Hybridization and Treebank Enrichment with Constraint-Based Representations

Philippe Blache, Stéphane Rauzy

Hybridization and Treebank Enrichment with Constraint-Based Representations

Philippe Blache, Stéphane Rauzy

Aix-Marseille Université & CNRS
LPL, 5 Avenue Pasteur, 13100 Aix-en-Provence
{blache;rauzy}@lpl-aix.fr

Abstract

We present in this paper a method for hybridizing constituency treebanks with constraint-based descriptions and enrich them with an evaluation of sentence grammaticality. Such information is calculated thanks to a two-steps technique consisting in : (1) constraint grammar induction from the source treebank and (2) constraint evaluation for all sentences, on top of which a grammaticality index is calculated. This method is theoretically-neutral and language independent. Because of the precision of the encoded information, such enrichment is helpful in different perspectives, for example when designing psycholinguistics experiments such as comprehension or reading difficulty.

1. Introduction

Besides syntactic description and NLP tools development, treebanks also play an important role in more specific perspectives such as ambiguity resolution (Koller and Thater, 2010), discourse analysis (Webber, 2009) or spoken language description (Wouden et al., 2002). More and more frequently, treebanks are used for interdisciplinary studies, in particular in psycholinguistics, bridging the gap between experimental studies (e.g. eye-tracking) and linguistic resource description (Keller, 2004; Demberg and Keller, 2008; Tomanek et al., 2010).

These different works share the fact that they rely on different types of information (morphology, syntax, semantics, or prosody), encoded at different levels (word forms, categories, sentences) and possibly incomplete. The problem is that a classical representation in terms of constituency hierarchy is not the right level of description for these new tasks (parsing spoken languages, building difficulty models, etc.), in particular because failing in representing partial structures, ill-formed constructions, etc.

Developing treebanks with a finer granularity of syntactic description is then necessary. Constraint-based representations are well equipped in such perspective: constraints can be very precise, possibly not connected to each others and may bring together different levels of representation. Treebanks bearing such precise information would then be of great help. Unfortunately, constraint parsers are of great complexity and often suffer from over-generation.

We propose in this paper to bypass this problem: instead of building constraint-based treebanks from scratch, we propose an hybridization technique building the constraint-based representation starting from a constituency-based one. Knowing syntactic structure (the original tree) dramatically reduces the search space required when building the constraint representation. The interest is that this technique is entirely automatic. It consists first in inducing a constraint grammar from the source treebank and then to build the constraint-based representation thanks to a set of constraint solvers exploiting this grammar. This technique, on top of being efficient, is *generic*: it is independent from any linguistic formalism as well as from the language: it can be applied to any constituency treebank. Moreover, other kinds of information derived from the constraint-based representation, such as grammaticality level, can also enrich the structure, opening the way to new applications, for example in psycholinguistics.

After a brief presentation of the main characteristics of the constraint-based representation, the grammar induction process is described. It consists in gathering all the possible realizations of the different categories of the corpus. The result is a large context-free grammar on top of which the constraint grammar is generated. The third section presents the hybridization mechanism which build the constraint treebank starting from the constituency. The application of this process to the *French Treebank* (Abeillé, 2003) is described and some results are discussed. The last section describes a treebank enrichment: the evaluation of the grammaticality completes the description of the different categories realized in the treebank.

2. Constraint-Based Syntactic Representation

Phrase-structure representations use a unique explicit relation, hierarchy, that encode constituency information. All other information such as linear order, headedness, dependency, etc. are implicit. On the opposite, constraint-based representations encode explicitly all these relations, making it possible to verify their level of satisfaction and to evaluate precisely the structure grammaticality. Such syntactic representation syntax has been experimented in different projects (Blache, 2005). In terms of parsing, the technique consists in considering relations as *constraints*, satisfaction being the core of the parsing process. We propose to represent syntactic information by means of six different types of constraints that describe phrase-level constituents:

- *Linearity*: linear precedence relations between the constituents of a phrase
- *Obligation*: possible heads of a phrase

- *Dependency*: dependency relations between the constituents
- *Uniqueness*: constituents that cannot be repeated in a phrase
- *Requirement*: mandatory cooccurrence between categories
- *Exclusion*: cooccurrence restriction between categories

