Property Grammars: a Fully Constraint-Based Theory

Philippe Blache

Property Grammars: A Fully Constraint-Based Theory

Philippe Blache

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

Abstract. This paper presents the basis of *Property Grammars*, a fully constraint-based theory. In this approach, all kinds of linguistic information is represented by means of constraints. The *constraint system* constitutes then the core of the theory: it is the grammar, but it also constitutes, after evaluation for a given input, its description. *Property Grammars* is then a non-generative theory in the sense that no structure has to be build, only constraints are used both to represent linguistic information and to describe inputs. This paper describes the basic notions used in *PG* and proposes an account of long-distance dependencies, illustrating the expressive power of the formalism.

1 Introduction: Constraints in Property Grammars

Constraints are usually used in linguistics as a filtering process. They represent fine-level information in order to rule-out some unwanted structures. According to the theory, constraints can more or less play an intensive role, as it is the case with *HPSG* (see [Sag99]), *Optimality Theory* (see [Prince93]), Constraint *Dependency Grammars* (see [Maruyama90]), or in other kinds of formalisms such as *Dynamic Syntax* (see [Kempson02]). We propose to extend the use of constraints to the representation and the treatment different of linguistic information.

This approach presents several advantages, in particular the possibility of representing (and treating) separately different linguistic information. This idea was initially implemented in GPSG (see [Gazdar85]) which proposes, by means of the ID/LP formalism, to distinguish hierarchy from linear order whereas this information is merged in classical phrase-structure grammars. The introduction of constraints completed this tendency in offering the possibility of representing directly some specific information such as cooccurrence restriction. We propose, in *Property Grammars* (cf. [Blache00]) to represent separately each type of linguistic information by means of a specific constraint. In this way, constraints can represent all kinds of linguistic information.

One consequence of this characteristic is that constraints can be evaluated independently from each other. In generative theories, syntactic information is represented within a system in which all information has to be interpreted with respect to the entire system. In other words, a phrase-structure rule, which is the basic representation level in generative approaches, is not evaluable in itself. It takes its meaning in the global derivational process, and has no specific value outside of it. This is what Pullum

in his lectures (cf. also [Huddleston02]) call the *holistic* aspect of generative theories. And this is one of the main drawbacks of such theories. For example, taking into account the different uses of a language (for example spoken language) usually leads to deal with partial information. This means the necessity of interpreting sub-parts of the description system, which is incompatible with the generative model. An alternative approach consists in representing linguistic information in a decentralized way. In this case, each information is autonomous in the sense that it can be evaluated independently from the entire system. Constraints can then constitute an interesting alternative to classical approaches both for theoretical and computational reasons. Because of its flexibility (representation of partial information, no need of entire structures and non-holistic representation), a fully constraint-based approach, as the one presented here, makes it possible to describe any kind of input. Moreover, it becomes possible to take into account the contextual aspect of the linguistic phenomena: the same object or the same set of constituents can have different uses, different interpretations according to their environment. In the same perspective, interpreting a specific construction usually involve information coming from different sources (morphology, syntax, pragmatics, etc.). This way of thinking linguistic information is adopted in some recent works, in particular *Construction Grammars* (see [Fillmore98]).

This paper proposes a new presentation of the Property Grammars approach (hereafter noted *PG*) that integrates the notion of construction. The first section describes the basic units in PG. These objects, called constructions, represent both local and global information, the global one being represented in terms of constraints. The second section presents the notion of property in syntax. This section shows in what sense any kind of linguistic information can be represented by means of constraints. The last section addresses the question of long distance dependencies, illustrating the fact that using constraints and constructions constitutes an efficient and simple way of representing linguistic information.

2 Constructions in PG

The notion of construction, as presented in the Construction Grammars paradigm (see [Fillmore98], [Kay99]), consists in bringing together different parts of linguistic information, making it possible to describe specific or marked phenomena in terms of interacting properties. More precisely, a construction specifies a set of constituents plus different values or relations that these constituents bear. Any kind of information (in particular coming from syntax and semantics) can be used in such structures. All objects can be described in terms of constructions: lexical categories as well as phrases or specific syntactic phenomena. One of the interests of this notion is the possibility of representing directly contextual phenomena. We propose in the following a PG version of the notion of construction.

