A Robust and Efficient Parser for Non-Canonical Inputs

Philippe Blache

A Robust and Efficient Parser for Non-Canonical Inputs

Philippe Blache
CNRS & Université de Provence
29, Avenue Robert Schuman
13621 Aix-en-Provence, France
pb@lpl.univ-aix.fr

Abstract

We present in this paper a parser relying on a constraint-based formalism called Property Grammar. We show how constraints constitute an efficient solution in parsing non canonical material such as spoken language transcription or e-mails. This technique, provided that it is implemented with some control mechanisms, is very efficient. Some results are presented, from the French parsing evaluation campaign *EASy*.

1 Introduction

Parsing spoken languages and non canonical inputs remains a challenge for NLP systems. Many different solutions have been experimented, depending on the kind of material to be parsed or the kind of application: in some cases, superficial information such as bracketing is enough whereas in other situations, the system needs more details. The question of robustness, and more generally the parsing strategy, is addressed differently according to these parameters. Classically, three families of solutions are proposed:

- Reducing the complexity of the output
- Controlling the parsing strategy
- Training and adapting the system to the type of input

In the first case, the idea consists in building structures with little information, even underspecified (which means the possibility of building partial structures). We find in this family the different shallow parsing techniques (see for example [Hindle83], [Abney96]). Unsurprisingly, the use of statistical methods is very frequent and efficient in this kind of application (see [Tjong Kim Sang00] for some results of a comparison between different shallow parsers). Generally, such parsers (being them symbolic or not) are deterministic and build non recursive units. In some cases, they can also determine relations between units.

The second family contains many different techniques. The goal is to control a given parsing strategy by means of different mechanisms. Among them, we can underline three proposals:

- Implementing recovering mechanisms, triggering specific treatments in case of error (cf. [Boulier05])
- Controlling the parsing process by means of probabilistic information (cf. [Johnson98])
- Controlling deep parsers by means of shallow parsing techniques (cf. [Crysmann02], [UszKoreit02], [Marimon02])

The last kind of control mechanism consists in adapting the system to the material to be parsed. This can be done in different ways:

- Adding specific information in order to reduce the search space of the parsing process. This kind of information can appear under the form of ad hoc rules or information depending on the kind of data to be treated.
- Adapting the resources (lexicon, grammars) to the linguistic material

These different strategies offer several advantages and some of them can be used together. Their interest is that the related questions of robustness and efficiency are both taken into account. However, they do not constitute a generic

solution in the sense that something has to be modified either in the goal, in the formalism or in the process. In other words, they constitute an additional mechanism to be plugged into a given framework.

We propose in this paper a parsing technique relying on a constraint-based framework being both efficient and robust without need to modify the underlying formalism or the process. The notion of constraints is used in many different ways in NLP systems. They can be a very basic filtering process as proposed by *Constraint Grammars* (see [Karlsson90]) or can be part to an actual theory as with *HPSG* (see [Sag03]), the *Optimality Theory* (see [Prince03]) or *Constraint Dependency Grammars* (cf. [Maruyama90]). Our approach is very different: all information is represented by means of constraints; they do not stipulate requirements on the syntactic structure (as in the above cited approaches) but represent directly syntactic knowledge. In this approach, robustness is intrinsic to the formalism in the sense that what is built is not a structure of the input (for example under the form of a tree) but a description of its properties. The parsing mechanism can then be seen as a satisfaction process instead of a derivational one. Moreover, it becomes possible, whatever the form of the input, to give its characterization. The technique relies on constraint relaxation and is controlled by means of a simple left-corner strategy. One of its interests is that, on top of its efficiency, the same resources and the same parsing technique is used whatever the input.

After a presentation of the formalism and the parsing scheme, we describe an evaluation of the system for the treatment of spoken language. This evaluation has been done for French during the evaluation campaign Easy.

2 Property Grammars: a constraint-based formalism

We present in this section the formalism of Property Grammars (see [Bès99] for preliminary ideas, and [Blache00], [Blache05] for a presentation). The main characteristics of Property Grammars (noted hereafter PG), is that all information is represented by means of constraints. Moreover, grammaticality does not constitute the core question but become a side effect of a more general notion called characterization: an input is not associated to a syntactic structure, but described with its syntactic properties.