A complete grammar can be represented by means of such constraints: each phrase-level category is described by a set of constraints between constituents. Parsing an input comes to evaluating for each category the set of constraints that describes it. Concretely, for a given category and a given set of constituents, the mechanism consists in satisfying all constraints, for example verifying that within the set of constituents, the head is realized (*obligation*) or that *linearity* constraints hold. At the end of the evaluation process, the parser has built a set of evaluated constraints. As a consequence, parsing two different realizations of a same category (in other words two different sets of constituents) will result in different sets of evaluated constraints (that can possibly be violated). The final set of evaluated constraints for a given input (also called a *characterization*) forms a description graph, as illustrated in figure 1 (we will use in the remaining of the paper examples from the *French Treebank*).
We can see in this graph how constraints represent explicitly the different kinds of syntactic information. In particular, it illustrates the fact that the number of evaluated constraints can be very different from one constituent to another. This property, together with the fact that constraints can be violated, is of central interest because describing precisely the syntactic relations, not only in terms of hierarchy. Such a rich representation makes it possible to *quantify* these two aspects of the syntactic structure: density and quality. We describe in the following a method for enriching constituency treebanks with such information, independently form the language.

3. Constraint Grammar Induction from Constituency Treebanks

Even if some effort have been done in terms of homogenizing the different syntactic annotation schemes (Abeillé, 2003), the encoded information can be very different from one treebank to another. For example functional annotation can be more or less precise or dependent from the chosen formalism (compare for example (Bies et al., 1995), (Abeillé et al., 2003), (Telljohann et al., 2004) or (Böhmová et al., 2003)). Still, constituency treebanks contains by definition, on top of the morpho-syntactic level, the hierarchical structure. We present in this section a procedure acquiring automatically the different constraints corresponding to the implicit grammar of the treebank. The mechanism is based on the analysis of all possible constructions of the different categories which corresponds, in terms of context-free grammars, to the set of the possible right-hand sides of non-terminal categories.

Calculating the construction sets consists for all non-terminal categories *XP* in traversing the treebank and identifying its daughters. The result, noted $RHS(XP)$, is made of ordered subsets of categories.
Let's note in the following $\prec$ the precedence relation between two categories into a construction. The set of constraints is then calculated for each non terminal category *XP* as follows:

- *Constituency*: for each non-terminal category *XP*, its set of constituents, noted *const(XP)*, is the set of categories participating to the constructions in *RHS(XP)*. Let's note that the tagset used in the constraint grammar to be built can be adapted at this stage: categories can be, according to the needs, more precise or at the opposite more general than that of the initial tagset.

- *Linearity*: the precedence table is built in 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 cannot 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 cooccur 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 in exclusion. This is a strong interpretation, that leads to over-generate the number of such constraints. However, it is the only way to identify it 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 $excl(c_i, c_j)$

Besides this direct acquisition from the treebanks, two other constraint types require explicit formulation:

- *Obligation*: the heads of a phrase. Identified as the minimal set of compulsory constituents. Usually, this set is identified by means of specific sets of rules (cf. (Lin, 1998)). Note that multiple heads are allowed.