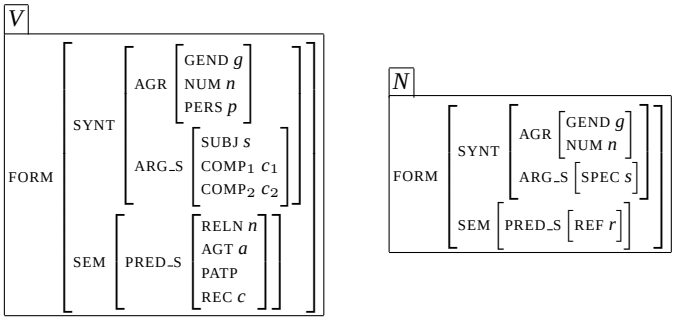
Generally speaking, it is necessary to distinguish between two different types of information: the one specific to the object (also called local information, or form) and the relations between the different components of the object, expressed in terms of properties or constraints (see next section). A construction in PG is then a triple ¡L, F,

P¿ in which L is the label of the construction, F its local information and P the properties representing relations between its components:

Label
FORM (*feature structure*)
PROPERTIES (*constraints*)

The form of the construction is represented by a feature structure. This structure is not fixed; the ones presented in this paper are a proposal among other possibilities. The important point is that this structure does not contain directly information about the constituents. Its role consists in instantiating different feature values, eventually by propagating some information from the relation side, as it will be shown later. In this version, we propose a basic repartition into syntactic and semantic features that can be completed by features coming from other domains. The structures are typed, which makes it possible to specify different forms according to the construction.

The following figures present the basic structures of two lexical constructions: the noun and the verb (we only show in these examples the Form part of the constructions). We make use in these structures of a very basic representation, making it possible to specify on the one hand the argument structure (a syntactic information) and a predicative structure (a semantic one). These values are not appropriated for all lexical constructions. For example, the adverb construction doesnt bear such structures. Moreover, when the feature is appropriated to a type, its form can vary (as it is the case between nouns and verbs).

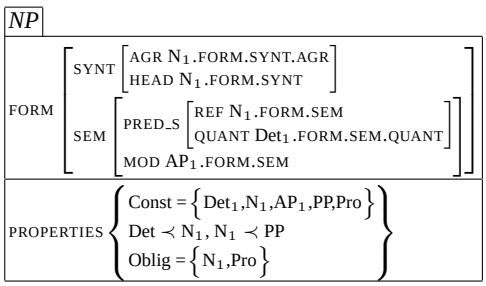


The argument structure is indicated in order to specify for a given construction its subcategorization schema. We will see that this information is also implemented by means of a property (requirement). At this stage, it is used in order to implement, if necessary, some semantic restrictions (for example lexical selection). In the case of the noun, this feature implements the subcategorization of the determiner as a specifier. As for the verb, we consider the subject and two complements as the basic structure. The predicative structure bears the basic roles. We consider for the verb the following ones: agent, patient and recipient, plus the relation name.

A construction is then a set of information describing a category, a phrase, a specific phenomenon. In other words, a construction is the general description of a particular phenomenon. The description of a given input is done by means of a set of constructions that are instantiated according to the input. An instantiated construction is called

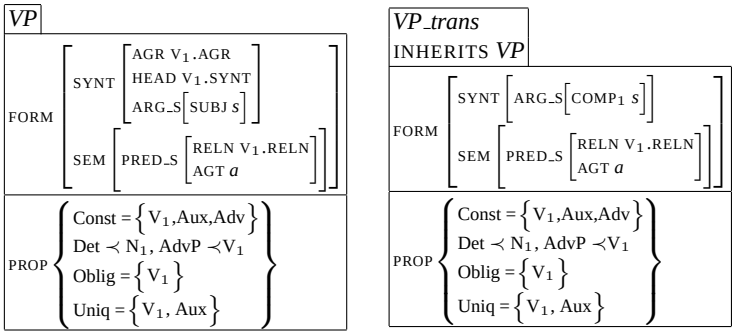
an object and can be used as a constituent of other constructions. More precisely, activating a construction results in inferring that the described object (referred by the label) is considered as realized. This means, from an operational point of view, that the object corresponding to the label is instantiated and added to the set of objects to be taken into consideration (this aspect will be detailed later). In the following, objects (or instantiated constructions) are indicated by means of an index.

