t in ( )) of a node without index inheriting descendants is necessarily unknown at the time its entry is created in a bottom up manner, it must be possible to add an intersemital pointer to an entry later on.

That is why a DI-entry for a node K in a CKY-cell requires an additional pointer to the entry for a descendant C, which contains the end-of-stack symbol of K and which eventually has to be supplemented by an intersemital continuation pointer. E.g. the entry

(14) $<B_1,f_2,(D,f_3,p,q),(C,f_t,r,s)>$ in $Z_{i,j}$

indicates that the next symbol f_3 below f_2 on the index stack belonging to B_1 can be found in cell $Z_{p,q}$ in the entry for the nonterminal D; the second quadruple (C,f_t,r,s) points to the descendant C of B_1carrying the last index f_t of $B_1$and containing a place where a continuation pointer to a neighbouring path can be added or has already been added.

To illustrate the extended CKY-algorithm, one of the more complicated cases of specifying an entry for the cell $Z_{i,j}$ is added below which dominates most of the other cases in time complexity:

```
....
FOR i:=n TO 1 DO
 FOR j:=i TO n DO
      FOR k:=i TO j-1 DO
        ................
        ................
        For each rule $A \rightarrow A_1fA_2$:
        if $<A_1,f,(B_1,f_1,p_1,q_1),(C_1,f_3,s_1,t_1)> \in Z_{i,k}$
            for some $B_1$, $C_1 \in N, f_1,f_3 \in F$,
            $p_1$, $q_1$ ($i \leq p_1 \leq q_1 \leq k$), $s_1,t_1$ ($i \leq p_1 \leq s_1 \leq t_1 \leq k$)
        and $<A_2,f_c,-,-> \in Z_{k+1,j}$ for some $f_c \in F$
        then 1. if
                 $<B_1,f_1,(B_2,f_2,p_2,q_2), X> \in Z_{p1,q1}$ for
                 some $B_2, \in N$, $f_2 \in F$, $p_2$, $q_2$
                 with $i \leq p_2 \leq q_2 \leq k$, and if $q_1 \leq p_2$, then
                 X=- , else $X=(C,f_t, u,v)$ for some
                 $C \in N$, $f_t \in F$, u,v ($p_1 \leq u \leq v \leq q_1$)
               then
                 $Z_{i,j} := Z_{i,j} \cup \{<A,f_1,(B_2,f_2,p_2,q_3),$
                                        $(A_2,f_c,k+1,j)>\}$
               else
                   if $<B_1,f_1,-,-> \in L_{p1,q1}$
                   $Z_{i,j} := Z_{i,j} \cup \{<A,f_1,(A_2,f_c,k+1,j),$
                                          $(A_2,f_c,k+1,j)>\}$
             2. if
                 $<C_1,f_3,-,-> \in L_{s1,t1}$
```

then
$Z_{s1,t1} :=$
$Z_{s1,t1} \cup \{<C_1, f_3, (A_2, f_c, k+1, j), ->\}$

The pointer $(A_2, f_c, k+1, j)$ in the new entry of $Z_{i,j}$ points to the cell of the node where the end of stack of the newly created node with noterminal A can be found. The same pointer $(A_2, f_c, k+1, j)$ appears in cell $Z_{s1,t1}$ as "supplement" in order to indicate where the stack of A is continued behind the end-of-stack of A_1. Note that supplemented quadruples of a cell $Z_{i,j}$ are uniquely identifiable by their form $<N, f_1, (C, f_2, r, s), ->$, i.e. the empty fourth component, and by the relation $j \le r \le s$. Supplemented quadruples cannot be used as entries for daughters of "active" nodes, i.e. nodes the entries of which are currently being constructed.

Let $a_1 \ldots a_n$ be the input. The number of entries of the form $<B, f_1, (D, f_2, p, q), (C, f_3, r, s)>$ ($f_1, f_2, f_3 \in F$, B,C, $D \in N$, $1 \le i, p, q, r, s, j \le n$) in each cell $Z_{i,j}$ will then be bounded by a polynomial of degree 4, i.e. $O(n^4)$. For a fixed value of i,j,k, steps like the one above may require $O(n^8)$ time (in some cases $O(n^{12})$). The three initial loops increase the complexity by degree 3.

References

[Aho, 1968] A. V. Aho. Indexed Grammars, *J.Ass.Comput.Mach.* 15, 647-671, 1968.

[Aho, 1969] A. V. Aho. Nested Stack Automata, *J.Ass.Comp.Mach.* 16, 383- , 1969.

[Gazdar, 1988] G. Gazdar. Applicability of Indexed Grammars to Natural Languages,in: U.Reyle and C.Rohrer (eds.)*Natural Language Parsing and Linguistic Theories,* 69-94, 1988.

[Joshi, Vijay-Shanker, and Weir, 1989] A. K. Joshi, K. Vijay-Shanker, and D. J. Weir. The convergence of mildly context-sensitive grammar formalisms. In T. Wasow and P. Sells (Eds.), *The processing of linguistic structure.* MIT Press, 1989.

[Kasami, 1965] T. Kasami. *An efficient recognition and syntax algorithm for context-free languages.*(Tech. Rep. No. AF-CRL-65-758). Bedford, MA: Air Force Cambridge Research Laboratory, 1965.

[Pereira, 1981] F. Pereira. Extraposition Grammars, in: *American Journal of ComputationalLinguistics*,7, 243-256, 1981.

[Pereira, 1983] F. Pereira. Logic for Natural Language Analysis, *SRI International,* Technical Note 275, 1983.

[Rizzi, 1982] L. Rizzi. *Issues in Italian Syntax,* Dordrecht, 1982.

[Stabler, 1987] E. P Stabler. Restricting Logic Grammars with Government-Binding Theory, *Computational Linguistics,* 13, 1-10, 1987.

[Takeshi, 1973] Hayashi Takeshi. On Derivation Trees of Indexed Grammars. *Publ.RIMS, Kyoto Univ.,* 9, 61-92, 1973.

[Vijay-Shanker, Weir, and Joshi, 1986] K. Vijay-Shanker, D. J. Weir, and A. K. Joshi. Tree adjoining and head wrapping. *11th International Conference on Comput. Ling.* 1986.

[Vijay-Shanker, Weir, and Joshi, 1987] K. Vijay-Shanker, D. J. Weir, A. K. Joshi. Characterizing structural descriptions produced by various grammatical formalisms. *25th Meeting Assoc.Comput. Ling.*, 104-111. 1987.

[Vijay-Shanker and Weir, 1991] K. Vijay-Shanker and David J. Weir. Polynomial Parsing of Extensions of Context-Free Grammars. In: Tomita, M.(ed.) *Current Issues in Parsing Technology,* 191-206, London 1991.

[Weir and Joshi, 1988] David J. Weir and Aravind K. Joshi. Combinatory Categorial Grammars: Generative power and relationship to linear context-free rewriting systems. *26th Meeting Assoc.Comput. Ling.*, 278-285, 1988.

[Younger, 1967] D. H. Younger. Recognition and parsing context-free languages in time n^3. *Inf. Control, 10,* 189-208.