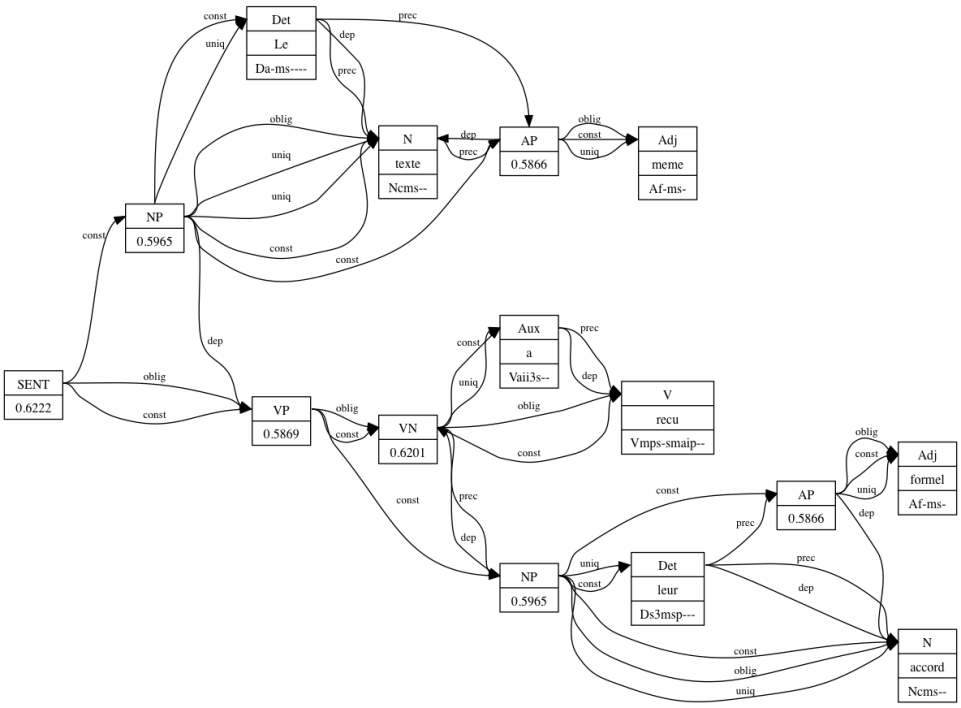

Figure 1: Description graph of the sentence"*The text itself had received their formal agreement*"

- *Dependency*: this constraint typically encodes grammatical relations. When present, they usually encodes complementation, modification or specification. Automatic acquisition in this case is dependent on the treebank and the way such relations are encoded.

The example figure 2 illustrates the acquisition process for the adjectival phrase. We work in this experiment on a subset of the *French Treebank* (Abeillé et al., 2003), made of 134,445 words, manually corrected. The figure 2 presents the list of *AP* constructions in this treebank. The total number of *rhs* is indicated in the right column, the total number of *AP* in the sub-corpus being 2,657. There are 56 different types of *rhs* (for sake of space, only the a subpart is mentioned here). We can note that the 5 most frequent *rhs* types represent 95% of the possible constructions (providing important indication when building probabilistic grammars). It is also interesting to note that in 84% of the cases, the *AP* is formed by a unique constituent, its head. On the opposite, complex constructions, made of more than two constituents are very rare.

In this encoding, without entering into the detail of the tagset, the qualificative adjective is encoded '*Af*', the others being noted '*A-*'. Coordination is indicated at the category level and treated as a feature (for example *NP:COORD*). Finally, punctuation marks correspond to the category *W*.

The extraction of the *AP* constraint system is presented figure 3. The first remark is that this constraint-based representation is very compact. This observation illustrates the fact that, as was observed with the ID/LP formalism in GPSG (Gazdar et al., 1985), a separate encoding of different types of syntactic information makes it possible to factorize a CFG rule-based representation. In fact, our approach systematizes the ID/LP one: we distinguish 6 different types of information where ID/LP takes into account 2 of them. One can see that this representation steps over a level of generalization, thanks to the factorization.

Constituents	Occ.	Constituents	Occ.
Af	1930	A- Ssub	1
A-	302	AdP Af Wm PP Wm PP	1
AdP Af	159	AdP Af Wm NP Wm Ssub	1
Af PP	63	Af Wm Ssub Wm PP	1
Af VPinf	19	AdP Wm AdP Af PP	1
AP Af	17	Af AdP	1
AdP Af Ssub	13	AdP Af AdP	1
AdP Af PP	8	AP Wm Cc AP Wm	1
A- PP	7	NP Af	1
Af Ssub	6	PP Af	1
AP A-	5	Af NP	1
AdP A-	4	AP AdP	1
AdP Af VPinf	3	AdP Wq Af Wq	1
Af PP:COORD	3	A- Wm A-	1
Af NP:COORD	2	AdP Af NP	1
AdP AdP Af	2	AdP Af NP PP VPinf	1
Af PP PP	2	Af Wm NP Wm PP	1

Figure 2: AP realizations in the FTB

Another important remark concerns frequency: it is not necessary to take into account all constructions under a certain frequency threshold. Generally speaking, constructions with at least 2 realizations in the treebank are reasonably representative. By another way, in case of conflict between two constraints, the most frequent one is chosen. It is the case in this example with the linearity constraint between AdP and A: all realizations but one satisfy the constraint $AdP \prec A$. We keep then this constraint in the

const	{AdP, A, VPinf, PP, Ssub, AP, NP}
lin	A $\prec$ {VPinf, Ssub, PP, NP, AP} AdP $\prec$ {A, Ssub, PP} AP $\prec$ {A, AdP} PP $\prec$ {Ssub}
dep	{AdP, VPinf, PP, Ssub, NP} $\rightsquigarrow$ A
uniq	{A, VPinf, Ssub}
oblig	{A}
excl	VPinf $\otimes$ {PP, Ssub}

