Model-Theory of Property Grammars with Features

Denys Duchier, Thi-Bich-Hanh Dao, Yannick Parmentier

Model-Theory of Property Grammars with Features

Denys Duchier
LIFO - Université d'Orléans
Orléans, France

Thi-Bich-Hanh Dao
LIFO - Université d'Orléans
Orléans, France

Yannick Parmentier
LIFO - Université d'Orléans
Orléans, France

firstname.lastname@univ-orleans.fr

Abstract

In this paper, we present a model-theoretic description of Property Grammar (PG) with features. Our approach is based on previous work of Duchier *et al.* (2009), and extends it by giving a model-theoretic account of feature-based properties, which was lacking in the description of Duchier *et al.*

On top of providing a formal definition of the semantics of feature-based PG, this paper also discusses the various possible interpretations of features (*e.g.*, within the requirement and agreement properties), and show how these interpretations are represented in our framework. This work opens the way for a constraint-based implementation of a parser for PG with features.

1 Introduction

Many formal descriptions of natural language syntax rely on rewriting systems (*e.g.*, Tree-Adjoining Grammar). They specify how to generate the syntactic structures (hence the strings) belonging to a given (natural) language, by applying successive derivations (rewritings). Such syntactic descriptions are called generative-enumerative syntax. They provide a procedural view of language that naturally leads to the development of parsing algorithms. Nonetheless, as advocated by Pullum and Scholz (2001), such descriptions fail in accounting for ungrammatical sentences, such as those regularly produced by humans.

An alternative description of syntax, called model-theoretic syntax, focuses on syntactic properties that the structures (and strings) of a language are supposed to follow (*e.g.*, Property Grammar).In other terms, such descriptions do not give any information about how to produce these structures, they "simply" give a declarative specification of them. The grammar can thus be seen as a set of constraints, and syntactic structures as models satisfying these constraints. If one allows for the violation of some specific constraints, it then becomes possible to account for ungrammatical sentences, that is, to build *quasi-models* that are linguistically motivated and formally computed.[1]

Duchier *et al.* (2009) proposed a model-theoretic semantics of Property Grammar (PG), where models are trees labeled with syntactic categories. Their formalization was then converted into a constraint optimization problem to implement a parser for PG (Duchier et al., 2010). In their formalization, the authors did not account for features, thus omitted some properties such as *agreement*[2]. In this paper, we propose to fill this gap, by giving a model-theoretic semantics of feature-based PG. This semantics makes it possible to implement a constraint-based parser for the full class of PG in a similar way to that of Duchier *et al.* (2010).

The paper is organized as follows. In section 2, we introduce (feature-based) PG. Then, in section 3, we present our logical specification of PG. Finally, in section 4, we discuss the different interpretations of feature-based properties and their representations in our specification.

2 Property Grammar

As mentioned above, Property Grammar (Blache, 2000) is a formalism belonging to model-theoretic syntax. It describes the relations between syntactic constituents in terms of local constraints (the so-called *properties*). These properties come from linguistic observations (*e.g.*, order between words, co-occurrence, facultativity, *etc*). In

[1] This ability to describe ungrammatical sentences by means of violable constraints is also present in Optimality Theory (Prince and Smolensky, 1997).

[2] In her PhD thesis, Guénot (2006) proposed to replace dependency (as introduced in Blache (2000)) with a more specialized property named agreement.

Proceedings of the 12th International Conference on Parsing Technologies, pages 75–79,
October 5-7, 2011, Dublin City University.

a first approximation, these properties can be seen as local constraints on categories labeling syntactic trees. A property $A : \psi$ specifies, for a given node labeled A, the constraint ψ to be applied on the categories of A's daughter nodes (written B, C hereafter). ψ is one of the following:

Obligation	$A : \triangle B$	at least one B
Uniqueness	$A : B!$	at most one B
Linearity	$A : B \prec C$	B precedes C
Requirement	$A : B \Rightarrow C$	if $\exists\ B$, then $\exists\ C$
Exclusion	$A : B \not\Rightarrow C$	not both B and C
Constituency	$A : S?$	all children $\in S$
Agreement	$A : B \rightsquigarrow C$	feat. constraints

As mentioned above, in PG, properties are not restricted to syntactic categories, they actually handle feature structures. That is, the above properties do not only constrain atomic categories labeling syntactic nodes, but feature-based labels. In order to give a logical specification of PG, we first need to formally define these feature-based properties.

