A COMPUTATIONAL MODEL OF LANGUAGE PERFORMANCE: DATA ORIENTED PARSING

RENS BOD*

Department of Computational Linguistics
University of Amsterdam
Spuistraat 134
1012 VB Amsterdam
The Netherlands
RENS@ALF.LET.UVA.NL

Abstract

Data Oriented Parsing (DOP) is a model where no abstract rules, but language experiences in the form of an analyzed corpus, constitute the basis for language processing. Analyzing a new input means that the system attempts to find the most probable way to reconstruct the input out of fragments that already exist in the corpus. Disambiguation occurs as a side-effect. DOP can be implemented by using conventional parsing strategies. Monte Carlo techniques are used to parse efficiently.

Introduction

This paper formalizes the model for natural language introduced in [Scha 1990]. Since that article is written in Dutch, we will translate some parts of it more or less literally in this introduction. According to Scha, the current tradition of language processing systems is based on linguistically motivated competence models of natural languages. The problems that these systems run into, suggest the necessity of a more performance oriented model of language processing, that takes into account the statistical properties of real language use. Therefore Scha proposes a system that makes use of an annotated corpus. Analyzing a new input means that the system attempts to find the most probable way to reconstruct the input out of fragments that already exist in the corpus.

* The author wishes to thank his colleagues at the Department of Computational Linguistics of the University of Amsterdam for many fruitful discussions, and, in particular, Remko Scha, Martin van den Berg, Kwee Tjoe Liong and Frederik Somsen for valuable comments on earlier versions of this paper.

The problems with competence grammars that are mentioned in Scha's article, include the explosion of ambiguities, the fact that human judgements on grammaticality are not stable, that competence grammars do not account for language change, and that no existing rule-based grammar gives a descriptively adequate characterization of an actual language. According to Scha, the development of a formal grammar for natural language gets more difficult as the grammar gets larger. When the number of phenomena one has already taken into account gets larger, the number of interactions that must be considered when one tries to introduce an account of a new phenomenon grows accordingly.

As to the problem of ambiguity, it has turned out that as soon as a formal grammar characterizes a non-trivial part of a natural language, almost every input sentence of reasonable length gets an unmanageably large number of different structural analyses (and semantical interpretations).[1] This is problematic since most of these interpretations are not perceived as possible by a human language user, while there are no systematic reasons to exclude them on syntactic or semantic grounds. Often it is just a matter of *relative* implausibility: the only reason why a certain interpretation of a sentence is not perceived, is that another interpretation is much more plausible.

Competence and Performance

The limitations of the current language processing systems are not suprising: they are the direct consequence of the fact that these systems implement Chomsky's notion of a competence grammar. The formal grammars that constitute the subject-matter of theoretical linguistics, aim at characterizing the competence of the language user. But the preferences language users have in the case of ambiguous sentences, are paradigm instances of performance phenomena.

In order to build effective language processing systems we must implement performance-grammars, rather than competence grammars. These performance grammars should not only contain information on the structural possibilities of the general language system, but also on details of actual language use in a language community, and of the language experiences of an individual, which cause this individual to have certain expectations on what kinds of utterances are going to occur, and what structures and interpretations these utterances are going to have.

There is an alternative linguistic tradition that has always focused on the concrete details of actual language use: the statistical tradition. In

[1] In [Martin 1979] it is reported that their parser generated 455 different parses for the sentence "List the sales of products produced in 1973 with the products produced in 1972".

this approach, syntactic structure is usually ignored; only 'superficial' statistical properties of a large corpus are described: the probability that a certain word is followed by a certain other word, the probability that a certain sequence of two words is followed by a certain word, etc. (Markov-chains, see e.g. [Bahl 1983]). This approach has performed succesfully in certain practical tasks, such as selecting the most probable sentence from the outputs of a speech recognition component. It will be clear that this approach is not suitable for many other tasks, because no notion of syntactic structure is used. And there are statistical dependencies within the sentences of a corpus, that can extend over an arbitrarily long sequence of words; this is ignored by the Markov-approach. The challenge is now to develop a theory of language processing that does justice to the statistical as well as to the structural aspects of language.

