BLACKBOARD-BASED DEPENDENCY PARSING

Valkonen, K., Jäppinen, H. and Lehtola, A.
KIELIKONE-project, SITRA Foundation
P.O.Box 329, SF-00121 Helsinki
Finland

ABSTRACT

This paper presents • blackboard-based computational modal for parting an Inflectional free word order language, like Finnish. The structure of sentences is described as partial dependency trees of depth one. Parsing becomes a nondetermlnistic search problem in the forest of partial parse trees. The search process is able to solve ambiguities and long-distance dependencies as well. Parsing is controlled by a blackboard system. A working parser for Finnish has been implemented based on the model.

1 INTRODUCTION

In our first approach, the parsing process is described as a sequence of local decisions (Nelimarkka et al. 1984). A pair of adjacent structures of an input sentence is connected if a valid binary dependency relation exists between them. In that first version of the parser dependency structures were modelled procedurally with finite two-way automata (Lehtola et al. 1985). Recently, we have developed a constraint-system formalism for dependency parsing (Jappinen et al. 1986). We also have augmented the model to cover long-distance dependencies. According to the augmented model a blackboard-based dependency parser AOP (Augmented Dependency Parser) has been implemented (Valkonen et al. 1987). In this paper we focus on the blackboard-based computational method.

In our model binary dependency relations specify constraints on argument structures. In functional schemata the structure of sentences is described as local dependent environments of regents. The goal is to find a matching local environment description for each word of an input sentence. As a side effect of the recognition corresponding partial dependency trees are built. The partial dependency trees are linked into a parse tree covering the whole sentence (Figure 1).

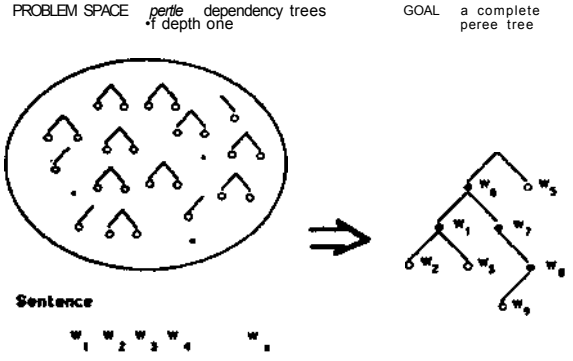

Figure 1. Parsing as a search process in a forest of partial dependency trees.

2 BLACKBOARD MODEL FOR DEPENDENCY PARSING

Blackboard is a popular problem-solving model for expert systems (Nii 1986). We have adopted that concept and utilized it for parsing purposes. Our blackboard model application is rather simple (Figure 2).

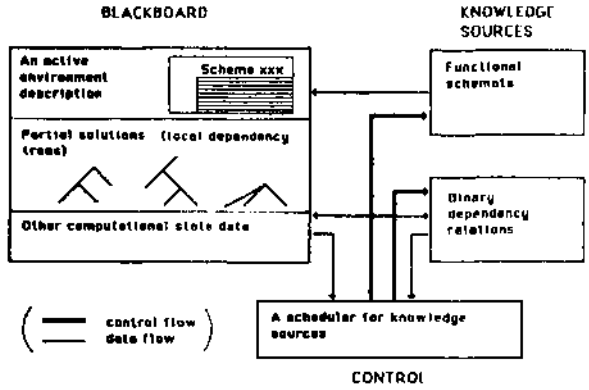

Figure 2. A blackboard model for dependency parsing.

There are three main components: a blackboard, a scheduler and knowledge sources. The blackboard contains the active environment description for regents. According to the structural knowledge in that environment description corresponding partial parse trees are built in the blackboard. Also other changes in the state of computation are marked in the blackboard.

Functional schemata and binary dependency relations are independent and separate knowledge sources; no communication happens between them. All data flow takes place through the blackboard. The module of knowledge to be applied is determined dynamically, one step at a time, resulting in the incremental generation of partial aolutions.

Binary relations are boolean expressions of the morphological and syntactic restrictions defining all the permitted dependency relations between two words in a sentence. In functional schemata a grammar writer has described local environments for regents using dependency relations. Only one of the schemata at a time is chosen as an active environment description for the current regent. The activated schema is matched with the environment of the regent by binary relation testa. Simultaneously a partial dependency tree Is built by corresponding dependency function applications. When a schema has been fully matched and the active regent bound to its dependents through function links, thee local partial dependency parse tree is complete.

A scheduler for knowledge sources controls the whole system. It monitors the changes on the blackboard and decides what actions to take next. The scheduler employs a finite two-way automaton for recognition of the dependents.

2.1 The control strategy for dependency parsing

For the formal definition of the parsing process we describe the input sentence as a sequence (c(1),c(2),...,c(i-1), c(i), c(i+1),...,c(n)) of word constituents. With each constituent c(i) a set (s(i,1),...,s(i,m)) of functional schemata is associated. The general parsing strategy for each word constituent c(i) can be modelled using a transition network. During parsing there are five possible computational states for each constituent c(i):

S1 The initial state. One of the schemata associated with c(i) is activated.