Let $\mathcal{F}$ be a finite set of features $\{f_1, \ldots, f_n\}$, where each feature f_i takes its values in a finite upper semilattice D_i. We write $\top_i$ for D_i's greatest element ($\top_i$ will be used in our specification to refer to features that do not apply within a given property). Since the syntactic category has a special status (it is mandatory within properties), we suppose that among the features f_i, there is one called `cat` to encode the category. Attribute-value matrices (AVM) of type $\mathcal{M} = [f_1{:}D_1, \ldots, f_n{:}D_n]$ also form a finite upper semilattice, equipped with the usual "product order" (written $\sqsubseteq$). This will allows us to compare AVM values. We write $\mathcal{M}_\downarrow$ for the minimal elements of $\mathcal{M}$. Within AVM values, we omit f_i if its value is $\top_i$. We also use AVM expressions, where features can be associated with variables (thus allowing for coreferences).[3] If S is an AVM expression, then S^v is the corresponding value obtained by replacing any occurrence of $f_i{:}X$ by $f_i{:}\top_i$ because f_i's value is constrained only by coreference equations. If S_0, S_1, S_2 are AVM expressions, then $E(S_0, S_1, S_2)$ is the set of coreference equations $(i, f) \doteq (j, g)$ for all $f{:}X$ in S_i and $g{:}X$ in S_j.

We can now define properties as being either of the form $S_0{:}r(S_1)$ or $S_0{:}r(S1, S2)$, where $S_0, S_1, S2$ are AVM expressions, and r one of the relations introduced above ($\triangle, \Rightarrow, \ldots$). That is, property literals are formed in one of the following ways (s_1 refers to a set of AVM expressions): $S_0 : \triangle S_1$, $S_0 : S_1!$, $S_0 : S_1 \prec S_2$, $S_0 : S_1 \Rightarrow S_2$, $S_0 : S_1 \not\Rightarrow S_2$, $S_0 : s_1?$, $S_0 : S_1 \rightsquigarrow S_2$. We write $\mathcal{P}$ for the set of all possible property literals over $\mathcal{F}$. Let $\mathcal{W}$ be a set of elements called *words*. A lexicon is a subset of $\mathcal{W} \times \mathcal{M}$ (that is, a lexicon maps words with AVM types). A *property grammar* G is a pair (P_G, L_G) where P_G is a set of properties (*i.e.*, a subset of $\mathcal{P}$) and L_G a lexicon.

When describing natural language, the properties of P_G are encapsulated within *linguistic constructions*, which typically describe syntactic constituents. As an illustration, consider Fig. 1 containing an extract of the PG for French of (Prost, 2008). In this figure, the `NP` construction describes noun phrases. It can be read as follows. In a noun phrase, there must be either a noun or a pronoun. If there is a determiner, a noun, a prepositional phrase or a pronoun, it must be unique. The determiner (if any) precedes the noun, pronoun, prepositional and adjective phrase (if any). A noun must come with a determiner, so does an adjective phrase with a noun. There cannot be both a noun and a pronoun. There must be gender and number agreements between the noun and the determiner.

3 Model-Theoretic Semantics

We will now extend the logical specification of PG of Duchier *et al.* (2009) using the above definition of feature-based properties.