The Synthesis of Syntax and Statistics

The idea that a synthesis between syntactic and statistical approaches could be useful has incidentally been proposed before, but has not been worked out very well so far. The only technical elaboration of this idea that exists at the moment, the notion of a probabilistic grammar, is of a rather simplistic nature. A probabilistic grammar is simply a juxtaposition of the most fundamental syntactic notion and the most fundamental statistical notion: it is an "old-fashioned" context free grammar, that describes syntactic structures by means of a set of abstract rewrite rules that are now provided with probabilities that correspond to the application-probabilities of the rules (see e.g. [Jelinek 1990]).

As long as a probabilistic grammar only assigns probabilities to individual rewrite rules, the grammar cannot account for all statistical properties of a language corpus. It is, for instance, not possible to indicate how the probability of syntactic structures or lexical items depends on their syntactic/lexical context. As a consequence of this, it is not possible to recognize frequent phrases and figures of speech as such - a disappointing property, for one would prefer that such phrases and figures of speech would get a high priority in the ranking of the possible syntactic analyses of a sentence. Some improvements can be made by applying the Markov-approach to rewrite rules, as is for instance found in the work of [Magerman 1991]. Nevertheless, any approach which ties probabilities to rewrite rules will never be able to accommodate all statistical dependencies. Optimal statistical estimations can only be achieved if the statistics are applied to different kinds of units than rewrite rules. It is interesting to note that also in the field of theoretical linguistics the necessity to use other kinds of structural units has been put forward. The clearest articulation of this idea is found in the work of [Fillmore 1988].

From a linguistic point of view that emphasizes the syntactic complexities caused by idiomatic and semi-idiomatic expressions, Fillmore et al. arrive at the proposal to describe language not by means of a set of rewrite rules, but by means of a set of *constructions*. A construction is a tree-structure: a fragment of a constituent-structure that can comprise more than one level. This tree is labeled with syntactic, semantic and pragmatic categories and feature-values. Lexical items can be specified as part of a construction. Constructions can be idiomatic in nature: the meaning of a larger constituent can be specified without being constructed from the meanings of its sub-constituents.

Fillmore's ideas still show the influence of the tradition of formal grammars: the constructions are schemata, and the combinatorics of putting the constructions together looks very much like a context free grammar. But the way in which Fillmore generalizes the notion of grammar resolves the problems we found in the current statistical grammars: if a construction-grammar is combined with statistical notions it is perhaps possible to represent all statistical information. This is one of the central ideas behind our approach.

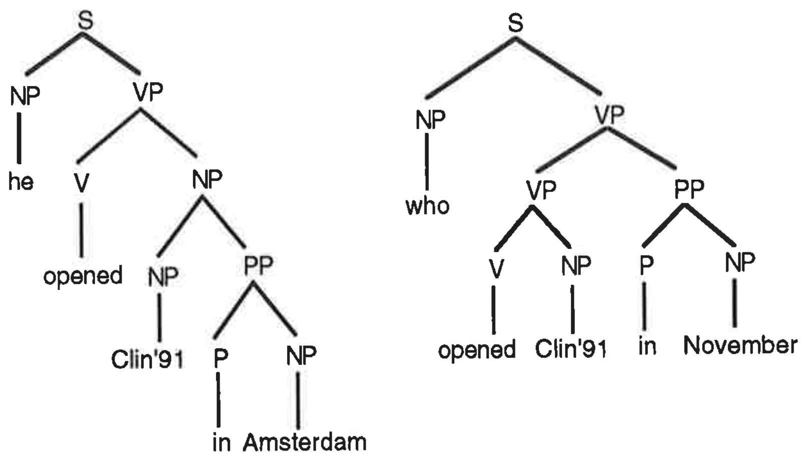
A New Approach: Data Oriented Parsing

The starting-point of our approach is the idea indicated above, that when a human language user analyzes sentences, there is a strong preference for the recognition of sentences, constituents and patterns that occurred before in the experience of the language user. There is a statistical component in language processing that prefers more frequent structures and interpretations to less frequently perceived alternatives.

