Representing Syntax by Means of Properties: a Formal Framework for Descriptive Approaches

Philippe Blache

Representing syntax by means of properties: a formal framework for descriptive approaches

Philippe Blache
CNRS & Aix-Marseille Université
Laboratoire Parole et Langage
blache@blri.fr

ABSTRACT

Linguistic description and language modelling need to be formally sound and complete while still being supported by data. We present a linguistic framework that bridges such formal and descriptive requirements, based on the representation of syntactic information by means of *local properties*. This approach, called *Property Grammars*, provides a formal basis for the description of specific characteristics as well as entire constructions. In contrast with other formalisms, all information is represented at the same level (no property playing a more important role than another) and independently (any property being evaluable separately). As a consequence, a syntactic description, instead of a complete hierarchical structure (typically a tree), is a set of multiple relations between words. This characteristic is crucial when describing unrestricted data, including spoken language. We show in this paper how local properties can implement any kind of syntactic information and constitute a formal framework for the representation of *constructions* (seen as a set of interacting properties). The *Property Grammars* approach thus offers the possibility to integrate the description of local phenomena into a general formal framework.

Keywords: syntax, constraints, linguistic theory, usage-based theories, constructions, Property Grammars

1 INTRODUCTION

The description and modelling of local language phenomena contributes to a better understanding of language processing. However, this data-driven perspective needs to provide a method of unifying models into a unique and homogeneous framework that would form an effective theory of language. Reciprocally, from the formal perspective, linguistic theories provide general architectures for language processing, but still have difficulty in integrating the variability of language productions. The challenge at hand is to test formal frameworks using a large range of unrestricted and heterogeneous data (including spoken language). The feasibility of this task mainly depends on the ability to describe all possible forms, regardless of whether they are well-formed (i.e. grammatical) or not. Such is the goal of the linguistic trend known as *usage-based* (Langacker 1987; Bybee 2010), which aims to describe how language works based on its concrete use. Our goal is to propose a new formal framework built upon this approach.

Moving away from the generative framework. Addressing the question of the syntactic description independently of grammaticality represents an epistemological departure from the generative approach in many respects. In particular, it consists in moving away from the representation of competence towards that of performance. Several recent approaches in line with this project consider grammar not as a device for generating language, but rather as a set of statements, making it possible to describe any kind of input, addressing at the same time the question of gradience in grammars (Aarts 2004; Blache and Prost 2005; Fanselow *et al.* 2005). To use a computational metaphor, this means replacing a *procedural approach* where grammar is a set of operations (rules), with a *declarative approach* where grammar is a set of descriptions. This evolution is fundamental: it relies on a clear distinction between linguistic knowledge (the grammar) and parsing mechanisms that are used for building a syntactic structure. In most current formalisms, this is not the case. For example, the representation of syntactic information with trees relies on the use of phrase-structure rules which encode both a syntactic relation (government) and operational information (the local tree to be used in the final structure). Such merging of operational information within the grammar can also be found in other formalisms. It is an important feature in

Tree-Adjoining Grammars (Joshi *et al.* 1975) in which the grammar is made of sub-parts of the final syntactic tree. It is also the case in *Dependency Grammars* (Tesnière 1959) with the projectivity principle (intended to control tree well-formedness) as well as in HPSG (Pollard and Sag 1994; Sag and Wasow 1999) and its feature percolation principles.

We propose disentangling these different aspects by excluding information solely motivated by the kind of structure to be built. In other words, linguistic information should be encoded independently of the form of the final representation. Grammar is limited then to a set of descriptions that are linguistic facts. As explained by Pullum and Scholz (2001), doing this enables a move away from *Generative-Enumerative Syntax* (GES) towards a *Model-Theoretic Syntax* (MTS) (Cornell and Rogers 2000; Blackburn and Meyer-Viol 1997; Blache 2007).

Several works are considered by Pullum and Scholz (2001) to exhibit the seeds of MTS, in particular around *HPSG* and *Construction Grammars* (Fillmore 1988; Kay and Fillmore 1999). These two approaches have recently converged, leading to a new framework called *Sign-Based Construction Grammars* (Sag 2012; Sag *et al.* 2012). SGBG is motivated by providing a formal basis for *Construction Grammars*, paving the way towards modelling language usage. It starts to fulfill the MTS requirements in that it proposes a monotonic system of declarative constraints, representing different sources of linguistic information and their interaction. However, there still remains a limitation that is inherent to HPSG: the central role played by *heads*. All information is controlled by this element, as the theory is *head-driven*. All principles are stipulated on the basis of the existence of a context-free skeleton, implemented by dominance schemas. As a consequence, the organization of the information is *syntacto-centric*: the interaction of the linguistic domains is organized around a head/dependent hierarchical structure, corresponding to a tree.

In these approaches, representing the information of a domain, and more to the point the interactions among the domains, requires one to first build the schema of mothers/daughters. Constraints are then applied as filters, so as to identify well-formed structures. As a side effect, no description can be given when no such structures can be built. This is a severe restriction both for theoretical and cognitive reasons: one of the requirements of MTS is to represent all linguistic do-

mains independently of each other (in what Pullum and Scholz (2001) call a *non-holistic* manner. Their interaction is to be implemented directly, without giving any priority to any of them with respect to the others. Ignoring this requirement necessarily entails a modular and serial conception of language processing, which is challenged now both in linguistics and psycholinguistics (Jackendoff 2007; Ferreira and Patson 2007; Swets *et al.* 2008). Evidence supporting this challenge includes: language processing is very often underspecified; linguistic information comes from different and heterogeneous sources that may vary depending on usage; the understanding mechanisms are often non-compositional; etc.

One goal of this paper is to propose an approach that accommodates such different uses of languageso as to be able to process canonical or non-canonical, mono- or multimodal inputs.

Describing any kind of input. Linguistic information needs to be represented separately when trying to account for unrestricted material, including non-canonical productions (for example in spoken language). The main motivation is that, whatever the sentence or the utterance to be parsed, it becomes then possible to identify its syntactic characteristics independently of the structure to be built. If we adopt this approach, we still can provide syntactic information partly describing the input even when no structure can be built (e.g. ill-formed realizations). In other words, it becomes possible to provide a description (in some cases a partial description) of an input regardless of its form.

This type of approach allows one to describe any type of sentence or utterance: it is no longer a question of establishing whether the sentence under question is grammatical or not, but rather of describing the sentence itself. This task amounts to deciding which descriptions present in the grammar are relevant to the object to be described and then to assessing them.

Grammar as set of constructions. One important advance for linguistic theories has been the introduction of the notion of *construction* (Fillmore 1988; Kay and Fillmore 1999). A construction is the description of a specific linguistic phenomenon, leading to a specific form-function pairing that is conventionalized or even not strictly predictable from its component parts (Goldberg 2003, 2009). These pairings result from the convergence of several properties or characteris-

tics, as illustrated in the following examples:

1. Covariational conditional construction
   `The Xer the Yer`: "*The more you watch the less you know*"
2. Ditransitive construction
   `Subj V Obj1 Obj2`: "*She gave him a kiss*"
3. Idiomatic construction: "*kick the bucket*"

Several studies and new methodologies have been applied to syntactic description in the perspective of modelling such phenomena (Bresnan 2007). The new challenge is to integrate these constructions, which are the basic elements of usage-based descriptions, into a homogeneous framework of a grammar. The problem is twofold: first, how to represent the different properties characterizing a construction; and second, how to represent the interaction between these properties in order to form a construction.

Our proposal. We seek an approach where grammars comprised of usage-based descriptions. A direct consequence is to move the question away from building a syntactic structure to describing the characteristics of an input. In concrete terms, grammatical information should be designed in terms of statements that are not conceived of with the aim of building a structure.

We propose a presentation of a theoretical framework that integrates the main requirements of a *usage-based* perspective. Namely, it first integrates constructions into a grammar and secondly describes non-grammatical exemplars. This approach relies on a clear distinction of operational and declarative aspects of syntactic information. A first step in this direction has been achieved with *Property Grammars* (Blache 2000; Blache and Prost 2014), in which a grammar is only made of properties, all represented independently of each other. *Property Grammars* offer an adequate framework for the description of linguistic phenomena in terms of interacting properties instead of structures. We propose going one step further by integrating the notion of construction into this framework. One of the contributions of this paper, in comparison to previous works, is a formal specification of the notion of construction, based on constraints only, instead of structures as in SBCG. It proposes moreover a computational method for recognizing them.

In the first section, we present a formal definition of the syntactic properties; these are used for describing any type of input. We then address more theoretical questions that constitute obstacles when trying to represent basic syntactic information independently of the rest of the grammar.[1] We explore in particular the consequences of representing relations between words directly, without the mediating influence of any higher-level structures or elements (i.e. without involving the notion of phrases or heads). Last, we describe how this framework can incorporate the notion of construction and detail its role in the parsing process.

2 NEW PROPERTIES FOR GRAMMARS

We seek to abstract the different types of properties that encode syntactic information. As explained above, we clearly separate the representation of such information from any pre-defined syntactic structure. In other words, we encode this information by itself, and not in respect to any structure: a basic syntactic property should not be involved in the building of a syntactic structure. It is thus necessary to provide a framework that excludes any notion of hierarchical information, such as heads or phrases: a property is a relation between two words, nothing more. Disconnecting structures and relations is the key towards the description of any kind of input as well as any type of construction.