Class of models. Following (Duchier et al., 2009), the strong semantics (*i.e.*, no property violation is allowed) of property grammars is given by interpretation over the class of syntactic trees τ. We write $\mathbb{N}_0$ for $\mathbb{N}\backslash\{0\}$. A *tree domain* D is a finite subset of $\mathbb{N}_0^*$ which is closed for prefixes and left-siblings; in other words, $\forall \pi, \pi' \in \mathbb{N}_0^*, \forall i, j \in \mathbb{N}_0$:

$$\pi\pi' \in D \quad \Rightarrow \quad \pi \in D$$
$$i < j \wedge \pi j \in D \quad \Rightarrow \quad \pi i \in D$$

A syntax tree $\tau = (D_\tau, L_\tau, R_\tau)$ consists of a tree domain D_τ, a labeling function $L_\tau : D_\tau \to \mathcal{M}_\downarrow$ assigning a *minimal* AVM value (w.r.t. $\sqsubseteq$) to each node, and a function $R_\tau : D_\tau \to \mathcal{W}^*$ assigning to each node its surface realization.

For convenience, we define the arity function $A_\tau : D_\tau \to \mathbb{N}$ as follows, $\forall \pi \in D_\tau$:

$$A_\tau(\pi) = \max\{0\} \cup \{i \in \mathbb{N}_0 \mid \pi i \in D_\tau\}$$

Instances. Following (Duchier et al., 2009), a property instance is a pair of a property and a tuple of nodes (paths) to which it is applied (see Fig. 2).

[3] In our PG specification, coreferences are only allowed within agreement properties.

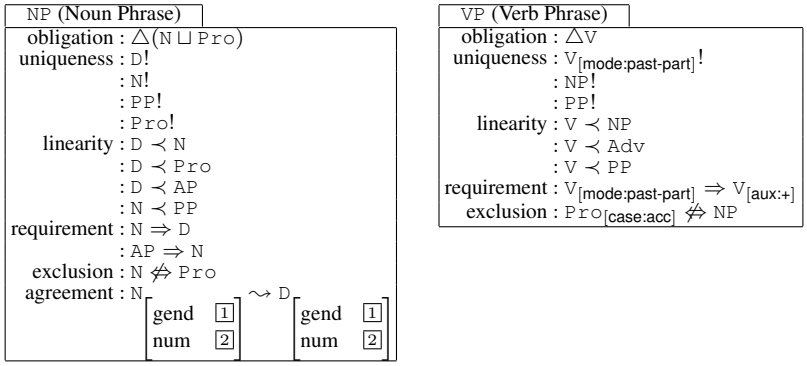

Figure 1: Extract of a Property Grammar for French

$$
\begin{aligned}
\mathcal{I}_\tau[\![G]\!] &= \cup\{\mathcal{I}_\tau[\![p]\!] \mid \forall p \in P_G\} \\
\mathcal{I}_\tau[\![S_0 : S_1 \prec S_2]\!] &= \{(S_0 : S_1 \prec S_2)@\langle \pi, \pi i, \pi j\rangle \mid \forall \pi, \pi i, \pi j \in D_\tau,\ i \neq j\} \\
\mathcal{I}_\tau[\![S_0 : \triangle S_1]\!] &= \{(S_0 : \triangle S_1)@\langle \pi\rangle \mid \forall \pi \in D_\tau\} \\
\mathcal{I}_\tau[\![S_0 : S_1!]\!] &= \{(S_0 : S_1!)@\langle \pi, \pi i, \pi j\rangle \mid \forall \pi, \pi i, \pi j \in D_\tau,\ i \neq j\} \\
\mathcal{I}_\tau[\![S_0 : S_1 \Rightarrow S_2]\!] &= \{(S_0 : S_1 \Rightarrow S_2)@\langle \pi, \pi i\rangle \mid \forall \pi, \pi i \in D_\tau\} \\
\mathcal{I}_\tau[\![S_0 : S_1 \not\Rightarrow S_2]\!] &= \{(S_0 : S_1 \not\Rightarrow S_2)@\langle \pi, \pi i, \pi j\rangle \mid \forall \pi, \pi i, \pi j \in D_\tau,\ i \neq j\} \\
\mathcal{I}_\tau[\![S_0 : s_1?]\!] &= \{(S_0 : s_1?)@\langle \pi, \pi i\rangle \mid \forall \pi, \pi i \in D_\tau\} \\
\mathcal{I}_\tau[\![S_0 : S_1 \rightsquigarrow S_2]\!] &= \{(S_0 : S_1 \rightsquigarrow S_2)@\langle \pi, \pi i, \pi j\rangle \mid \forall \pi, \pi i, \pi j \in D_\tau,\ i \neq j\}
\end{aligned}
$$

