Composition of Translation Schemes with D-Trees

Martin Flátek

Charles University, Faculty of Mathematics and Physics
Malá straná nám. 25
118 00 Prague
Czechoslovakia

Generative systems (GS) are defined in this paper as a composition of simple translation schemes with dependency trees. The following issues are discussed:
(a) explicative power of GS, (b) the time complexity for the analysis and synthesis for GS.

INTRODUCTION

A generative system for Czech was presented in Sgall [6]. The concept of a generative system was studied by Flátek [4] and Flátek and Sgall [5]. In this paper we use a similar approach as that presented by Hajičová, Flátek and Sgall in [3].

We define generative systems as a fundamental device for construction of grammars of natural languages. We give here some mathematical results to illustrate the usefulness of the new concept. We try first to formulate the necessary requirements on a grammar G of a natural language L. The grammar G must determine:

a) The set of all correct sentences of the language L. The set will be denoted by LC.

b) The set of the correct structural descriptions (SD) of the language L. SD represents all meanings of all sentences of LC.

c) The relation SH between LC and SD. The relation SH describes the ambiguity and the synonymy of L.

By a structural description we understand a dependency tree (D-tree).

The concept of a simple translation scheme from [1] is a generalisation of context-tree grammar. We introduce here a similar concept of a translation scheme, in this case as a generalisation of dependency grammar (see [2, 5]).

A generative system (GS) is defined as a sequence of translation schemes with a special asymmetric property.

We show that the explicative power of GS increases with the length of GS. We present results concerning on algorithm for the analysis and synthesis of GS and show that its time complexity is independent on the length of GS.

Moreover for a given GS we can construct a similar GS, for which a fast algorithm for synthesis exists.

Definitions.

Notation. The vocabulary, sets of nodes, edges and rules are here nonempty and finite sets.
Let \( R \) be a relation. We denote
\[
\text{Dom} (R) = \{a; \langle a, b \rangle \in R \} \text{ and }
\]
\[
\text{Range} (R) = \{b; \langle a, b \rangle \in R \}
\]
By \( f : U \rightarrow V \) we denote a total mapping from \( U \) into \( V \).

Def. A string over a vocabulary \( V \) is a triple \( S=(U,LR,o) \), where
\( U \) is a set of nodes, \( LR \) a linear ordering of \( U \), \( o:U \rightarrow V \). Let \( o(u)=A \). We say that \( A \) is the value of node \( u \). Let \( S=(U,LR,o) \),
\( S_1=(U_1,LR_1,o_1) \), \( S_2=(U_2,LR_2,o_2) \) be the strings and \( u \in U \). We say that \( S_2 \) is produced from \( S \) by replacing \( u \) by \( S_1 \), when the string \( S_1 \) is placed between the predecessor and the successor of node \( u \) and otherwise \( S_2 \) does not differ from \( S \). We denote as \( V \) the set of all nonempty strings over \( V \).

Def. Let \( S_1 = (U_1,LR_1,o_1) \), \( S_2 = (U_2,LR_2,o_2) \) be strings.
Let \( U_1 = \{u_1,\ldots,u_n\} \) and \( U_2 = \{u_2,\ldots,u_n\} \) and \( u_1,\ldots,u_n \) be in the ordering \( LR_1 \), and \( u_2,\ldots,u_n \) in the ordering \( LR_2 \) and
\( o_1(u_i) = o_2(u_i) \) for all \( i \) between 1 and \( n \). Then we say that \( S_1 \) and \( S_2 \) are equivalent.

We shall not distinguish between equivalent strings.

Def. A quintuple \( SR=(U,LR,B,r,o) \) is called a D-tree over \( V \), when
\( S(S(R))=(U,LR,o) \) is a string and \( o:U \rightarrow V \), \( B(S(R))=(U,B,r) \) is a tree with the root \( r \) and when the following condition holds: The nodes of every path in \( B(S(R)) \), which begins with a leaf, are nodes of a substring of \( S(S(R)) \). We say that \( S(S(R)) \) is a projection of \( SR \).

