Kimmo Kettunen

SITRA Foundation
P.O. Box 329
SF-00121 HELSINKI

ON MODELLING DEPENDENCY-ORIENTED PARSING

1. Introduction

Dependency grammars have not been in the mainstream of grammars whether we consider "traditional" grammars or grammars used in computational linguistics. In recent years, however, interest in dependency-oriented grammars has grown considerably. Text book writers and researchers even outside the dependency grammar tradition have laid more interest in dependency relations (cf. e.g. Lyons 1977, Matthews 1981, Miller 1985, Somers 1984).

Much work in dependency analysis seems to have been done in Slavonic and German languages and Japanese, which are all (highly) inflectional. Quite recently researchers of non-inflectional languages have also become interested in dependency-oriented models. It is probable that dependency analysis is suitable for both kinds of languages. Currently there is active work going on in some sort of automatic dependency analysis or formal modelling of dependency analysis at least in Czechoslovakia (e.g. Sgall, ed. 1984), Japan (Muraki & Shunji & Fukumochi 1985), BRD (Hellwig 1985), GDR (Kunze 1982), Great Britain (Hudson 1985, Fraser 1985), in the USA (Starosta 1985), Soviet Union (Urutyan & Simonyan 1983). EC´s machine translation project, EUROTRA, uses dependency relations in its sentence analysis (Johnson & King & des Tombe 1985).

This paper describes some basic linguistical characteristics of the parser, DADA, that has been implemented as a part of a database interface system for written Finnish queries (Nelimarkka et al 1983, 1984, Lehtola et al 1985). The parser is general and it is by now capable of analyzing a nontrivial subset of Finnish clauses. The basic idea of the parser is to provide analyzed sentences with syntactico-semantic structure. The structure that is given to an input clause is a functional case-labeled dependency structure. Dependency is stated and interpreted in functional labels which are then further interpreted using semantic roles. Therefore a superficial semantic representation is given to the analyzed sentence.

The following set lists salient features of our parser:

On modelling dependency-oriented parsing
Kimmo Kettunen
Proceedings of NODALIDA 1985, pages 113-120

1) Strength in grasping word-order variations of an inflectional language. This is due to the dependency grammar and to the implementation that employs two-way automata (cf. Levelt 1974).

2) Multidimensional analysis: full advantage of the rich inflectional morphology of Finnish is obtained. Rules of grammar are stated so that morphological knowledge as well as knowledge from higher strata may appear in them.

3) Parallel syntactic and case-semantic analysis or only syntactic analysis may be obtained as one wishes.

4) Semi-strict separation of linguistic knowledge and parsing mechanism. This is due to the high-level grammar description language, DPL, in which the grammar is written. The grammar and the parser in their present version have some 30 pages of DPL-description. That makes about 5500 lines of compiled elementary LISP-code.

5) The parser is strongly data-driven. Parsing proceeds bottom-up and is lexicon-based. Structures are built from words to larger constituents (cf. Winograd 1983).

6) All the time the parser has only a few rules that must be checked. The only hypothesis made are those, which come with expectations of the current stack. When rules are activated like this, the size of grammar will not affect the efficiency of the parser.

2. A sketch of the parsing process

Firstly we shall briefly sketch the overall parsing process. A morphological analysis precedes the parser. We have a morphological processor that analyzes the inflected words and gives the basic word forms and inflectional categories (cf. Jäppinen et al 1983). This is, of course, a prerequisite for parsing a highly inflected language, such as Finnish. The parser gets morphologically analyzed words with their lexical information. Lexical descriptions come from the parser´s own lexicon.

A disambiguator for ambiguous morphological output should also exist somewhere. One place for it could be after morphological analysis (cf. Karlsson 1985b) or the disambiguation could be done during the parse. So far we have none. For each morphologically ambiguous token of a word form the parser tries to find dependents. This leads to multiple work and possible misparses.

The DADA parser proceeds word-by-word in the input clause. The basic method of the parser is that it tries to recognize possible dependents for each token of a word category moving out from the nearest neighbour. As the parser is modelled with two-way automata, it can recognize subordinates both from the left and right context of the current input word. For each word category possible dependents are described in automaton networks.