Figure 2: Property instances of a grammar G on a syntactic tree τ

Pertinence. Since we created instances of all properties in P_G for all nodes in τ, we must distinguish properties which are truly pertinent at a node from those which are not. For this purpose, we define the predicate P_τ over instances as in Fig. 3. This evaluation of property pertinence extends (Duchier et al., 2009) by comparing AVM expressions.

Satisfaction. When an instance is pertinent, it should also (preferably) be satisfied. For this purpose, we extend the predicate S_τ over instances of (Duchier et al., 2009) as in Fig. 4. For agreement, satisfaction relies on satisfaction of coreference equations, defined as follows. We say that the triple of values M_0, M_1, M_2 satisfies the coreference equations of expressions S_0, S_1, S_2, and write $M_0, M_1, M_2 \models E(S_0, S_1, S_2)$, iff $M_i.f{=}M_j.g$ for all $(i, f){\doteq}(j, g)$ in $E(S_0, S_1, S_2)$. As in (Duchier et al., 2009), we write $I^0_{G,\tau}$ for the set of pertinent instances, $I^+_{G,\tau}$ for its subset that is satisfied, and $I^-_{G,\tau}$ for its subset that is violated:

$$
\begin{aligned}
I^0_{G,\tau} &= \{r \in \mathcal{I}_\tau[\![G]\!] \mid P_\tau(r)\} \\
I^+_{G,\tau} &= \{r \in I^0_{G,\tau} \mid S_\tau(r)\} \\
I^-_{G,\tau} &= \{r \in I^0_{G,\tau} \mid \neg S_\tau(r)\}
\end{aligned}
$$

Admissibility. A syntax tree τ is admissible as a candidate model for grammar G iff it satisfies the projection property, *i.e.* $\forall \pi \in D_\tau$:

$$
\begin{aligned}
&A_\tau(\pi) = 0 \text{ (leaf node)} \ \Rightarrow \quad \langle L_\tau(\pi), R_\tau(\pi)\rangle \in L_G \\
&A_\tau(\pi) \neq 0 \text{ (inner node)} \Rightarrow R_\tau(\pi) = \textstyle\sum_{i=1}^{i=A_\tau(\pi)} R_\tau(\pi i)
\end{aligned}
$$

where $\sum$ represents the concatenation of sequences. In other words, leaf nodes must conform to the lexicon, and inner nodes pass upward the ordered realizations of their daughters.

Strong and loose models. The definition of strong and loose models stated by Duchier *et al.* (2009) are applied directly in this extension. A syntax tree τ is a strong model of a property grammar G iff it is admissible and $I^-_{G,\tau} = \emptyset$. A syntax tree τ is a loose model of G iff it is admissible and it maximizes the ratio $F_{G,\tau}$ defined as $F_{G,\tau} = I^+_{G,\tau}/I^0_{G,\tau}$.

4 About the Interpretation of Features

Let us now discuss the model-theoretic semantics of feature-based PG introduced above, by looking at some examples. In particular, let us see what is the meaning of features and how do these affect property pertinence and satisfaction. Let us first consider the requirement property of VP in Fig. 1.

