PROPERTY GRAMMARS: A FLEXIBLE CONSTRAINT-BASED APPROACH TO PARSING

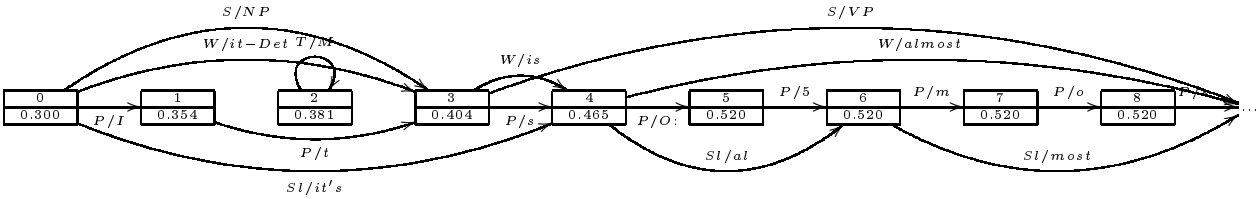
Philippe Blache & Jean-Marie Balfourier
LPL-CNRS, Université de Provence
29 Avenue Robert Schuman, 13621 Aix-en-Provence, France
{pb,balfourier}@lpl.univ-aix.fr

Abstract

We present in this paper a constraint-based parsing technique relying on *Property Grammar* in which all information is represented by means of constraints, the parsing process being a constraint satisfaction one. This technique is intrisically robust and allows the integration of different sources of linguistic knowledge.

1 Introduction

Interpreting a communication act requires to take into account all the aspects of linguistic analysis (prosody, morphology, syntax, semantics, pragmatics, etc..)



We propose in this paper a representation by means of graphs in which edges connect positions in the input signal. Each edge represent a relation and the linguistic structure can then be seen as a set of edges (possibly discontinuous). The example above represents such an annotation graph (cf. [Bird00] or [Blache00b]) in which different edges represent phonetic, morphologic or syntacitic information. This approach is fully incremental, in the sense given in [Johnson99]: the interpretation process is described in terms of interaction between properties describing the linguistic domains. It is thus necessary for each domain to be represented in the same form. We propose to specify edges by means of constraints. These constraints comes to building rules of new edges starting from sub-edges.

We present in this paper an application of this approach to parsing. The syntactic level can indeed illustrate the general mechanism. It is based on the result of a preliminary lexical analysis and aims at the constitution of the most total possible interpretation.

2 Property Grammars

Linguistic analysis can be described in terms of constraint interaction, as proposed in the constraint-based theories (see [Sag99] for a presentation of constraint satisfaction in such theories). Almost all modern linguistic theories make use of the notion of constraint. But in most of the cases, they

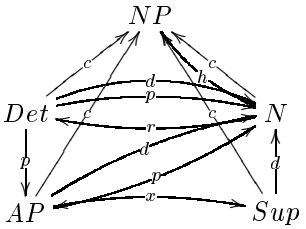
are simply used as a filtering mechanism as proposed for example by the Optimality Theory (see [Kager99]) or by Constraint Dependency Grammars (cf. [Schröder00]).

The formalism of Property Grammars (cf. [Blache00a]), by representing all the linguistic information by means of constraints, makes it possible to consider the analysis process as a constraint satisfaction one. In Property Grammars, constraints are stipulated over categories whereas most other approaches use constraints over structures (see for example [Schröder00] or [Duchier01]). In this last case, as it is usually the case in generative theories, one first have to build a structure and then to verify its satisfiability. In Property Grammars, satisfiability can be checked directly over the initial objects (lexical categories). The representation of syntactic information relies over a limited set of relations corresponding to the different types of properties: linearity, dependency, obligation, exclusion, exigency, constituency, uniqueness.