Def. Let \( SR_1=(U_1,LR_1,B_1,r_1,o_1) \) and \( SR_2=(U_2,LR_2,B_2,r_2,o_2) \) be D-trees. Let strings \( S(SR_1) \) and \( S(SR_2) \) be equivalent. Let \( f \) be a one-to-one mapping from \( U_1 \) on \( U_2 \), which preserves the ordering \( LR_1 \) to the ordering \( LR_2 \). Let \( f(r_1)=r_2 \) and let it hold that
\[
\langle u,v \rangle \in B_1 \text{ iff } \langle f(u),f(v) \rangle \in B_2.
\]
Then we say that \( SR_1 \) and \( SR_2 \) are equivalent. We shall not distinguish between equivalent D-trees.

Def. Let \( D=(U,LR,B,r,o), D_1=(U_1,LR_1,B_1,r_1,o_1) \) and
\( D_2=(U_2,LR_2,B_2,r_2,o_2) \) be D-trees and \( u \in U \). We say, that \( D_2 \) is produced from \( D \) by replacing \( u \) by \( D_1 \), when \( S(D_2) \) is produced from \( S(D) \) by replacing \( u \) by \( S(D_1) \) and the neighbours of \( r_1 \) in \( B(D_2) \) are the same as neighbours of \( u \) in \( B(D) \). Otherwise \( D_2 \) does not differ from \( D \).

Def. A translation scheme of type string - D-trees (\( TS\ [S,D] \ )) is a quadruple \( T=(VN,VT,S,P) \), where \( VN \) is the vocabulary of nonterminals, \( VT \) the vocabulary of terminals, \( VN \cap VT \neq \emptyset \), \( S \in VN \) and
\( P \) is a set of rules of the following type: \( LS \rightarrow A \rightarrow RS \), where
\( A \in VN \) (the middle of the rule) \( LS \) (the lefthand side) is a string over \( VN \cup VT \), \( RS \) the righthand side) is a D-tree over \( VN \cup VT \) and the following condition holds: When all nodes with terminals are erased from \( SR(S(R)) \) and \( LS \), then we get two equal strings.

Let \( p=LS \rightarrow A \rightarrow RS \). We write \( f \) \( LS_1,RS_1 \xrightarrow{p} \{LS_2,RS_2\} \) when
\( (i) \); \( (ii) \) the leftmost nonterminal node of \( LS_1 \) is some \( u \) with the value \( A \) and \( (iii) \) \( LS_2 \) is produced from \( LS_1 \) by replacing \( u \) by \( LS \) and \( RS_2 \) is produced from \( RS_1 \) by replacing \( v \) by \( RS \).

\( p \xrightarrow{\star} \) is denoted as \( \xrightarrow{\star} \) and \( \xrightarrow{\ast} \) is the transitive closure of \( \xrightarrow{\star} \).

We denote as \( TR(T) = \{[LS,RS]; [S,S] \xrightarrow{\star} [LS,RS], LS,S (RS) \in VT \} \).
Remark. Analogically as a translation scheme of the type string - string was defined, also definitions of the type string - D-tree (TS $[S,S]$) or of the type D-tree - D-tree (TS $[D,D]$) can be given.

By TS $[S,S]$ the lefthand side and righthand side of a rule is always a string. By TS $[D,D]$ both sides of a rule are always D-trees.

As TS we denote the set of all translation schemes of all the three types.

Def. Let $T_1,...,T_n$ be a sequence of TS. We denote as $TR(T_1,...,T_n)=TR(T_1).TR(T_2)...TR(T_n)$. The main definition of this paper is the following:

Def. A generative system (GS) is a sequence $T_1,...,T_n$ of TS, where $TR(T_1,...,T_n)$ is a relation between strings and D-trees and for every $[d_1,d_2] \in TR(T_n)$ there exists a $s_1,s_2$, such that $s_1,d_2 \in TR(T_1,...,T_n)$. The set $AN(T_1,...,T_n)=\{[v,d] \in TR(T_1,...,T_n) \}$ is called the analysis of $v$. The set $SF(T_1,...,T_n)=\{[s,d] \in TR(T_1,...,T_n) \}$ is called the full synthesis of D-tree $d$.