Unlike most syntactic formalisms, we limit grammar to those aspects that are purely descriptive, excluding *operational* information. Here, the grammatical information as well as the structures proposed for representing syntactic knowledge are not determined by how they may be used during analysis. We want to avoid defining (e.g. as in constituency-based grammars) a phrase-structure rule as a step in the derivational process (corresponding to a sub-tree). In this case, the notions of projection and sisterhood eclipse all other information (linear order, co-occurrence, etc.), which becomes implicit. Likewise, in *dependency grammars*, a dependency relation corresponds to a branch on the dependency tree. In this context, sub-categorization or mod-

[1] Pullum and Scholz (2001) emphasize this characteristic as a requirement for moving away from the holistic nature of generative grammars.

ification information becomes dominant and supersedes other information which, in this case too, generally becomes implicit. This issue also also affects modern formalisms, such as HPSG (Pollard and Sag 1994; Sag and Wasow 1999; Sag 2012) which strictly speaking does not use phrase-structure rules but organizes syntactic information by means of principles in such a way that it has to percolate through the heads, building as a side-effect a tree-like structure.

Our approach, in the context of *Property Grammars* (hereafter PG) consists in identifying the different types of syntactic information in order to represent them separately. At this stage, we will organize grammatical statements around the following types of syntactic information:

- the *linear order* that exists among several categories in a construction
- the *mandatory co-occurrence* between two categories
- the *exclusion of co-occurrence* between two categories
- the impossibility of *repeating* a given category
- syntactic-semantic *dependency* between two categories (generally a category and the one that governs it)

This list of information is neither fixed nor exhaustive and could be completed according to the needs of the description of specific languages, for example with adjacency properties, completing linearity, or morphological dependencies.

Following previous formal presentations of *Property Grammars* (Duchier *et al.* 2010; Blache and Prost 2014) we propose the following notations: x, y (lower case) represent individual variables; X, Y (upper case) are set variables. We note $C(x)$ the set of individual variables in the domain assigned to the category C (cf. Backofen *et al.* (1995) for more precise definitions). We use the set of binary predicates for linear precedence ($\prec$) and equality ($\approx$).

2.1 *Linearity*

In PG, word order is governed by a set of linearity constraints, which are based on the clause established in the ID/LP formalism (Gazdar *et al.* 1985). Unlike phrase-structure or dependency grammars, this

information is, therefore, explicit. The linearity relationship between two categories is expressed as follows:

$$Prec(A,B) : (\forall x, y)[(A(x) \wedge B(y) \rightarrow y \not\prec x)] \quad (1)$$

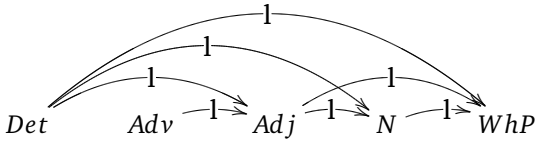
This is the same kind of linear precedence relation as proposed in GPSG (Gazdar *et al.* 1985). If the nodes x and y, respectively of category A and B, are realized[2], then y can not precede x.

For example, in a nominal construction in English, we can specify the following linearity properties:

$$Det \prec Adj; \quad Det \prec N; \quad Adj \prec N; \quad N \prec WhP; \quad N \prec Prep \quad (2)$$

Note that, in this set of properties, relations are expressed directly between the lexical categories (the notion of phrase-structure category is no longer used). As such, the $N \prec Prep$ property indicates precedence between these two categories regardless of their dependencies. This aspect is very important and constitutes one of the major characteristics of PG: all properties can be applied to any two items, including when no dependency or subcategorization link them.

The following example illustrates all the linearity relationships in the nominal construction "*The very old reporter who the senator attacked*" (the relative clause is not described here):



(3)

In this example, the linearity properties between two categories are independent of the *rection* (government) relations that these categories are likely to have. The linearity between *Det* and *Adj* holds even if these two categories have other dependencies (for example between the *Adj* and a modifier such as *Adv*). In theory, it could even be possible that a word dependent from the second category of the relation is realized before the first one: as such, there is no projectivity in

[2] A word or a category is said to be *realized* when it occurs in the sentence to be parsed.

these relations[3]. The same situation can be found for non-arguments: a linearity can be directly stipulated for example between a negative adverb and a verb. This is an argument in favour of stipulating properties directly between lexical categories rather than using phrase structures.

In addition to the representation of syntactic relations, properties may be used to instantiate attribute values. For example, we can distinguish the linearity properties between the noun and the verb, depending on whether *N* is *subject* or *object* by specifying this value in the property itself:

$$N[subj] \prec V; \qquad V \prec N[obj] \tag{4}$$

As we shall see, all properties can be used to instantiate certain attribute values. As is the case in *unification grammars*, attributes can be used to reduce the scope of a property by limiting the categories to which it can be applied. Generally speaking, a property (playing the role of a constraint) has a dual function: control (limiting a definition domain) and instantiation (assigning values to variables, by unification).

2.2 *Co-occurrence*

In many cases, some words or categories must co-occur in a domain, which is typically represented by sub-categorization properties. For example, the transitive schema for verbs implies that a nominal object (complement) must be included in the structure. Such co-occurrence constraint between two categories x and y specifies that if x is realized in a certain domain, then y must also be included. This is formally represented as follows:

$$Req(A,B) : (\forall x, y)[A(x) \rightarrow B(y)] \tag{5}$$

If a node x of category A is realized, so too is a node y of category B. The co-occurrence relation is not symmetric.

As for verbal constructions, a classical example of co-occurrence concerns nominal and prepositional complements of ditransitive verbs, which are represented through the following properties:

[3] Such a phenomenon does not exist in languages with fixed word order such as English or French.

$$V \Rightarrow N; \quad V[dit] \Rightarrow Prep \tag{6}$$

As described in the previous section, a property is stipulated over lexical categories, independently of their dependents and their order.

It should be noted that co-occurrence not only represents complement-type relations, it can also include co-occurrence properties directly between two categories independently from the head (thus regardless of rection relations). For example, the indefinite determiner is not generally used with a comparative superlative[4]:

(1) a. The most interesting book of the library
*b. *A most interesting book of the library*

In this case, there is a co-occurrence relation between the determiner and the superlative, which is represented by the property:

$$Sup \Rightarrow Det[def] \tag{7}$$

Furthermore, this example shows that we can also specify variable granularity properties by applying general or more specific categories by means of attribute values.

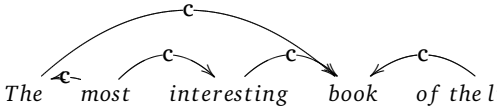
A key point must be emphasized when using co-occurrence properties: the notion of head does not play a preponderant role in our approach. Moreover, we do not use sets of constituents within which, in constituency-based grammar, the head is distinct and indicates the type of projection. Classically in syntax, the head is considered to be the governing category, which is also the minimum mandatory component required to create a phrase. This means that the governed components must be realized together with the head. As such, this information is represented by properties establishing co-occurrence between the head and its complements. Defining a specific property that identifies the head is, therefore, not necessary.

In the case of nominal construction, the fact that N is a mandatory category is stipulated by a set of co-occurrence properties between the complements and the adjuncts to the nominal head:

[4] This constraint is limited to comparative superlatives. In some cases the use of an indefinite determiner entails a loss of this characteristic. In the sentence "*In the crowd, you had a former fastest man in the world.*" the superlative becomes absolute, identifying a set of elements instead of a unique one.

$$Det \Rightarrow N[common]; \quad Adj \Rightarrow N; \quad WhP \Rightarrow N; \quad Prep \Rightarrow N \qquad (8)$$

The set of co-occurrence properties for the nominal construction described so far can be represented by the following graph:



(9)

We shall see later how the conjunction between co-occurrence and dependency properties is used to describe the syntactic characteristics of a head, without the need for other types of information. As such (unlike previous versions of PG), using specific properties for describing the head is not required.

At this stage, we can note that different solutions exist for representing non-headed constructions, for example when no noun is realized in a nominal construction. As we will see later, all constraints are violable. This means that a nominal construction without a noun such as in "*The very rich are different from you and me*" can be described with a violation of the co-occurrence properties stipulated above. This comes to identify a kind of implicit relation, not to say an empty category. Another solution consists in considering the adjective as a possible head of the nominal construction. In such case, the grammar should contain another set of co-occurrence and dependency properties that are directly stipulated towards the adjective instead of the noun.

2.3 *Exclusion (co-occurrence restriction)*

In some cases, restrictions on the possibilities of co-occurrence between categories must be expressed. These include, for example, cases of lexical selection, concordance, etc. An exclusion property is defined as follows:

$$Excl(A,B) : (\forall x)(\nexists y)[A(x) \wedge B(y)] \qquad (10)$$

When a node x of category A exists, a sibling y of category B can not exist. This is the *exclusion* relation between two constituents, that corresponds to the co-occurrence restriction in GPSG. The following

properties show a few co-occurrence restrictions between categories that are likely to be included in nominal constructions:

$$Pro \otimes N; \quad N[prop] \otimes N[com]; \quad N[prop] \otimes Prep[inf] \tag{11}$$

These properties stipulate that, in a nominal construction, the following can not exist simultaneously: a pronoun and a noun; a proper noun and a common noun; nor a proper noun and an infinitive construction introduced by a preposition.

Likewise, relative constructions can be managed based on the syntactic role of the pronoun. A relative construction introduced by a subject relative pronoun, as indicated in the following property, can not contain a noun with this same function. This restriction is compulsory in French, where relative pronouns are case marked:

$$WhP[subj] \otimes N[subj] \tag{12}$$

It is worth noting that a particularity of this type of property is that it can only be verified when the entire rection domain is known. We will discuss later the different cases of constraint satisfiability, which depend on their scope.

