LOCALLY GOVERNED TREES AND DEPENDENCY PARSING

Jäppinen, Harri, Lassila, Eero, and Lehtola,Aarno
KIELIKONE-project
SITRA Foundation
.P.O. Box 329, 00121 Helsinki, FINLAND

SUMMARY

This paper describes the notion of locally governed trees as a model of structurally restricted dependency structures of sentences. An abstract machine and its supporting software for the building of locally governed trees is introduced. The rest of the paper discusses how unambiguous, well-formed locally governed trees can be parsed in linear: time when certain structural constraints are in force,

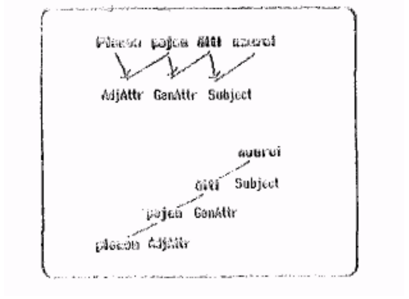
INTRODUCTION

The phrase structure rule is a widely used primitive notation in literature when syntactic structures of sentences are discussed in a rigorous manner. A majority of syntactic parsing programs also utilize phrase structure rules in one way or another.

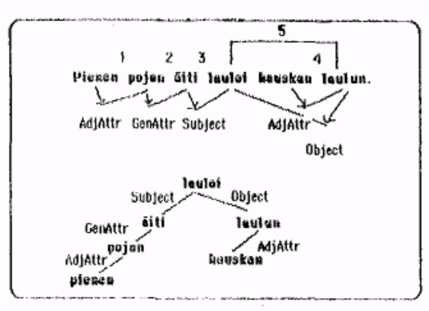
Phrase structure rules reflect the immediate constituent analysis of sentences. Each rule names a constituent and its specified ordered elements on the. lower level. Its primitive relations are therefore part-of-a-whole and concatenation. In parsing, phrase structure rules are used to search a hierarchical constituent organization of the word string of a sentence. Phrase structure rules discover the hierarchical organization of a sentence, but they do not tell which words are the heads of the phrases (save the X-bar theory /Jackendoff 1977/) nor do they further specify the types of the structural relations.

Dependency grammars, in contrast, indicate the binary relations that hold between the words in sentences /Hays 1964, (Gaifman 1965, Robinson 1970, Anderson 1971, Hellwig 1986, Starosta 1986/. Neither non-terminal symbols nor phrase structure rules have any role to play because constituents are not looked for. ft parser which employs dependency rules (rather than phrase structure rules) makes the heads and the types of binding relations explicit, but does not indicate the hierarchical constituent configurations of sentences explicitly. We argue that dependency grammars suit better than phrase structure rules to non-configurational, free-word-order languages.

Insofar as dependency relations are local (that is, they hold between adjacent words or trees) and destructive (that is, a recognized dependant is removed promptly from processing) deterministic parsing in linear time often results. Fig. la illustrates this point for a simple intransitive-verb Finnish sentence "Pienen pojan aiti nauroi." (A/the small boy's mother laughed).



a) A monotonous dependency tree



b) A non-monotonous dependency tree

Fig. 1. Parsing dependency trees

Linear time is preserved in parsing also in many typical Finnish non-monotonous dependency trees if, as a default control rule, a word attempts first to govern its left neighbor. This strategy is natural for Finnish as most modifiers are of prepositional type, This rule was already implicit in Fig. la. (There are of course exceptions which must overrule this default strategy. For example, prepositions have their dependants, not their regents, on their right side.)

Finnish sentences have typically SVO structure. Fig. 1b shows the parsing of an ordinary transitive-verb sentence "Pienen pojan äiti lauloi hauskan laulun" (A/the small boy's mother sang a merry song). Parsing steps are indicated by the numbers between the words.

