SEMANTIC PARSING AS GRAPH LANGUAGE TRANSFORMATION - A MULTIDIMENSIONAL APPROACH TO PARSING HIGHLY INFLECTIONAL LANGUAGES

Eero Hyvönen
Helsinki University of Technology
Digital Systems Laboratory
Otakaari 5A
02150 Espoo 15
FINLAND

ABSTRACT

The structure of many languages with "free" word order and rich morphology like Finnish is rather configurational than linear. Although non-linear structures can be represented by linear formalisms it is often more natural to study multidimensional arrangement of symbols. Graph grammars are a multidimensional generalization of linear string grammars. In graph grammars string rewrite rules are generalized into graph rewrite rules. This paper presents a graph grammar formalism and parsing scheme for parsing languages with inherent configurational flavor. A small experimental Finnish parsing system has been implemented (Hyvönen 1983).

1 A SIMPLE GRAPH GRAMMAR FORMALISM WITH A CONTROL FACILITY

In applying string grammars to parsing natural Finnish several problems arise in representing complex word structures, argeements, "free" word ordering, discontinuity, and intermediate depencies between morphology, syntax and semantics. A strong, multidimensional formalism that can cope with different levels of language seems necessary. In this chapter a graph grammar formalism based on the notions of relational graph grammars (Rajlich 1975) and attributed programmed graph grammars (Bunke 1982) is developed for parsing languages with configurational structure.

Definition 1.1 (relational graph, r-graph)

Let ARCS, NODES, and PROPS be finite sets of symbols. A relational graph (r-graph) RG is pair RG = (EDGES, NP) consisting of a set of edges

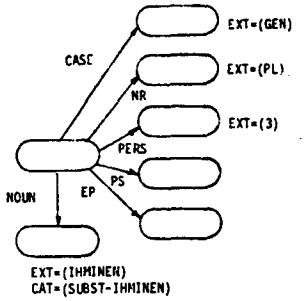
EDGES $\subseteq$ ARCSxNODESxNODES

and a function NP that associates each node in EDGES to a set of labeled property values:

NP: NODESxPROPS -> PVALUES

PVALUES is the set of possible node property values. They are represented as sets of symbols or lists.

Example: Figure 1.1 depicts the morphological r-graph representation of Finnish word "ihmisten" (the humans') and its edges as a list. EXT-property expresses the set of symbols the node currently refers to (extension); CAT tells the syntactico-semantic category of the node.



((NOUN N1 N2)
(CASE N1 N3)
(NR N1 N4)
(PERS N1 N5)
(PS N1 N6)
(EP N1 N7))

Fig. 1.1. Morphological r-graph representation of word "ihmisten" (the humans).

Definition 1.2 (r-production)

An r-production RP is a pair:

RP = (LS, RS)

LS (left side) and RS (right side) are r-graphs. An RP is said to be applicable to an r-graph G iff $EDGES_{LS} \subseteq EDGES_G$ and the values in NP_{LS} are subsets of corresponding values in NP_G for each node in LS.

Definition 1.3 (direct r-derivation)

The direct r-derivation of r-graph H from r-graph G via an r-production RP = (LS, RS) is defined by the following algorithm:

Algorithm 1.1 (Direct r-derivation)

Input: An r-graph G and
an r-production RP = (LS, RS)
Output: An r-graph H derived via RP
from G

```
PROCEDURE Direct-r-derivation:
   BEGIN
      IF RP is applicable to G (see text)
         THEN
            EDGES_G := EDGES_G - EDGES_LS
            H := G ⊎ RS
            RETURN H
         ELSE
            RETURN "Not applicable"
   END
```

Here ⊎ is an operation defined for two r-graphs RG1 and RG2 as follows:

$H = RG1 \uplus RG2$

iff

$EDGES_H = EDGES_{RG1} \cup EDGES_{RG2}$ and

$NP_H(ni, propj) = NP_{RG2}(ni, propj)$ for any property propj in every node ni in RG2.

Time complexity: Direct r-derivations are essentially set operations and can be performed efficiently. By using a hash table the expected time complexity is O(n) with respect to the size of the production (it does not depend on the size of the object graph). The worst case complexity is O(n**2).

Example: Figure 1.2 represents an r-production and figure 1.3 its application to an r-graph. We have designed a meta-production description facility for r-productions by which match-predicates can be attached to nodes and arcs in order to test and modify node properies. The instantiation of a meta-production is found context-dependently while matching the production left side. It is also possible to specify some special modifications to the derivation graph by meta-productions.

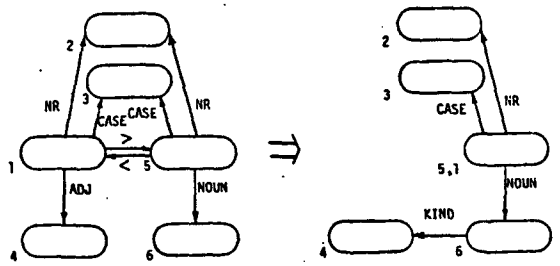


Fig. 1.2. Production ADJ-ATTR to identify adjective attributes.