2.4 *Uniqueness*

Certain categories can not be repeated inside a rection domain. More specifically, categories of this kind can not be instantiated more than once in a given domain. This property is defined as follows:

$$Uniq(A) : (\forall x, y)[A(x) \wedge A(y) \rightarrow x \approx y] \tag{13}$$

If one node x of category A is realized, other nodes y of the same category A can not exist. Uniqueness stipulates that constituents can not be replicated in a given construction. Uniqueness properties are common in domain descriptions, although their importance depends upon the constructions to which they belong. The following example describes the uniqueness properties for nominal constructions:

$$Uniq = \{Det, Rel, Prep_{[inf]}, Adv\} \tag{14}$$

These properties are archetypal for the determiner and the relative pronoun. They also specify here that it is impossible to replicate

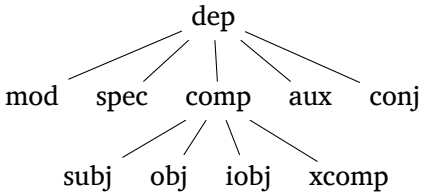


Figure 1:
The hierarchy of the *dependency* relation

a prepositional construction that introduces an infinitive ("*the will to stop*") or a determinative adverbial phrase ("*always more evaluation*").

Uniqueness properties are encoded by a loop:

$$\underset{\text{The}}{\overset{\text{u}}{\circlearrowright}} \qquad \underset{book}{} \qquad \underset{that}{\overset{\text{u}}{\circlearrowright}} \qquad \underset{I\ read}{} \tag{15}$$

2.5 *Dependency*

The dependency relation in *PG* is in line with the notion of syntactic-semantic dependency defined in *Dependency Grammars*. It describes different types of relations between two categories (complement, modifier, specifier, etc.). In terms of representation, this relation is arbitrarily oriented from the dependent to the head. It indicates the fact that a given object complements the syntactic organization of the target (usually the governor) and contributes to its semantic structure. In this section, we we leave aside semantics and focus on the syntactic aspect of the dependency relation.

Dependency relations are type-based and follow a type hierarchy (Figure 1); note that this hierarchy can be completed according to requirements of specific constructions or languages.

Since the dependency relation is organized as a type hierarchy, it is possible to describe a dependency relation at the most general level (the root of the hierarchy) or at any sub-level, depending on the required precision. Each of these types and/or sub-types corresponds to a classic syntactic relation (Figure 2).

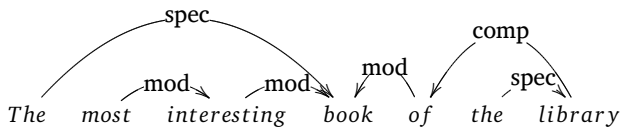
Dependency relations (noted $\rightsquigarrow$) possibly bear the dependency sub-type as an index. The following properties indicate the dependency properties applied to nominal constructions:

$$Det \rightsquigarrow_{spec} N[com]; \qquad Adj \rightsquigarrow_{mod} N; \qquad WhP \rightsquigarrow_{mod} N \tag{16}$$

Figure 2: The sub-types of the *dependency* relation

dep	generic relation, indicating dependency between a constructed component and its governing component
mod	modification relation (typically an adjunct)
spec	specification relation (typically *Det-N*)
comp	the most general relation between a head and an object (including the subject)
subj	dependency relation describing the subject
obj	dependency relation describing the direct object
iobj	dependency relation describing the indirect object
xcomp	other types of complementation (for example between N and $Prep$)
aux	relation between the auxiliary and the verb
conj	conjunction relation

The following example illustrates some dependencies into a nominal construction:



(17)

In this schema, we can see the specification relations between the determiners and the corresponding nouns, and the modification relations between the adjectival and prepositional constructions as well as between the adverb and the adjective inside the adjectival construction.

Feature control: The types used in the dependency relations, while specifying the relation itself, also provide information for the dependent element. In PG, the dependency relation also assigns a value to the FUNCTION attribute of the dependent. For example, a *subject* dependency between a noun and a verb is expressed by the following property:

$$N[subj] \rightsquigarrow_{subj} V \tag{18}$$

This property instantiates the function value in the lexical structure [FUNCTION *subject*]. Similarly, dependency relations, as with other properties, make it possible to control attribute values thanks to unification. This is useful, for example, for agreement attributes that are often linked to a dependency. For instance, in French, a gender and number agreement relation exists between the determiner, the adjective and the noun. This is expressed in the following dependencies:

$$Det[agr_i] \rightsquigarrow_{spec} N[agr_i]; \quad Adj[agr_i] \rightsquigarrow_{mod} N[agr_i] \qquad (19)$$

Formal aspects: Unlike dependency grammars, this dependency relation is not strict. First of all, as the dependencies are only a part of the syntactic information, a complete dependency graph connecting all the categories/words in the sentence is not required. Moreover, dependency graphs may contain cycles: certain categories may have dependency relations with more than one component. This is the case, for example, in relative constructions: the relative pronoun depends on the main verb of the construction (a complementation relation with the verb of the relative, regardless whether it is the subject, direct object, or indirect object). But it is also a dependent of the noun that it modifies.

In PG, a cycle may also exist between two categories. Again, this is the case in the relative construction, between the verb and the relative pronoun. The relative pronoun is a complement of the main verb of the relative. It is also the target of the dependency relation originating from the verb. This relation indicates that the verb (and its dependencies) will play a role in establishing the sense of the relative construction. In this case, the dependency relation remains generic (at the higher level of the type hierarchy). The dependency properties of the relative construction stipulate:

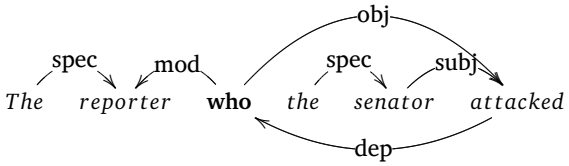
$$WhP_{[comp]} \rightsquigarrow_{comp} V; \quad WhP \rightsquigarrow_{mod} N; \quad V \rightsquigarrow_{dep} WhP \qquad (20)$$

It should be noted that the dependency relation between *WhP* and V bears the *comp* type. This generic type will be specified in the grammar by one of its sub-types $subj$, obj or $iobj$, each generating

Figure 3: Characteristics of the *dependency* relation

Antisymmetric:	if $A \rightsquigarrow_x B$, then $B \not\rightsquigarrow_x A$
Antireflexive:	if $A \rightsquigarrow B$, then $A \neq B$
Antitransitive:	if $A \rightsquigarrow_x B$ and if $B \rightsquigarrow_x C$ then $A \not\rightsquigarrow_x C$

different properties (in particular exclusion) for the relative. The following schema illustrates an example of a relative construction, with the particularities of having a double dependency for the WhP, and the cycle WhP-V:



(21)

As we can see, the dependency graph in PG (as with the other properties) is not necessarily connected or cycle-free. Figure 3 summarizes the main characteristics of the dependency relation.

Note that these relations are stipulated taking into account the precise type of the dependency relations: they are true only for a given type, but not as a general rule. For example, a symmetric complementation relation can not exist (if A is a complement of B, then B can not be a complement of A). However, a cycle can appear when the dependency types are different (as seen above for $V - WhP$ dependencies).

Apart from the type-based restrictions, properties are identical to those found in dependency grammars. One important features in PG is that the dependency graph is not necessarily connected and does not necessarily have a unique root.

Furthermore, we can see that when two realized categories (i.e. each corresponding to a word in the sentence) are linked by a property, they are usually in a dependency relation, directly or otherwise. Formally speaking, this characteristic can be expressed as follows:

Let $\mathscr{P}$ a relation expressing a PG property, let x, y and z categories:

$$\text{If } x \mathscr{P} y, \text{ then } x \rightsquigarrow y \vee y \rightsquigarrow x \vee [\exists z \text{ such that } x \rightsquigarrow z \wedge y \rightsquigarrow z] \quad (22)$$

Finally, dependency relations comprise two key constraints, ruling out some types of dual dependencies:

- A given category can not have the same type of dependency with several categories:

$$\text{If } x \rightsquigarrow_{dep_i} y, \text{ then } \nexists\, z \text{ such that } y \neq z \wedge x \rightsquigarrow_{dep_i} z \qquad (23)$$

Example : $Pro_i \rightsquigarrow_{subj} V_j$; $\quad Pro_i \rightsquigarrow_{subj} V_k$
The same pronoun can not be subject of two different verbs.

- A given category can not have two different types of dependencies with the same category:

$$\text{If } x \rightsquigarrow_{dep_i} y, \text{ then } \nexists\, dep_j \neq dep_i \text{ such that } x \rightsquigarrow_{type\_dep_j} y \qquad (24)$$

Example : $Pro_i \rightsquigarrow_{obj} V_j$; $\quad Pro_i \rightsquigarrow_{subj} V_j$
A given pronoun can not simultaneously be the subject and object of a given verb.

Note that such restrictions apply for dependencies at the same level in the dependency type hierarchy. In the above example, this is the case for *subj* and *obj*: such dual dependency can not exist. Also note that these constraints do not rule out licit double dependencies such as that encountered in control phenomena (a same subject is shared by two verbs) or in the case of the relative pronoun which is both the modifier of a noun and the complement of the verb of the relative:

$$WhP \rightsquigarrow_{comp} V; \qquad WhP \rightsquigarrow_{mod} N \qquad (25)$$

In this case, the relation types represent dependencies from both inside and outside the relative clause.

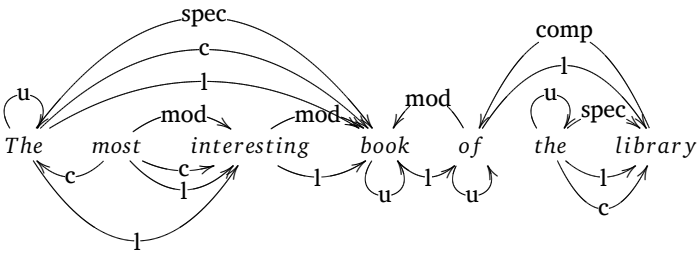
2.6 *A comprehensive example*

Each property as defined above corresponds to a certain type of syntactic information. In PG, describing the syntactic units or linguistic phenomena (chunks, constructions) in the grammar consists in gathering all the relevant properties into a set. Table 1 summarizes the properties describing the nominal construction.