$$
\begin{aligned}
P_\tau((S_0 : S_1 \prec S_2)@\langle\pi, \pi i, \pi j\rangle) &\equiv (L_\tau(\pi) \sqsubseteq S_0^v) \wedge (L_\tau(\pi i) \sqsubseteq S_1^v) \wedge (L_\tau(\pi j) \sqsubseteq S_2^v) \\
P_\tau((S_0 : \triangle S_1)@\langle\pi\rangle) &\equiv L_\tau(\pi) \sqsubseteq S_0^v \\
P_\tau((S_0 : S_1!)@\langle\pi, \pi i, \pi j\rangle) &\equiv (L_\tau(\pi) \sqsubseteq S_0^v) \wedge (L_\tau(\pi i) \sqsubseteq S_1^v) \wedge (L_\tau(\pi j) \sqsubseteq S_1^v) \\
P_\tau((S_0 : S_1 \Rightarrow S_2)@\langle\pi, \pi i\rangle) &\equiv (L_\tau(\pi) \sqsubseteq S_0^v) \wedge (L_\tau(\pi i) \sqsubseteq S_1^v) \\
P_\tau((S_0 : S_1 \not\Rightarrow S_2)@\langle\pi, \pi i, \pi j\rangle) &\equiv (L_\tau(\pi) \sqsubseteq S_0^v) \wedge ((L_\tau(\pi i) \sqsubseteq S_1^v) \vee (L_\tau(\pi j) \sqsubseteq S_2^v)) \\
P_\tau((S_0 : s_1?)@\langle\pi, \pi i\rangle) &\equiv L_\tau(\pi) \sqsubseteq S_0^v \\
P_\tau((S_0 : S_1 \rightsquigarrow S_2)@\langle\pi, \pi i, \pi j\rangle) &\equiv (L_\tau(\pi) \sqsubseteq S_0^v) \wedge (L_\tau(\pi i) \sqsubseteq S_1^v) \wedge (L_\tau(\pi j) \sqsubseteq S_2^v)
\end{aligned}
$$

Figure 3: Property pertinence on a syntactic tree τ

$$
\begin{aligned}
S_\tau((S_0 : S_1 \prec S_2)@\langle\pi, \pi i, \pi j\rangle) &\equiv i < j \\
S_\tau((S_0 : \triangle S_1)@\langle\pi\rangle) &\equiv \vee\{(L_\tau(\pi i) \sqsubseteq S_1^v) \mid 1 \leq i \leq A_\tau(\pi)\} \\
S_\tau((S_0 : S_1!)@\langle\pi, \pi i, \pi j\rangle) &\equiv i = j \\
S_\tau((S_0 : S_1 \Rightarrow S_2)@\langle\pi, \pi i\rangle) &\equiv \vee\{(L_\tau(\pi j) \sqsubseteq S_2^v) \mid 1 \leq j \leq A_\tau(\pi)\} \\
S_\tau((S_0 : S_1 \not\Rightarrow S_2)@\langle\pi, \pi i, \pi j\rangle) &\equiv (L_\tau(\pi i) \not\sqsubseteq S_1^v) \vee (L_\tau(\pi j) \not\sqsubseteq S_2^v) \\
S_\tau((S_0 : s_1?)@\langle\pi, \pi i\rangle) &\equiv L_\tau(\pi i) \sqsubseteq x \quad \text{for some } x \text{ in } s_1 \\
S_\tau((S_0 : S_1 \rightsquigarrow S_2)@\langle\pi, \pi i, \pi j\rangle) &\equiv L_\tau(\pi), L_\tau(\pi i), L_\tau(\pi j) \models E(S_0, S_1, S_2)
\end{aligned}
$$

Figure 4: Property satisfaction on a syntactic tree τ

This property states that, within a verb phrase, a past-participle requires an auxiliary. That is, in a model, a V node labeled with [mode:past-part] must come with a sister V node labeled with [aux:+]. As shown in Fig. 3, for this property to be *pertinent* for a couple of nodes $\langle\pi, \pi i\rangle$ with π the mother node of πi, these need to have category VP and V respectively, and πi needs to be labeled with [mode:past-part] ($L_\tau(\pi i) \sqsubseteq S_1^v$). For this property to be *satisfied*, a sister node of πi, say πj, needs to be labeled with [aux:+] ($L_\tau(\pi j) \sqsubseteq S_2^v$), as shown in Fig. 4. In other words, the `cat` and `mode` features affect pertinence and `aux` satisfaction.