Definition 1.4 (r-graph grammar and r-graph language)

An r-graph grammar (RGG) is a pair:

RGG = (PROD, START)

PROD is a set of r-productions and START is a set of r-graphs.

An r-graph language (RGL) generated by an r-graph grammar is the set of all derivable r-graphs from any r-graph in START by any sequence of applicable r-productions of PROD:

$RGL = \{R\text{-graph} \mid START \Rightarrow_{PROD} R\text{-graph}\}$

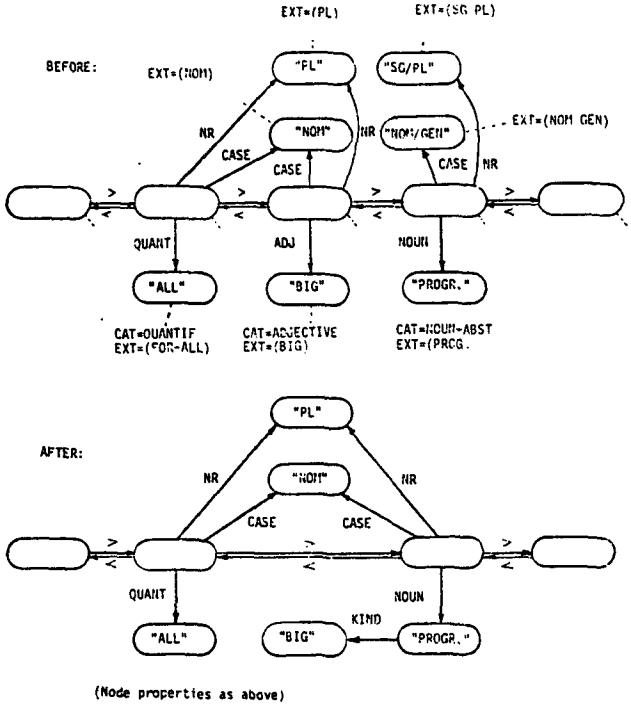


Fig. 1.3. The effect of applying production ADJ-ATTR (fig. 1.2) to an r-graph.

Definition 1.5 (controlled r-graph grammar)

A controlled r-graph grammar (CRG) is a pair:

CRG = (CG, RGG)

CG is an r-graph called control graph (c-graph). Its interpretation is defined very much in the same way as with ATN-networks. The actions associated to arcs are direct r-derivations (def. 1.3). RGG is an r-graph grammar (def. 1.4).

Example: Figure 1.4 illustrates a c-graph expressing potential attribute configurations of nouns belonging to category NOUN-HUMAN. Adjective, pronoun and genetive attributes and a quantifier may be identified by corresponding r-productions (the meaning of (READWORD)- and (PUT-LAST)-arcs is not relevant here).

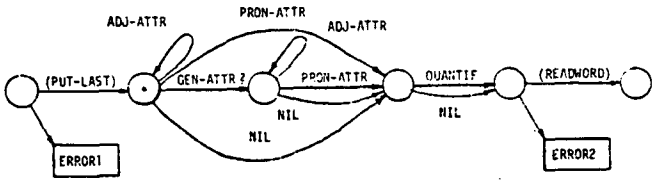


Fig. 1.4. A control graph expressing attribute configurations of syntactico-semantic word category NOUN-HUMAN.

Definition 1.6 (Controlled graph language)

A controlled graph language (CGL) corresponding to a controlled r-graph grammar CRG = (CG, RGG) is the set of r-graphs derived by the CG using the start graphs START and the productions of the grammar RGG.

2 A GRAPH GRAMMAR PARSING SCHEME

2.1 Function and structure

Figure 2.1 depicts a RGG-based parsing scheme that we have applied to natural language parsing. Roughly spoken, the input of the parser, i.e. the set START of a CRG, is the morphological representation(s) of a sentence. The output is a set of corresponding semantic deep case representations. Parsing is seen as a multidimensional transformation between the morphological and semantic levels of a language. These levels are seen as graph languages. The parser essentially defines a "meaning preserving" mapping from the morphological representations of a sentence into its semantic representations. The transformation is specified by a controlled r-graph grammar. The control graph is not predefined but is constructed dynamically according to the individual words of the current sentence. During parsing morphological and semantic representations are generated in parallel as words are read from left to right.

2.2 Specification of the morphological and semantic graph languages

Morphological level. The morphological representation of a sentence consists of star-like morphological representations of the words (fig. 1.1) that are glued together by sequential >- and <-relations (fig. 1.3).

Semantic level. The semantic representation of a sentence consists of a semantic deep case structure corresponding to the main verb. Deep case constituents have their own semantic case structures corresponding to their main words.