The following example illustrates the NP construction. The form of the construction is specified by means of a feature structure indicating some syntactic and semantic properties (again, this structure can be modified in function of specific requirements). We will present in the next section more precisely the notion of property. One can already note in this representation the relations between the constituents and feature values, in terms of structure sharing. For example, the semantic referential feature takes as value the semantic value of the noun N1, one of the constituents of the construction.



Properties will be presented in detail in the next section. Roughly speaking, the one presented here describes the list of constituents, the linearity and the possible heads of the construction. References to feature values are indicated by means of a path in the object. In order to simplify notations, paths are indicated in reduced forms when there is no ambiguity (for example N.AGR instead of N.FORM.SYNT.AGR). In the same way, for simplicity reasons, we note an object by its label instead of the entire path towards the constituency property (for example, we note N_1 instead of NP.PROPERTIES.CONST.N_1). We can also notice the use of a MOD feature in order to represent modification which comes from the adjective (or, for the VP, from different modalities such as modal verbs, negation, etc.).

As it is usually the case, constructions are typed; the types being structured into a hierarchy. A type specifies for a given construction some characteristics in terms of features (the appropriate features in HPSG) and relations. The inheritance hierarchy is explicit: a construction specifies from what other construction it inherits. In the example of the figure (3), the subject-auxiliary construction is a specification of the verb phrase in the sense that all properties of the VP also apply to this construction. Some of them are overridden, which means that a property can be replaced by another one which is more specific (as it is the case in an object-oriented paradigm).



3 The Properties

The basic syntactic information corresponds to some regularities that can be observed and described separately. The kind of information presented in the following list illustrates this point:

– words follow a certain (partial) order
– some words are mutually exclusive in a close context
– some words systematically coocur in a close context
– some words cannot be repeated in a close context
– the form of some words co-varies in a close context
– the realization of some words is more facultative than some others

In PG, the idea consists in specifying each kind of such information by means of a distinct constraint. The remaining of the section proposes a presentation of the different types of constraints (see also [vanRullen03] for a slightly different presentation of the notion of property). We use for the property description the following definitions:

- let $\mathcal{K}$ be a set of categories, let $\mathcal{A}$ be a subset of categories corresponding to a given input,
- let $pos(\mathcal{C}, \mathcal{A})$ be a function that returns the position of $\mathcal{C}$ in $\mathcal{A}$,
- let $card(\mathcal{C}, \mathcal{A})$ be a function that returns the number of elements of type $\mathcal{C}$ in $\mathcal{A}$,
- let $\{\mathcal{C}_1, \mathcal{C}_2\} \in \mathcal{K}$,
- let $comp(\mathcal{C}_1, \mathcal{C}_2)$ be a function that verifies the semantic compatibility of $\mathcal{C}_1$ and $\mathcal{C}_2$ and complete the semantic structure of $\mathcal{C}_2$ with that of $\mathcal{C}_1$.

3.1 Precedence

The linear precedence relation stipulates explicitly the order relations between the objects. These relations are valid inside a context.

Def: $\mathcal{C}_1 \prec \mathcal{C}_2$ is satisfied for $\mathcal{A}$ iff $pos(\mathcal{C}_1, \mathcal{A}) < pos(\mathcal{C}_2, \mathcal{A})$

This property makes it possible to represent explicitly what is often done implicitly (as for example in HPSG).

3.2 Constituency

This relation indicates for a given construction the set of categories (or constructions) that are used in its description. This is then a classical description of constituency. However, in PG, this notion plays a very secondary role: it is theoretically possible to use any kind of constituent for any construction. The only necessity is to distinguish between a construction made only with licit constituents and another that could also contain other constituents, violating then this constraint. The important point is that this constraint does not have any priority over other constraints as in classical phrase structure grammars. It is even possible not to use it without any deep consequence on the parsing process itself.