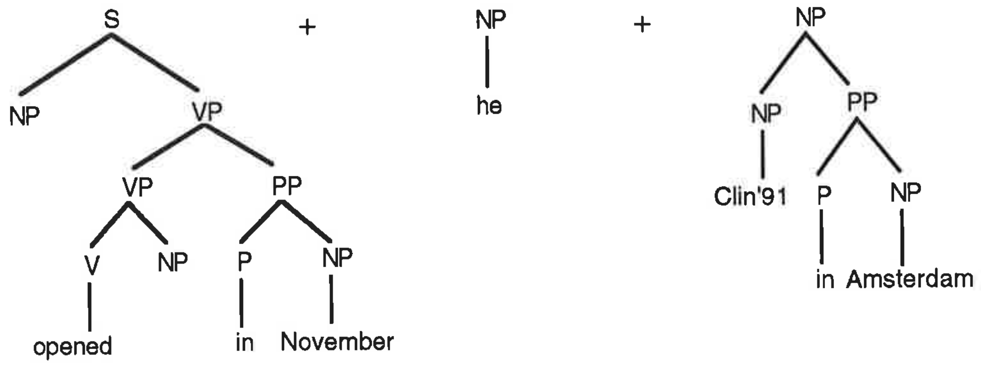
The information we ideally would like to use in order to model the language performance of a natural language user, comprises therefore an enumeration of all lexical items and syntactic/semantic structures ever experienced by the language user, with their frequency of occurrence. In practice this means: a very large corpus of sentences with their syntactic analyses and semantic interpretations. Every sentence comprises a large number of constructions: not only the whole sentence and all its constituents, but also the patterns that can be abstracted from the analyzed sentence by introducing 'free variables' for lexical elements or complex constituents.

Parsing then does not happen by applying grammatical rules to the input sentence, but by constructing an optimal analogy between the input sentence and as many corpus sentences as possible. Sometimes the system shall need to abstract away from most of the properties of the trees in the corpus, and sometimes a part of the input is found literally in the corpus, and can be treated as one unit in the parsing process. Thus the system tries to combine constructions from the corpus so as to reconstruct the input sentence. The preferred parse is obtained by maximizing the probability that arbitrary combinations of constructions generate a parse of the input sentence. Finally, the preferred parse is added to the corpus, bringing it into a new 'state'.

To illustrate the basic idea, consider the following extremely simple example. Assume that the whole corpus consists of only the following two trees:



Then the input sentence *he opened Clin'91 in Amsterdam in November* can be analyzed as an *S* by combining the following constructions from the corpus:



The Model

In order to come to formal definitions of *parse* and *preferred parse* we first specify some basic notions.

Labels

We distinguish between the set of *lexical labels L* and the set of *non-lexical labels N.* Lexical labels represent words. Non-lexical labels represent syntactic and/or semantic and/or pragmatic information, depending on the kind of corpus being used. We write $\mathfrak{L}$ for $L \cup N$.

String

Given a set of *labels* $\mathfrak{L}$, a *string* is an n-tuple of elements of $\mathfrak{L}$: $\langle l_1,...,l_n \rangle \in \mathfrak{L}^n$. An *input string* is an n-tuple of elements of L: $\langle l_1,...,l_n \rangle \in L^n$. A *concatenation* $*$ can be defined on strings as usual: $\langle a,...,b \rangle * \langle c,...,d \rangle = \langle a,...,b,c,...,d \rangle$.