In this approach, a syntactic description, instead of being organized around a specific structure (for example a tree), consists in a set of independent (but interacting) properties together with their status (satisfied or violated). The graph in the figure below illustrates the PG description of the nominal construction: "*The most interesting book of the library*".

Table 1: Properties of the nominal construction

$Det \prec \{Det, Adj, WhP, Prep, N\}$	$Det \rightsquigarrow_{spec} N$
$N \prec \{Prep, WhP\}$	$Adj \rightsquigarrow_{mod} N$
$Det \Rightarrow N[com]$	$WhP \rightsquigarrow_{mod} N$
$\{Adj, WhP, Prep\} \Rightarrow N$	$Prep \rightsquigarrow_{mod} N$
$Uniq = \{Pro, Det, N, WhP, Prep\}$	$Pro \otimes \{Det, Adj, WhP, Prep, N\}$
	$N[prop] \otimes Det$



(26)

In PG, a syntactic description is therefore the graph containing all the properties of the grammar that can be evaluated for the sentence to be parsed. As illustrated in the example, this property graph represents explicitly all the syntactic characteristics associated to the input; each is represented independently of the others.

3 BRINGING CONSTRUCTIONS INTO PROPERTY GRAMMARS

A *construction* is defined as the convergence of several properties. For example, the ditransitive construction is, among other features, characterized by the fact that the argument roles are filled by two nominal objects in a specific order. The first step towards the recognition of a construction consists in identifying such basic properties. At this stage, no other process but the spotting of the properties needs to be used. This means that all properties should be identified directly and independently of the rest of the grammar. For example, in the case of the ditransitive construction, this consists in identifying the linear order between the nominal objects.

The issue, then, is to describe such local and basic properties, without relating them to any higher level information. As a consequence, we propose a representation in which all properties are self-contained (as presented in the previous section) in the sense that their

evaluation should not depend on the recognition of other elements or structure. However, the two classical means of representing syntactic information (*constituency* or *dependency*) consist either in structuring higher-level groups (phrases in the case of constituency-based grammars) or assigning a specific role to the head in the definition of a branching structure (in the case of dependency grammars). In this section, we explore in greater details these aspects and their consequences when trying to represent basic properties directly. Our analysis is built around three questions: the notion of syntactic group, the status of the head, and the kind of information to be encoded in the lexicon for the representation of basic properties.

3.1 *Constructions as sets of properties*

Constituency-based approaches rely on the definition of syntactic properties in terms of belonging: a syntactic object is first characterized by its set of constituents. This approach offers several advantages in describing the distributional properties of syntactic groups, for example. Moreover, it constitutes a direct framework for controlling the scope of local properties (such as linearity or cooccurrence restriction): they are valid within a *domain* (a phrase).

Using this notion of domain proves interesting for constraint-based frameworks in which a phrase is described by a set of categories to which several constraints apply (offering a direct control of the scope of constraints). However, such an approach requires the organization of syntactic information into two separate types, forming two different levels: on the one hand, the definition of the domain (the set of categories, the phrase) and, on the other hand, their linguistic properties. In terms of representation (in the grammar), this means giving priority to the definition of the domain (the identification of the set of constituents, for example by means of rules or schemas). The constraints come on top of this first level, adding more information. In terms of parsing, the strategy also follows this dual level organization: first recognizing the set of categories (for example *Det, N, Rel, ...* for the *NP*), then evaluating constraint satisfaction.

The problem with this organization is that it gives priority to a certain type of information, namely constituency, that is motivated by operational matters (the representation and the construction of the syntactic structure) more than by linguistic considerations: sisterhood

in itself does not provide much syntactic knowledge or, more precisely, is too vague in comparison with the syntactic properties binding two categories (such as co-occurrence, restriction, dependency, etc.). Moreover, this organization constitutes a severe drawback: a linguistic description is only possible when the first level (identification of the set of categories) is completed. In other words, it is necessary to build a phrase before being able to evaluate its properties. This approach does not fit with the notion of construction for several reasons. First, a construction is not necessarily composed of adjacent constituents. A constituency-based grammar can not handle such objects directly. Moreover, constructions can be formed with a variable structure (elements of varying types, non-mandatory elements, etc.), due to the fact that they encode a convergence of different sources of information (phonology, morphology, semantics, syntax, etc.). An organization in terms of constituents relies on a representation driven by syntax, which renders impossible a description in terms of interaction of properties and domains as is the case with construction-based approaches.

Our goal is to integrate a multi-domain perspective, based on a description in terms of constructions, that is capable of dealing with any kind of input (including ill-formed or non-canonical realizations). We propose a representation of the linguistic information in terms of properties that are all at the same level. In other words, all information needs to be represented in the same manner, without any priority given to one type of information over another. No domain, set of categories or phrase should be built before being able to describe the linguistic characteristics of an input: a linguistic property should be identified directly, independently of any other structure.

As a consequence, properties need to be represented as such in the grammar (i.e. independently of any notion of constituency) and used directly during parsing (i.e. without needing to build a set of categories first). This goal becomes possible provided that the scope of the property is controlled. One way to do this consists in specifying precisely the categories in relation. Two types of information can be used with this perspective: the specification of certain features (limiting the kinds of objects to which the property can be applied), and the use of an HPSG-like category index (making it possible to specify when two categories from two properties refer to the same object).

As such, integrating the notion of construction should not make use of the notion of constituency but rather favour a description based on direct relations between words (or lexical categories). Thus, we fall in line with a perspective that is akin to dependency grammars, except for the fact that we intend to use a larger variety of properties to describe the syntax and not focus exclusively on dependency. In the remainder of this section we will present a means of representing constructions only using such basic properties.

3.2 *The question of heads: to have them or not (to have them)?*

The notion of head plays a decisive role in most linguistic theories: syntax is usually described in terms of government or dependency between a head and its dependents. In *constituency-based grammars*, the head bears a special relation to its projection (the root of the local tree it belongs to). In *dependency grammars*, a head is the target of the relations from the depending categories. The role of the head can be even more important in lexicalized theories such as LFG or HPSG. In this case, the head is also an operational element in the construction of the syntactic structure: it represents the site through which all information (encoded by features) percolates. All exocentric syntactic relations (between a phrase constituent and another component outside this phrase) are expressed as feature values which, as a result of a number of principles, move from the source constituent to the target, passing through the head.

A direct consequence is that when heads play a central role, syntactic information needs to be represented in a strictly hierarchical manner: as the head serves as a gateway, it is also a *reduction* point from which all information relating to the head's dependents may be accessed. Such a strict hierarchical conception of syntax has a formal consequence: the syntactic structure must be represented as a hierarchical (or a tree-like) structure in which every component (word, category, phrase, etc.) is dependent on a higher-level element. Such a syntactic organization is not suited for the description of many phenomena that we come across in *natural* language. For example, many constructions have no overt head:

(2) a. *John sets the red cube down and takes the black.*
b. *First trip, New York.*

c. *Monday, washing, Tuesday, ironing, Wednesday, rest.*

Example (2a) presents a classical elision as part of a conjunction: the second NP has no head. This is also the case in the nominal sentences in examples (2b) and (2c), which correspond to binary structures where each nominal component holds an argumentative position (from the semantic point of view) without a head being realized. We already gave some elements for the analysis of non-headed constructions in the second section. In the case of the last two examples, little information can be given at the syntactic level; it mainly comes from the interaction of morphology, prosody and discourse. The solution in PG (not developed in this paper) consists in implementing interaction constraints for controlling the alignment of properties coming from the different domains (Blache and Prévot 2010).

This raises the issue of structures that can be adapted to the representation of linguistic relations outside the head/dependent relation. The example of collective nouns in French illustrates such a situation:

(3) a. *un ensemble de catégories* (a set of categories)
b. **un ensemble des catégories* (a set of-plu categories)
c. *l'ensemble de catégories* (the set of categories)
d. *l'ensemble des catégories* (the set of-plu categories)

If a collective noun is specified by an indefinite determiner, then the complex category preposition-determiner *de* ("of") – which, in this case, is a partitive – can only be used in its singular form. This construction is controlled by the exclusion property:

$$Det_{[ind]} \otimes \{Prep + Det_{[plu]}\} \tag{27}$$

Inside a nominal construction with a collective noun, we have a direct constraint between the type of determiner (definite or indefinite) and the preposition agreement feature without any mediation of the head. In order to be complete, this property has to be restricted to those determiners specifying a collective noun. This is implemented by a co-indexation mechanism between categories, that will be described later on in the paper.

Generally speaking, the head plays a fundamental role in specifying the sub-categorization or the argument structure. It is not, however, necessary to give it an operational role when constructing the

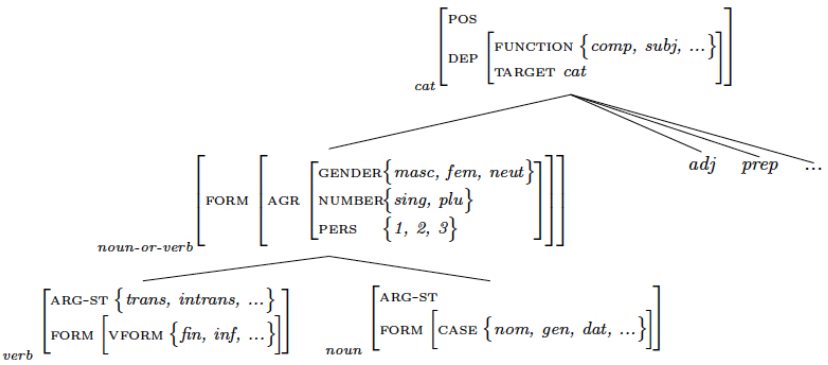


Figure 4: Inheritance in nominal and verbal categories

syntactic structure. We shall see that the head, even with no specific role, can be identified only as being the category to which all dependency relations converge.