Remark. Let $GSI=T_1,...,T_n$ be a GS. Then $Range(TR(T_1)) \subseteq Dom(TR(T_2)) \subseteq ... \subseteq Range(TR(T_{n-1})) \subseteq Dom(TR(T_n))$.

We call this property of GSI an asymmetric property of GS.

Def. Let $GSI$ be a GS. We say that the function $MS$ is a function of minimal synthesis of GSI, if the following conditions are fulfilled:

a) $MS^{-1} \subseteq TR(GSI)$

b) $Dom(MS)=Range(TR(GSI))$.

Def. D-grammar (DG) is a T $\in$ TS $[S,D]$, where $T=(VN,VT,S,P)$ and for every $p \in P, p=LSA$ there holds, that $LS=P(RS)$.

Def. We denote $DR_o=\{TR(T); T \in DG\}$ and $DR_j=\{TR(T_1,...,T_j); T_1,...,T_j \in GSI \}$ for $j \in N$. For $j \in N \cup \{0\}$ we write $IDR_j=\{F \in DR_j; F$ is a function $\}$.

Note. We need also one more concept. It is the concept of an h-morphic generative system for another one.

Def. Let $V_1,V_2$ be two alphabets and $h:V_1 \rightarrow V_2$. Let $S_1=(U_1,LR_1,B_1,r_1,o_1)$, $S_2=(U_2,LR_2,B_2,r_2,o_2)$ be two strings, where $o_1:U_1 \rightarrow V_1$, $o_2:U_2 \rightarrow V_2$. We say that a tuple $(z,h)$ is an h-morphism from $S_1$ to $S_2$, when $h(U_1) \rightarrow U_2$ is a one-to-one mapping which preserves the ordering on nodes and for every $u \in U_1$ there holds that $h(o_1(u))=o_2(f(u))$. We say that $S_1$ is h-morphic for $S_2$, if there exists an h-morphism from $S_1$ to $S_2$.

Def. Let $D_1=(U_1,LR_1,B_1,r_1,o_1)$ and $D_2=(U_2,LR_2,B_2,r_2,o_2)$ be D-trees. Let $(z,h)$ be an h-morphism $S(D_1)$ to $S(D_2)$. We let there hold that $(u,v) \in D_1$ iff $(t(u),t(v)) \in D_2$ and $t(x) \approx 2$.

We say that $(z,h)$ is an h-morphism from $D_1$ to $D_2$. We say that $D_1$ is h-morphic to $D_2$, when there exists an h-morphism from $D_1$ to $D_2$.

Def. Let $T_1=(VN_1,VT_1,S_1,P_1)$ and $T_2=(VN_2,VT_2,S_2,P_2)$ be TS. Let $h:VN_1 \cup VT_1 \rightarrow VN_2 \cup VT_2$, where $h(VN_1)=VN_2, h(VT_1)=VT_2$.

Let there exist a one-to-one mapping $MF$ from $P_1$ on $P_2$ such, that if $p=LS_1 \rightarrow A_1 \rightarrow RS_1$ and $MF(p)=LS_2 \rightarrow A_2 \rightarrow RS_2$, then $LS_1$ is h-morphic to $LS_2$, $RS_1$ is h-morphic to $RS_2$ and $h(A_1)=A_2$.

We then say, that $T_1$ is h-morphic for $T_2$. 
Def. Let $GSI = T_1, \ldots, T_n$ and $GS2 = T_2, \ldots, T_{2n}$ both be GS. Let $T_1$ be $h_1$-morphic to $T_2$, $T_2$ $h_2$-morphic to $T_2, \ldots$ and so on to $n$; we say then, that $GSI$ is $h$-morphic for $GS2$, where $h = (h_1, \ldots, h_n)$.