Figure 3: AP properties

final description.
In our experiment, as presented figure 4, the grammar contains a total of 493 constraints extracted from the treebank (taking into account only constructions with more than one realization). There are important differences between the constructions with respect to the number of constraints as well as their distribution. The *NP*, because of the great variability of its possible realizations, requires a high number of constraints. As for the constraint types, linearity is the most frequent, illustrating the fact that word order in French is highly constrained. It is also interesting to note that the size of the constraint set is not directly dependent from the number of constituents (compare *AdP, Ssub* and *VP*).
The following example, taken from the FTB, illustrates the evaluation of the constraint grammar for the AP "*plus économique que politique* (more economical than politic)":

const	{AdP, A, Ssub}
lin	A $\prec$ Ssub AdP $\prec$ A
dep	{AdP, Ssub} $\rightsquigarrow$ A
uniq	{A, Ssub}
oblig	{A}
excl	VPinf $\otimes$ Ssub

This example shows the interest of a constraint-based description which provides many precise information not directly accessible in a constituency-based representation. We will see in the last section of the paper the importance of such description for different applications.

4. Enriching Treebanks with a Constraint-Based Description

Our treebank enrichment consists in building an hybrid syntactic representation, one (the original) being purely constituency-based, the second being constraint-based. Generally, building a constraint-based representation as described in section 2 is a computationally complex process, highly non-deterministic, in particular due to constraint relaxation. However, the task in our case is to enrich an existing treebank. The problem consists to evaluate the constraint system for each node of the original tree instead of building an entire graph description starting from the initial set of words. Concretely, the process comes to traverse for each sentence its original tree. At each node, the constraint set describing the corresponding category is evaluated thanks to different constraint solvers, presented in the following in terms of set operations.

We note $|E|$ the cardinality of the set E; $\mathcal{C}$ the ordered set of constituents of the category taken into account; $\mathcal{C}_{i..j}$ the sublist of $\mathcal{C}$ between positions i and j; c_i a constituent of $\mathcal{C}$ at position i; n the number of different constituents belonging to $\mathcal{C}$.
Constraints are of two types: those specifying a set (obligation, uniqueness) and those defining relations between sets of categories. In the first case, we note $\mathcal{S}_{cx}$ the set of categories specified by the constraint of type cx. In the second case, we note $\mathcal{L}_{cx}$ and $\mathcal{R}_{cx}$ respectively the left and right parts of the constraint cx.

- *Obligation*: this operation consists in verifying the presence of one of the obligatory categories specified in the obligation constraints. In terms of sets, this means that the intersection between the set of realized constituents $\mathcal{C}$ and the set of categories specified in the obligation constraint:

  $|\mathcal{C} \cap \mathcal{S}_{oblig}| > 0$

- *Linearity*: the verification of the linear precedence constraints consists in verifying that when a category belonging to a left-hand side of a linearity constraint is realized, then no category of the right-hand side can be realized in the sublist of the current category list of constituents preceding it:

  $\forall c_i \in \mathcal{L}_{lin},\ \nexists c_j \in \mathcal{R}_{lin}$ such that $c_j \in \mathcal{C}_{1..k} \wedge c_i \in \mathcal{C}_{k+1..n}$

- *Uniqueness*: this constraints verifies that the specified categories are not repeated, which means that the intersection of the category and the set of realized constituents is not greater than one:

  $\forall c_i \in \mathcal{S}_{uniq}, |c_i \cap \mathcal{C}| \leqslant 1$

- *Requirement*: when one category of a LHS of this constraint is realized, then one of its RHS should too:

  $\forall c_i \in \mathcal{L}_{req} \wedge c_j \in \mathcal{R}_{req},\ c_i \in \mathcal{C} \Rightarrow c_j \in \mathcal{C}$

- *Exclusion*: when one category of a LHS of this constraint is realized, then no category of its RHS should be present:

  $\forall c_i \in \mathcal{L}_{req} \wedge c_j \in \mathcal{R}_{req},\ c_i \in \mathcal{C} \Rightarrow c_j \notin \mathcal{C}$

Thanks to these mechanisms, a constraint-based annotation can be built on top of the constituency structure. Concretely, this mechanism makes it possible to build a parallel treebank as well as an hybrid one. In the first case, two different sets of syntactic annotations are built: one representing the constituency representation, the other the constraint-based one, both of them being aligned at the word level. Another type of representation of the treebank consists in enriching the constituency structure: the description of non-terminal categories (the nodes of the tree) is completed by a set of relation between its constituents (their daughters). The example in figure 6 illustrates this second approach.
Enriching a constituency tree consists in calculating the set of constraints of the FTB constraint grammar that are satisfied for each node of the tree. As presented above, the result consists in a characterization, which is the set of constraints