3.3 Obligation

Among the set of constituents, some of them are required. This notion of obligation recovers partially that of head. However, it is always possible to have different categories in this set.

Def: *Oblig*($\mathcal{C}_1$) is satisfied for $\mathcal{A}$ iff $card(\mathcal{C}_{1,A}) = 1$

This notion of obligation makes it possible to identify specific elements that constitutes in a certain sense the core of the construction. However, at the difference with the notion of head, we do not consider that the status of obligatory element has consequences on the government and semantic relations. It is possible for some non-head elements within a construction to have direct relations. For example, within a NP, determiners can have some restrictions on the adjuncts. At the semantic level, it can also be the case that a dependency exists between two elements that are not heads. This is another argument in favor of a separate representation of the various types of information.

3.4 Uniqueness

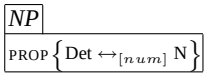
This property indicates the categories that can be realized only once into a construction.

Def: *Uniq*(C_1) is satisfied for $\mathcal{A}$ iff $card(\mathcal{C}_1, \mathcal{A}) \leq 1$

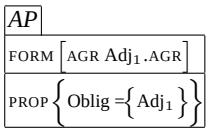
This property has to be explicit for every elements, including the obligatory ones.

3.5 Agreement

This property is used to implement different kinds of agreement. The classical agreement constraint consists in stipulating that two constituents of a construction must share the same agreement features. This is represented by a relation specifying the kind of features involved in the agreement and the objects concerned by the agreement. For example, the number agreement between the determiner and the noun is implemented as follows:



In some cases, the agreement value has to be propagated from a lexical construction to a phrasal one, as for the agreement between adjective and nouns in French which is stipulated between the N and the AP of a NP construction. In this case, the propagation has to be implemented between the Adj and the AP constructions, which is directly implemented as follows:



3.6 Requirement

This property specifies a mandatory cooccurrence between categories. This is in theory a relation between sets of categories.

Def: $C_1 \Rightarrow \mathcal{C}_2$ is satisfied for $\mathcal{A}$ iff $\mathcal{C}_1 \notin \mathcal{A}$ or $\mathcal{C}_2 \in \mathcal{A}$

This property is at the basis of the representation of the government relations. It can be used to implement any kind of cooccurrency, using syntactic or semantic information.

3.7 Exclusion

As the opposite of the requirement, exclusion stipulates cooccurrence restriction between categories. This property can be specified between sets of categories.

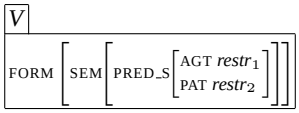
Def: $\mathcal{C}_1 \not\Rightarrow \mathcal{C}_2$ is satisfied for $\mathcal{A}$ iff $\{\mathcal{C}_1, \mathcal{C}_2\} \cup \mathcal{A} \neq \{\mathcal{C}_1, \mathcal{C}_2\}$

3.8 Dependency

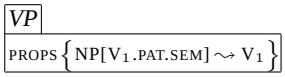
This property stipulates a semantic dependency between different objects of the construction. It is used to indicate how an object can fill the predicative structure of a governor. It is important to note that dependency relations can be expressed between any constituent of the construction, there is no specific need of specifying a semantic head.

Def: $\mathcal{C}_1 \rightsquigarrow \mathcal{C}_2$ is satisfied for $\mathcal{A}$ iff $comp(\mathcal{C}_1, \mathcal{C}_2)$ holds

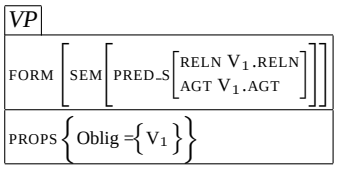
On top of this role, dependency is also used in order to implement semantic restrictions on the argument structure. This is done directly by specifying the corresponding semantic feature in the restricted object. For example, if a verb specifies a restriction on its arguments, it has to be indicated in the corresponding lexical construction:



This information is then used by a dependency property between the complement and the verb, specified into the VP construction:



This notation stipulates unification between the SEM values (more precisely the restriction features in the semantic structure) of the verb and its direct NP complement. In the same way, in order to implement the restriction between a verb and a subject, it is necessary to propagate the subject restriction from the V to the VP construction. It is done straightforwardly, without using any mechanism or propagation principle, by structure sharing in the construction itself:



A grammar in PG is then a set of constructions representing objects at any levels. As seen above, each construction contains a set of properties (or constraints). We call local constraint system (noted LCS) the set of properties specific to a construction. But constraints can also be taken globally, independently from the construction view. In this case, the set of all constraints without any direct reference to the construction they belong to, forms a set called global constraint system (noted GCS). This kind of polymorphism of constraints is at the basis of the parsing mechanism described in the next section.