Tree

Given a set of labels $\mathfrak{L}$, the set of *trees* is defined as the smallest set *Tree* such that

if $l \in \mathfrak{L}$, *then* $\langle l, \langle\rangle \rangle \in Tree$

if $l \in \mathfrak{L}$, $t_1,...,t_n \in Tree$, *then* $\langle l, \langle t_1,...,t_n \rangle \rangle \in Tree$

For a set of trees *Tree* over a set of labels $\mathfrak{L}$, we define a function *root*: $Tree \rightarrow \mathfrak{L}$ and a function *leaves*: $Tree \rightarrow \mathfrak{L}^n$ by

for $n \geq 0$, $root(\langle l, \langle t_1,...,t_n \rangle \rangle) = l$

for $n > 0$, $leaves(\langle l, \langle t_1,...,t_n \rangle \rangle) = leaves(t_1) * ... * leaves(t_n)$

for $n = 0$, $leaves(\langle l, \langle\rangle \rangle) = \langle l \rangle$

Corpus

A *corpus C* is a multiset of trees, in the sense that any tree can occur zero, one or more times. Ideally, a corpus represents all past language experiences of a natural language user. The *leaves* of every tree in a corpus is an element of L^n: it constitutes the string of words of which that tree is the analysis that seemed most appropriate for understanding the string in the context in which it was uttered.

Constructions

In order to define the *Constructions* of a tree, we need two additional notions: *Subtrees* and *Patterns*,

$$Subtrees(\langle l,\langle t_1,\ldots,t_n\rangle\rangle) \;=\; \{\langle l,\langle t_1,\ldots,t_n\rangle\rangle\} \;\cup\; (\bigcup_{i=1}^{n} Subtrees(t_i))$$

$$Patterns(\langle l,\langle t_1,\ldots,t_n\rangle\rangle) \;=$$

$$\{\langle l,\langle\rangle\rangle\} \;\cup\; \{\langle l,\langle u_1,\ldots,u_n\rangle\rangle \;/\; \forall i\in[1,n]:\, u_i\in Patterns(t_i)\}$$

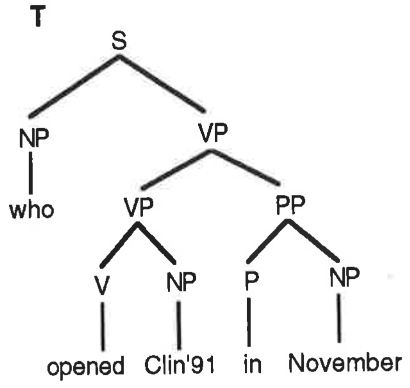
$$Constructions(T) \;=\; \{t \;/\; \exists u\in Subtrees(T):\, t\in Patterns(u)\}$$

We shall use the following notation for a construction of a tree in a corpus:

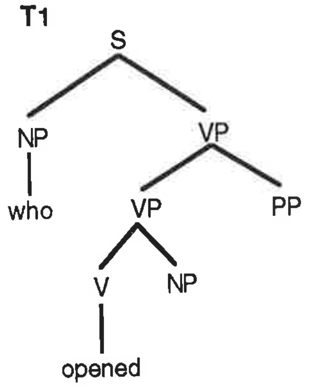
$$t\varepsilon C \;=_{def}\; \exists u\in C:\, t\in Constructions(u)$$

Example: consider tree T. The trees T_1 and T_2 are constructions of T, while T_3 is not.