	AdP	AP	NP	PP	SENT	Sint	Srel	Ssub	VN	VNinf	VNpart	VP	VPinf	VPpart	Total
const	10	7	13	7	8	8	7	10	6	7	7	10	9	8	115
dep	5	6	18	5	3				5	5	6	8			59
exc	1	2	44		2	6	3	3							61
req			6							4	4				14
lin	18	10	36	6	5	4	7	14	11	6	7	24	13	7	165
oblig	1	1	4	1	1	1	1	1	1	3	2	1	1	1	20
uniq	4	3	10	3	3	4	4	1	2	4	5	3	7	6	59
	39	22	131	22	22	23	22	29	25	29	31	46	30	22	493

Figure 4: Distribution of the constraints in the FTB-acquired grammar

```
<category label="SENT" sample_index="0" sentence_index="0:16" node_index="0:16:0">
   <category label="NP" features="NP:SUJ" node_index="0:16:1">
      <category label="Det" features="Da-ms----" node_index="0:16:2" form="Le" lemma="le"/>
      <category label="Noun" features="Ncms--" node_index="0:16:3" form="texte" lemma="texte"/>
      <category label="AP" features="AP" node_index="0:16:4">
         <category label="Adj" features="Af-ms-" node_index="0:16:5" form="mme" lemma="mme"/>
      </category>
   </category>
   <category label="VP" features="VP" node_index="0:16:6">
      <category label="VN" features="VN" node_index="0:16:7">
         <category label="Aux" features="Vaii3s--" node_index="0:16:8" form="avait" lemma="avoir"/>
         <category label="Verb" features="Vmps-smaip--" node_index="0:16:9" form="reu" lemma="recevoir"/>
      </category>
      <category label="NP" features="NP:OBJ" node_index="0:16:10">
      <category label="Det" features="Ds3msp---" node_index="0:16:11" form="leur" lemma="leur"/>
      <category label="Noun" features="Ncms--" node_index="0:16:12" form="accord" lemma="accord"/>
      <category label="AP" features="AP" node_index="0:16:13">
         <category label="Adj" features="Af-ms-" node_index="0:16:14" form="formel" lemma="formel"/>
      </category>
   </category>
</category>
<category label="Pct" features="Wd" node_index="0:16:15" form="." lemma="."/>
```

Figure 5: Example of a tree in the FrenchTreeBank

that can be evaluated for this specific realization. The example figure 6 shows the enrichment for the subject *NP* . In this encoding, the characterization is a set of constraints encoded by the elements $<$ `property` $>$. In this representation, all constraints are encoded as relations between two nodes. In the case of set constraints (for example uniqueness constraints that specifies the categories that cannot be repeated), the corresponding evaluated constraint is encoded as a relation between the node and the category. Each element contains 4 attributes: the type of the corresponding constraint, its source and target and the result of its satisfaction (true or false). In this example, the characterization represents linearity *Det* $\prec$ *N, Det* $\prec$ *AP*, dependencies between *Det, AP* and *N*, etc. As mentioned above, the interest of such representation lies in the fact that it offers a precise description of the different syntactic properties.

Moreover, each constraint is evaluated independently and can be satisfied or possibly violated. This means that such representation can also encode non-grammatical, partial or ill-formed constructions. Let's imagine for example that in our example, the noun would precede the determiner. The only difference in the characterization would then be the value of the attribute `sat`, set to *false* in the corresponding constraint:

```
<prop type="lin" srce="0:16:2" tget="0:16:3" sat="f"/>
```

This characteristic is interesting when describing specific data such as second-language acquisition, spoken language, pathological productions, etc.

Table 1 recaps some figures of the FTB sub-treebank described above and the application to the enrichment procedure. The first table indicates the number of categories observed in the treebank. As already underlined, *NP* is by far the most frequent category, followed by *PP*. Moreover, as mentioned above, *NP* has a great variability of realizations, in comparison to other categories, which has also consequences on the distribution of the constraint types. The second table indicates the total number of evaluated constraints for the treebank, indicated per type. In this case too, we can observe a great difference in the distribution at a first glance. However, this aspect mainly comes from the frequency of the *NP* that uses a lot of exclusion and uniqueness constraints.