S2 Left dependents are searched for c(i).

S3 c(i) is waiting for the building of the right context.

S4 Right dependents are searched for c(i).

S5 The final state. The schema associated with c(i) has been fully matched and becomes inactive. c(i) is the head of the completed (partial) dependency tree.

At any time, only one schema is active, i.e. only one constituent c(i) may be in state S2 or S4. Only a completed constituent (one in state S5) is allowed to be bound as a dependent for a regent. There may be a number of constituents simultaneously in state S3. We call these pending constituents (implemented as a stack PENDING).

Binding is stated as mapping f(c(i),c(j)) -> c(i)' where c(i)' stands for the regent c(i) after it has bound the dependent c(j). Function f is defined by the corresponding binary relation.

The parsing process starts with c(1) and proceeds to the right. Initially all constituents c(1),..,c(n) are in the state S1. A sentence is well formed if in the end of the parsing process the result is a single constituent that has reached the state S5 and contains all the other constituents bound in its dependency tree. For each constituent c(i) the parsing process can be described by the following five steps. Parsing begins from step 1 with i,k = 1.

1) A schema candidate s(i,k) associated with c(i) is activated, i.e. the constituent c(i) takes the role of a regent. Following the environment description in s(i,k), dependents for c(i) are searched from its immediate neighbourhood. Go to step 2 with j = i-1.

2) The search of left dependents. There are two subcases:

2a) There are no left neighbours (j = 0), none is expected for c(i), or c(j) (j < i) exists and is in state S3. Go to step 3 with j = j+1.

2b) c(j) (j < i) exists and is in state S5. Binary relation tests are done. In case of a success the mapping f(c(i), c(j)) -> c(i)' takes place. Repeat step 2 with j = j-1 and c(i) = c(i)'.

3) Building the right context of the regent. There are two subcases:

3a) There are no right neighbours (j > n) or none is expected for c(i). Go to step 5.

3b) c(j) (j > i) exists. Go to step 1 with c(i) = c(i+1) and PENDING = push (c(i), PENDING).

4) The search of right dependents. Binary relation tests are done. In case of success the mapping f(c(i), c(j)) -> c(i)' takes place. Repeat step 3 with j = j+1 and c(i) = c(i)'.

5) The final state. There are two subcases:

5a) The environment description has been matched. If no unbound c(j)'s (j < i or j > i) remain the sentence is parsed. If c(i+1) exists go to the step 1 with i = i+1. If c(i+1) doesn't exist or the steps following the previous case returned a failure, go to step 4 with c(i) = pop (PENDING).

5b) The environment description has not been matched. If another schema for c(i) exists (k < m), go to step 1 with k = k+1. Otherwise return a failure.

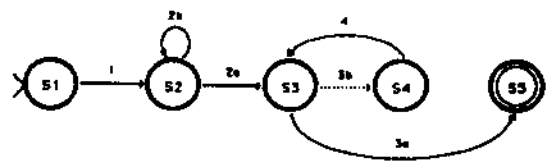

Figure 3. The transition network model of the control strategy.

2.2 The implementation of the control strategy

The control system has two levels: the basic level employs a general two-way automaton and the upper level uses a blackboard system. There is a clear correspondence between the grammar description and the control system: the two-way automaton makes local decisions according to the binary relations. These local decisions are controlled by the blackboard system which utilizes the environment descriptions written in the schemata.

To account for ambiguities there are three kinds of backtracking points in the control system. Backtracking may be done in regard to the choice of dependency functions, homographic word forms, or associated schemata (only the last case was expressed in section 2.1). Backtracking is chronological.

The solution of long-distance dependencies takes place in two phases. First an element which may have moved from the domain of its original regent to the domain of another one is recognized (by special CAPTURE functions). Such elements are assigned to a special list of distant elements. Then, in the binding phase, the original regent binds a distant element from the list. For more details, see Valkonen et al. (1987).

The strategy of local decisions controlled by global knowledge of the input sentence yields a strongly data-driven, left-to-right and bottom-up parse whereby partial dependency trees are built proceeding from middle to out.

