Towards a formal model of natural language description based on restarting automata with parallel DR-structures*

Markéta Lopatková, František Mráz, and Martin Plátek

Charles University in Prague, Czech Republic
lopatkova@ufal.mff.cuni.cz
frantisek.mraz@mff.cuni.cz
martin.platek@mff.cuni.cz

Abstract. *We provide a formal model of a stratificational dependency approach to natural language description. This formal model is motivated by an elementary method of analysis by reduction, which serves for describing correct sentence analysis. The model is based on enhanced restarting automata that assign a set of parallel dependency structures to every reduction of an input sentence. These structures capture the correspondence of dependency trees on different layers of linguistic description, namely layer of surface syntax and layer of language meaning.*
The novelty of this contribution consists in (i) the extension of enhanced restarting automata in order to produce tree structures with several interlinked layers and (ii) the application of these automata to the stratificational description of a natural language.

1 Introduction

Formal modeling of syntactic structure of a natural language, its syntactic analysis as well as synthesis, has an important impact on an insight into the characteristic features of the language and into the needs of its explicit description.

We attempt to provide a formal model for natural language description which would adequately reflect linguistic methods and makes it possible to formulate and refine linguistic observations and thus deepen the understanding of the language.

The proposed formal model is based on an elementary method of analysis by reduction. The *analysis by reduction* (RA henceforth, see [1, 2], here Section 1.2), serves for describing correct reductions of natural language sentences (particularly for languages with free word order) on several linguistic layers (see Section 1.1).

The proposed model is based on the concept of enhanced restarting automata that assign a set of dependency structures (DR-structures) to every reduction of an input sentence; DR-structures can capture a set of dependency trees representing sentence on different layers of linguistic description in a parallel way. The novelty of this approach consists in the formal presentation of the stepwise parallel composition of tree structures on different language layers.

In [2], natural language description is modeled as a formal string to string translation using a suitable model of restarting automata. [3] introduces a class of enhanced restarting automata with an output consisting of a single DR-tree. Here we discuss a model which is able to represent several parallel dependency structures and thus to capture relations between syntactic structures on different layers derived from RA.

1.1 Functional Generative Description

The theoretical linguistic basis for our research is provided by the Functional Generative Description (FGD in the sequel, see esp. [4]). FGD is characterized by its stratificational and dependency-based approach to the language description.

The *stratificational approaches* split language description into layers, each layer providing complete description of a (disambiguated) sentence and having its own vocabulary and syntax. We use the version of FGD that distinguishes four layers of description:[1]

- t**-layer** (tectogrammatical layer) capturing deep syntax, which comprises language meaning in a form of a dependency tree; the core concepts of this layer are dependency, valency, and topic-focus articulation, see esp. [4];
- a**-layer** (analytical layer) capturing surface syntax in a form of a dependency tree (non-projective in general);
- m**-layer** (morphological layer) capturing morphology (string of triples [word form, lemma, tag]);
- w**-layer** (word layer) capturing individual words and punctuation marks in a form of a simple string.

There are one-to-one correspondences between w- and m-layer (we leave aside small exceptions here)

* The paper reports on the research supported by the grants of GAČR No. P202/10/1333, P103/10/0783, and 405/08/0681. It is carried under the project of MŠMT ČR No. MSM0021620838.

[1] We adopt the notation of the Prague Dependency Treebank [5], a large corpus of Czech sentences, which uses FGD as its theoretical basis.

and between *m*- and *a*-layer; individual symbols of these three layers (surface layers in the sequel) reflect individual 'surface' words and punctuation marks. On the other hand, individual symbols of *t*-layer reflect only lexical words (so called function words, as, e.g., prepositions, auxiliary verbs, are captured as attributes of lexical words); moreover, surface ellipses (as, e.g., elided subject in Czech) are restored as nodes on *t*-layer.

Similarly as in other stratificational approaches, see e.g. [6], the layers are ordered; the lowest one being the simplest *w*-layer, the highest being the most abstract *t*-layer.

FGD as a *dependency-based approach* captures both surface and deep syntactic information in a form of dependency structures. Words (i.e., their *a*- and *t*-correlates, respectively) are represented as nodes of the respective trees, each node being a complex unit capturing the lexical, morphological and syntactic features; relations among words are represented by oriented edges [7]. The dependency nature of these representations is important particularly for languages with relatively high freedom of word order; it also complies with the shift of focus to deep syntactic representation for which dependency approach is commonly used.

The following example illustrates description of a sentence at four layers of FGD (slightly simplified). Such a description expresses all necessary linguistic information on a disambiguated sentence.

Sídlo dnes mohla mít ve státě Texas.
residence - today - could - have - in - state - Texas
'She (= elided Sb) could reside in the state of Texas today.'