SENT	1 471
NP	8 127
AP	2 632
VP	2 550
VN	2 628
PP	4 124
AdP	1 733
Srel	508
Ssub	476
Sint	352
VPinf	917
VPpart	618
VNinf	863
VNpart	616

lin	27 367
obl	32 602
dep	21 971
exc	89 293
req	11 022
uni	38 007

Table 1: Number of constraints by category and type

It is interesting to have a closer look at the distribution of the different evaluated constraints by category. The results for the FTB sub-treebank are presented in figure 7. Note that the number of constraints in the grammar is not directly correlated with the number of evaluated constraint (which is expected) but also to the frequency of the category: *PP* is a very frequent category with an

```
<category label="NP" features="NP:SUJ" node_index="0:16:1">
   <category label="Det" features="Da-ms----" node_index="0:16:2" form="Le" lemma="le"/>
   <category label="Noun" features="Ncms--" node_index="0:16:3" form="texte" lemma="texte"/>
   <category label="AP" features="AP" node_index="0:16:4">
      <category label="Adj" features="Af-ms-" node_index="0:16:5" form="mme" lemma="mme"/>
   </category>
   <characterization>
      <property type="lin" source="0:16:2" target="0:16:3" sat="p"/>
      <property type="lin" source="0:16:2" target="0:16:4" sat="p"/>
      <property type="req" source="0:16:3" target="0:16:2" sat="p"/>
      <property type="dep" source="0:16:2" target="0:16:3" sat="p"/>
      <property type="dep" source="0:16:4" target="0:16:3" sat="p"/>
      <property type="oblig" source="0:16:1" target="0:16:3" sat="p"/>
      <property type="uniq" source="0:16:1" target="0:16:2" sat="p"/>
   </characterization>
</category>
```

Figure 6: Example of a FTB enriched tree for the NP

average number of constraints in the grammar, but represents only 7% of the number of evaluated constraints. This figure is to be compared to that of the *SENT* category, which represents 31% of the total. However, and this very clear when comparing with the *NP*, the frequency of exclusion and uniqueness constraints, which is highly variable, mainly explains this observation.

More interestingly, the respective role of constraint types for each category can be measured when putting together these different information. In particular, the frequency of the constraint type in the treebank for a given category has to be balanced with its frequency in the grammar: a constraint type very frequent in the treebank and with few instance in the grammar will play a more important role in the syntactic description. In other words, the respective weights of constraint types for each category can be automatically evaluated thanks to these figures. We propose the following formula:

$$weight(cx) = \frac{freq_{\text{tbank}}(cx)}{freq_{\text{gram}}(cx)} \quad (1)$$

Figure 8 presents the application of this measure to the FTB. We can observe for example that for the *NP*, even though the evaluation of the exclusion constraint is much more frequent than others, its relative importance is balanced with respect to others types such as requirement or linearity, which is expected from a syntactic point of view.

5. Enriching Treebanks with Grammaticality Information

We present in this section the application of a grammaticality evaluation technique making use of to the constraint-based enrichment presented in the previous section. Such information, as mentioned in the introduction, can be of great help in particular for psycholinguistics experiments.

One of the characteristics of our constraint-based representation is that it is possible to quantify the number of constraints and their relative importance. This evaluation have been described in (Blache et al., 2006) and relies on the study of the set of evaluated constraints. The method proposes different scoring terms on top of which a grammaticality index is calculated.

We note in the following N_c^+ the amount of constraints satisfied by the constituent c, N_c^- the constraints violated, N_c^+, and E_c the total number of constraints that received an evaluation (i.e. $N_c^+ + N_c^-$). We note T_c the total amount of constraints (evaluated or not) specifying the category c. Constraints being weighted, we note W_c^+ (respectively W_c^-) the sum of the weights assigned to the constraints satisfied (respectively violated) by the constituent c. The different terms are calculated as follows:

- *Satisfaction/Violation Ratio*: SR_c (resp. VR_c) is the number of satisfied constraints (resp. violated) divided by the number of evaluated constraints:
$$SR_c = \frac{N_c^+}{E_c} \qquad VR_c = \frac{N_c^-}{E_c}$$
- *Completeness Index*: number of evaluated constraints divided by the total number of constraints for the category c: $CI_c = \frac{E_c}{T_c}$
- *Quality Index*: distribution of the weights of satisfied and violated constraints: $QI_c = \frac{W^+ - W^-}{W^+ + W^-}$
- *Precision Index*: The *Index of Precision* for the constituent c is defined as the following ratio: $PI_c = k \cdot QI_c + l \cdot SR_c + m \cdot CI_c$
  These *adjustment coefficients* (k, l, m) are used as variable parameters for tuning up the model.