Property	Definition	Example
Constituency (*const*)	Set of categories constituting a phrase.	*Const(NP) = {Det, AP, N, PP, Sup, Pro}*
Head (*head*)	Set of compulsory, unique categories (heads).	*Head(NP) = {N, Pro}*
Uniqueness (*uniq*)	Set of categories which cannot be repeated in a phrase.	*Uniq(NP) = {Det, N, AP, PP, Sup, Pro}*
Requirement ($\Rightarrow$)	Cooccurrency between sets of categories.	$N[com] \Rightarrow Det$
Exclusion ($\not\Leftrightarrow$)	Restriction of cooccurrence between sets of categories.	$AP \not\Leftrightarrow Sup$
Linearity ($\prec$)	Linear precedence constraints.	$Det \prec N$
Dependency ($\rightsquigarrow$)	Dependency relations between categories.	$Adj \rightsquigarrow N$

We call in the following *characterization* the state of the constraint system after evaluation: it contains for a given set of categories the set of satisfied properties (noted P^+) and the set of violated ones (noted P^+).

(1)



An interesting aspect of this approach is the direct equivalence between a constraint system and a graph: each property describing a category can correspond to what we call a *constraint graph*. In the following, we call the described category the *root* of the graph. The figure (1) represents a subset of properties describing the *NP*: each property is represented as an edge between categories.

In this representation, c stands for constituency, d for dependency, x for exclusion, p stands for precedence, h stands for head, r stands for requirement. It is possible to represent a syntactic structure as a set of graphs that can be connected to each other with any kind of relation (including none). Any input can then be associated with a description: parsing an ill-formed sentence can then lead to a set of graphs. During parsing, a graph is built on the basis of relevant constraint graphs. It indicates all the properties satisfied by a given set of categories. Such a graph is called *description graph*. A set of categories participates to the characterization of a phrase when such a graph can be built.

3 The parsing level

In this framework, parsing comes to build the syntactic edges by means of constraint graphs specified in a property grammar. An edge is created when a set C of categories (lexical or not) can characterize an upper-level category XP. A positive characterization means that C satisfy all the properties describing XP. In terms of graphs such a characterization is represented by a description graph. An edge is built when such a graph exists and connects the first and last categories of C. It is labeled with XP, the description graph representing the properties characterizing XP plus a set of features.

The parsing mechanism itself consists in selecting a set of edges and verifying its satifiability with

respect to the constraint graphs of the grammar. Such a verification comes in the end to building a new edge. The new arc is labeled, taking into consideration all phrasal categories likely to contain the categories of this initial set. Once this category is determined, a new edge is created which constitutes a valid interpretation if it satisfies the set of relevant constraints in the grammar. As far as parsing ill-formed inputs is concerned, it is interesting to note that it is possible to build new edges with non positive characterization by relaxing constraints. More precisely, the edges built by the parser can be restricted to the positively characterized categories (parsing only grammatical inputs). But they can also describe categories that violate some constraints. The algorithm schema can be described as follows:

```
1.  current level is 1
2.  repeat
3.      for each input node (from first to last)
4.          for each edge e_i (from current level - 1 to level 0) which begins in current node
5.              for each constraint graph rooted by P containing e_i
6.                  initialize a new edge E with label P including e_i
7.                  Build(E)
8.                      if (E contains at least one edge of current level - 1)
                        and SAT(E)
                        then create a copy of E at the current level
9.                      for each edge e_j of lower level (in increasing order)
10.                         Build(E + e_j)
                        end for each
                    end Build
                end for each
            end for each
        end for each
11.     increment current level
12. until no node is created at the previous level
```

In this algorithm, determining the category of a new edge comes to find a corresponding graph of constraints (cf. instruction 5). It is possible, in spite of using an entire set of constraints, to use a single one, for example dependency or constituency. In this case, one simply have to replace the instruction 5 with :

5. **for each** P such that $e_i \in$ *const(P)*

It is also possible to rule out some cases by filtering the selection of possible sub-edges to be added to the current one with the instruction:

9-bis. **if** (e_j can follow E (i.e beginning on the following node)) **and** ($e_j \in$ *const(E)*)