This paper elaborates the locality principle in dependency parsing. First, we specify the ideas of local government and locally governed trees. Then we describe a basic machine and its supporting software as an implementation of the locality principle for parsing arbitrary locally governed trees. The parsing system has been implemented for Finnish. Occasionally our: parser invokes expensive search because no prerequisites restrict trees (save locality). We discuss how parsing can be speeded up into linear time if certain natural structural constraints are in force.

LOCAL GOVERNMENT, LOCALLY GOVERNED TREES, AND DEPENDENCY PARSING

The ideological underpinning of local dependency parsing is to focus on adjacent word pairs and see if a binary dependency relation holds between them. The words of a sentence have various attributes in our parser. Some of the attributes have been extracted by a morphological preprocessor /Jappinen and Ylilammi 1986/, while others are tagged during the parsing process.

Local Government

Let <w_1 w_2...w_n> be an ordered list of words. We say that a word w_i locally governs another word w_j iff j = i-1 or i+1 and w_i R w_j, where R is a binary dependency relation such that w_i is the governor (or the regent) of the pair and w_j is the dependant. In other words, a word locally governs another one if they are adjacent (at the moment of the testing) and a dependency relation holds between them.

The governor alone represents its government: once a local government has been established between two adjacent words the dependant is linked with the governor and disappears thereafter from sight. An elementary destructive processing step takes place, reducing the number of visible words by one (shown by arrows in Fig. 1).

Government is transitive. If w_i locally governs w_j, and w_j locally governs w_k, then w_i governs w_k. Government is also antisymmetric and irreflexive.

Locally Governed Trees

Due to the destructive processing step explained above, a governor gets a new neighbor immediately after a local government has been built. This new neighbor qualifies for a local government as well. A single word may therefore locally govern a number of other words, and two initially distant words may later on establish a local government between themselves. If a word is the governor of several words simultaneously, we say that it governs a locally governed tree of depth one (LGT-1). In fact, we can view a (binary) local government as a LGT-1 having just a single branch.

LGT-1's are elementary trees. Relational trees which preserve the locality principle and can reach arbitrary depth are called locally governed trees (LGT). LGT's are defined recursively as follows:

i. Any LGT-1 is a LGT.
ii. A tree formed by a word which locally governs LGT's is itself a LGT.

Let $<w_1\ w_2\ ...\ w_n>$ be a sequence of words. If there exists a LGT which governs all the words, the LGT is a parse tree of the words. Figure 1 portrays two parse trees.

Parsing Strategy

In the implemented parser the parsing strategy is based on the following two control principles: (1) parsing focuses first on the leftmost word (the initial word principle); (2) the parser always tries first to establish a focused word as a governor to its left neighbor and then shifts focus to the right neighbor (preferred direction principle).

The resulting parsing strategy is a left-corner-up strategy. The strategy is tuned to efficiently bind prepositional attributes as dependants.

THE MACHINE

We have designed and implemented a parsing system for LGT's. The underlying abstract machine has one focus register and two stacks which hold the left and the right contexts of the focused word, respectively /Nelimarkka et al. 1985/. Locality is enforced by permitting a focused word to bind dependants only from top of either stack - the left stack being preferred. The machine has also instructions for contextual testing. These tests may penetrate the stacks.

THE SOFTWARE

A high-level language FUNDPL (Functional Dependency Parsing Language) was designed for parsing locally governed trees /Jappinen et al. 1986b/. A precursor was a more procedural language DPL /Nelimarkka et al. 1985/. The system includes a compiler and supporting programming environment for the developmental work /Lehtola et al. 1985/.

A grammar description has three parts in FUNDPL. The initial part declares data types. The second part describes valid binary dependency relations. For each named binary relation the user specifies valid word pairs using morphological and/or lexical attribute values. The notation permits concise use of boolean operations on attributes.

The third part of a grammar description defines a set of functional schemata. Functional schemata have both declarative and procedural readings. From the declarative point of view, functional schemata define a set of valid LGT-1's. Each schema describes a regent and its possible local governments.