We may generalize that during the parse local dependencies are sought first: each word category gathers dependents of its own type. When each non-verbal category has finished gathering its dependents and it is labeled as +Phrase (i.e. it governs > 0 dependents), it may be matched to the global structure of the clause. In non-elliptic sentences (sentences containing a finite verb) the parsing is always finished when some global dependent (dependent of the main verb) is attached and the working stack is empty.

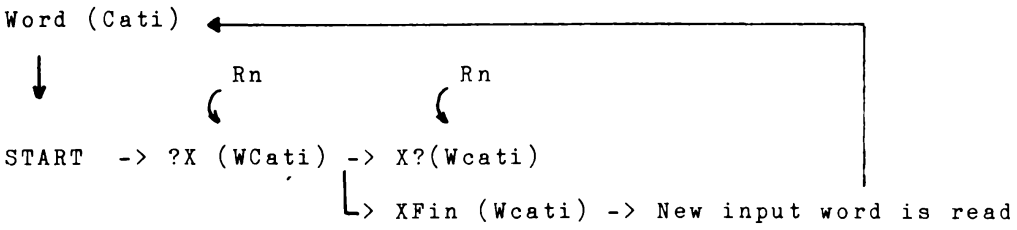

Fig.1: a simplified description of the flow of parsing

Here ?X and X? refer to left and right-hand states, respectively. START is the initial state that sends the analysis to the proper automaton network. Rn´s are dependency relations.

Dependency is stated in two steps: for each word-class a set of possible dependents is determined by function names in states. Possible orderings of dependent and regent are stated in left and right-side automaton descriptions.

Dependency relations are the only rules that are used. A dependency relation concerns a pair of words (C, D) which have to fullfill the requirements stated in the rule. Rules are written as conditions or constraints that have to hold between C and D. Schematically rules are stated as follows:

```
((C = < MorphC SyntCat ConstFeat SyntFeat SemCat SemFeat
                                  FrameCat FrameFeat >)

    ->

((D = < MorphC  SyntCat ConstFeat SyntFeat SemCat SemFeat
                                  FrameCat FrameFeat >)))
```

where

MorphC = inflectional knowledge
SyntCat = syntactic category
SyntFeat = a bundle of syntactic features
SemCat = name of the semantic class
SemFeat = a bundle of distinctive semantic features
ConstFeat = knowledge that is figured out during the parse and
is stated in binary features
FrameCat = frame category of the verb
FrameFeatc= frame features of the verb
C = regent candidate
D = dependent candidate

Features and categories can be written in disjunctive and conjunctive statements to form rules of grammar. When a match between C and D succeeds D is attached to C and labeled with the proper syntactic function label and (possibly) with semantic case role label.

Our kind of description is rather far from the Haysian classical dependency notation. Whereas Hays describes order and restrictions in the same formula (Hays 1964), we have different descriptions for each. Especially the word order restrictions are currently described rather clumsily. Possible word orders are blurred into paths of the automaton network. As a linguistic description this is not satisfying and a new way for describing the word order should be found. The second major problem is that lexical knowledge is stated in two places. At present automaton descriptions work as a kind of valency lexicons which give the possible dependents of each word-category. But it is obvious that this information should be given separately in the lexicon.

3. Assigned structures

The parser builds labeled dependency trees which have the following characteristics:

- the linear order of the surface clause is preserved in the trees

- heads and their modifiers are attached directly without any non-terminal symbols

- dependency trees have labels of two kinds: syntactic function labels and case role labels. Syntactic functions are the main objects that are created by the dependency relations. Case role labels are further interpretations of these functional relations.

4. Dependency

As we know two elements of a sentence are directly related in a dependency characterization if one DEPENDS on (conversely, is GOVERNED by) the other. The relationship is transitive, irreflexive and antisymmetric. A dependency relation is said to hold between two elements in certain circumstances. One member of this relation is called the GOVERNOR (or head or regent), the other the DEPENDENT (or modifier) (cf. e.g. Hudson 1980a, 1984, Mel´cuk 1979, Kunze 1975).