PG makes it possible to represent syntactic information in a decentralized way and at different levels. Instead of using sub-trees as with classical generative approaches, PG specifies directly constraints on features, categories or set of categories, independently of the structure to which they are supposed to belong. This characteristic is fundamental in dealing with partial, underspecified or non canonical data. It is then possible to stipulate relations between two objects, independently from their position in the input or into a structure. The description of the syntactic properties of an input can then be done very precisely, including the case of non canonical or non grammatical input. We give in the remaining of the section a brief overview of GP characteristics

All syntactic information is represented in PG by means of constraints (also called properties). They stipulate different kinds of relation between categories such as linear precedence, imperative co-occurrence, dependency, repetition, etc. There is a limited number of types of properties. In the technique described here, we use the following ones:

- Linear precedence: $Det < N$ (a determiner precedes the noun)
- Dependency: $AP \rightarrow N$ (an adjectival phrase depends on the noun)
- Requirement: $V[inf] \Rightarrow to$ (an infinitive comes with *to*)
- Exclusion: $seems \neq ThatClause[subj]$ (the verb *seems* cannot have *That* clause subjects)
- Uniqueness : $Uniq_{NP}\{Det\}$(the determiner is unique in a *NP*)
- Obligation : $Oblig_{NP}\{N, Pro\}$(a pronoun or a noun is mandatory in a *NP*)

This list can be completed according to the needs or the language to be parsed. In this formalism, a category, whatever its level is described with a set of properties, all of them being at the same level and none having to be verified before another.

Parsing a sentence in PG consists in verifying for each category the set of corresponding properties in the grammar. More precisely, the idea consists in verifying for each constituent subset its relevant constraints (i.e. the one applying to the ele-

ments of the subset). Some of these properties can be satisfied, some other can be violated. The result of this evaluation, for a category, is a set of properties together with their evaluation. We call such set the characterization of the category. Such an approach makes it possible to describe any kind of input.

Such flexibility has however a cost: parsing in PG is exponential (cf. [VanRullen05]). This complexity comes from several sources. First, this approach offers the possibility to consider all categories, independently from its corresponding position in the input, as possible constituent for another category. This makes it possible for example to take into account long distance or non projective dependencies between two units. Moreover, parsing non canonical utterances relies on the possibility of building characterizations with satisfied and violated constraints. In terms of implementation, a property being a constraint, this means the necessity to propose a constraint relaxation technique. Constraint relaxation and discontinuity are the main complexity factors of the PG parsing problem. The technique describe in the next section propose to control these aspects.

3 Parsing in PG

Before a description of the controlled parsing technique proposed here, we first present the general parsing schemata in PG. The process consists in building the list of all possible sets of categories that are potentially constituents of a syntactic unit (also called *constructions*). A characterization is built for each of this set. Insofar as constructions can be discontinuous, it is necessary to build all possible combinations of categories, in other words, the subsets set of the categories corresponding to the input to be parsed, starting from the lexical categories. We call *assignment* such a subset. All assignments have then, theoretically, to be evaluated with respect to the grammar. This means, for each assignment, traversing the constraint system and evaluating all relevant constraints (i.e. constraints involving categories belonging to the assignment). For some assignments, no property is relevant and the corresponding characterization is the empty set: we say in this case that the assignment in *non productive*. In other cases, the characterization is formed with all the evaluated properties, whatever their status (satisfied or not). At the first stage, all constructions contain only lexical categories, as in the following example:

Construction	*Assignment*	*Characterization*
AP	{Adv, Adj}	{Adv < Adj; Adv → Adj; ...}
NP	{Det, N}	{Det < N; Det → N; N ≠ Pro; ...}

An assignment with a productive characterization entails the instantiation of the construction as a new category; added to the set of categories. In the previous examples, *AP* and *NP* are then added to the initial set of lexical categories. A new set of assignments is then built, including these new categories as possible constituents, making it possible to identify new constructions. This general mechanism can be summarized as follows:

```
Initialization
   ∀ word at a position i:
      create the set c_i of its possible
      categories
   K ←  {c_i | 1<i<number of words}
   S ←  set of subsets of K
Repeat
   ∀ S_i ∈ S
      if S_i  is a productive assignment
        add k_i the characterization
             label to K
   S ←  set of subsets of K
Until new characterization are built
```

This parsing process underlines the complexity coming from the number of assignments to be taken into account: this set has to be rebuilt at each step (i.e. when a new construction is added).