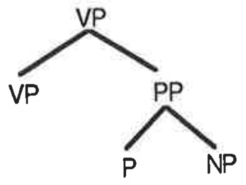
T



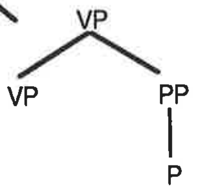
T1



T2



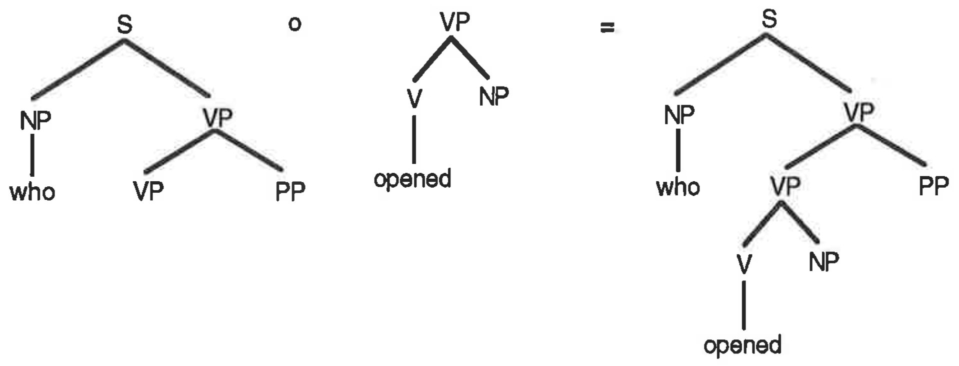
~~T3~~



Composition

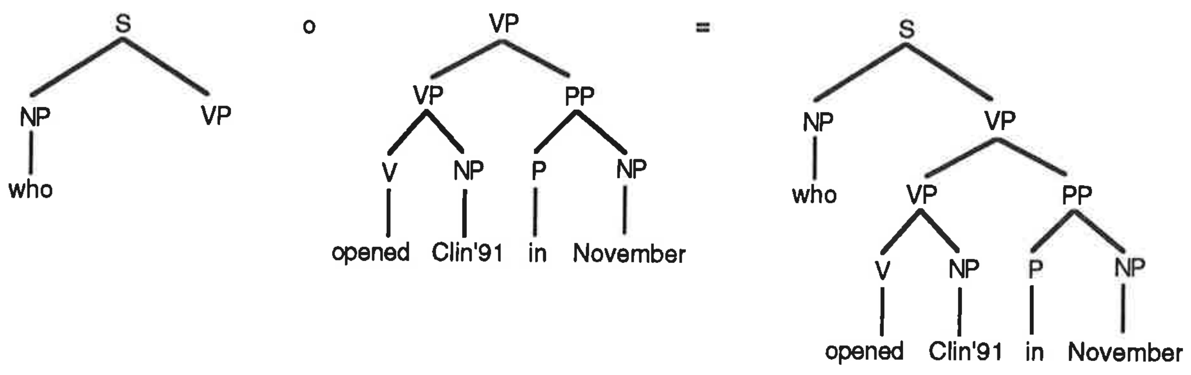
If t and u are trees, such that the *leftmost non-lexical leaf* of t is equal to the *root* of u, then $t \circ u$ is the tree that results from substituting this leaf in t by tree u. The partial function $\circ: Tree \times Tree \rightarrow Tree$ is called *composition.* We will write $(t \circ u) \circ v$ as $t \circ u \circ v$, and in general $(..((t_1 \circ t_2) \circ t_3) \circ ..) \circ t_n$ as $t_1 \circ t_2 \circ t_3 \circ ... \circ t_n$.