Figure 1 shows the deep syntactic tree on *t*-layer, the surface non-projective syntactic tree on *a*-layer, the string of triples [word form, lemma, tag] on *m*-layer and the string of wordforms on *w*-layer. The dotted lines interconnect corresponding nodes. We focus on the non-trivial relation between *a*-layer and *t*-layer here; (i) preposition *ve* 'in' as well as noun *státě* 'state' in the *a*-tree are linked to the single *t*-symbol representing lexical word *stát* 'state'; (ii) similarly, both modal verb *mohla* 'could' and lexical verb *mít* 'have' are represented as the single *t*-node *mít* 'to have' (information on modal verb is preserved as the attribute 'poss'); as a result, the non-projective *a*-tree is transformed to the projective *t*-tree; (iii) moreover, subject elided in a surface sentence is restored in the *t*-tree (the node with the symbol starting with #PersPron), thus this node has no counterpart on the *a*-layer.

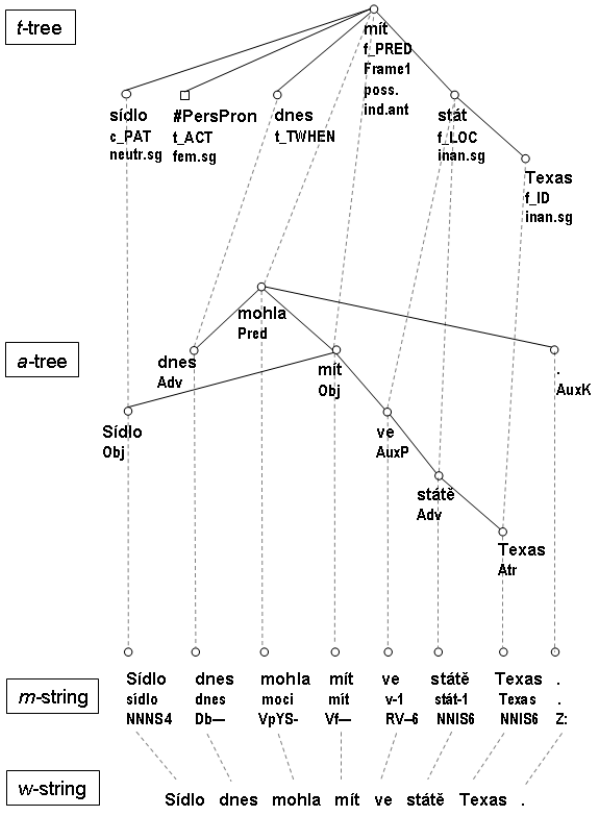


Fig. 1. Parallel representation on *t*-, *a*-, *m*- and *w*-layers of the sample sentence according to FGD.

1.2 Basic principles of analysis by reduction

Analysis by reduction is based on a stepwise simplification of an analyzed sentence. It defines possible sequences of reductions (deletions) in the sentence – each step of RA is represented by (i) deleting at least one word of the input sentence, or (ii) by replacing an (in general discontinuous) substring of a sentence by a shorter substring. Consequently, it is possible to derive formal dependency relations between individual sentence members based on the possible order(s) of reductions.

Using RA we analyze an input sentence (*w*-layer) enriched with the metalanguage information from the *m*-, *a*- and *t*-layer. Symbols on different layers representing a single word of an input sentence are processed simultaneously.

The principles of RA can be summed up in the following observations:

- The fact that a certain word (or a group of words) can be deleted implies that this word (or group of words) *depends in* RA on one of the words retained in the simplified sentence; the latter being called *governing word(s) in* RA.
- Two words (or groups of words) can be deleted in an arbitrary order if and only if they are *mutually independent in* RA.

– In order to ensure adequate modeling of natural language meaning (on t-layer), certain groups of words have to be deleted in a single step (e.g., valency frame evoking words[2] and their (valency) complementaions [1]); such words is said to constitute a *reduction component*. Even in such cases it is usual to determine governing-dependent pairs on the layer of surface syntax (a-layer). In such a case, it is necessary to define special rules for particular language phenomena.

When simplifying input sentence, it is necessary to apply certain elementary principles assuring adequate analysis:

– **principle of correctness:** simplified sentence must be correct; this principle is applied on all layers of language description;
– **principle of completeness:** simplified sentence must be complete with respect to its valency structure, i.e., each frame evoking word must be 'saturated' on the t-layer [1];
– **principle of shortening:** at least one 'surface' word (i.e., its correlates on w-, m- and a-layer) must be deleted in each step of RA;
– **principle of layers:** each step of RA must concern all symbols, i.e., symbols from all layers, representing a particular processed word.
– **principle of minimality:** each step of RA must be 'minimal' – any potential reduction step concerning less symbols in the sentence would violate the principle of completeness.

These principles imply that in a single reduction step, either (i) item(s) representing a single free modification or (ii) items representing valency complementations of a single frame evoking word together with their governing word are processed.

The sentence is simplified until so called *core predicative structure* is reached (typically formed by sentence predicate and its valency complementations).

The basic principles of RA are exemplified on several reduction steps of our sample Czech sentence from Section 1.1; they illustrate how the sentence is simplified and how the fragments of its DR-structure (a- and t-trees) are built.