As explained above, each assignment has to be evaluated. This process comes to build a characterization formed by the set of its relevant properties. A property p is relevant for an assignment A when A contains categories involved in the evaluation of p. In the case of unary properties constraining a category c, the relevance is directly known. In the case of n-ary properties, the situation is different for positive or negative properties. The former (e.g. cooccurrence constraints) concern two realized categories. In this case, c_1 and c_2 being these categories, we have $\{c_1, c_2\} \subset A$. In the case of negative properties (e.g. cooccurrence restriction), we need to have either $c_1 \notin A$ or $c_2 \notin A$.

When a property is relevant for a given A, its satisfiability is evaluated, according to the prop-

erty semantics, each property being associated to a solver. The general process is described as follows:

```
Let G the set of properties in the gram-
mar, let A an assignment

∀ p_i ∈ G, if p_i is relevant
     Evaluate the satisfiability of p_i
       for A
     Add p_i and its evaluation to the
       characterization C of A
Check whether C is productive
```

In this process, for all assignments, all properties have to be checked to verify their relevance and eventually their satisfiability.

The last aspect of this general process concerns the evaluation of the productivity of the characterization or an assignment. A productive assignment makes it possible to instantiate the corresponding category and to consider it as realized. A characterization is obviously productive when all properties are satisfied. But it is also possible to consider an assignment as productive when it contains violated properties. It is then possible to build categories, or more generally constructions, even for non canonical forms. In this case, the characterization is not entirely positive. This process has to be controlled. The basic control consists in deciding a threshold of violated constraints. It is also possible to be more precise and propose a hierarchization of the constraint system: some types of constraints or some constraints can play a more important role than others (cf. [Blache05b]).

A controlled version of this parsing schema, implemented in the experimentation described in the next section, takes advantage of the general framework, in particular in terms of robustness implemented as constraint relaxation. The process is however controlled for the construction of the assignment.

This control process relies on a left-corner strategy, adapted to the PG parsing schema. This strategy consists in identifying whether a category can start a new phrase. It makes it possible to drastically reduce the number of assignments and then control ambiguity. Moreover, the left corner suggests a construction label. The set of properties taken into consideration when building the characterization is then reduced to the set of properties corresponding to the label. These two controls, plus a disambiguation of the lexical level by means of an adapted POS tagger, render the parsing process very efficient.

The left corner process relies on a *precedence table*, calculated for each category according to the precedence properties in the grammar. This table is built automatically in verifying for each category whether, according to a given construction, it can precede all the other categories. The process consists in verifying that the category is not a left member of a precedence property of the construction. If so, the category is said to be a possible left corner of the construction. The precedence table contains then for each category the label of the construction for which it can be left corner.

During the process, when a category is a potential left corner of a construction *C*, we verify that the *C* is not the last construction opened by a left corner. If so, a new left corner is identified, and *C* is added to the set of possible constituents (usable by other assignments). Moreover, the characterization of the assignment beginning with c_i is built in verifying the subset of properties describing *C*.

The generation of the assignments can also be controlled by means of a co-constituency table. This table consists for each category, in indicating all the categories with which it belongs to a positive property. This table is easily built with a simple traversal of the constraint system. Adding a new category c_i to an assignment *A* is possible only when c_i appears as a co-constituent of a category belonging to *A*.

```
S initial set of lexical categories
Identification all the left corners
For all C, construction opened by a left
       corner c_i with G' the set of
       properties describing C
  Build assignments beginning by c_i
  Build characterizations verifying G'
```

The parsing mechanism described here takes advantage of the robustness of PG. All kind of input, whatever its form, can be parsed because if the possibility of relaxing constraints. Moreover, the control technique makes it possible to reduce the complexity of the process without modifying its philosophy.