4 How do things work?

As explained in the introduction, our perspective is definitely descriptive in the sense that a grammar is conceived as a tool for describing realized utterances. In such a perspective, the mechanisms have to be described starting from an input. We present in this section the general architecture for PG explaining how descriptions of an input can be built.

4.1 Characterization

One of the benefits of a constraint system is that it is possible to interpret its state after evaluation. It is even the most important operation in PG. The mechanism consists, for a given input, to identify and evaluate all the possible constraints of the grammar. More precisely, starting from the input, all the possible categories (the lexical constructions)

corresponding to the lexical items of the utterance are activated. This notion of activation is central in PG, we will explain it in more details in the next section. Activating a construction results in instantiating the corresponding object. At this stage, this means that the initial set of objects is the set of the lexical constructions corresponding to the words of the utterance.

Example: "*Time flies like an arrow*."
- Set of initial categories Σ = {N1, Adj1, N2, V2, V3, Det4, N5}

Starting from this set, the idea consists in trying to identify the possible constructions describing the input. For that, we take into consideration all the possible subsets of lexical categories of the input with one restriction: no two categories referring to the same position in the string can belong to the same subset.

Each of this subset is called an assignment, referring to the fact that we try to find all assignments that satisfy a constraint system (in other words a construction). Such an assignment can be then be composed by categories that are not necessarily juxtaposed. Obviously, such a technique can be computationally very expensive, but such an enumerative process can obviously take advantage of the constraint-based approach in order to control the process.

Example: Σ = {N1, Adj1, N2, V2, V3, Det4, N5}
- Assignments $\mathcal{A}$ = {{N1,N2}; {N1,V2}; {N1,V2,V3}; {V3,Det4,N5}; {N1,V3}, ...}

For each assignment, the process consists in traversing the global constraint system, in other words the set of all constraints specified in the grammar. For each constraint, there are three cases: the constraint cannot be evaluated and is considered as not relevant, the constraint is evaluated and satisfied, the constraint is evaluated and violated. All the relevant constraints are stored into a set called *characterization*[1]. It is possible to adopt different strategies at this stage. One consists in building only positive characterizations (i.e. characterizations only containing satisfied constraint). The other possibility, which is much more powerful and well adapted to the treatment of unrestricted inputs, including spoken language material, consists in building characterizations also containing violated constraints. This makes it possible to describe any kind of input, whatever its form. Obviously, relaxing constraints has a computational cost (cognitively as well as for an implementation), but it is the only way to explain why, even for native speakers, the surface form can vary even in violating some constraints. Each assignment, after this first stage, can be associated to a set of evaluated properties: a characterization is formed by an assignment plus a set of evaluated properties. For simplicity reasons, in the following examples, we represent some properties by means of sets. For example, the notation *Uniq(Det, N, Adj)* replaces the three corresponding properties of unicity.

Example: "*the book red*"

[1] Marie-Laure Gunot in her PhD work suggests to add a third type of constraints in the characterization: the one for which some categories (not all) involved in the constraint do not belong to the considered assignment. These constraints cannot be evaluated because not all constituents are present, but they can be said to be semi-activated.

Assignment	*Properties*
{Det1, N2}	P^+={Det$\prec$N; N$\Rightarrow$Det; Uniq(Det,N); Oblig(N)}
{Det1, N2, Adj3}	P^+={Det$\prec$N; N$\Rightarrow$Det; Uniq(Det, N, Adj); Oblig(N)}
.	P^-={Adj$\prec$N}

Among the different possible assignments, two are presented in this example. The first is positively characterized for linearity, the two realized constituents are unique, there is a cooccurrence between the common noun and the determiner, a noun (one of the obligatory constituents) is realized. The second assignment presents almost the same set of satisfied constraints but also has a non empty set of violated constraint.