3.3 *The structure of lexical entries*

As in unification grammars, the lexical information is highly important. Nonetheless, the lexicalization of syntactic information (emphasized in theories such as LFG or HPSG) is more limited in PG. In particular, the lexicon does not play a direct role in the construction of the syntactic structure; rather, all information is borne by the properties. Lexical information, although rich, is only used on the one hand to control the scope of the properties (as described above) and on the other hand to instantiate the subcategorization or the specific dependencies that one category can have with others.

In general, a lexical entry is associated with an attribute-value matrix which basically contains the category, agreement, morpho-syntactic features, sub-categorization list and grammatical function (when relevant). This structure can be enriched with other features, for example those describing semantics, phonology, etc. It can also be completed depending on the category, with more specific information such as mood, tense, person, or the valence feature that gives the list of arguments required.

Figure 4 summarizes the main features of nominal and verbal categories. It represents a type hierarchy, while the subtypes inherit appropriate features from the higher-level types.

The most general type, *cat*, comprises features appropriate to the description of all categories: the category label as well as the descrip-

tion of its dependency with other categories. This relation is described by the type of the dependency and the target value of the relation. In the above example, the lower level subtypes describe the features appropriated to *N* and *V*: both categories take agreement. Moreover, the verb has an argument structure which specifies its valence as well as its form attributes. As for the noun, it is associated with case features.

3.4 *The role of features*

Properties are relations between two lexical categories (that may potentially have other dependencies). For example, a linear property such as $V \prec N[obj]$ indicates that the verb precedes the direct object. This relation holds regardless of the other dependency relations of V and N. However, in this example, specifying the function value is mandatory: without such, the property would not be valid ($V \prec N$ is not licit as such in English).

The instantiation of feature values of a category involved in a property reduces its definition domain and, as a side effect, the scope of the property. Moreover, with all properties being independent of each other, it is necessary to provide as much information as possible to identify precisely the categories to be linked. Representing a property in this way renders them absolute, in the manner of *Optimality Theory* (Prince and Smolensky 1993) in which all constraints are universal. In this approach, a property can be evaluated directly, without needing any knowledge of the context or the rest of the syntactic structure. This condition is imperative when trying to consider a grammar as a set of properties.

We present two series of examples illustrating how feature instantiation helps in controlling the application of a property.

Control by feature values. The specification of feature values in properties can be used in order to describe certain phenomena directly. For example, the argument structure can be described by means of linearity and dependency properties, assigning subcategorization and case feature values:

$$\begin{array}{ll} V \Rightarrow N_{[subj]}; & V_{[trans]} \Rightarrow N_{[obj]} \\ V_{[intrans]} \otimes N_{[obj]}; & V_{[ditrans]} \Rightarrow N_{[iobj]} \end{array} \tag{28}$$

Likewise, the different possible constructions of the relative in French can be described by specifying the case of the relative pronoun:

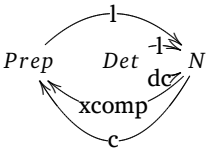
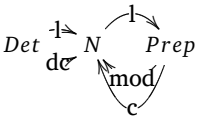
Construction	*Properties*	*Example*	*Property graph*
Prepositional	$Prep \prec N$ $N \rightsquigarrow_{xcomp} Prep$	*"on the table"*	
Nominal	$N \prec Prep$ $Prep \rightsquigarrow_{mod} N$	*"the book on ..."*	

Table 2: Relative ordering depending on the construction

$$\begin{array}{ll} WhP_{[nom]} \otimes N_{[subj]}; & WhP_{[nom]} \rightsquigarrow_{subj} V \\ WhP_{[acc]} \otimes N_{[obj]}; & WhP_{[nom]} \rightsquigarrow_{obj} V \end{array} \tag{29}$$

These properties stipulate that the nominative relative pronoun *qui* ("who") excludes the possibility to realize a subject within the relative construction and specifies a subject-type dependency relation between the relative pronoun and the verb. The same type of restriction is specified for the accusative pronoun *que* ("which") and could also be extended to the dative pronoun *dont* ("of which"/"of whom"). These properties implement the long-distance dependency between *WhP* and the "gap" in the argument structure of the main verb.

Control by co-indexation. We illustrate here the possibility of controlling the application of properties thanks to the co-indexation of the categories involved in different properties. The following example describes the relative order between *Prep* and N, which is governed by the type of construction in which they are involved: the preposition precedes the noun in a prepositional construction whereas it follows it into a nominal one. Table 2 presents a first description of these different cases, illustrated with an example.

As such, it is necessary to specify the *linearity* and *dependency* properties between *Prep* and N according to the construction they belong to. In order to distinguish between these two cases, we specify the syntactic functions. The following feature structures specify the dependency features of N, illustrating here the cases of the N subject of a V or complement of a $Prep$:

$$\text{(a) N} \left[\text{DEP} \left[\begin{array}{l}\text{FUNCTION } mod \\ \text{TARGET } V\end{array}\right]\right] \qquad \text{(b) N} \left[\text{DEP} \left[\begin{array}{l}\text{FUNCTION } xcomp \\ \text{TARGET } Prep\end{array}\right]\right] \tag{30}$$

Figure 5:
Features specification in properties

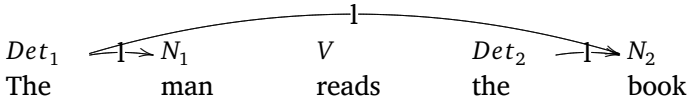
Construction type	Constraints
Nominal	$N_i \prec Prep\begin{bmatrix}\text{FCT } mod\\ \text{TGT } N_i\end{bmatrix}$ $Prep\begin{bmatrix}\text{FCT } mod\\ \text{TGT } N_i\end{bmatrix} \rightsquigarrow_{mod} N_i$
Prepositional	$Prep_i \prec N\begin{bmatrix}\text{FCT } xcomp\\ \text{TGT } Prep_i\end{bmatrix}$ $N\begin{bmatrix}\text{FCT } xcomp\\ \text{TGT } Prep_i\end{bmatrix} \rightsquigarrow_{xcomp} Prep_i$

Using this representation, the distinction between the two cases of dependency between N and $Prep$ relies on the specification of the function and target features of the categories (Figure 5). Moreover, a co-indexation makes it possible to link the properties.

These properties stipulate an order and a dependency relation; these are determined by the syntactic roles. In a nominal construction, the noun precedes the prepositional construction that modifies it, whereas the preposition precedes the noun in the other construction. Two classical mechanisms, based on unification, are used in these properties: first, the specification of the dependency attribute controls the application of the properties (the N following $Prep$ is its complement, the $Prep$ that follows N modifies it). Moreover, index unification (marked by the use of the same index i in the previous examples) ensures that the category is identical across all relations: the co-indexation of the categories in the different properties imposes a reference to the same object.

4 REPRESENTING AND PROCESSING CONSTRUCTIONS

Syntactic information is usually defined with respect to a specific domain (a set of categories). For example, the precedence property between *Det* and *N* only makes sense within a nominal construction. The following example illustrates this situation, showing the possible relations corresponding to the linearity property $Det \prec N$. These relations are represented regardless of any specific domain (i.e. between all the determiners and nouns of the sentence). Same-category words are distinguished by different indexes:



(31)

In this example, the relation $Det_1 \prec N_2$ connects two categories that clearly do not belong to the same domain. More generally, the subsets of categories $\{Det_1, N_1\}$ and $\{Det_2, N_2\}$ form possible units, unlike $\{Det_1, N_2\}$. The problem is that, as explained in the previous section, properties need to be assessed and evaluated independently of any a priori knowledge of a specific domain: a property in the grammar is not specifically attached to a set of categories (a phrase or a dependent). However, linguistic description relies mainly on the identification of local phenomena that corresponds to the notion of *construction* such as that specified in *Construction Grammars* (Fillmore 1988). It is, therefore, necessary to propose an approach fulfilling both requirements: the representation of properties independently and the description of local phenomena as sets of properties.

We propose to examine two perspectives: one concerning the grammatical representation and the other the question of parsing. The first perspective leads to a definition of constructions in terms of an interaction of properties. The latter presents the mechanisms for recognizing a construction on the basis of topological characteristics of the property graph (representing set of evaluated properties).

4.1 *In grammar: construction = set of properties*

Grammars organize syntactic information on the basis of structures to which different relations can be applied. In phrase-structure grammars, the notion of *phrase* implicitly comprises the definition of a domain (the set of constituents) in which the relations are valid. This notion of domain also exists in theories like HPSG, using generic tree schemas that are completed with the subcategorization information borne by lexical entries (both pieces of information together effectively correspond to the notion of constituency). Dependency grammars, in contrast, integrate syntactic information in the dependency relation between a head and its dependents. In both cases, the question of the scope of syntactic relations relies on the topology of the structures: a relation is valid inside a local tree. Therefore, a domain

typically corresponds to a set of categories that share common properties.

Our approach relies on a *decentralized* representation of syntactic information by means of relations that can be evaluated independently of the entire structure. In other words, any property can be assessed alone, without needing to evaluate any other. For example, the assessment of linearity between two categories is done without taking into account any other information such as subcategorization. In this case, we can evaluate the properties of a construction without having to create a syntactic tree: *PG* is based on a *dynamic* definition of the notion of construction. This means that all properties are assessed separately, a construction being the set of independently evaluated properties.[5]

In *Construction Grammars*, a construction is defined by the interaction of relations originating from different sources (lexical, syntactic, semantic, prosodic, etc.). This approach makes it possible to describe a wide variety of facts, from lexical selection to syntactico-semantic interactions (Goldberg 2003; Kay and Fillmore 1999; Lambrecht 1995). A construction is then intended as a linguistic *phenomenon* that is comprised of syntactic units as well as other types of structures such as multi-word expressions, specific turns, etc. The notion of construction is, therefore, more general than that of syntactic unit and not necessarily based on a structured representation of information (e.g. a tree).