3 PARSING EXAMPLE

To visualize our discussion, a full trace of parsing the sentence "Älä eksy metsässä" (Don't get lost in a forest) appears in Figure 4. Parsing starts from the left (an arrow). Next line indicates the selected schema and dependents that are tested. The first word "Älä" is identified as a negative imperative verb with no dependents (schema DummyVP ok). The imperative verb "eksy" (to get lost) is then tried by the schema N-ImperIntrVP. The binary relation Negation holds between the two verbs, and the corresponding dependency function adjoins them. The other functions fail. Dependents are searched next from the right context. The control proceeds to the word "metsässä" (forest). For that word no dependents are found and the system returns to the unfinished regent "eksy". The schema N-ImperIntrVP has only two relations remaining: Subject and Adverbial. The word "metsässä" is bound as an adverbial. The schema has been fully matched and the input sentence is completely parsed.

```
> älä eksy metsässä.

=>  (älä)  (eksy)  (metsässä)
Schema: DummyVP ()
DummyVP ok
(älä)  =>  (eksy)  (metsässä)
Schema: N-ImperIntrVP (Negation Subject Adverbial)
Negation ok
Subject failed
Adverbial failed
((älä) eksy)  =>  (metsässä)
Schema: TrivialSP (DefPart R)
DefPart failed
TrivialSP ok
  returning to unfinished constituent...
((älä) eksy)  <=  (metsässä)
Schema: N-ImperIntrVP (Subject Adverbial)
Subject failed
Adverbial ok
N-ImperIntrVP ok
=>  ((älä) eksy (metsässä)) PARSED

The parse took 0.87 seconds CPU-time on VAX-11/751.
```

Figure 4. An example of parsing.

4 COMPARISON

The notion of unification has recently emerged as a common descriptive device in many linguistic theories like FUG, PATR-II and HPSG (Kay 1985, Shieber 1986). Another popular approach has been to apply attribute grammars originally developed as a theory for formal languages (Knuth 1968). LFG and DCG can be viewed as attribute grammar systems. The trend has been towards strictly declarative descriptions of syntactic structure. Syntactic rules are often expressed in the form of complex feature sets.

Our ADP system also uses features, but there is neither unification nor correspondence to attribute grammars. Where FUG and the others use unification, there ADP uses a pattern matching via binary relation tests for local decisions. After binding the regent solely represents the constituents hanging below (however, in some cases certain features must be raised). Functional schemata are independent, local dependency environment descriptions of regents. Through blackboard approach we have gained a more flexible control: the blackboard system can conveniently take into account global knowledge of the sentence.

5 CONCLUSIONS

According to declarative word environment descriptions in schemata partial solutions are built in the blackboard. Local decisions controlled by global knowledge of the input sentence has made it possible to find solutions for problems that are difficult to solve in traditional parsing systems. ADP finds all solutions for an ambiguous sentence. An augmented search process covers long-distance dependencies as well.

ADP has been implemented in Franz Lisp. Experiments with a non-trivial set of Finnish sentence structures has been performed on a VAX 11/751 system. An average time for parsing a six word sentence is less than 2.0 seconds for the first parse. At the moment the grammar description covers common sentence structures quite well. There are 66 binary relations, 188 functional schemata and 1800 lexicon entries. The lexicon of the morphological analyzer (Jäppinen and Ylilammi 1986) contains 35 000 word entries.

We argue that our blackboard-based computational model also gives a good basis for parallel parsing. There should be an own processor for each word of the input sentence. The partial dependency trees would be built parallel and sent to the main process that links them into a parse tree covering the whole sentence.

ACKNOWLEDGEMENTS

This research has been supported by SITRA Foundation. We also would like to thank Matti Ylilammi, who has suggested many improvements to our parsing model.

REFERENCES

Jäppinen, H., Lehtola, A. and Valkonen, K. "Functional Structures for Parsing Dependency Constraints." in Proc. COLING86/ACL. Bonn, 1986, pp. 461-463.

Jäppinen, H. and Ylilammi, M. "Associative Model of Morphological Analysis: an Empirical Inquiry." Computational Linguistics 12:4 (1986) 257-272.

Kay, M. "Parsing in functional unification grammar." in Dowty, Karttunen and Zwicky (Eds.), Natural Language Parsing. Cambridge University Press, 1985.

Knuth, D. "Semantics of Context-Free Languages." Mathematical Systems Theory 2(1968a), pp. 127-145.

Lehtola, A., Jäppinen, H. and Nelimarkka, E. "Language-based Environment for Natural Language Parsing." in Proc. 2nd European Conf. of ACL. Geneve, 1985, pp. 98-106.

Nelimarkka, E., Jäppinen, H., and Lehtola, A. "Parsing an inflectional free word order language with two-way finite automata." in Proc. 6th European Conf. on Artificial Intelligence. Pisa, 1984, pp. 167-176.

Nii, H. "Blackboard Systems: The Blackboard Model of Problem Solving and the Evolution of Blackboard Architectures." AI Magazine 7:2 (1986), 38-53, 7:3 (1986), 82-106.

Shieber, S. An Introduction to Unification-Based Approaches to Grammar. CSLI Lecture Notes Series, No. 4, 1986.

Valkonen, K., Jäppinen, H., Lehtola, A. and Ylilammi, M. "Declarative Model for Dependency Parsing - A View into Blackboard Methodology." in Proc. 3rd European Conf. of ACL. Copenhagen, 1987, 8 p. (in print).