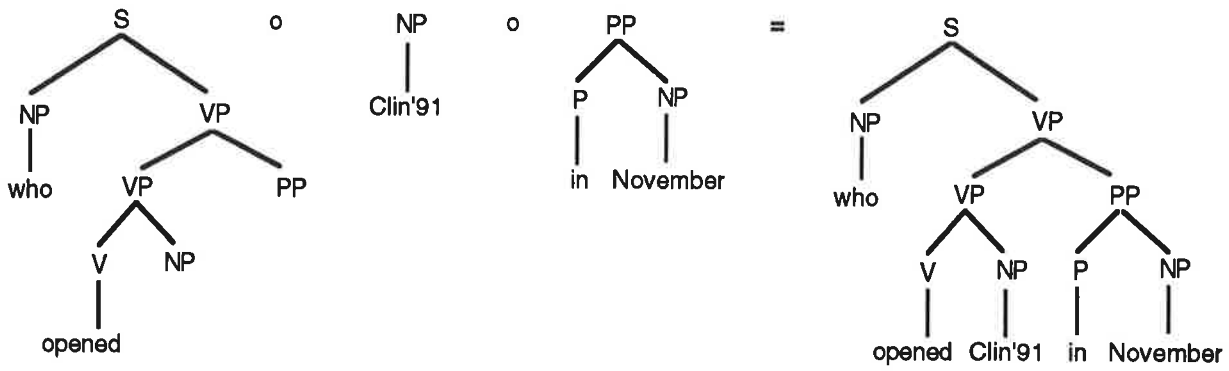
Example:



Parse

Tree T is a parse of input string s with respect to C, iff $leaves(T) = s$ and there are constructions $t_1,...,t_n \varepsilon C$, such that $T = t_1 \circ ... \circ t_n$. A tuple $\langle t_1,...,t_n \rangle$ of such constructions is said to *generate* parse T of s. Note that different tuples of constructions can generate the same parse, as is shown in the next example:





Etcetera...

The set of parses of *s* with respect to *C*, *Parse(s,C)*, is given by

$$Parse(s,C) \;=\; \{T \in Tree \;/\; leaves(T)=s \wedge \exists t_1,\ldots,t_n \,\varepsilon\, C\!: T=t_1\circ\ldots\circ t_n\}$$

The set of tuples of constructions that generate a parse *T*, *Tuples(T,C)*, is given by

$$Tuples(T,C) \;=\; \{\langle t_1,\ldots,t_n\rangle \;/\; t_1,\ldots,t_n \,\varepsilon\, C \wedge t_1\circ\ldots\circ t_n=T\}$$

Probability

An input string can have several parses and every such parse can be generated by several different combinations of constructions from the corpus. What we are interested in, is, given an input string *s*, the probability that arbitrary combinations of constructions from the corpus generate a certain parse T_i of *s*. Thus we are interested in the *conditional* *probability* of a parse T_i given *s*, with as *probability space* the set of constructions of trees in the corpus.

Let T_i be a parse of input string *s*, and suppose that T_i can exhaustively be generated by *k* tuples of constructions: $Tuples(T_i,C) = \{\langle t_{11},..,t_{1n_1}\rangle, \langle t_{21},..,t_{2n_2}\rangle, ..., \langle t_{k1},..,t_{kn_k}\rangle\}$. Then T_i occurs iff $\langle t_{11},...,t_{1n_1}\rangle$ **or** $\langle t_{21},...,t_{2n_2}\rangle$ **or** .... **or** $\langle t_{k1},...,t_{kn_k}\rangle$ occur, and $\langle t_{h1},...,t_{hn_h}\rangle$ occurs iff t_{h1} **and** t_{h2} **and** .... **and** t_{hn_h} occur ($h \in [1,k]$). Thus the *probability* of T_i is given by

$$P(T_i) = P(\, (t_{11} \cap ... \cap t_{1n_1}) \cup .... \cup (t_{k1} \cap ... \cap t_{kn_k})\,)$$