4 Evaluation

We experimented this approach during the French evaluation campaign EASy (cf. [Paroubek05]). The test consisted in parsing several files containing various kinds of material: literature, newspaper, technical texts, questions, e-mails and spoken language. The total size of this corpus is one million words. Part of this corpus was annotated with morpho-syntactic (POS tags) and syntactic annotations. The last one provides bracketing as well as syntactic relations between units. The annotated part of the corpus represents 60,000 words and constitutes the gold standard.

The campaign consisted for the participants to parse the entire corpus (without knowing what part of the corpus constituted the reference). The results of the campaign are not yet available concerning the evaluation of the relations. The figures presented in this section concern constituent bracketing. The task consisted in identifying minimal non recursive constituents described by annotation guidelines given to the participants. The different categories to be built are: GA (adjective group: adjective or passed participle), GN (nominal group: determiner, noun adjective and its modifiers), GP (prepositional group), GR (adverb), NV (verbal nucleus: verb, clitics) and PV (verbal propositional group).

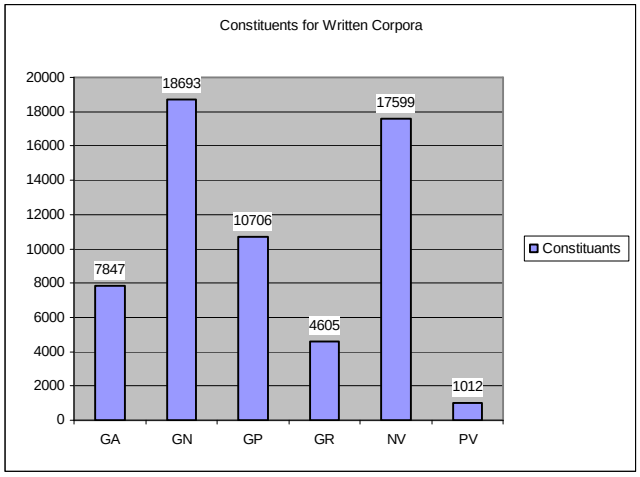
Our system parses the entire corpus (1 million words) in 4 minutes on a PC. It presents then a very good efficiency.

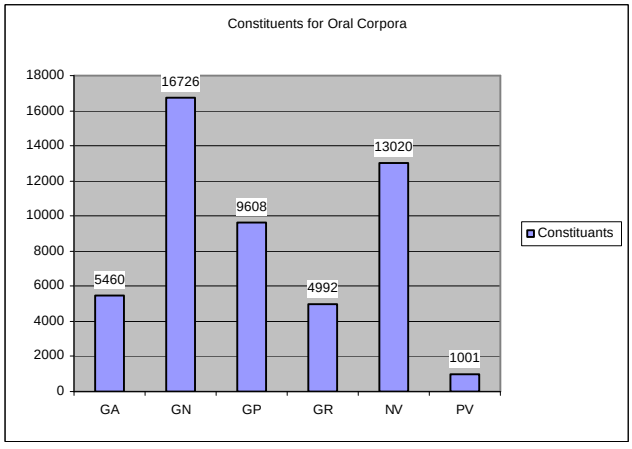
We have grouped the different corpora into three different categories: written texts (including newspapers, technical texts and literature), spoken language (orthographic transcription of spontaneous speech) and e-mails. The results are the following:

	Precision	*Recall*	*F-mesure*
Written texts	77.78	82.96	79.84
Spoken language	75.13	78.89	76.37
E-Mails	71.86	79.06	74.42

These figures show then very stable results in precision and recall, with only little loss of efficiency for non-canonical material. When studying more closely the results, some elements of explanation can be given. The e-mail corpus is to be analyzed separately: many POS tagging errors, due to the specificity of this kind of input explain the difference. Our POS-tagger was not tuned for this kind of lexical material.

The interpretation of the difference between written and oral corpora can have some linguistic basis. The following figures give quantitative indications on the categories built by the parser. The first remark is that the repartition between the different categories is the same. The only main difference concerns the higher number of nucleus VP in the case of written texts. This seems to support the classical idea that spoken language seems to use more nominal constructions than the written one.





The problem is that our parser encounters some difficulties in the identification of the NP borders. It very often also includes some material belonging in the grammar given during the campaign to AP or VP. The higher proportion of NPs in spoken corpora is an element of explanation for the difference in the results.