Let us now consider the agreement property of `NP` in Fig. 1. Such a property ensures that, within a noun phrase, there are gender and number agreements between the determiner and the noun. For this property to be pertinent, we only consider the categories of the triple of nodes $\langle\pi, \pi i, \pi j\rangle$ (*i.e.*, omitting variables), see Fig. 3. For it to be satisfied, one need the coreferences to hold ($L_\tau(\pi), L_\tau(\pi i), L_\tau(\pi j) \models E(S_0, S_1, S_2)$). Here, all but the `cat` feature affect satisfaction.

Let us finally consider the following property:

$$
\texttt{VP} : \texttt{V}\begin{bmatrix} \text{mode} & \text{past-part} \\ \text{gend} & \boxed{1} \\ \text{num} & \boxed{2} \\ \text{pers} & \boxed{3} \end{bmatrix} \rightsquigarrow \texttt{Pro}\begin{bmatrix} \text{case} & \text{acc} \\ \text{gend} & \boxed{1} \\ \text{num} & \boxed{2} \\ \text{pers} & \boxed{3} \end{bmatrix}
$$

which constrains the gender, number and person agreements between a past-participle and an accusative pronoun (*e.g.*, ***je l'ai aimée*** / I loved her). For this property to be pertinent at a triple of nodes $\langle\pi, \pi i, \pi j\rangle$, one needs (a) π to have category VP ($L_\tau(\pi) \sqsubseteq S_0^v$), (b) πi to have category V and to be labeled with [mode:past-part] ($L_\tau(\pi i) \sqsubseteq S_1^v$), and (c) πj to have category Pro and to be labeled with [case:acc] ($L_\tau(\pi j) \sqsubseteq S_2^v$). For it to be satisfied, one needs the additional constraint that the coreferences hold ($L_\tau(\pi), L_\tau(\pi i), L_\tau(\pi j) \models E(S_0, S_1, S_2)$). In this example, the property mixes features affecting pertinence (`cat`, `mode`, `case`) and features affecting satisfaction (`gend`, `num`, `pers`). Thanks to our definition of AVM, and of $\sqsubseteq$ only checking for ground values, and $\models$ checking for coreferences, our representation supports the various interpretations of features.

5 Conclusion

We presented a model-theoretic semantics of PG that supports the various interpretations of features. Forthcoming work concerns the implementation of a PG parser by converting this semantics into a constraint optimization problem following Duchier *et al.* (2010). The motivation behind this is to provide the linguist with a device to implement her/his theories and check the logicial consequences of these on syntactic analyzes.

References

Philippe Blache. 2000. *Constraints, Linguistic Theories and Natural Language Processing*. Lecture Notes in Artificial Intelligence Vol. 1835. Springer-Verlag.

Denys Duchier, Jean-Philippe Prost, and Thi-Bich-Hanh Dao. 2009. A model-theoretic framework for grammaticality judgements. In *Conference on Formal Grammar (FG2009)*, Bordeaux, France, July.

Denys Duchier, Thi-Bich-Hanh Dao, Yannick Parmentier, and Willy Lesaint. 2010. Property Grammar Parsing Seen as a Constraint Optimization Problem. In *Proceedings of the 15th International Conference on Formal Grammar (FG 2010)*, Copenhagen, Denmark.

Marie-Laure Guénot. 2006. *Éléments de grammaire du français pour une théorie descriptive et formelle de la langue*. Ph.D. thesis, Université de Provence.

Alan Prince and Paul Smolensky. 1997. Optimality: From neural networks to universal grammar. *Science*, 275(5306):1604–1610, March.

Jean-Philippe Prost. 2008. *Modelling Syntactic Gradience with Loose Constraint-based Parsing*. Cotutelle Ph.D. Thesis, Macquarie University, Sydney, Australia, and Université de Provence, Aix-en-Provence, France, December.

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, pages 17–43, Berlin. Springer Verlag.