Sídlo dnes mohla mít ve státě Texas.
residence - today - could - have - in - state - Texas
'She (= elided Sb) could reside in the state of Texas today.'

[2] A frame evoking word is a lexical word (verb, noun, adjective or adverb) that requires a set of syntactico-semantic complementations, as e.g. the verb *to give* requires three complementations, namely actor (ACT, who gives something), patient (PAT, what is given) and addressee (ADDR, to whom something is given).

w-layer	m-layer	a-layer	t-layer
Sídlo	sídlo.NNNS4	Obj	PAT
			ACT
dnes	dnes.Db- - -	Adv	TWHEN
mohla	moci.VpYS-	Pred	
mít	mít.Vf- - -	Obj	PRED.Frame1.poss
ve	v-1.RV- -6	AuxP	
státě	stát.NNIS6	Adv	LOC
Texas	Texas.NNIS6	Atr	ID
.	..Z:	AuxK	

Fig. 2. Representation of the sample sentence at four layers of FGD (simplified).

Figure 2 presents a (simplified) representation of the sample sentence at four layers of FGD. Each column captures one layer of language description (w-, m-, a- and t-layer, respectively, see Section 1.1). Rows capture information related to particular words and punctuation marks (one or more rows for an individual word/punctuation mark, depending on its surface and deep word order, see [2]).

We can see that the sentence contains a verbonominal predicate (the predicate consisting of the light verb *mít* 'to have' and the noun *sídlo* 'residence'); this predicate evokes two valency positions, actor (ACT) and local modification (LOC); the nominal part of the predicate is analyzed as patient (PAT) of the verb in FGD (encoded as Frame1 ... ACT PAT LOC).

There are two possible orders of reductions: **(1)** In the first reduction step of RA, the word *Texas* is reduced – the simplified sentence is grammatically correct and it is complete (i.e., its valency structure is complete). The respective symbols on all layers (interlinked by dotted lines in Figure 1) are processed simultaneously: those on w- and m-layers are deleted; the a-symbol is analyzed as depending on the a-symbol for the preceding noun *stát* 'state' (as its syntactic attribute, Atr); further, the t-symbol for *Texas* is analyzed as depending on the t-symbol for *stát* 'state' (ID indicates a proper name).
(2) Alternatively, RA may start with the reduction step processing the word *dnes* 'today'. Again, respective symbols on w- and m-layers are deleted; based on surface syntactic and morphological categories, the a-symbol and t-symbol are included in the a- and t-structures, respectively.

After processing the words *Texas* and *dnes*, RA continues with the predicate and its complementations. As they form a reduction component, they must be processed in a single step on the t-layer (otherwise a principle of completeness on the t-layer is violated). Thus the sentence is simplified on the surface layers (i.e., w-, m- and a-layers) first and only when all va-

lency complementations are reduced there, the frame evoking predicate and its complementations can be processed on the *t*-layer. Let us describe the analysis of this component in more detail.

First, a prepositional group *ve+státě* 'in (the) state' is identified in the sentence – it is processed in a single step on 'surface' layers (its *w*- and *m*- symbols are deleted and *a*-symbols are included into the *a*-structure as adverbial modification of the verb *mít* 'to have'). The whole prepositional group will be represented as a single *t*-node *stát*, see Figure 1. It will be identified as a local valency complementation LOC on the *t*-layer. Second, the noun *sídlo* 'residence' is processed on the surface layers and marked as a valency PAT complementation on the *t*-layer. Third, the symbol for elided subject, which is restored on the *t*-layer, is marked as ACT valency complementation. All the valency complementations of the predicate are identified now, the valency frame Frame1 is saturated.

Next, we focus on the modal verb *mohla* '(she) can'. On the *a*-layer, this is a governing node of the lexical verb *mít*; the respective edge is created, which results in a non-projective *a*-tree (the symbols on the surface layers are deleted). On the other hand, modal verbs are accounted functional verbs in FGD and thus represented as attributes of lexical verbs in *t*-trees (*mít* in our case, value 'pass' in the respective *t*-node). Thus the non-projectivity is eliminated in the *t*-tree.

Now the predicate with its complementations ACT, PAT and LOC (Frame1) can be identified as the core predicative structure on the *t*-layer, the relevant edges for individual valency complementations are created and the simplified sentence is accepted (in the accepting step, the final full stop is processed on the surface layers).

2 Restarting automata

First, we introduce a relevant type of a simple restarting automaton – sRL-automaton – rather informally. From technical reasons, we do it in a slightly different way than in [8].