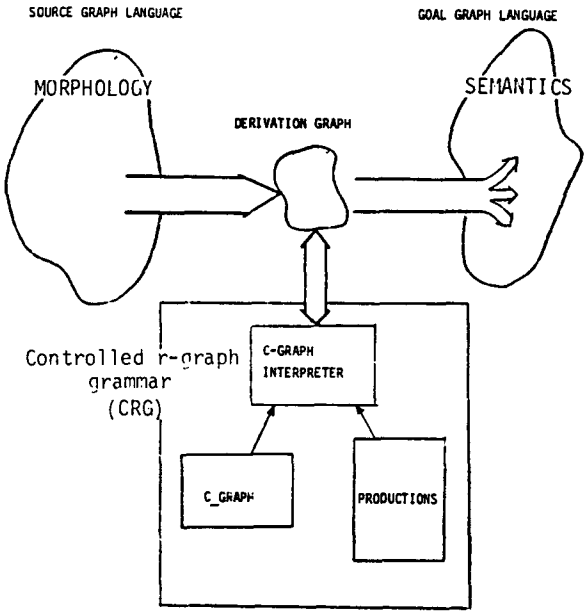


Fig. 2.1. A parsing scheme for transforming graph languages.

Example: Figure 2.2 illustrates the semantic representation of question " Kuka luennoitsija on luennoinut jonkun seminaarimaisen kurssin tietojenkäsittelyteoriasta syksyllä 1981" ("Which lecturer has lectured some seminar-type course on computer science in the autumn 1981").

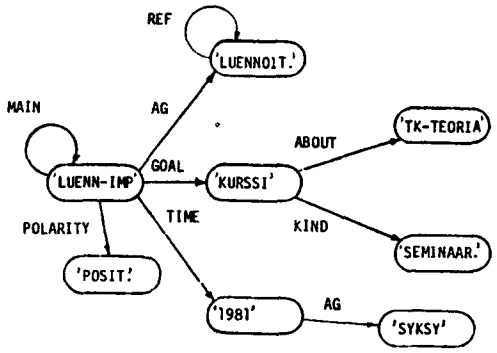


Fig. 2.2. Semantic graph representation of a Finnish question. Node properties are not shown.

2.3 Specification of the graph language transformation

The transformation is specified by an agenda of prioritized c-graphs. Initially, the agenda consists of a set of sentence independent "transformational" c-graphs (that, for example, transform passive clauses into active ones) and

sentence dependent c-graphs corresponding to the syntactico-semantic categories of the individual words in the sentence. For example, the c-graph of fig. 1.4 corresponds to nouns belonging to category NOUN-HUMAN. It tries to identify semantic case constituents by the productions corresponding to the arcs. Fig. 1.2 illustrates the production ADJ-ATTR (adjective attribute) used in the c-graph of fig. 1.4. The interpretation of the production is: If there is an adjective preceeding a noun in the same case and number the words are in semantic KIND relation with each other. As a whole, the agenda constitutes a modular, sentence dependent c-graph.

Parsing is performed by interpreting the agenda. Different strategies could be applied here; the structure of the c-graphs depend on the choice. In our experimental system parsing is performed by interpreting the first c-graph in the agenda. The c-graphs are defined in such way that they intepret each other and glue morphological representations of words into the derivation graph (arcs (READWORD) and (PUTLAST) in fig. 1.4) until a grammatical semantic representation (or in ambiguous cases several ones) is reached.

2.4 Linguistic and computational motivations

Most influential linguistic theories and ideas behind our parser are dependence grammar, semantic case grammar, and the notion of "word expert" parsing. The idea is that the c-graphs of word categories actively try to find the dependents of the main words and identify in what semantic roles they are (cf. the ADJ-ATTR-production of fig. 1.2). In some cases it it useful to assign active role to dependents. The c-graphs serve as illustrative linguistic descriptions of the syntactico-semantic features of word categories and other fenomena.

Computationally, our formalism and parsing scheme gives high expressive power but its time complexity is not high. Only potentially relevant productions are tried to use during parsing. Graphs are illustrative and can be used to express both procedural and declarative knowledge. New word category models can be added to the parser rather independently from the other models.

Our small experimental graph grammar parser for Finnish (Hyvönen 1983) is still liguistically quite naive containing some 150 lexical entries, 50 productions, and 50 c-graphs. A larger subset of Finnish needs to be modelled in order to evaluate the approach properly. We are currently developing the graph grammar approch further by generalizing the formalism into hierarchic graphs. By this way, for example, large graph structures could be manipulated more easily as single entities and identical structures could have different interpretations in different contexts. Also, a more elaborate coroutine based control structure for interpreting the c-graphs is under developement. We feel that the idea of seeing parsing as a multidimensional transformation of relational graphs in stead of as a delinearization process of a string into a parse tree is worth investicating further.

3 ACKNOWLEDGEMENTS

Thanks are due to Rauno Heinonen, Harri Jäppinen, Leo Ojala, Jouko Seppänen and the personnel of Digital Systems Laboratory for fruitful discussions. Finnish Academy, Finnish Cultural Foundation, Siemens Foundation, and Technical Foundation of Finland have supported our work financially.

4 REFERENCES

Bunke H. (1982): Attributed graph grammars and their application to schematic diagram interpretation. IEEE Trans. of pattern analysis and machine intelligence, No 6, pp. 574-582.

Hyvönen E. (1983): Graph grammar approach to natural language parsing and understanding. Proceedings of IJCAI-83, Karlsruhe.

Rajlich V. (1975): Dynamics of discrete structures and pattern reproduction. Journal of computer and system sciences, No 11, pp. 186-202.