4.2 Construction activation

The second stage of the process consists in traversing again the grammar, adopting this time the constructional view. Each characterization is compared to the local constraint systems of the grammar (the property sets attached to the constructions). When a LCS is subsumed by a characterization, the corresponding construction is said to be activated, which means that it is added to the set of object to take into consideration for the description. At this stage, the LCS is re-evaluated for the assignment in order to take into account (if necessary) some constraints specific to the construction or higher-level constraints. As mentioned before, it is not necessary to satisfy all constraints of a LCS in order to activate the corresponding construction, some can be violated

Example: "Dans la marine tu as short blanc chaussettes blanches"
(*In the navy you have white short, white socks*)

In this utterance (taken from a spoken French corpus), both NP complements violate a requirement constraint: in the French NP, determiners are mandatory with a common noun. Both characterizations contain then a violated constraint, but in spite of this, the NP construction is activated.

The same assignment can activate several constructions. Some of them are compatible (typically the one belonging to the same inheritance hierarchy), some others are not. In theory, all the possible constructions are activated, without taking into account the whether they are concurrent or not. When several activated constructions belong to the same branch of the type hierarchy, only the most specialized (the lowest in the hierarchy) is activated: the inheritance relation replaces the necessity of activating the entire hierarchy. When two constructions belong to different branches of the type hierarchy, both are activated and form a potential constituent of higher-level constructions.

When a construction is activated, the corresponding constituent is considered as realized and added to the set of constituents (in fact the set of labels referring to constructions). This set was initially formed by the constructions describing the lexical categories, progressively, this set also contains any level constructions. From this set, the same activating process is applied. This means that all the possible assignments are generated, with the same restriction as for lexical categories: no two constructions covering totally or partially the same substring of the input can belong to the same assignment.

All these new assignments are to their turn characterized. The process is repeated until no new construction can be activated. At the end, the input is described by means of several constructions. When all constructions are positively characterized, there exists one construction (corresponding to the label S) that covers the entire input (provided that such construction belongs to the grammar, which is not imperative, unlike in phrase-structure grammars). Otherwise, it can be the case that the input is described by means of several distinct constructions. In other words, the syntactic description of an input can be formed with a set of juxtaposed but non connected descriptions. This kind of representation is very useful for the description of unrestricted texts.

5 Unbounded dependencies

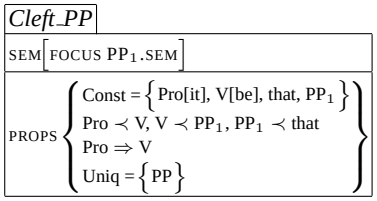
Unbounded dependency is one of the difficult phenomena to take into account whatever the theory. Its description makes use of some local properties describing the origin of the dependency (for example a cleft) plus other relations for the dependency itself between the origin and its site. Usually, the mechanism consists in propagating some information through a structure. This kind of process requires, by definition, to build a structure covering the entire input, or at least the domain of the dependency. As explained before, this is not always possible, in particular when the input is not canonical; in this case, several non connected constructions are used for the description, which forbids any propagation. Moreover, the propagation requires specific mechanisms (implementing the propagation itself as well as the control of the process).

In PG, any specific information is represented by means of constraints or set of constraints attached to a construction. The problem is that only part of the information is possibly represented by means of one construction. One solution could have been to describe the long distance dependency in terms of discontinuity. The problem, as with all other formalisms, is that the form of the construction, in particular its constituents, is not known a priori. Another solution has then to be found.

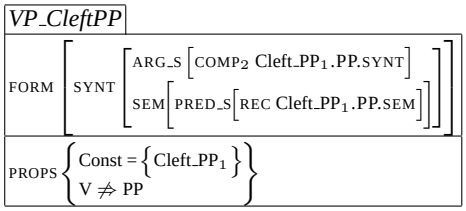
In the general case, dependencies are introduced by specific syntactic phenomena, with morpho-syntactic marks such as cleft, relatives, etc. In this situation, a specific construction is given for the description of what can be identified as the origin of the dependency. But, together with this construction, one can already identify some specificities or restrictions concerning the site of the dependency. The following example presents the case of a cleft PP.