An sRL-*automaton* M is (in general) a nondeterministic machine with a finite-state control Q, a finite characteristic alphabet Γ, and a head (window of size 1) that works on a flexible tape delimited by the left sentinel ¢ and the right sentinel \$ (¢, \$ $\notin \Gamma$). For an input $w \in \Gamma^*$, the initial tape inscription is ¢w\$. To process this input, M starts in its initial state q_0 with its window over the left end of the tape, scanning the left sentinel ¢. According to its transition relation, M performs the following operations in the individual steps:

- *moves to the right or to left* – shift the head one position to the right or to the left;
- dl – deletes the visited symbol and shifts the head on its right neighbor;
- wr[b] – rewrites the visited symbol by the symbol b;
- pb – serves for marking (putting a pebble on) the visited item only: marked items are used as nodes in DR-trees (in any other aspect it is an empty operation, see later);
- *accept* – halts the computations and accepts the input word.

Of course, neither the left sentinel ¢ nor the right sentinel \$ must be deleted. At the right end of the tape M either halts and accepts, or it halts and rejects, or it *restarts*, that is, it places its window over the left end of the tape and reenters the initial state. It is required that prior to the first restart step and also between any two restart steps, M executes at least one delete operation. During each step, M can change its internal state.

We can see that any finite computation of M consists of certain phases. A phase, called a *cycle*, starts in a restarting configuration, the window is moved along the tape by performing its operations until a restart operation is performed and thus a new restarting configuration is reached. If no further restart operation is performed, each finite computation necessarily finishes in a halting configuration – such a phase is called a *tail*. We assume that no delete and rewrite operation is executed in a tail computation.

The notation $u \vdash_M^c v$ denotes a reduction performed during a cycle of M that begins with the tape inscription ¢u\$ and ends with the tape inscription ¢v\$; the relation $\vdash_M^{c^*}$ is the reflexive and transitive closure of $\vdash_M^c$.

A string $w \in \Gamma^*$ is *accepted* by M, if there is an accepting computation which starts in the restarting configuration with the tape inscription ¢w\$ and ends by executing the *accept* operation. By $L_C(M)$ we denote the language consisting of all words accepted by M; we say that M *recognizes (accepts) the characteristic language* $L_C(M)$.

Further we will refer to a sRL-automaton M as a tuple $M = (\Gamma, ¢, \$, R(M), A(M))$, where Γ is a characteristic vocabulary (alphabet), ¢ and \$ are sentinels not belonging to Γ, $R(M)$ is a finite set of restarting instructions over Γ and $A(M)$ is a finite set of accepting meta-instructions over Γ.

Remark: sRL-automata are two-way nondeterministic automata which allow to check whole input sentence prior to any changes. It resembles linguist who can read the whole sentence first and reduce the sentence in a correct way afterward. We choose nondeterministic model to enable various orders of reductions. This can serve for verification of independency between individual parts of a sentence.

Similarly as in [8], we use a restricted type of sRL-automata for which the number of rewrite, delete and pebble operations made per cycle is limited by a constant. Such sRL-automata can be described by (meta)-instructions, which describe the moves of the head and the changes of the states implicitly. Each cycle of M is described by a single *restarting instructions* over Γ of the following form:

$I_{\mathrm{R}} = (\mathrm{¢} \cdot E_0, [a_1]_1 o_1, E_1, [a_2]_2 o_2, E_2, \ldots, E_{s-1}, [a_s]_s o_s, E_s \cdot \$, \mathsf{Restart})$, where

- $E_0, E_1, \ldots, E_s$ $(s > 0)$, are regular languages over Γ (usually represented by regular expressions);
- $o_1, \cdots, o_s \in \{\mathrm{dl}, \mathrm{pb}, \mathrm{wr}[b]\}$, where $b \in \Gamma$.
- The symbols $a_1, a_2, \ldots, a_s \in \Gamma$ correspond to the symbols on which the corresponding operations $\{o_1, \cdots, o_s\}$ are executed.

Let us define auxiliary function $o : \Gamma \to \Gamma$ for each operation $o \in \{\mathrm{dl}, \mathrm{pb}, \mathrm{wr}[b]\}$:

- $\mathrm{pb}(a_i) = a_i$,
- $\mathrm{dl}(a_i) = \lambda$, and
- $\mathrm{wr}[b](a_i) = b$.

When trying to execute I_{R} starting from a tape inscription $\mathrm{¢}w\$$, M will get stuck (and so reject), if w does not admit a factorization of the form $w = v_0 a_1 v_1 a_2 \ldots v_{s-1} a_s v_s$ such that $v_i \in E_i$ for all $i = 0, \ldots, s$. On the other hand, if w admits factorizations of this form, then one of them is chosen nondeterministically, and $\mathrm{¢}w\$$ is transformed (reduced) into $\mathrm{¢}v_0 o_1(a_1) v_1 \cdots v_{s-1} o_s(a_s) v_s\$$.

Tails of accepting computations are described by *accepting instructions* over Γ of the form

$I_{\mathrm{A}} = (\mathrm{¢} \cdot E_0, [a_1]_1, E_1, [a_2]_2, E_2, \ldots, E_{s-1}, [a_s]_s,\ E_s \cdot \$, \mathsf{Accept})$, where individual E_i are regular languages over Γ.