Examples

Example 1.
Let us have an example of a translation scheme.
Let $T3 = (\{S, S1, S2, S3\}, \{a, b, c\}, S, P3)$ and

$P3$: $a S1 \leftarrow S \rightarrow S3$
$\quad a S1 \leftarrow S1 \rightarrow S3$
$\quad c S2 \leftarrow S1 \rightarrow S3$
$\quad a S2a \leftarrow S2 \rightarrow S3$

It holds that:
$[S, S3] \rightarrow [a S1, S3] \rightarrow [aa S1, S3] \rightarrow [aa, S3]$

We can see that $\text{Dom}(\text{TR}(T3)) = \{a^n c a^n c a^j; n, j \in \mathbb{N}\} \cup \{a^j c a^n c a^n; j, n \in \mathbb{N}\}$ and that $\text{TR}(T3)$ is a function.

Example 2.
We present in this example some interesting set of translation schemes.
$G4 = (\{S, A\}, \{a, c\}, S, P4)$ where

$\text{F4}$: $\text{caac} \leftarrow S \rightarrow \text{caac}$
$\quad \text{aa} \leftarrow A \rightarrow \text{aa}$
$\quad c \leftarrow A \rightarrow c$

Then $\text{TR}(G4) = \{[c a^n c a^n c, o a^n c a^n c]; n \in \mathbb{N}\}$
$T5 = (\{S, A\}, \{a, c\}, S, P5)$
$\text{F5}$: $\text{ca} \leftarrow S \rightarrow \text{ca}$
$\quad \text{ca} \leftarrow A \rightarrow \text{ca}$
$\quad \text{aa} \leftarrow A \rightarrow \text{aa}$
$\quad c \leftarrow A \rightarrow c$
and let 

\[ R_4(k) = \text{TR}(G_4, T_5, \ldots, T_5), \]

\[ k \text{-times} \]

then 

\[ R_4(k) = \{ [c a^n : a^2, (c a^n)^2 : c] : n \in N \} \]

and if 

\[ \text{TR}(G_4, T_5, \ldots, T_5, T_6) = I T(k) \]

\[ (k-1) \text{-times} \]

then 

\[ R_4(k) = \{ [a, b] : [a, c] \in I T(k) \text{ and } b = S(c) \} \]

Results.

**Assertion 1.** For \( j \geq 0 \) it holds that \( DR_j \subseteq DR_{j+1} \) and 
\[ 1DR_j \subseteq 1DR_{j+1}. \]

**Notation.** \( |s| \) denotes the length of the string \( s \), which is the card \( (U) \), where \( U \) is the set of nodes of \( s \).

**Assertion 2.** Let \( GS_1 \) be a generative system.

a) Then there exists an algorithm that computes for every string \( v \) the set \( AN(GS_1, v) \) (analysis) with the time complexity bound by a function \( KL \). 
\[ |v|^3 \cdot \max \{ \text{card}(AN(T_1, \ldots, T_j; v)) \} \]

where \( KL \) depends only on \( GS_1 \).

b) Then there exists an algorithm that computes for every \( D \)-tree \( d \) the set \( ST(GS_1, d) \) (full synthesis) with the time complexity bound by function \( K_2 \). 
\[ |S(d)|^3 \cdot \text{card}(ST(GS_1)), \]

where \( K_2 \) depends only on \( GS_1 \).