Two intuitive ideas determine the existence of a dependency relation between the governor and the dependent:

i) The governor expects to find certain types of dependents in its neighbourhood. This expectation is inherent to the element that constitutes the governor, and may be a piece of dictionary knowledge.

ii) The governor is felt to be modified by the dependent (not vice versa).

Johnson & King & des Tombe (1985) have discussed some basic problems concerning the dependency construction. It is useful to briefly state their points and consider our formalism in those respects.

The basic representational principles may be stated followingly (cf. also Robinson 1970):

i) There is one-to-one correspondence between leaves of the tree and primitive elements of the construction represented by the tree.

ii) There is one-to-one correspondence between nonterminal nodes of the tree and constructions of the text.

iii) Labellings on the nodes express the categories of primitive elements, constructions, and dependency relations.

As Johnson & King & des Tombe point out, this is elegant but empirically wrong and must be augmented. There are two classes of problems to the basic representational theory. Firstly it implies that no unit can be a member of more than one construction. Secondly it implies that every unit must be a member of at least one construction (except the text itself). But this is not the case. In a sentence like "John tried to swim" "John" is member of two constructions (Hudson calls this modifier-sharing) and this cannot be represented in a tree form. Accordingly in the sentence "Tom went to Paris and Hanna to London" "went" is a governor for two constructions (head-sharing). The Eurotra formalism has introduced a speacial notion of EMPTY ELEMENTS to handle these phenomena. These are shadow elements of their antecedents, i.e. the elements that participate in more than one construction. The empty elements in trees are leaves that do not correspond to anything at all in the text (the one-to-one correspondence is no longer valid).

There are some further problems. In some constructions there exist elements that are not dependent on anything in the clause. In "Frankly, I do not care a bit" "frankly" does not seem to be dependent on any word in the clause. For these situations a notion of TRANSCONSTRUCTIONALS is introduced in the Eurotra formalism. These are handled in a way that makes them as if they were dependents in the construction they are related to intuitively. A special label, pseudodependency, is attached to them.

Such problems are of course existent also in Finnish. Especially the problem of modifier-sharing is common in rather simple clauses already. Different kinds of infinitive constructions are typical examples of the phenomenon. Clauses such as "Poika haluaa analysoida kiviä" ("The boy wants to analyze stones") cannot be properly handled in our parser at present. Some new methods for handling these phenomena should be added either to the parser or at least in post parser analysis of sentences. At present the constructions of the parser are based only on the naively elegant one-to-one correspondence principle.

REFERENCES

Fraser, Norman 1985. A Word Grammar Parser. M.Sc. thesis, Department of Computer Science. University College, London.

Hays, David G. 1964. Dependency Theory: a Formalism and Some Observations. Language 40: 511 - 525.

---------- 1966. Parsing. In Hays, D.G. (ed.). Readings in Automatic Language Processing. American Elsevier Publishing Company, New York: 73 - 82.

Hellwig, Peter 1985. Program System PLAIN. "Programs for Language Analysis and Inference. Examples of Application." Computing Unit and Linguistics and International Studies Department. University of Surrey, Guildford.

Heringer, Hans Jürgen & Strecker, Bruno & Wimmer, Rainer 1980. Syntax. Fragen - Lösungen - Alternativen. Wilhelm Fink Verlag, München.

Hudson, Richard 1976. Arguments for a Non-transformational Grammar. The University of Chicago Press, Chicago.

---------- 1980a. Constituency and Dependency. Linguistics 18: 179 - 198.

---------- 1980b. Daughter-dependency Grammar. In Hans-Heinrich Lieb (ed.). Oberflächensyntax und Semantik. Linguistische Arbeiten 93. Max Niemeyer Verlag, Tübingen.

---------- 1983. Word Grammar. In Hattori, Shiro & Inoue, Kazuko 1983 (ed.). Proceedings of the XIIIth International Congress of Linguists, Tokyo: 89 - 101.

---------- 1984. Word Grammar. Basil Blackwell, Oxford.