Finally, the global *Grammaticality Index* (GI_c) is a function of the previous indexes. It is defined recursively as follows, where c is a constituent and c_i is a nested constituent of c:

$$GI_c = PI_c \cdot \overline{GI_{c_i}} = PI_c \cdot \frac{\sum_{i=1}^{Z_c} GI_{c_i}}{Z_c}$$

Besides the constraint description, each node of the treebank can also be enriched with its grammaticality evaluation as presented in the figure 9.

6. Conclusion

We have presented in this paper a method for enriching constituency treebanks with a constraint-based representation, which offers the interest to propose a very precise representation of syntactic information on top of which automatic grammaticality evaluation can be calculated. Such constraint-based representation has been shown to be adapted to the description of non-canonical input (for example spoken language).

This technique is generic in the sense that it is independent from the source formalism and can be applied to any constituency-based treebank. Grammar induction only depends on the analysis

	AdP	AP	NP	PP	SENT	Sint	Srel	Ssub	VN	Vninf	Vnpart	VP	Vpinf	Vppart	
dep	41	320	7 730	3 871	7 114	0	0	8	854	143	24	1 866	0	0	21 971
exc	4	157	57 232	0	31 820	54	18	8	0	0	0	0	0	0	89 293
req	0	0	6 451	0	4 489	0	0	0	0	72	10	0	0	0	11 022
lin	9	360	9 336	3 895	8 960	0	2	475	965	143	24	2 329	709	160	27 367
obl	1 732	2 562	8 010	3 942	7 073	270	486	463	2 620	863	616	2 523	838	604	32 602
uniq	1 733	2 589	11 586	3 942	10 385	286	506	463	680	144	640	2 642	1 614	797	38 007
	3519	5 988	100 345	15 650	69 841	610	1 012	1 417	5 119	1 365	1 314	9 360	3 161	1 561	22 0262

Figure 7: Distribution of the evaluated constraints in the FTB sub-treebank

	AdP	AP	NP	PP	SENT	Sint	Srel	Ssub	VN	Vninf	Vnpart	VP	Vpinf	Vppart
dep	0.0676	0.2271	0.5050	0.7420	0.4753	-	-	0.1129	0.6340	0.4610	0.0731	0.8971	-	-
excl	0.0330	0.2229	1.5296	-	3.1892	0.2361	0.0949	0.0376	-	-	-	-	-	-
req	-	-	1.2643	-	-	-	-	-	-	0.2901	0.0457	-	-	-
lin	0.0041	0.1460	0.3050	0.6222	0.3592	-	0.0045	0.4789	0.3256	0.3841	0.0626	0.3732	0.3623	0.2196
oblig	14.2734	7.2735	2.3548	3.7783	1.4178	7.0820	7.6838	6.5349	9.7246	4.6364	5.6256	9.7038	5.5672	5.8040
unic	3.5704	2.4501	1.3624	1.2594	0.6939	1.5003	1.6000	6.5349	1.2620	0.5802	2.3379	3.3872	1.5318	1.0941

Figure 8: Weights of the constraint types in the FTB

of the realizations of the different constituents. As a consequence, the different constraints can be generated directly from the original constituency representation. Starting from such grammar, the annotation process itself is entirely automatic. This ensure the consistency of the encoding as well as the reusability of the process. Moreover, it is language independent, the constraint grammar being automatically acquired from the original treebank.

Several works can take advantage from this kind of resources. In particular, grammaticality evaluation makes it possible to compare the different realizations of a same construction as well as quantify its "prototypicity": a high grammatical score usually comes from the fact that the corresponding construction contains redundant information, reinforcing its categorization. This kind of information is useful for example in discourse relations identification, speaker involvement evaluation, etc.

As an example, two on-going experiments rely on such treebanks. First, eye-tracking data are on the process to be acquired for the *French Treebank*, making it possible to look for correlation between grammaticality and difficulty. A second project concerns cross-linguistic study of constraint-based representation, applied to English, Chinese and Arabic. The idea consists here in acquiring a constraint grammar for each of these languages and to compare the description (and the grammaticality) of a same construction through languages.

7. References

A. Abeillé, L. Clément, and Toussenel F. 2003. Building a treebank for french. In A. Abeillé, editor, *Treebanks*, Kluwer, Dordrecht.

A. Abeillé, editor. 2003. *Treebanks: Building and Using Parsed Corpora*. Dordrecht: Kluwer.

A. Bies, M. Ferguson, K. Katz, and R. MacIntyre. 1995. Bracketing guidelines for treebank ii style. Department of Computer and Information Science, University of Pennsylvania.

P. Blache, B. Hemforth, and S. Rauzy. 2006. Acceptability prediction by means of grammaticality quantification. In *ACL-44: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, July.