For our linguistic application (i.e., modeling FGD), we consider the accepting instructions with finite E_i's only.

A tail performed by the instruction I_{A} starts with the inscription on the tape $\mathrm{¢}z\$$; if $z \in E_0 a_1 \cdots a_s E_s$, then M accepts z (and the whole computation as well). Otherwise the computation halts with rejection. This special form of accepting instruction is introduced with regard to the future enhancements of restarting automata.

The class of all sRL-automata are denote as sRL.

A crucial role in our applications has the following property of restarting automata.

(Correctness Preserving Property) A sRL-automaton M is *correctness preserving* if $u \in L_C(M)$ and $u \vdash_M^{\mathrm{c}*} v$ imply that $v \in L_C(M)$.

It is already known that all deterministic sRL-automata are correctness preserving. On the other hand, it is easy to design a nondeterministic sRL-automaton which is not correctness preserving (see [8]). We consider only the correctness preserving automata in the sequel in order to model adequately the analysis by reduction.

2.1 Restarting automata enhanced with DR-structures

In this section we introduce *enhanced restarting automata*, so called sERL-automata. During their computations, these automata build structures consisting of deleted, rewritten, or marked items and of directed edges between pairs of such items.

Enhanced restarting automata sERL-automata were formally introduced in a restricted form in [3]. Their formal definition is rather long and very technical. So we prefer to use an informal description of the model and concentrate on their possible applications. In contrast to sRL-automata, there can be attached a so called DR-structure to any item of the tape of an sERL-automaton.

A DR-structure is a slight generalization of DR-trees used in [7]. It is an oriented graph $D = (V, H_d)$, where V is a finite set of nodes, and H_d is a finite set of edges. A *node* $u \in V$ is a tuple $u = [i, j, a]$, where a is a symbol assigned to the node, i, j are natural numbers; i represents the horizontal position of the node u, j represents the vertical position of u (it is equal to 0 or to the number of nodes with the same horizontal position i from which there are oriented paths to u). An *edge* h of D is an ordered pair of nodes $h = (u, v)$. We define two types of edges:

- *Oblique edge:* $h = (u, v)$, where $u = [i_u, j_u, a]$, $v = [i_v, j_v, b]$ and $i_u \neq i_v$;
- *Vertical edge*: $h = (u, v)$, where $u = [i_u, j_u, a]$, $v = [i_v, j_v, b]$ and $i_u = i_v$, $j_v = j_u + 1$;

We say that $D = (V, H_d)$ is a DR-tree if the graph D is an oriented tree (with a single root, in which all maximal paths in D end).

Each sERL-automaton M_{e} is actually an sRL-automaton with enhanced instructions. An enhanced instruction is a pair $I_{\mathrm{e}} = (I, G)$ consisting of an instruction I of a sRL-automaton and an acyclic graph G representing the required structure for symbols processed – deleted, rewritten or marked (pebbled) – during the application of the instruction I. The restrictions put on the set of edges of G are described below together with the application of an enhanced instruction $I_{\mathrm{e}} = (I, G)$, where $G = (U, H)$, on a tape containing $\mathrm{¢}w\$$.

All the symbols on the tape are stored in so called items which are actually nodes of a DR-structure. I.e., a symbol x is stored in a node $[i, j, x]$, where i is

its horizontal position. Initially, horizontal position of n-th input symbol of the input equals n, $j = 0$ and there are no edges between the items.

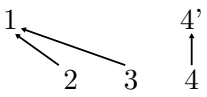
Restarting instruction. If I is a restarting instruction $I = (¢ \cdot E_0, [a_1]_1 o_1, E_1, [a_2]_2 o_2, E_2, \ldots, E_{s-1}, [a_s]_s o_s, E_s \cdot \$, \mathsf{Restart})$ then $o_1, \cdots, o_s \in \{\mathrm{dl}, \mathrm{pb}, \mathrm{wr}[b]\}$ ($b \in \Gamma$) are the operations performed on symbols $\{a_1, \cdots, a_s\}$. Let $o_{i_1} = wr[b_{i_1}], \ldots, o_{i_r} = wr[b_{i_r}]$, for some $r \geq 0$, be all rewrite operations from $\{o_1, \ldots, o_s\}$. Let $G = (U, H)$, where $U = \{1, 2, \ldots, s, i'_1, i'_2, \ldots, i'_r\}$, and let the nodes correspond to the symbols $a_1, \ldots, a_s$ and symbols $b_{i_1}, \ldots, b_{i_r}$, respectively. An edge $(u,v) \in H$ is of one of the following forms:

1. deleting: $u = i$ corresponds to a deleted symbol a_i (hence $o_i = \mathrm{dl}$, $1 \leq i \leq s$) and $v = j$ for some $j \in U$, $j \neq i$. Let us note that the deleting edge is always oblique.
2. rewriting: $u = i$ corresponds to a rewritten symbol a_i (hence $o_i = \mathrm{wr}[b_{i_j}]$, $b_{i_j} \in \Gamma$, $1 \leq i \leq s$, $1 \leq j \leq r$) and $v = i'_j$. Let us note that the rewriting edge is always vertical.