In shortened form:

$$P(T_i) = P(\bigcup_{p=1}^{k} (\bigcap_{q=1}^{n_p} t_{pq}))$$

The events t_{pq} are <u>not</u> mutually exclusive, since constructions can overlap, and can include other constructions. The general formula for the joint probability of events E_i is given by:

$$P(\bigcap_{i=1}^{n} E_i) = \prod_{i=1}^{n} P(E_i | E_{i-1} ... E_1)$$

The formula for the probability of combination of events E_i (that are not independent) is given by (see e.g. [Harris 1966]):

$$P(\bigcup_{i=1}^{k} E_i) = \sum_{i} P(E_i) - \sum_{i_1<i_2} P(E_{i_1} \cap E_{i_2}) + \sum_{i_1<i_2<i_3} P(E_{i_1} \cap E_{i_2} \cap E_{i_3}) - ....$$

$$+/- \; P(E_1 \cap E_2 \cap ... \cap E_k)$$

We will use Bayes' decomposition formula to derive the *conditional probability* of T_i given *s*. Let T_i and T_j be parses of *s*; the conditional probability of T_i given *s*, is then given by:

$$P(T_i|s) = \frac{P(T_i)P(s|T_i)}{P(s)} = \frac{P(T_i)P(s|T_i)}{\Sigma_j \, P(T_j)P(s|T_j)}$$

Since $P(s/T_j)$ is 1 for all j, we may write

$$P(T_i/s) = \frac{P(T_i)}{\Sigma_j P(T_j)}$$

A parse T_i of s with maximal conditional probability $P(T_i/s)$ is called a *preferred parse* of s.

Implementation

Several different implementations of DOP are possible. In [Scholtes 1992] a neural net implementation of DOP is proposed. Here we will show that conventional rule-based parsing strategies can be applied to DOP, by converting constructions into rules. A construction can be seen as a production rule, where the lefthand-side of the rule is constituted by the root of the construction and the righthand-side is constituted by the leaves of the construction. The only extra condition is that of every such rule its corresponding construction should be remembered in order to generate a parse-tree for the input string (by composing the constructions that correspond to the rules that are applied). For a construction t, the corresponding production rule is given by

$$root(t) \rightarrow leaves(t)$$

In order to calculate the *preferred parse* of an input string by maximizing the conditional probability, all parses with all possible tuples of constructions must be generated, which becomes highly inefficient. Often we are not interested in all parses of an ambiguous input string, neither in their exact probabilities, but only in which parse is the preferred parse. Thus we would like to have a strategy that estimates the top of the probability hierarchy of parses. This can be achieved by using *Monte Carlo* techniques (see e.g. [Hammersley 1964]): we estimate the preferred parse by taking *random samples* from the space of possibilities. This will give us a more effective approach than exhaustively calculating the probabilities.

Abstractions

It might happen that for some input string no parse can be generated. For instance when a word of the input does not exist in the corpus; or when the structures in the corpus cannot account for all structural properties of the input. Even if we include all analyses of sentences ever heard or spoken in our corpus, there will be well-formed utterances that cannot be parsed adequately in the way we described so far. This is simply because actual natural languages change in time and no rigid model can ever account for this. A natural language user hearing a new word in an utterance can immediately analyze the whole utterance, even if he does not understand the meaning of the word. He can assign a syntactic category to the word by matching its context with similar contexts in his (passive) memory, abstracting for a moment from the unknown word.

A performance model that wants to account for this phenomenon, should therefore leave open the possibility to make *abstractions* from words, as well as from features, categories, semantic formulas. Abstracting from a word, or more generally a label, means that this label is treated as a wildcard, in the sense that it can match with any other label. A reasonable constraint on abstractions is that they are as minimal as possible, otherwise we would abstract from all labels all the time, finding the same, most probable structure for all input strings.

The parsing process does now not change significantly: the system attempts to find the most probable way to parse an input by composing constructions from the corpus, with a minimal number of possible mismatches between the labels in the composition operation. Once a preferred parse is derived, the corpus is updated with that parse. A new word or structure is thus 'learned' by the corpus and a new use of it in the input does not involve any new abstractions. DOP shows how language change follows from language use, by updating the corpus with the structures attributed to the utterances.