PG provides an adequate framework for the representation of constructions. First, a syntactic description is the interaction of several sources of information and properties. Moreover, *PG* is a constraint-based theory in which each piece of information corresponds to a constraint (or property). The description of a construction in a *PG* grammar is a set of properties connecting several categories. This definition gives priority to the relations instead of their arguments, which means that prior definition of the set of constituents involved in the construction is not necessary.[6] As a consequence, the notion of constraint scope is not directly encoded: each property is specified independently and

[5] A direct implementation of this mechanism consists in assessing all the possible properties, for all the combinations of words/categories, which is exponential. Different possibilities of controlling this complexity exists, such as delayed evaluation or probabilistic selection.

[6] In previous versions of *PG*, all categories belonging to a construction were indicated in a list of constituents.

the grammar is a set of constructions, each described by a set of properties.

The following example illustrates the encoding of the ditransitive construction, focusing on the relation between the type of categories (N or *Prep*), their linear order and their function:

(1) $V_{[ditrans]} \Rightarrow N_{[obj]}$
(2) $V_{[ditrans]} \Rightarrow X_{[iobj]}$
(3) $N_{[iobj]} \prec N_{[obj]}$
(4) $N_{[obj]} \prec Prep_{[iobj]}$
(5) $N_{[obj]} \rightsquigarrow_{obj} V_{[ditrans]}$
(6) $N_{[iobj]} \rightsquigarrow_{iobj} V_{[ditrans]}$
(7) $Prep_{[iobj]} \rightsquigarrow_{iobj} V_{[ditrans]}$

The two first co-occurrence properties stipulate that the ditransitive verb governs a nominal object plus an indirect object of unspecified category encoded by X (that could be, according to the rest of the properties, either a nominal or a prepositional construction). Linearity properties stipulate that in the case of a double nominal construction, the nominal indirect object should precede the direct object. Otherwise, the direct object precedes the indirect prepositional construction. Finally, the dependency relations instantiate, according to their function, the type of the dependency with the verb.

4.2 *In analysis : construction = government domain*

The theoretical and naïve parsing principle in PG consists in evaluating all properties that may exist between all categories corresponding to the words in a sentence. This set of properties contains considerable noise: most of the properties evaluated in this way link categories which do not belong to the same domain. The issue is to elicit the constructions existing in this set. Concretely, the set of properties forms a graph from which the connected categories may correspond to a construction. In the following, we put forward a formal characterisation of the notion of construction in terms of graph topology.

Generally speaking, two types of properties can be distinguished, based on the number of categories they involve:

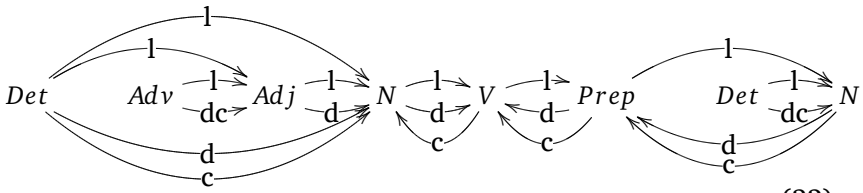
- Binary properties, where two categories are connected: linearity, dependency, co-occurrence
- Unary properties: uniqueness, exclusion

Unary relations, because of their specificity, do not have any features that may be used to identify the construction. On the contrary,

Figure 6: Constructions corresponding to maximal complete subgraphs

$Adv-Adj$	Adjectival construction
$Det-Adj-N$	Nominal construction
$N-V$	Subject/verb construction
$V-Prep$	Verb/indirect object construction
$Prep-N$	Prepositional construction
$Det-N$	Nominal construction

the three types of binary properties are the basis of the domain identification mechanism. The following graph illustrates the characterization of the sentence "*A very old book is on the table.*" :



(32)

It is noteworthy that in this graph, it is possible to identify several subgraphs in which all the categories are interconnected. Formally, they are referred to as being *complete*: a complete graph is a graph where all nodes are connected. In this example, the nodes labelled by *Adv* and *Adj* form a complete subgraph: both categories are connected. On the other hand, the set of categories $\{Det, Adv, Adj\}$ does not form a complete subgraph, the *Det* and *Adv* categories being disconnected.

Furthermore, when eliciting a construction, it is necessary to take into account all the categories of a same constraint network. For example, the *Adj* and *N* nodes could form a complete subgraph, but it would be a subset of a more complete subgraph $\{Det, Adj, N\}$ subset. As a consequence, we only take into consideration *maximal complete subgraphs*.

The maximal complete subgraphs in the previous example correspond to the subsets of the following nodes (Figure 6) to which we have associated a construction type.

As such, based on a graph topology, we can identify constructions for which the following definition can be given:

Definition: *A construction is a maximal complete subgraph of the property graph.*

Concretely, these subsets correspond to syntactic units. Yet, where classical approaches rely on the definition of constructions a priori in the grammar, this definition proposes a dynamic and a posteriori description. This is fundamental: it makes it possible to describe any type of sentence, regardless of its grammaticality. Analyzing a sentence consists in interpreting the property graph. This structure may contain constructions that lead directly to a semantic interpretation. But it can also be the case that the property graph contains subparts that are not necessarily connected with the rest of the sentence. This situation occurs with ungrammatical sentences.

At this stage, exhibiting the set of relevant constructions for the description of a sentence consists in identifying, among the set of maximal complete subgraphs, those that cover the set of words: in the optimal case, the set of nodes of the exhibited constructions corresponds to the set of words in the sentence. Note that in theory, constructions can overlap, which means that the same node could belong to different constructions. This characteristic is useful when combining different domains of linguistic description, including prosody, discourse, etc. However, when studying a single domain, for example syntax, it is useful to reduce overlapping: a category belonging to a construction can contribute to another construction provided it is its head. The task is therefore to exhibit the optimal set of constructions, covering the entire input.

5 PARSING BY SATISFYING CONSTRAINTS

Parsing a sentence S consists in firstly determining and evaluating the set of properties relevant for the input and secondly in exhibiting the constructions. In the second stage, it is necessary to establish all the partitions[7] of the suite of categories that correspond to S. The issue is to know which parts correspond to a construction and whether an *optimal* partition exists.

In the first stage, an operational semantics describing conditions of satisfiability must be assigned to the properties. In this perspective, we introduce some preliminary notions:

[7] A partition of S is a set of non-empty parts of P, disjoint two-by-two, and that cover S.

Figure 7: Operational semantics of properties

- `Uniqueness`: $Uniq_x$ holds in C iff $\forall y \in C$, then $x \neq y$
- `Exclusion`: $x \otimes y$ holds in C iff $\forall z \in C$, then $z \neq y$
- `Co-occurrence`: $x \Rightarrow y$ holds in C iff $C \cap \{y\} \neq \emptyset$
- `Linearity`: $x \prec y$ holds in C iff $pos(x, C) < pos(y, C)$

- **Set of property categories** : Let p be a property. We define a function **Cat**(**p**) building the set of categories contained in p. For example, $Cat(Det \prec N) = \{Det, N\}$.
- **Applicable properties** : Given a grammar G and a set of categories C, the set of *C-applicable properties* is the set of all the properties of G in which the categories of C appear. More specifically, a property p is *applicable* when its evaluation becomes possible. Two types of properties can be distinguished: those requiring the realization of all the categories they involve (uniqueness, linearity and dependency) and the properties needing at least one of their categories to be evaluated (co-occurrence and exclusion). As such, we have:
  Definition: Let $p \in G$:
  - if $p \in \{uniq, lin, dep\}$, p is an *applicable property* for C iff $\forall c \in Cat(p)$, then $c \in C$
  - if $p \in \{cooc, excl\}$, p is an *applicable property* for C iff $\exists c \in Cat(p)$, such that $c \in C$
- **Position in the string** : We define a function **Pos**(**c**, **C**), returning the rank of c in the category suite C

An operational semantic definition may be assigned to each property as in Figure 7 (C being a set of categories).

These definitions provide the conditions of satisfiability of the different properties. It now becomes possible to illustrate how the description of the syntactic structure can be built.

The construction of the syntactic description (called the *characterization*) of a construction consists in evaluating the set of its applicable properties. In more general terms, parsing a sentence consists in evaluating all the relevant properties and then determining the corresponding constructions.

Formally, let S be the set of categories of a sentence to be parsed, let $Part_S$ be a partition of S, let p be one subpart of $Part_S$, let $Prop_p$ be the

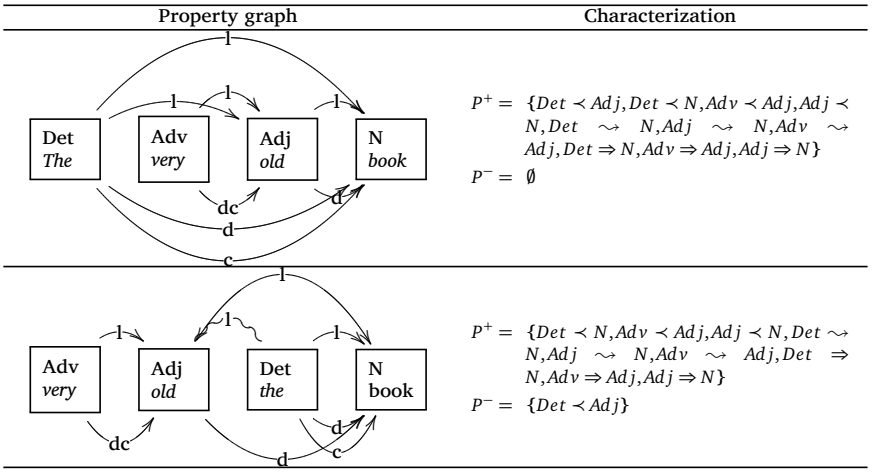


Figure 8: Property graphs and their characterizations

set of applicable properties of p. The categories belonging to p part are instantiated: their feature values, as determined by the corresponding lexical entries, are known insofar as they correspond to the words of the sentence to be parsed. The properties in $Prop_p$ stipulate constraints in which the categories are fully instantiated (by the unification of the categories of the properties in the grammar and those realized in the sentence). We define $Sat(Prop_p)$ as the constraint system formed by both applicable properties and the state of their satisfaction after evaluation (true or false).