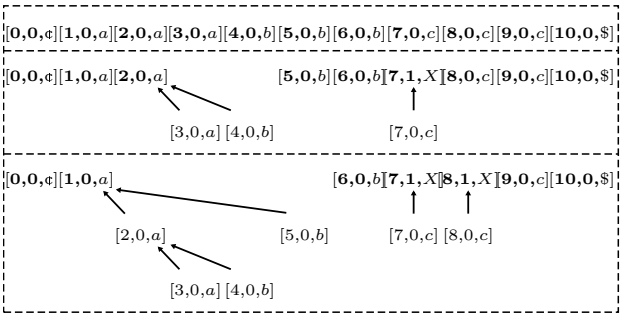
An application of I_{e} on a word w consists in:

1. Choosing a factorization of $¢w\$$ of the form $w = v_0 a_1 v_1 a_2 \ldots v_{s-1} a_s v_s$ such that $v_i \in E_i$ for all $i = 0, \ldots, s$. On the other hand, if w does not admit any factorization of this form, then I cannot be applied on w.
2. Rewriting the tape containing $¢w\$$ into the tape containing $¢v_0 o_1(a_1) v_1 \cdots v_{s-1}\ o_s(a_s) v_s\$$.
3. For each edge $e \in H$ a new edge is inserted into the current DR-structure.
   If $e = (i, j)$ is a deleting edge, an oblique edge from the item containing deleted a_i into the item containing the symbol corresponding to j (either a_j, if $1 \leq j \leq s$, or b_j, when $j \in \{i'_1, \ldots, i'_r\}$) is inserted.
   If $e = (i, j)$ is a rewriting edge, a vertical edge from the item containing a_i into the item containing b_j is inserted ($j \in \{i'_1, \ldots, i'_r\}$) and the vertical position of the new item containing b_j is set to the value by $q + 1$, where q is the vertical position of the item containing a_i.
   If there was a DR-structure attached to some of the deleted of rewritten cell, the structure is preserved and combined into a larger graph.

Example 1: Let $I_1 = ((¢ \cdot a^*, [a]_1 \mathrm{pb}, \lambda, [a]_2 \mathrm{dl}, \lambda, [b]_3 \mathrm{dl}, b^* X^*, [c]_4 \mathrm{wr}[X], c^* \cdot \$, \mathsf{Restart}), D_1)$ be an enhanced restarting instruction with the graph D_1 of the following form:



Then two consecutive applications of I_1 on the word *aaabbbccc* will result in the following sequence of tape contents (in the figure the tape content consists of the bold items depicted in the upper horizontal part of a picture for a particular configuration):

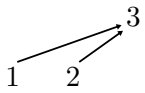


Accepting instruction. If I is an accepting instruction $I = (¢ \cdot E_0, [a_1]_1 \mathrm{pb}, E_1, [a_2]_2 \mathrm{pb}, E_2, \ldots, E_{s-1}, [a_s]_s \mathrm{pb}, E_s \cdot \$, \mathsf{Accept})$ then the symbols $\{a_1, \cdots, a_s\}$ are pebbled and they correspond to the nodes in $U = \{1, 2, \ldots, s\}$. All edges $(u, v) \in H$ are oblique and have the same properties as deleting edges in graphs enhancing restarting instructions.

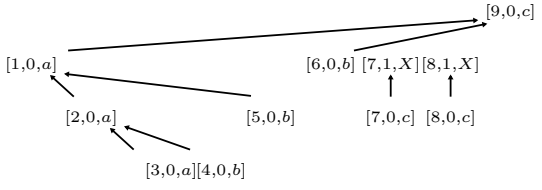
An application of $I_{\mathrm{e}} = (I, G)$, with $G = (U, H)$, on a word w consists in:

1. Choosing a factorization of $¢w\$$ of the form $w = v_0 a_1 v_1 a_2 \ldots v_{s-1} a_s v_s$ such that $v_i \in E_i$ for all $i = 0, \ldots, s$. On the other hand, if w does not admit any factorization of this form, then I cannot be applied on w.
2. For each edge $e = (i, j)$ in H a new oblique edge is inserted into the current DR-structure. The inserted edge starts from the item containing a_i and ends in the item containing symbol a_j.
   If there was a DR-structure attached to some of the connected items, the structure is preserved and combined into the resulting graph.