P. Blache. 2005. Property grammars: A fully constraint-based theory. In H. Christiansen et al., editor, *Constraint Solving and Language Processing*, volume LNAI 3438. Springer.

A. Böhmová, J. Hajič, E. Hajičová, and B. Vidová-Hladká. 2003. The prague dependency treebank: A three-level annotation scenario. In A. Abeillé, editor, *Treebanks: Building and Using Parsed Corpora*, pages 103–127. Kluwer.

V. Demberg and F. Keller. 2008. Data from eye-tracking corpora as evidence for theories of syntactic processing complexity. In *Cognition*, volume 109, Issue 2, pages 193–210.

G. Gazdar, E. Klein, Pullum G., and I. Sag. 1985. *The Logic of Typed Feature Structures*. Blackwell.

F. Keller. 2004. The entropy rate principle as a predictor of processing effort: An evaluation against eye-tracking data. *Proceedings of the conference on empirical methods in natural language processing*, 317(324):324.

A. Koller and S. Thater. 2010. Computing weakest readings. In *ACL '10: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

D. Lin. 1998. A dependency-based method for evaluating broad-coverage parsers. *Natural Language Engineering*, 4(2):97–114.

H. Telljohann, H. Hinrichs, and Kübler S. 2004. The tüba-d/z treebank - annotating german with a context-free backbone. In *4th International Conference on Language Resources and Evaluation*.

K. Tomanek, U. Hahn, S. Lohmann, and J. Ziegler. 2010. A cognitive cost model of annotations based on eye-tracking data. *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1158–1167.

B. Webber. 2009. Genre distinctions for discourse in the penn treebank. In *47th Annual Meeting of the ACL*, pages 674–682.

T. van der Wouden, H. Hoekstra, M. Moortgat, B. Renmans, and I. Schuurman. 2002. Syntactic analysis in the spoken dutch corpus. In *3rd International Conference on Language Resources and Evaluation*, pages 768–773.

```
<category label="SENT" sample:index="0" sentence:index="0:16" node:index="0:16:0">
   <category label="NP" features="NP:SUJ" node:index="0:16:1">
      <category label="Det" features="Da-ms----" node:index="0:16:2" form="Le" lemma="le"/>
      <category label="Noun" features="Ncms--" node:index="0:16:3" form="texte" lemma="texte"/>
      <category label="AP" features="AP" node:index="0:16:4">
         <category label="Adj" features="Af-ms-" node:index="0:16:5" form="mme" lemma="mme"/>
         <indices grammaticality="0.5866" sat:ratio="1.0" completeness="0.1176" quality:index="1.0" precision="0.5847"/>
      </category>
      <indices grammaticality="0.5993" sat:ratio="1.0" completeness="0.1525" quality:index="1.0" precision="0.6011"/>
   </category>
   <category label="VP" features="VP" node:index="0:16:6">
      <category label="VN" features="VN" node:index="0:16:7">
         <category label="Aux" features="Vaii3s--" node:index="0:16:8" form="avait" lemma="avoir"/>
         <category label="Verb" features="Vmps-smaip--" node:index="0:16:9" form="reu" lemma="recevoir"/>
         <indices grammaticality="0.6201" sat:ratio="1.0" completeness="0.2105" quality:index="1.0" precision="0.6284"/>
      </category>
      <category label="NP" features="NP:OBJ" node:index="0:16:10">
         <category label="Det" features="Ds3msp---" node:index="0:16:11" form="leur" lemma="leur"/>
         <category label="Noun" features="Ncms--" node:index="0:16:12" form="accord" lemma="accord"/>
         <category label="AP" features="AP" node:index="0:16:13">
            <category label="Adj" features="Af-ms-" node:index="0:16:14" form="formel" lemma="formel"/>
            <indices grammaticality="0.5851" sat:ratio="1.0" completeness="0.1176" quality:index="1.0" precision="0.5847"/>
         </category>
         <indices grammaticality="0.5981" sat:ratio="1.0" completeness="0.1525" quality:index="1.0" precision="0.6011"/>
      </category>
      <indices grammaticality="0.5871" sat:ratio="1.0" completeness="0.1111" quality:index="1.0" precision="0.5816"/>
   </category>
   <category label="Pct" features="Wd" node:index="0:16:15" form="." lemma="."/>
   <indices grammaticality="0.6224" sat:ratio="1.0" completeness="0.2142" quality:index="1.0" precision="0.6302"/>
</category>
```

Figure 9: Example of tree enriched with grammaticality