4 Some results

A property grammar parser has been developed and tested for French. The grammar covers main syntactic phenomena such as relatives, sentence complements, adverbial phrases, noun clauses, clefts or coordinations. The parser has been evaluated over a set of 622 sentences (18,083 words) from the newspaper "Le Monde" tagged and disambiguated by Talana (see http://www.talana.linguist.jussieu.fr). The parser generates a total of 37,821 edges (around 60.8 edges per sentence). The following results shows in the first column (*Generated edges*) the edge distribution according to the categories. The second column (*MC edges*) indicates the number of edges that participate to a solution (a maximal coverage). The percentage in the sub-row indicates the proportion of edges actually used w.r.t. the

total amount. In the third column (*Root edges*), the number of edges constituting a maximal coverage is indicated. The percentage in the sub-row indicates the proportion of such edges w.r.t the number of edges participating to a solution. The fourth column (*Used edges*) indicates the number of edges used at a higher level (not necessarily in a final solution). Finally, the last column indicates the average depth of each phrase participating to a solution.

Category	Generated Edges		MC edges		Root edges		Used edges		Depth
VP	10181	26.9%	583	5.7%	189	29.0%	483	55.6%	3.96
NP	10042	26.6%	798	7.9%	88	13.5%	223	24.2%	3.15
PP	6500	17.2%	1454	22.4%	40	6.1%	58	10.5%	4.03
AP	5138	13.6%	766	14.9%	28	4.3%	104	20.4%	3.66
S	3743	9.9%	294	7.9%	272	41.8%	248	87.3%	5.52
AdvP	1004	2.7%	485	48.3%	0	0.0%	30	38.5%	1.00
Rel	858	2.3%	126	14.7%	25	3.8%	10	19.6%	3.90
Circ	338	0.9%	228	67.5%	9	1.4%	4	12.1%	6.25
Sup	17	0.0%	0	0.0%	0	0.0%	1	50.0%	1.00

The parser always proposes a description for an input in terms of satisfied and non satisfied properties. This means that a solution is a P that covers the entire input. In the case where such a P cannot be found, a partial description of the input can be given anyway. This particularity is obviously important when parsing ill-formed input.

5 Conclusion

Property grammars allow (1) a direct representation of all information by means of constraints and (2) the use of constraint satisfaction for parsing without needing any other mechanism. This means that, whatever its form (i.e. even for non grammatical utterances), the system can build a characterization of an input. Moreover, a set of constraints can be interpreted as a graph. Such a characteristics is important with respect to robustness: the linguistic structure is no more a hierarchical one (i.e. a tree-like structure) and allows the representation of partial or non connected structures.

References

[Bird00] Bird S., D. Day, J. Garofolo, J. Henderson, C. Laprun, & M. Liberman (2000) "ATLAS: A flexible and extensible architecture for linguistic annotation", in proceedings of *LREC*.

[Blache00a] Blache P. (2000) "Constraints, Linguistic Theories and Natural Language Processing", in *Natural Language Processing*, D. Christodoulakis (ed.), Lecture Notes in Artificial Intelligence, Springer.

[Blache00b] Blache P. & D.J. Hirst (2000) "Multi-level Annotation for Spoken-Language Corpora", in proceedings of *ICSLP-2000*.

[Duchier99] Duchier D. & S. Thater (1999) "Parsing with Tree Descriptions: a constraint based approach", in proceedings of *NLULP'99*.

[Duchier01] Duchier D. & R. Debusmann (2001) "Topological Dependency Trees: A Constraint-based Account of Linear Precedence", in proceedings of *ACL'01*.

[Hirst98] Hirst D. (1998), "Intonation in British English", in Hirst D. & A. Di Cristo (eds) *Intonation Systems*, Cambridge University Press.

[Johnson99] Johnson D. & S. Lappin. (1999) *Local Constraints vs. Economy*, CSLI.

[Kager99] Kager R. (1999) *Optimality Theory*, Cambridge University Press.

[Sag99] Sag I. & T. Wasow (1999), *Syntactic Theory. A Formal Introduction*, CSLI.

[Schröder00] I. Schröder, W. Menzel, K. Foth & M. Schulz (2000), "Modeling Dependency Grammars with Restricted Constraints", in journal *TAL*, 41:1.