5 Conclusion

The first results obtained during the evaluation campaign described in this paper are very interesting. They illustrate the relevance of using symbolic approaches for parsing non-canonical material. The technique described here makes it possible to use the same method and the same resources whatever the kind of input and offers the possibility to do chunking as well as deep analysis. Moreover, such techniques, provided that they are implemented with some control mechanisms, can be very efficient: our parser treat more than 4,000 words per second. It constitutes then an efficient tool capable of dealing with large amount of data. On top of this efficiency, the parser has good results in terms of bracketing, whatever the kind of material parsed. This second characteristics also shows that the system can be used in real life applications.

In terms of theoretical results, such experimentation shows the interest of using constraints. First, they makes it possible to represent very fine-level information and offers a variety of control mechanisms, relying for example on the possibility of weighting them. Moreover, constraint relaxation techniques offer the possibility of building categories violating part of syntactic description of the grammar. They are then particularly well adapted to the treatment of non canonical texts. The formalism of Property Grammars being a fully constraint-based approach, it constitutes an efficient solution for the description of any kind of inputs.

Reference

[Abney 96] Abney S. (1996) "Partial Parsing via Finite-State Calculus", in proceedings of ESSLLI'96 Robust Parsing Workshop

[Bès99] Bès G. (1999) "La phrase verbale noyau en français", in *Recherches sur le français parlé*, 15, Université de Provence.

[Blache00] Blache P. (2000) "Constraints, Linguistic Theories and Natural Language Processing", in *Natural Language Processing*, D. Christodoulakis (ed), LNAI 1835, Springer-Verlag

[Blache05a] Blache P. (2005) "Property Grammars: A Fully Constraint-Based Theory", in *Constraint Solving and Language Processing,* H. Christiansen & al. (eds), LNAI 3438, Springer

[Boullier 05] Boullier P. & B. Sagot (2005) "Efficient and robust LFG parsing: SxLfg", in Proceedings of *IWPT '05*.

[Crysmann02] Crysmann B. A. Frank, B. Kiefer, S. Müller, G. Neumann, J. Piskorski, U. Schäfer, M. Siegel, H. Uszkoreit, F. Xu, M. Becker & H. Krieger (2002) "An Integrated Architecture for Shallow and Deep Processing", in proceedings of *ACL-02*.

[Frank03] Frank A., M. Becker, B. Crysmann, B. Kiefer & U. Schäfer (2003) "Integrated Shallow and Deep Parsing: TopP meets HPSG", in proceedings of *ACL-03*.

[Hindle83] Hindle D. (1983) *User manual for Fidditch, a deterministic parser*, Technical memorandum 7590-142, Naval Research Laboratory.

[Johnson98] Johnson M. (1998) "PCFG Models of Linguistic Tree Representations'", in *Computational Linguistics*, 24:4.

[Karlsson90] Karlsson F. (1990) "Constraint grammar as a framework for parsing running texts", in proceedings of *ACL-90*.

[Marimon02] Marimon M. (2002) "Integrating Shallow Linguistic Processing into a Unification-Based Spanish Grammar", in proceedings of *COLING-02*.

[Maruyama90] Maruyama H. (1990) "Structural Disambiguation with Constraint Propagation'", in proceedings of ACL'90.

[Paroubek05] Paroubek P., L. Pouillot, I. Robba & A. Vilnat (2005) "EASy : campagne d'évaluation des analyseurs syntaxiques", in proceedings of the workshop *EASy, TALN-2005*.

[Prince93] Prince A. & Smolensky P. (1993) "Optimality Theory: Constraint Interaction in Generative Grammars", Technical Report RUCCS TR-2, Rutgers Center for Cognitive Science.

[Tjong Kim Sang00] Tjong Kim Sang E. & S Buchholz (2000) "Introduction do the CoNLL-2000 Shared Task: Chunking", in proceedings of *CoNLL-2000*.

[Uszkoreit02] Uszkoreit H. (2002) "New Chances for Deep Linguistic Processing", in proceedings of *COLING-02*.

[VanRullen05] Van Rullen T. (2005), *Vers une analyse syntaxique à granularité variable*, PhD Thesis, Université de Provence.