Figure 5 presents two examples of nominal constructions along with their characterizations; the second example contains a linear constraint violation between Det and Adj:

This example illustrates a key aspect of *Property Grammars*: their ability to describe an ill-formed sentence. Furthermore, we also note that in this description, in spite of the property violation, the nominal construction is characterized by a large number of satisfied constraints. This characteristic allows one to introduce a crucial element for usage-based grammars: *compensation* phenomena between positive and negative information. We know that constraint violation can be an element of difficulty for human or automatic processing. The idea is that the violation of constraints can be compensated by the satisfaction of some others. For example, the violation of a precedence constraint can be compensated by the satisfaction of co-occurrence and dependency ones. PG offers the possibility to quantify these com-

pensation effects, on the basis of complexity evaluation (Blache *et al.* 2006; Blache 2011).

One important question when addressing the question of parsing is that of ambiguity. The problem is twofold: how to represent ambiguity and how to deal with it. With syntactic information being represented in terms of graphs, it is theoretically possible to represent different types of attachment at the same time. It is possible to have in the property graph two dependency relation of the same type, which are then mutually exclusive. The control of ambiguity resolution can be done classically, thanks to preference options implemented by property weights.

6 AN APPLICATION TO TREEBANKING

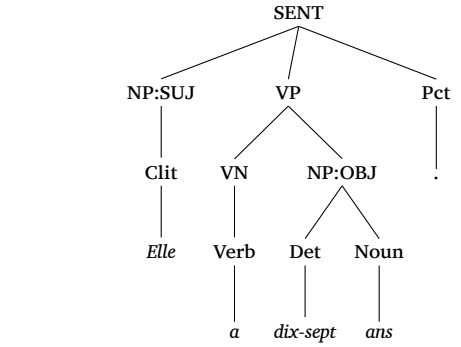
The use of treebanks offers a direct framework for the experimentation and the comparison of syntactic formalisms. Most of them have been developed using classical constituency or dependency-based representations. They have then to be adapted when studying more specific proposals. We present in this section an approach making it possible to extract properties from existing treebanks.

Most of the properties presented in this paper can be extracted automatically under some conditions, following a method presented in Blache *et al.* (2016). This is in particular the case with linearity, uniqueness, co-occurrence and exclusion, on which we focus in this section. The three first properties can be inferred fully automatically, the last one has to be filtered manually after its automatic extraction. The mechanism consists of two steps:

1. Extraction of the implicit context-free grammar
2. Generation of the properties from the CFG

In order to validate the approach, we have tested the method on several treebanks that offer different representations. We used first a set of four large constituency-based treebanks: the *Penn Treebank* (Marcus *et al.* 1994) itself, the *Chinese Treebank* (Xue *et al.* 2010), the *Arabic Treebank* (Maamouri *et al.* 2003), and the *French Treebank* (Abeillé *et al.* 2003). In a second stage, we have applied property extraction to the *Universal Dependencies Treebank* (Nivre *et al.* 2015). We offer a brief overview of this ongoing work presently.

6.1 *Extracting the implicit CFG from a treebank*



SENT → **NP:SUJ VP** *Pct*
NP:SUJ → *Clit*
VP → **VN NP:OBJ**
VN → *Verb*
NP:OBJ → *Det Noun*

Figure 9: Constituent tree and inferred CFG rules

The extraction of a context-free grammar (CFG) from a constituency treebank is based on a simple method described in Charniak (1996). Each internal node of a tree is converted into a rule in which the left-hand side (LHS) is the root and the right-hand side (RHS) is the sequence of constituents. The implicit grammar is composed of the complete set of rules. Figure 9 shows the syntactic tree associated to the French sentence *Elle a dix-sept ans* (“She is seventeen”), together with the corresponding CFG rules.

We applied a similar approach to dependency treebanks. In this case, a root node (LHS of a rule) is a head, while the constituents (RHS) form its list of dependents, following the projection order by which the head is added (encoded with the symbol *)

Figure 10 illustrates the dependency tree of the same sentence as in Figure 9 with the extracted CFG rules.

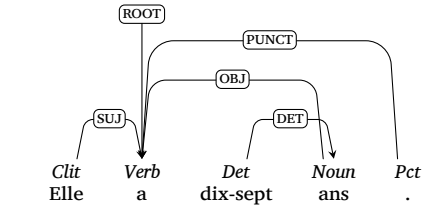
6.2 *Generating the properties*

Using these grammars, it is straightforward to extract the properties that we consider in this experiment, which we describe in 11.

6.3 *First results*

The treebanks and the generated resources are serialized as XML; this facilitates editing and visualization. We have developed software to

Figure 10: Dependency tree and inferred CFG rules



Linearity: the precedence table is built while verifying – for each category preceding another category into a construction (or a right-hand side) – whether this relation is valid throughout the set of constructions	$\forall\, rhs_m \in RHS(XP)$ **if** $((\exists\, (c_i, c_j) \in rhs_m \mid c_i \prec c_j)$ **and** $(\nexists\, rhs_n \in RHS(XP) \mid (c_i, c_j) \in rhs_n \wedge c_i \prec c_j))$ **then** add $prec(c_i, c_j)$
Uniqueness: the set of categories that can not be repeated in a right-hand side	$\forall\, rhs_m \in RHS(XP)$ $\forall\, (c_i, c_j) \in rhs_m$ **if** $c_i \neq c_j$ **then** add $uniq(c_i)$
Requirement: identification of two categories that co-occur systematically in all constructions of an *XP*	$\forall\, rhs_m \in RHS(XP)$ $bool \leftarrow ((c_i \in rhs_m) \wedge (c_j \in rhs_m))$ **if** *bool* **then** add $req(c_i, c_j)$
Exclusion: when two categories never co-occur in the entire set of constructions, they are supposed to be mutually exclusive; this is a strong interpretation, and it causes us to overgenerate such constraints, but it is the only way to identify this phenomenon automatically	$\forall\, rhs_m \in RHS(XP)$ $bool \leftarrow \neg((c_i \in rhs_m) \wedge (c_j \in rhs_m))$ **if** *bool* **then** add $excl(c_i, c_j)$

Figure 11: Implementation of the properties

view the different types of information: treebanks, tagset, extracted grammar, rules, and properties. Each type of information is associated with a link to a corresponding example in the treebank. Figure 6.3 illustrates some properties of a NP extracted from the *Chinese Treebank*.

In our interface, the left part of the window lists the set of cat-

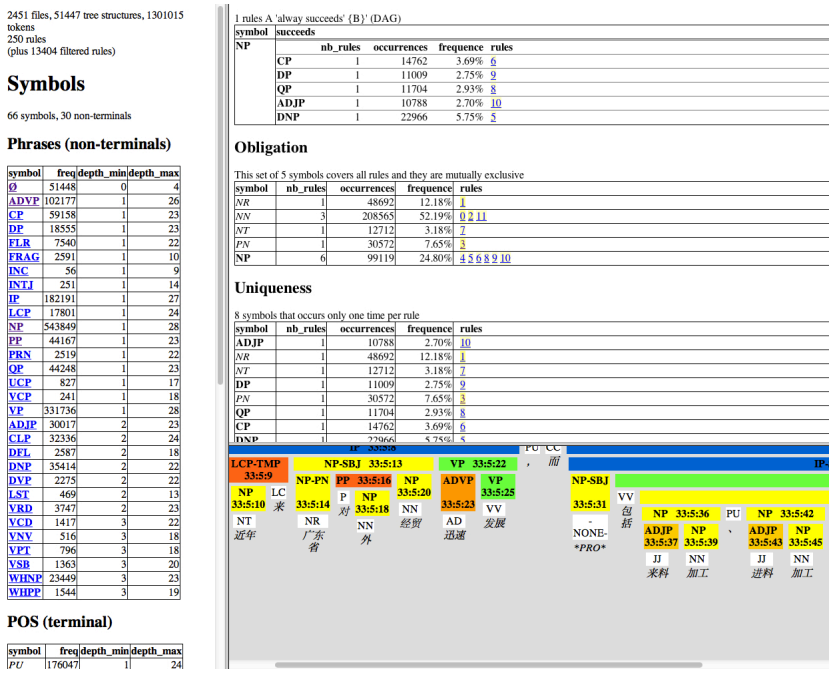


Figure 12: Properties from the *Chinese Treebank*

egories of the grammar, together with frequency information. Non-terminals are hyperlinked to their syntactic description (corresponding PS-rules and properties). This information is displayed in the top right of the window. Each property (in this example *Obligation* and *Uniqueness*) comes with the set of rules starting from which it has been generated. Links to the different occurrences of the corresponding trees in the treebank are also listed. The lower right side of the window contains a graphical representation of the tree structure.

7 CONCLUSION

Describing linguistic phenomena by means of atomic, low-level, and independent properties makes possible the joining of formal and descriptive linguistics. We are now in position to propose a general account of language processing, capable of integrating the description of local phenomena into a global architecture and making it possible to benefit from the best of the descriptive and formal approaches.