A local government is either mandatory or optional, and an optional one may recur. By default the surface ordering of local governments is free. Sometimes stringent ordering constraints exist between local governments; sometimes it is advantageous to give probabilistic information about the ordering of positionally free governments. Such structural information may be written in a schema.

Schemata have also procedural reading which is yet another distinguishing feature from phrase structure rules. A schema actively controls the build-up of the LGT-1 it represents. From the procedural viewpoint a schema monitors function calls of local governments using blackboard control regime /Valkonen et al. 1987/.

THE SEARCH PROBLEM OF PARSING ARBITRARY LGT'S

To discover a parse tree for an arbitrary LGT is *a* complicated search process even in a bottom-up strategy (in top-down problems would be worse). The basic problem is this: how does an algorithm know on which level in the hierarchy a given word belongs to? That is, when parsing proceeds from left to right and an attempt is made to establish the right neighbor of a governor as a dependant, the link is possible only if that word is not a governor of a yet incomplete LGT. Our left-corner-up strategy occasionally has to invoke complex search for this reason.

If a language constrains the structures of its possible LGT's, LGT's become computationally much more economical devices. The problem discussed above does not arise with constituent grammars and phrase structure rules because these rules indicate hierarchy implicitly through the naming of the constituents.

CONSTRAINED LGT'S

Finnish is a highly inflectional, agglutinating language. Both verbs and nominals have numerous distinct surface forms which distinguish between different syntactic functions the words can have in sentences. Word forms carry, among other things, such syntactic information which in configurational languages is indicated by the precedence relation. Word order in Finnish is relatively free.

The basic Finnish sentence configuration is SVO: a subject LGT is followed by a verb, an object LGT, and possible adverbial LGT's. Topicalization, wh-movement, and other movements create variations to this basic configuration.

The shape of nominal LGT's is markedly distorted. They have almost all modifiers on their left hand side forcing them to lean to the right. The most important modifiers are adjectival and genitive. Adjectival attributes modify the head noun iteratively, as in the phrase (1).

(1) Nuori pitkä viehättävä tyttö
Young tall charming girl

Genitive attributes, themselves nominals, modify head nominals recursively, as in the phrase (2).

(2) Tytön isän työnantajan auto
Girl (gen) father (gen) employer (gen) car
(A/the girl's father's employer's car)

Other prepositional modifiers for nouns are quantifiers and demonstrative pronouns. Prepositional modifier types can be mixed (under certain restrictions) as in the phrase (3).

(3) Tämän nuoren viehättävän tytön vanha kiero isä
This young charming girl (gen) old crooked father (nom)
(This young charming girl's old crooked father)

Prepositionality of Finnish is also demonstrated by the fact that postpositions are common but prepositions rare. Nouns have also occasional postpositional nominal modifiers, but these modifiers can be governed only by the maximal nominal heads of a LGT (the governors which fill the valencies of verbs) or by another postpositionally modifying LGT. For example, the nominal phrase (4)

(4) suuren miehen pieni auto talon takana
big man (gen) small car (nom) house (gen) behind
(a/the big man's small car behind the house)

has the LGT shown in Fig. 2. The postpositionally modifying adverbial LGT "talon takana" (behind the house) cannot modify the genitive attribute: *suuren miehen talon takana pieni auto.

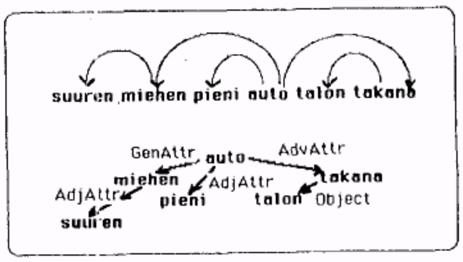


Fig. 2. Postpositional modifier.