---------- 1985. A Prolog Implementation of Word Grammar. Mimeo, October 85, University College, London.

Johnson, Rod & King, Maghi & des Tombe, Louis 1985. A Multilingual System under Development. Computational Linguistics 11: 155 - 169.

Jäppinen, Harri & Lehtola, Aarno & Nelimarkka, Esa & Ylilammi, Matti 1983. Morphological Analysis of Finnish: a Heuristic Approach. Helsinki University of Technology, Digital Systems Laboratory, report B26.

Karlsson, Fred 1985a (ed.) Computational Morphosyntax. Report on Research 1981 - 1984. Publications of the Department of General Linguistics. No. 13. University of Helsinki.

---------- 1985b. Parsing Finnish in Terms of Process Grammar. In Karlsson 1985a (ed.): 137 - 176.

Kunze, Jürgen 1975. Abhängigkeitsgrammatik. Studia Grammatica XII. Akademie-Verlag, Berlin.

---------- 1982a (Hrg.). Automatische Analyse des Deutschen. Akademie-Verlag, Berlin.

---------- 1982b. Einführung. In Kunze 1982a: 17 - 34.

Lehtola, Aarno & Jäppinen, Harri & Nelimarkka, Esa 1985. Language-based Environment for Natural Language Parsing. Proceedings of the Second Conference of the European Chapter of the Association for Computational Linguistics, USA: 98 - 106.

Levelt, W.J.M. 1974. Formal Grammars in Linguistics and Psycholinguistics. Volume II: Applications in Linguistic Theory. Mouton, The Hague.

Lyons, John 1977. Semantics 2. Cambridge University Press, Cambridge.

Matthews, P.M. 1981. Syntax. Cambridge University Press, Cambridge.

Mel´cuk, Igor 1979. Studies in Dependency Syntax. Linguistica Extranea, Studia 2. Karoma Publishers, Ann Arbor.
Miller, J. 1985. Semantics and Syntax: Parallels and Connections. Cambridge University Press, Cambridge.
Muraki, Kazunori & Ichiyama, Shunji & Fukumochi, Yasumoto 1985. Augmented Dependency Grammar: a Simple Interface between the Grammar Rule and the Knowledge. Proceedings of the Second Conference of the European Chapter of the Association for Computational Linguistics, USA: 198 - 204.
Nelimarkka, Esa & Lehtola, Aarno & Jäppinen, Harri 1984a. A Computational Model of Finnish Sentence Structure. In Anna Sågvall Hein (ed.). Föredrag vid De Nordiska Datalingvistik-dagarna 1983, Uppsala: 169 - 177.
---------- 1984b. Parsing an Inflectional Free Word Order Language with Two-way Finite Automata. In Shea, Tim O. (ed.). ECAI-84: Advances in Artificial Intelligence. Elsevier Science Publishers: 167 - 176.
Platek, M. & Sgall, J. 1984. A Dependency Base for Linguistic Description. In Sgall 1984 (ed.): 63 - 98.
Robinson, Jane J. 1970. Dependency Structures and Transformational Rules. Language 46: 259 - 285.
Sgall, Petr 1984 (ed.). Contributions to Functional Syntax, Semantics and Language Comprehension. Linguistic & Literary Studies in Eastern Europe vol. 16. John Benjamins, Amsterdam.
Somers, Harald 1984. On the Validity of the Complement-adjunct Distinction in Valency Grammar. Linguistics 22: 507 - 530.
Starosta, Stanley 1985. The End of Phrase Structure as We Know It. Series A, Paper no. 147. L.A.U.D.T., Linguistic Agency of Duisburg.
Urutyan, R.L. & Simonyan, S.L. 1983. Analysis of Equivalence in Language by Means of D-grammars. In Tiits, M. (ed.). Symposium on Grammars of Analysis and Synthesis and Their Representation in Computational Structures. Academy of Sciences of the Estonian S.S.R., Tallinn: 108 - 109.
Winograd, Terry 1983. Language as a Cognitive Process. Volume I, Syntax. Addison Wesley, Massachusetts.