Example: "*It is on the table that John put the book.*"

The grammar contains a specific construction describing the cleft itself, with its morpho-syntactic properties:



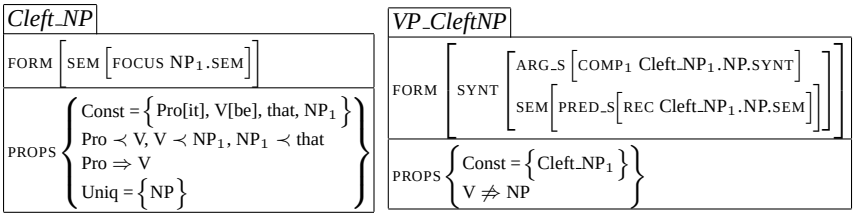
As it is the case with other constructions, the identification of the corresponding properties activates the construction *Cleft_PP*. It is now possible to use this new object in the rest of the description. In the case of a cleft *PP*, the only possibility is to have the *PP* extracted from a *VP*. In this case, it is necessary first to indicate that the *PP* fills a syntactic and semantic argument in the *VP* structure and second to forbid to realize another *PP* in the same *VP*. We create for this a specific construction specific to a *VP* from which the *PP* has been extracted. We call this construction *VP_CleftPP*, it inherits from the *VP* type and is represented as follows:



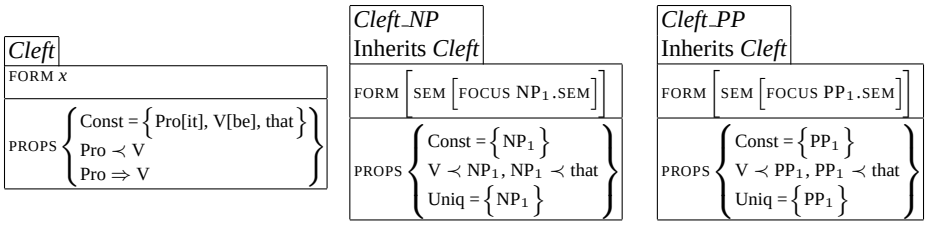
This construction is activated when a *Cleft_PP* is realized. It is not necessary to indicate more information than that or to use a specific propagation mechanism through the structure in order to implement relations between the cleft elements and the site of the dependency. In the case of a cleft *PP*, the site of the dependency is a *VP* semantically and syntactically compatible with such an argument and that has not yet realized this argument. More precisely, this *VP* should have a verbal constituent licensing a *PP* as complement (i.e. as a SYNT | ARG_S | COMP_2 value) and as a recipient argument of the predicative structure (i.e. the SEM | PRED_S | REC value). The link between the extracted element and the arguments values that it fills in the site of the dependency (the *VP*) is implemented by the fact that the $\textit{Cleft\_PP}_1$ is the only constituent of the construction. This object, noted $\textit{Cleft\_PP}_1$ in the construction, contains by definition a *PP* as constituent, the one that has been extracted. The link between the semantic and syntactic characteristics of this *PP* and the corresponding feature values to be filled in the *VP* in order to specify the dependency is then directly implemented.

Moreover, the exclusion constraint between the verb and another *PP* also has to be satisfied. The fact that a *PP* can only be extracted from a *VP* is controlled by the constituent property of the *VP_CleftPP* construction. During the parse, it can be the case that different VP construction can be a potential site for the dependency as soon as it satisfies the corresponding constraints and restrictions. This is a case of ambiguity that necessitates information coming from other domains (for example pragmatics or prosody), but no more information from the morpho-syntactic domain has to be given at this level.

The same kind of description can be given for a *NP* complement extraction. In this case as well, the cleft part of the construction is described by means of the *Cleft_NP* construction indicating the morpho-syntactic properties. This construction is to its turn the only constituent of the *VP_CleftNP* construction which indicates how the *NP* that has been cleft fills the different syntactic and semantic roles of the *VP*.