Example 2: Let $I_2 = ((¢, [a]_1 \mathrm{pb}, \lambda, [b]_2 \mathrm{pb}, X^*, [c]_3 \mathrm{pb}, \$, \mathsf{Accept}), D_2)$ be an enhanced accepting instruction with the graph D_2 of the following form:



Then after applying I_2 on the resulting DR-structure from Example 1 we obtain the following final DR-structure:



Computation of enhanced restarting automata. A *(restarting) configuration* $C = (T, D)$ *of a computation by* $M_e = (\Gamma, ¢, \$, ER(M_e), EA(M_e))$ consists of the set T of items representing current tape content and a DR-structure D. By an application of an enhanced instruction I, the tape content is changed and the DR-structure can grow. The initial configuration $C_0 = (T_w, D_\emptyset)$ for an input word $w = x_1 \dots x_n$ consists of the set of items representing the initial content of the tape $T_w = \{[0, 0, ¢]\} \cup \{[0, i, x_i] \mid i = 1, \dots, n\} \cup \{(n+1, 0, \$]\}$ and the empty DR-structure $D_\emptyset = (\emptyset, \emptyset)$. By application of an enhanced instruction I, a configuration C is transformed onto a new configuration C'.

An input word w is *accepted* by M_e if there exists a computation starting with the initial configuration for w which ends in a configuration C_a by an application of an accepting enhanced instruction. The result of the computation is the DR-structure of C_a.

Similarly as for sRL-automata, we define the characteristic language of M_e as $L_C(M_e) = \{w \mid w \in \Gamma$ and M_e accepts $w\}$. Moreover, DR-language of M_e is the set $\mathsf{DR}(M_e) = \{D \mid D$ is a result of some accepting computation of $M_e\}$.

2.2 Enhanced automata with several layers

First, we introduce a technical notion of *projection*. Let Σ and Γ ($\supset \Sigma$) be alphabets. The projection from Γ^* onto Σ^* denoted as Pr^Σ is the morphism defined as $a \mapsto a$ (for $a \in \Sigma$) and $A \mapsto \lambda$ (for $A \in \Gamma \setminus \Sigma$). Similarly, we define the projection of languages: $Pr^\Sigma : \mathcal{P}(\Gamma^*) \mapsto \mathcal{P}(\Sigma^*)$.

Similarly as above, we introduce a *projection for* DR*-structures*. Let D be a DR-structure from a DR-language over an alphabet Γ, $Pr^\Sigma(D)$ denotes a DR-structure over Σ that is obtained from D by removing all nodes with (at least one) symbol from $\Gamma \setminus \Sigma$ and all edges incident to that nodes. A projection may be in an obvious way extended onto projection of a DR-language over Γ.

Let $\Sigma_1, \dots, \Sigma_j$, for some $j \geq 1$, be a sequence of pairwise disjoint alphabets and $\Gamma = \Sigma_1 \cup \dots \cup \Sigma_j$. We say that sERL-automaton $M_j = (\Gamma, ¢, \$, ER(M_j), EA(M_j))$ is *an enhanced simple* sERL*-automaton with* j *layers* ((j)-sERL-automaton for short) if the following assumptions are fulfilled:

- M_j is *correctness preserving*;
- M_j is allowed to rewrite a symbol from Σ_i (for $1 \leq i \leq j$) by a symbol from the same sub-vocabulary Σ_i, only. We refer to the symbols from Σ_i as the symbols on layer i (or i-symbols).

3 FGD as a (4)-sERL-automaton

Our ultimate goal is to model FGD – we consider a (4)-sERL-automaton M_{FD} to be a suitable formal frame for this linguistic theory.

Let $M_{\mathsf{FD}} = (\Gamma, ¢, \$, ER(M_{\mathsf{FD}}), EA(M_{\mathsf{FD}}))$; Γ consists of four parts $\Gamma = \Sigma_w \cup \Sigma_m \cup \Sigma_a \cup \Sigma_t$ which correspond to the respective layers of FGD (Section 1.2). Recall that the symbols from individual layers can be quite complex.

A *language of layer* $\ell \in \{w, m, s, t\}$ *accepted by* M_{FD} is obtained as a projection of the characteristic language onto Σ_ℓ, i.e., $L_\ell(M_{\mathsf{FD}}) = Pr^{\Sigma_\ell}(L_C(M_{\mathsf{FD}}))$.

The characteristic language $L_C(M_{\mathsf{FD}})$ contains input sequences (over Σ_w) interleaved with metalanguage information in the form of symbols from $\Sigma_m \cup \Sigma_a \cup \Sigma_t$. Then, the language of correct sentences of the studied natural language is $L_w = Pr^{\Sigma_w}(L_C(M_{\mathsf{FD}}))$. In our case, it defines the set of correct Czech sentences.

Similarly, a DR*-language of layer* ℓ *accepted by* M_{FD} is obtained as $\mathsf{DR}_\ell(M_{\mathsf{FD}}) = Pr^{\Sigma_\ell}(\mathsf{DR}(M_{\mathsf{FD}}))$, $\ell \in \{w, m, s, t\}$. Let us note that $\mathsf{DR}_w(M_{\mathsf{FD}})$ and $\mathsf{DR}_m(M_{\mathsf{FD}})$ are empty (L_w and L_m are string languages). Further, $\mathsf{DR}_a(M_{\mathsf{FD}})$ and $\mathsf{DR}_t(M_{\mathsf{FD}})$ are languages of DR-trees. Each DR-tree $T \in \mathsf{DR}_t(M_{\mathsf{FD}})$ is *projective* (with respect to its descendants); that is, for each node n of the DR-tree T all its descendants (including also the node n) constitute a contiguous sequence in the horizontal ordering of nodes of the tree T. On the other hand, trees from $\mathsf{DR}_a(M_{\mathsf{FD}})$ can be in general non-projective.