Usage-based theories describe language starting from the data, identifying different linguistic phenomena and gathering them into a set of descriptions. In the same perspective, *Construction Grammars* represent phenomena in terms of constructions. We have defined in this paper a framework, *Property Grammars* (PG), that represents all syntactic information with properties that can interact. *PG* has the advantage of being very flexible: properties are local and independent of each other, able to represent any local relation between words or categories. This characteristic solves the issue raised by Pullum and Scholz (2001), showing the limits of a *holistic* approach in grammars, in which all statements are dependent on each other (for example, a phrase-structure rule is not considered in and of itself, but rather as a step in the derivation process corresponding to a piece of the final syntactic tree). In *PG* all information is described by means of properties; these can remain local or can interact with other properties.

PG thus offers a formal framework for representing *constructions*, which are considered as a set of interacting properties. It also constitutes a homogeneous approach integrating both views of syntactic description: a usage-based one, aimed at describing specific phenomena; and a formal one that proposes a general organization in terms of grammars. Moreover, a syntactic description given in terms of properties makes it possible to describe ill-formed inputs: a property graph is not necessarily connected, and can even contain violated properties.

As a perspective, on top of being an adequate framework for a precise description of unrestricted linguistic material, *Property Grammars* also offer a framework for an evaluation of the quality of syntactic information associated to an input, based on an analysis of the syntactic description (the quantity and the importance of satisfied properties, their coverage, etc.). This also paves the way towards a cognitive account of language processing, capable of evaluating the relative importance of local phenomena within a general description.

REFERENCES

Bas Aarts (2004), Modelling Linguistic Gradience, *Studies in Language*, 28(1):1–49.

Anne Abeillé, Lionel Clément, and François Toussenel (2003), Building a Treebank for French, in A. Abeillé, editor, *Treebanks*, Kluwer, Dordrecht.

Rolf BACKOFEN, James ROGERS, and K. VIJAY-SHANKER (1995), A First-Order Axiomatization of the Theory of Finite Trees, *Journal of Logic, Language, and Information*, 4(1).

Philippe BLACHE (2000), Constraints, Linguistic Theories and Natural Language Processing, in D. CHRISTODOULAKIS, editor, *Natural Language Processing*, volume 1835 of *Lecture Notes in Artificial Intelligence (LNAI)*, Springer-Verlag.

Philippe BLACHE (2007), Model Theoretic Syntax is not Generative Enumerative Syntax with Constraints: at what Condition?, in *Proceedings of CSLP07*.

Philippe BLACHE (2011), Evaluating Language Complexity in Context: New Parameters for a Constraint-Based Model, in *CSLP-11, Workshop on Constraint Solving and Language Processing*.

Philippe BLACHE, Barbara HEMFORTH, and Stéphane RAUZY (2006), Acceptability Prediction by Means of Grammaticality Quantification, in *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pp. 57–64, Association for Computational Linguistics, Sydney, Australia, `http://www.aclweb.org/anthology/P/P06/P06-1008`.

Philippe BLACHE and Laurent PRÉVOT (2010), A Formal Scheme for Multimodal Grammars, in *Proceedings of COLING-2010*.

Philippe BLACHE and Jean-Philippe PROST (2005), Gradience, Constructions and Constraint Systems, in Henning CHRISTIANSEN, Peter Rossen SKADHAUGE, and Jorgen VILLADSEN, editors, *Constraint Solving and Language Processing - CSLP 2004*, volume 3438 of *Lecture Notes in Artificial Intelligence (LNAI)*, pp. 74–89, Springer, Roskilde, Denmark.

Philippe BLACHE and Jean-Philippe PROST (2014), Model-Theoretic Syntax: Property Grammar, Status and Directions, in P. BLACHE, H. CHRISTIANSEN, V. DAHL, D. DUCHIER, and J. VILLADSEN, editors, *Constraints and Language*, pp. 37–60, Cambridge Scholar Publishers.

Philippe BLACHE, S. RAUZY, and G. MONTCHEUIL (2016), MarsaGram: an Excursion in the Forests of Parsing Trees, in *Proceedings of LREC16*.

Patrick BLACKBURN and Wilfried MEYER-VIOL (1997), Modal Logic and Model-Theoretic Syntax, in M. DE RIJKE, editor, *Advances in Intensional Logic*, pp. 29–60, Kluwer.

Joan BRESNAN (2007), Is Syntactic Knowledge Probabilistic? Experiments with the English Dative Alternation, in Sam FEATHERSTON and Wolfgang STERNEFELD, editors, *Roots: Linguistics in Search of Its Evidential Base*, pp. 75–96, Mouton de Gruyter.

Joan BYBEE (2010), *Language, Usage and Cognition*, Cambridge University Press.

Eugene CHARNIAK (1996), Tree-bank Grammars, in *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pp. 1031–1036.

Thomas CORNELL and James ROGERS (2000), Model Theoretic Syntax, in Cheng L. LAI-SHEN and R. SYBESMA, editors, *The Glot International State of the Article Book I*, Holland Academic Graphics.

Denys DUCHIER, Thi-Bich-Hanh DAO, Yannick PARMENTIER, and Willy LESAINT (2010), Property Grammar Parsing Seen as a Constraint Optimization Problem, in *Proceedings of Formal Grammar 2010*, pp. 82–96.

Gisbert FANSELOW, Caroline FÉRY, Ralph VOGEL, and Matthias SCHLESEWSKY, editors (2005), *Gradience in Grammar: Generative Perspectives*, Oxford University Press, Oxford.

Fernanda FERREIRA and Nikole D. PATSON (2007), The 'Good Enough' Approach to Language Comprehension, *Language and Linguistics Compass*, 1(1).

Charles J. FILLMORE (1988), The Mechanisms of "Construction Grammar", in *Proceedings of the Fourteenth Annual Meeting of the Berkeley Linguistics Society*, pp. 35–55.

Gerald GAZDAR, Ewan KLEIN, Geoffrey PULLUM, and Ivan SAG (1985), *Generalized Phrase Structure Grammars*, Blackwell.

Adele E GOLDBERG (2003), Constructions: a New Theoretical Approach to Language, *Trends in Cognitive Sciences*, 7(5):219–224.

Adele E GOLDBERG (2009), The Nature of Generalization in Language, *Cognitive Linguistics*, 20(1):1–35.

Ray JACKENDOFF (2007), A Parallel Architecture Perspective on Language Processing, *Brain Research*, 1146(2-22).

Aravind JOSHI, Leon LEVY, and M. TAKAHASHI (1975), Tree Adjunct Grammars, *Journal Computer Systems Science*, 10(1).

Paul KAY and Charles FILLMORE (1999), Grammatical Constructions and Linguistic Generalizations: the *What's X doing Y?* Construction, *Language*, 75(1):1–33.

Knud LAMBRECHT (1995), Compositional vs. Constructional Meaning: The Case of French "comme-N", in M. SIMONS and T. GALLOWAY, editors, *SALT V*.

Ronald LANGACKER (1987), *Foundations of Cognitive Grammar, vol. 1 : Theoretical Prerequisites*, Stanford University Press.

Mohamed MAAMOURI, Ann BIES, Hubert JIN, and Tim BUCKWALTER (2003), Arabic Treebank, Technical report, Distributed by the Linguistic Data Consortium. LDC Catalog No.: LDC2003T06.

Mitchell P. MARCUS, Beatrice SANTORINI, and Mary Ann MAREINKIEWICZ (1994), Building a Large Annotated Corpus of English: the Penn Treebank, *Computational Linguistics*, 19(2):313–330.

Joakim Nivre, C. Bosco, J. Choi, M.-C. De Marneffe, T. Dozat, R. Farkas, J. Foster, F. Ginter, Y. Goldberg, J. Hajic, J. Kanerva, V. Laippala, A. Lenci, T. Lynn, C. Manning, R. Mcdonald, A. Missilä, S. Montemagni, S. Petrov, S. Pyysalo, N. Silveira, M. Simi, A. Smith, R. Tsarfaty, V. Vincze, and D. Zeman (2015), Universal Dependencies 1.0., Technical report, http://hdl.handle.net/11234/1-1464.

Carl Pollard and Ivan Sag (1994), *Head-driven Phrase Structure Grammars*, Center for the Study of Language and Information Publication (CSLI), Chicago University Press.

Alan Prince and Paul Smolensky (1993), *Optimality Theory: Constraint Interaction in Generative Grammars*, Technical Report RUCCS TR-2, Rutgers Optimality Archive 537.

Geoffrey Pullum and Barbara Scholz (2001), On the Distinction Between Model-Theoretic and Generative-Enumerative Syntactic Frameworks, in Philippe de Groote, Glyn Morrill, and Christian Rétoré, editors, *Logical Aspects of Computational Linguistics: 4th International Conference*, number 2099 in Lecture Notes in Artificial Intelligence, pp. 17–43, Springer Verlag, Berlin.

Ivan Sag (2012), Sign-Based Construction Grammar: An Informal Synopsis, in H. Boas and I. Sag, editors, *Sign-Based Construction Grammar*, pp. 69–200, CSLI.

Ivan Sag, Hans Boas, and Paul Kay (2012), Introducing Sign-Based Construction Grammar, in H. Boas and I. Sag, editors, *Sign-Based Construction Grammar*, pp. 1–30, CSLI.

Ivan Sag and T. Wasow (1999), *Syntactic Theory. A Formal Introduction*, CSLI.

Benjamin Swets, Timothy Desmet, Charles Clifton, and Fernanda Ferreira (2008), Underspecification of Syntactic Ambiguities: Evidence from Self-Paced Reading, *Memory and Cognition*, 36(1):201–216.

Lucien Tesnière (1959), *Éléments de syntaxe structurale ("Elements of structural syntax")*, Klincksieck.

Nianwen Xue, Zixin Jiang, Xiuhong Zhong, Martha Palmer, Fei Xia, Fu-Dong Chiou, and Meiyu Chang (2010), Chinese Treebank 7.0, Technical report, Distributed by the Linguistic Data Consortium. LDC Catalog No.: LDC2010T07.