In order to simplify notations, the different cleft constructions can be organized into a type inheritance hierarchy making it possible to group the common characteristics of the cleft construction as follows:



6 Conclusion

During the last decade, the place of the notion constraints moved from the peripheral to the core of linguistic theories. Constraints are now not only a control mechanism used in order to refine analysis, but they can be conceived as an approach in itself. In fully constraints-based theories such as Property Grammars, there is no distinction between the representation and the control levels, which also means the possibility of a direct interpretation (cf. [Dahl04]). Constraints are used in order to represent the information and to build the structure: a grammar is a constraint system, an analysis is the state of the constraint system after evaluation for a given input. Such system can contain both satisfied and violated constraints, which makes the method adapted for dealing with unrestricted inputs, included spoken language material.

One of the challenges now is to show in what extend fully constraint-based approaches such as Property Grammars can constitute an actual language theory. Many aspects are to be explored in this perspective. First, it is necessary to explain in an homogeneous framework how the different linguistic domains (prosody, pragmatics, etc.) can interact. In our approach, any kind of constraints can be stipulated in the LCS. This means that each construction can have in its property part constraints describing the different domains. The interest of using constraints is that there is no need to express interaction in terms of correspondences between structures. The second kind of work, under experimentation for PG (see [Blache04]), concerns cognitive aspects. In particular, it seems possible to quantify some element of information useful in the characterization of the notion of complexity. In both cases, preliminary results show that constraints are not only a powerful way of representing linguistic information, they can also constitute the basis of a cognitive theory.

References

[Bès99] Bès G & P. Blache (1999) “Propriétés et analyse d’un langage”, in proceedings of *TALN’99*.

[Blache01] Blache P. & J-M. Balfourier (2001). “Property Grammars: a Flexible Constraint-Based Approach to Parsing”, in proceedings of *IWPT-2001*.

[Blache00] Blache P. (2000). “Constraints, Linguistic Theories and Natural Language Processing”, in *Natural Language Processing*, D. Christodoulakis (ed), Lecture Notes in Artificial Intelligence 1835, Springer-Verlag

[Blache04] lache P. & J.-P. Prost (2004) “Gradience, Constructions and Constraint Systems, in proceedings of the *CSLP Workshop*

[Croft03] Croft W. & D. Cruse (2003) *Cognitive Linguistics*, Cambridge University Press.

[Dahl04] Dahl V. & P. Blache (2004) “Implantation des Grammaires de Proprits en CHR”, in proceedings of *JFPLC’04*.

[Fillmore98] Fillmore C. (1998) “Inversion and Contructional Inheritance”, in *Lexical and Constructional Aspects of Linguistic Explanation*, Stanford University.

[Gazdar85] azdar G., E. Klein, G. Pullum & I. Sag (1985), *Generalized Phrase Structure Grammar*, Blackwell.

[Goldberg95] Goldberg A. (1995) *Constructions: A Construction Grammar Approach to Argument Structure*, Chicago University Press.

[Huddleston02] uddleston R. & G. Pullum (2002) *The Cambridge Grammar of English Language*, Cambridge University Press.

[Kay99] Kay P. & C. Fillmore (1999) “Grammatical Constructions and Linguistic Generalizations: the *what’s x doing y* construction”, Language.

[Keller03] Keller F. (2003) “A probabilistic Parser as a Model of Global Processing Difficulty”, in proceedings of *ACCSS-03*

[Kempson01] empson R., W. Meyer-Viol & D. Gabbay (1999) Dynamic *Syntax*, Blackwell.

[Langacker99] Langacker R. (1999), *Grammar and Conceptualization*, Walter de Gruyter.

[Pollard94] Pollard C. & I. Sag (1994), *Head-driven Phrase Structure Grammars*, CSLI, Chicago University Press.

[Maruyama90] aruyama H. (1990), “Structural Disambiguation with Constraint Propagation”, in proceedings of *ACL’90*.

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

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

[vanRullen03] . VanRullen, M.-L. Gunot & E. Bellengier (2003) Formal Representation of Property Grammars, in proceedings of ESSLLI *Student Session*