AN EFFICIENT PARSING ALGORITHM FOR LGT'S

The basic left-corner-up algorithm can be modified so that it hierarchically first builds nominal LGT's without postpositional modifiers, then LGT's governed by prepositions and postpositions, then nominal LGT's with postpositional modifying nominal LGT's, and finally the LGT governed by the finite verb. The structural constraints of LGT's prune search, and it can be proved that the algorithm then parses unambiguous sentences in linear time. The following restrictions are assumed:

i. Adjectives, quantifiers, and adverbs have only prepositional modifiers,
ii. Nouns have postpositional modifiers only on the maximal level. On lower levels they have only prepositional modifiers.

AMBIGUITY AND WELL-FORMEDNESS

The modified algorithm presumes that LGT's are unambiguous. None of the bound dependants should not qualify as a dependant to any other governor than the one chosen. Because the algorithm removes dependants after binding, it cannot cope with alternative relations.

Albeit rich morphology greatly helps to make unique distinctions between different binary relations in Finnish, it leaves some residual ambiguity. The most prominent example is caused by the genitive surface case. That case signals either accusative case, the object of a sentence, or possession. The governor of an adverbial may also be ambiguous. The basic algorithm solves ambiguity by backtracking.

In their "pure" form both algorithms parse only well-formed LGT's. There are, however, some well known syntactic phenomena which cannot be represented by LGT's. TG-theory postulates certain transformations which result in long-distance dependencies. In modern SB-theory these displacement operations go under the general rubric "move-alpha".

For example, certain fronting movements (wh-movement and topicalization) remove an element and may transport it across clause boundaries onto a landing site in the beginning of the main sentence. A LGT which originally was governed locally becomes distant to its governor and is no more within its reach.

The algorithm can be augmented to handle long-distance fronting movements. At one point the algorithm has built nominal and adverbial LGT's. The valencies for a verb are filled first locally and, if a filler cannot be found, a search is made from the fronted LGT's. The resulting LGT is not well-formed.

CONCLUSION

We have introduced the notions of local government and locally governed trees to express restricted dependency structures. Their computational counterparts are binary dependency relations (for local government) and functional schemata (for locally governed trees of depth one). We then briefly mentioned a parsing system which we have implemented for parsing dependency structures of Finnish sentences. We then discussed how the algorithm can be augmented into a multilevel model which takes into account varying structural constraints in different levels of sentence hierarchies. Linear parsing time ensues for unambiguous well-formed locally governed trees.

REFERENCES

Anderson, J.M., Dependency and grammatical functions. Foundations of Language 7, 1971, 30-37.
Gaifman, H., Dependency systems and phrase-structure systems. I Information and control 8, 1965, 304-337.
Hays, D., Dependency theory: a formalism and some observations. language 40, 1964, 511-525.
Hellwig, P., Dependency unification grammar. COLING'86, Bonn 1986.
Hudson, R., Arguments for a Non-transfornational Grammar. The University of Chicago Press, 1976.
Jackendoff, R., X-bar Syntax: A Study of Phrase Structure. Linguistic Inquiry, Monograph Series Two, The MIT Press, 1977.
Jäppinen, H. and Ylilammi, M., Associative model of morphological analysis: an empirical inquiry. Computational Linguistics, Vol. 12, No. 4, 1986a.
Jäppinen, H., Lehtola, A., and Valkonen, K., Functional structures for parsing dependency constraints. Proc. COLING86, Bonn, 1986b.
Lehtola, A., Jäppinen, H., and Nelimarkka, E., Language-based environment for natural language parsing. Proc. 2nd EUROACL, Geneve, 1985.
Nelimarkka, E., Jäppinen, H., and Lehtola, A., Parsing an inflectional free word order language with two-way finite automata. In O'Shea, T. (Ed.), Advances in Artificial Intelligence. North-Holland, 1985.
Robinson, J., Dependency structure and transformational rules. Language, Vol. 46, No. 2, 1970.
Starosta, S. and Nomura, H., Lexicase parsing: a lexicon-driven approach to syntactic analysis. COLING'86, Bonn, 1986.
Valkonen, K., Jäppinen, H., and Lehtola, A., Blackboard-based dependency parsing. IJCAI87, Milan, 1987.