The DR-language $\mathsf{DR}_t(M_{\mathsf{FD}})$ represents the set of meaning descriptions in FGD whereas and $\mathsf{DR}_a(M_{\mathsf{FD}})$ models the set of (surface) syntactic trees.

Let us note that $L_t(M_{\mathsf{FD}})$ is designed as a deterministic context-free language. Readers familiar with restarting automata can see that $L_C(M_{\mathsf{FD}})$ is a deterministic context-sensitive language and $L_w(M_{\mathsf{FD}})$ is a context-sensitive language.

We have not mentioned so far the edges from DR-structures from $\mathsf{DR}(M_{\mathsf{FD}})$ which have the edges with nodes (their symbols) in different layers. These edges serve for connecting the corresponding lexical units on different layers (see dotted lines in Figure 1 and are obtained by applications of extended restarting meta-instructions of M_{FD}.

Here such an edge connects nodes on neighboring layers only. I.e., this edges connect w-layer nodes to m-layer nodes, m-layer nodes to a-layer nodes, a-layer nodes to t-layer nodes, and nothing else (see Figure 1).

Concluding remarks In this paper, encouraged by [9, 10], we extend the formal model of natural language description based on FGD so that it outputs neither lists of words nor lists of symbols but the

so called reduction language and a set of complex DR-structures. We plan to study the relation between reduction languages and DR-languages more deeply in the near future.

The novelty of this contribution consists in (i) the extension of enhanced restarting automata in order to produce tree structures with several interlinked layers and (ii) the application of these automata to the stratificational description of a natural language. Moreover, we outline a formalization of the basic methodology of FGD in terms derived from the automata theory and from the theory of parsing schemata as well. We envisage that the proposed methodology is not FGD-specific and that similar approach can be used to obtain a formal frame for other language descriptions, as e.g. those presented in [6] and [11].

References

1. M. Lopatková, M. Plátek, V. Kuboň: *Modeling syntax of free word-order languages: dependency analysis by reduction.* In: Matoušek, V., Mautner, P., Pavelka, T., (eds), Proceedings of Text, Speech and Dialogue International Conference, TSD 2005, Volume 3658 of LNAI, Berlin Heidelberg, Springer-Verlag (2005) 140–147.
2. M. Lopatková, M. Plátek, P. Sgall: *Towards a formal model for functional generative description, analysis by reduction and restarting automata.* The Prague Bulletin of Mathematical Linguistics, **87**, 2007, 7–26.
3. M. Plátek, F. Mráz, M. Lopatková: *Restarting automata with structured output and functional generative description.* In: Dediu, A., Fernau, H., Martin-Vide, C. (eds), Proceedings of the Fourth International Conference Language and Automata Theory and Applications, LATA 2010, Berlin Heidelberg, Springer-Verlag, 2010, 500–511.
4. P. Sgall, E. Hajičová, J. Panevová: *The Meaning of the Sentence in Its Semantic and Pragmatic Aspects.* Reidel, Dordrecht (1986).
5. J. Hajič, E. Hajičová, J. Panevová, P. Sgall, P. Pajas, J. Štěpánek, J. Havelka, M. Mikulová, M.: *Prague Dependency Treebank 2.0.* Linguistic Data Consortium, Philadelphia, PA, USA, 2006.
6. I.A. Mel'čuk: *Dependency syntax: theory and practice.* State University of New York Press, Albany, 1988.
7. T. Holan, V. Kuboň, K. Oliva, M. Plátek: *Two useful measures of word order complexity.* In: Polguere, A., Kahane, S. (eds), Processing of Dependency-Based Grammars: Proceedings of the Workshop (COLING/ACL'98), Montreal, ACL, 1998, 21–28.
8. H. Messerschmidt, F. Mráz, F. Otto, M. Plátek: *Correctness preservation and complexity of simple RL-automata.* In: Implementation and Application of Automata, Volume 4094 of LNCS, Berlin Heidelberg, Springer-Verlag, 2006, 162–172.
9. R. Gramatovici, C. Martín-Vide: *Sorted dependency insertion grammars.* Theor. Comput. Sci., **354** (1), 2006, 142–152.
10. S. Bensch, F. Drewes: *Millstream systems.* Report UMINF 09.21, Umeå University, 2009.
11. J. Kunze: *Abhängigkeitsgrammatik.* Volume XII of Studia Grammatica. Akademie Verlag, Berlin, 1975.