Discussion

Although DOP has not yet been tested thoroughly[2], we can already predict some of its capabilities. In DOP, the probability of a parse depends on all tuples of constructions that generate that parse. The more different ways in which a parse can be generated, the higher tends to be its probability. This implies that a parse which can (also) be generated by relatively large constructions is favoured over a parse which can only be generated by relatively small constructions. This means that

[2] Corpora that will be used to test DOP, incude the Tosca Corpus, built at the University of Nijmegen, and possibly the Penn Treebank, built at the University of Pennsylvania.

prepositional phrase attachments and figures of speech can be processed adequately by DOP.

As to the problem of language acquisition, this might seem problematic for DOP: with an already analyzed corpus, only adult language behaviour can be simulated. The problem of language acquisition is in our perspective the problem of the acquisition of an initial corpus, in which non-linguistic input and pragmatics should play an important role.

An additional remark should be devoted here to formal grammars and disambiguation. Much work has been done to extend rule-based grammars with selectional restrictions such that the explosion of ambiguities is constrained considerably. However, to represent semantic and pragmatic constraints is a very expensive task. No one has ever succeeded in doing so except in relatively small grammars. Furthermore, a basic question remains as to whether it is possible to formally encode all of the syntactic, semantic and pragmatic information needed for disambiguation. In DOP, the additional information that one can draw from a corpus of hand-marked structural annotations is that one can by-pass the necessity for modelling world knowledge, since this will automatically enter into the disambiguation of structures by hand. Extracting constructions from these structures, and combining them in the most probable way, taking into account all possible statistical dependencies between them, preserves this world knowledge in the best possible way.

In conclusion, it may be interesting to note that our idea of using past language experiences instead of rules, has much in common with Stich's ideas about language ([Stich 1971]). In Stich's view, judgements of grammaticality are not determined by applying a precompiled set of grammar rules, but rather have the character of a perceptual judgement on the question to what extent the judged sentence 'looks like' the sentences the language user has in his head as examples of grammaticality. The concrete language experiences of the past of a language user determine how a new utterance is processed; there is no evidence for the assumption that past language experiences are generalized into a consistent theory that defines the grammaticality and the structure of new utterances univocally.

References

[Bahl 1983]: Bahl, L., Jelinek, F. and Mercer, R., 'A Maximum Likelihood Approach to Continuous Speech Recognition', in: *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-5, No.2.

[Fillmore 1988]: Fillmore, C., Kay, P. and O'Connor, M., 'Regularity and idiomaticity in grammatical constructions: the case of let alone', *Language* **64**, p. 501-538.

[Hammersley 1964]: Hammersley, J.M. and Handscomb, D.C., *Monte Carlo Methods*, Chapman and Hall, London.

[Harris 1966]: Harris, B., *Theory of Probability*, Addison-Wesley, Reading (Mass).

[Jelinek 1990]: Jelinek, F., Lafferty, J.D. and Mercer, R.L., *Basic Methods of Probabilistic Context Free Grammars*, Yorktown Heights: IBM RC 16374 (#72684).

[Magerman 1991]: Magerman, D. and Marcus, M., 'Pearl: A Probabilistic Chart Parser',in: *Proceedings of the European Chapter of the ACL'91*, Berlin.

[Martin 1979]: Martin, W.A., *Preliminary analysis of a breadth-first parsing algorithm: Theoretical and experimental results* (Technical Report No. TR-261). MIT LCS.

[Scha 1990]: Scha, R., 'Language Theory and Language Technology; Competence and Performance' (in Dutch), in: Q.A.M. de Kort & G.L.J. Leerdam (eds.), *Computertoepassingen in de Neerlandistiek*, Almere: Landelijke Vereniging van Neerlandici. (LVVN-jaarboek)

[Scholtes 1992]: Scholtes, J. C. and Bloembergen, S., 'The Design of a Neural Data-Oriented Parsing (DOP) System', *Proceedings of the International Joint Conference on Neural Networks 1992*, Baltimore.

[Stich 1971]: Stich, S.P., 'What every speaker knows', in: *Philosophical Review* **80**, p.476-496.