**Assertion 3.** Let \( GS_1 \) be a GS. Then there exists an \( h \)-morphic generative system \( GS_2 \) for \( GS_1 \) and an algorithm that for every \( D \)-tree \( d \) computes \( ST(GS_2, d) \) with a time complexity bound by function \( K \). 
\[ |S(d)| \cdot \text{card}(ST(GS_1, d)) \]

where \( K \) depends only on \( GS_2 \) and 
\[ \text{Dom}(TR(GS_1)) = \text{Dom}(TR(GS_2)) \]

**Assertion 4.** Let \( GS_1 \) be a GS. Then there exists an \( h \)-morphic generative system \( GS_2 \) for \( GS_1 \) and an algorithm such that for every \( D \)-tree \( d \) computes \( MS(d) \) with a time complexity bound by function \( K \). 
\[ |S(d)| \cdot \text{card}(ST(GS_1, d)) \]

where \( MS \) is the function of minimal synthesis of \( GS_2 \), 
\[ \text{Dom}(TR(GS_1)) = \text{Dom}(TR(GS_2)) \] and \( K \) depends only on \( GS_2 \).

Remarks.

**Remark to Assertion 1.** We sketch here a proof of Ass. 1. We see that \( DR_0 \subseteq DR_1 \) and 
\[ 1DR_0 \subseteq 1DR_1. \] Dikovskij and Modina have shown in [2], that \( TR(T_3) \) from Example 1 cannot be in \( DR_0 \). We see that \( T_3 \) is a TS. Thus 
\[ DR_0 \nsubseteq DR_1. \] Since \( TR(T_3) \) is a function, we see that 
\[ 1DR_0 \nsubseteq 1DR_1. \]
In the Example 2 we have shown that \( IT(k) \subseteq IDR_k \). From the results on composition of pushdown transducers (PST) in \( \mathcal{J} \) and from the equivalence theorem between TS’s and PST’s from \( \mathcal{J} \), it follows, that \( IT(k+1) \subseteq DR_k \). Thus \( DR_j \subseteq DR_{j+1} \) and \( IDR_j = IDR_{j+1} \).

Remark to Assertion 2.
The algorithm for analysis and synthesis for a GS is based on the idea of Cocke-Younger-Kasami algorithm. For a sequence of simple translation schemes of the type string-string the algorithm is presented in Suchomel [7]. The difference between the upper boundary of the time complexity of the full synthesis and analysis is given by the asymmetric property of a GS.

Remark to Assertion 3.
The basic idea of the proof is a construction of a new GS to GS1. The new GS, denoted GS2, has full information in the alphabets for a straightforward algorithm for a full synthesis.

Remark to Assertion 4.
The idea of the proof is analogous to that of Assertion 3. When we have a partition of \( \text{Dom}(TR(GS1)) \) in the classes of synonymous sentences, the function of minimal synthesis chooses always only one representant of his class. Therefore the algorithm can be so fast.

Conclusion remarks.
When formulating a grammar for natural language, we can use with advantage the modularity of GS. We have shown that the time complexity of the analysis and synthesis for \( DR_j \), \( j \geq 2 \) is independent on \( j \). Otherwise the explicative power of \( DR_j \) is increasing with \( j \). We have also shown, that to any generative system there can be constructed an h-morphic generative system with the full information for a fast algorithm of the minimal synthesis.

References.
[1] Aho A.V.-Ullman J.D.: The Theory of Parsing Translation and Compiling, Vol.1: Parsing (Prentice-Hall, Englewood Cliffs, 1972)
[2] Dikovskij A.J.-Modina L.S.: O trech typach odnomnočnosti kon-těkstnysvobodnych jazykov, in Matematičeskaja lingvistika i teoria algoritmov, Kalinin 1978.
[3] Hajiové E.-Plátěk M.-Sgall P.: Komunikace s počítačem v Šestině [Man-machine communication in Czech] in SOFSEM (VVS Bratislava 1980)
[4] Plátěk M.: On serial and parallel compositions of PST’S Thesis Faculty of mathematics and physics, Charles University, Prague (December 1979)
[5] Plátěk M.-Sgall P.: A scale of context sensitive languages: Application to natural language, Information and Control, vol38 N. 1 (1978)
[6] Sgall P.: Generativní popis jazyka a česká deklinač (Academia Prague 1967)
[7] Suchomel K.: Generativní systémy a rospožnávání jazyků [Generative systems and language recognition] Diploma work, Faculty of mathematics and physics, Charles University, Prague (April 1961)