Given that PS-based parsing has been exhaustively studied in the last few decades, PS-based systems have shown to be superior to SH-based ones in terms of efficiency. Nevertheless, the adoption of a PS-based parsing strategy implies some considerable adjustments to the HPSG description language as well as to the linguistics assumed for handling constituency structure.

In PS-based systems, the HPSG immediate dominance/linear precedence (ID/LP) format for syntactic constituency must be converted to the PS rules format. This implies that the disjunctive schemata of the Immediate Dominance Principle (ID Principle) together with the Constituent Ordering Principle (CO Principle) are removed from the set of implicative constraints and their constraining effect is taken over by a convenient design of PS rules.

However, these drawbacks have to be weighed up against the fact that in configurational languages such as English or Portuguese, this adaptation from ID/LP format to PS format turns out to be considerably straightforward. On the other hand, from the SH-based systems listed above, CUF and TFS are no longer maintained and ConTROLL was not publicly available at the time we started developing the workbench. Also, the only SH-based system that may approximately match PS systems in terms of efficiency, TDL, is embodied in a complex multi-component platform which greatly reduces the portability of any workbench developed in it and hence possibility of incorporating extensions. Given this, the options available to us were ALE, ProFIT or CL-ONE.

From this group of three systems, CL-ONE has to be discarded given that, at its current stage of development, it does not present adequately stable behavior (Gregor Erbach, p.c.). ProFIT was therefore the chosen system, in preference to ALE. ProFIT handles rich type systems, it is highly portable, and since it relies directly on the unification procedure of Prolog, it is reportedly the most efficient of the systems listed above (the data in tables (2) and (3) were taken from Backofen et al., 1996 and Bolc et al., 1996).

**ProFIT**

ProFIT is an extension of Prolog with features, inheritance and templates. It was developed by Gregor Erbach at the Computational Linguistics Department of the University of Saarland, Saarbrücken, Germany, and a thorough presentation of it can be found in Erbach, 1995a.
The basic rationale of ProFIT is precisely to take the best profit from the advances both in the logic grammars tradition and in the grammatical theorizing tradition that uses sorted features and inheritance devices. This means that ProFIT was designed with the aim of deriving maximum benefit from the improvements in processing efficiency and expressive conspicuousness achieved in both of these research traditions.

Instead of resorting to a specific feature term unification algorithm implemented on top of Prolog, ProFIT compiles all sorted feature terms into a Prolog term representation. This enables the built-in Prolog term unification to be used, which makes ProFIT “5 to 10 times faster than systems which implement an unification algorithm on top of Prolog” (cf. Erbach, 1995, p.186).

The set of ProFIT programs is thus a superset of Prolog programs, and a ProFIT program consists not only of clauses but also of a data type declaration (sort hierarchy plus appropriateness conditions on the features of the sorts). This means that sorted feature terms, as defined below, can then be used together with Prolog terms.

(4) BNF of ProFIT terms:

\[
PFT := \begin{cases} 
\text{Sort} & [1. \text{Term of a sort Sort}] \\
\text{Attribute!PFT} & [2. \text{Attribute-Value Pair}] \\
\text{PFT \& PFT} & [3. \text{Conjunction of terms}] \\
\text{PROLOGTERM} & [4. \text{Any Prolog term}] \\
\text{FinDomTerm} & [5. \text{Boolean combinations of atoms}] \\
@\text{Template} & [6. \text{Template call}] \\
`\text{PFT}` & [7. \text{Quoted term not translated}] \\
``\text{PFT}` & [8. \text{Main functor not translated}] \\
>>>\text{Attribute!PFT} & [9. \text{Search for an attribute}] \\
\text{Sort}>>>\text{Attribute!PFT} & [10. \text{Abrev:>Sort&>>>Attribute!PFT}] \\
\text{PFT or PFT} & [11. \text{Disjunction}] 
\end{cases}
\]

Given this, a language engineering program written in ProFIT can include any type of parser from the logic grammar framework. All the techniques developed for the optimization of logic grammars efficiency can therefore be applied to improve the performance of grammars written in ProFIT.
The parser we linked to our ProFIT program is a left-corner bottom-up one, described in Pereira and Shieber, 1986, pp.178ff, and repeated below:

\[\text{(5)}\]

\[
\begin{align*}
\text{parse}(\text{Phrase}) \rightarrow & \ \text{leaf}(\text{Subphrase}), \\
& \ \text{lc}(\text{Subphrase, Phrase}).
\end{align*}
\]

\[
\begin{align*}
\text{leaf}(\text{Cat}) \rightarrow & \ [\text{Word}], \\
& \ \{\text{word}(\text{Word, Cat})\}.
\end{align*}
\]

\[
\begin{align*}
\text{leaf}(\text{Phrase}) \rightarrow & \ \{\text{Phrase} \longrightarrow []\}.
\end{align*}
\]

\[
\begin{align*}
\text{lc}(\text{Phrase, Phrase}) \rightarrow & \ [].
\end{align*}
\]

\[
\begin{align*}
\text{lc}(\text{SubPhrase, SuperPhrase}) \rightarrow & \ \{\text{Phrase} \longrightarrow [\text{SubPhrase}|\text{Rest}]\}, \\
& \ \text{parse}_\text{rest}(\text{Rest}), \\
& \ \text{cstrt}(\text{List}), \\
& \ \text{lc}(\text{Phrase, SuperPhrase}).
\end{align*}
\]

\[
\begin{align*}
\text{parse}_\text{rest}([]) \rightarrow & \ [].
\end{align*}
\]

\[
\begin{align*}
\text{parse}_\text{rest}([\text{Phrase}\mid\text{Phrases}]) \rightarrow & \ \text{parse}(\text{Phrase}), \\
& \ \text{parse}_\text{rest}(\text{Phrases}).
\end{align*}
\]

In view of the above comments relating to using PS-based systems for implementing HPSG particular grammars, we restricted ourselves to use PS rules, with format \text{A} \longrightarrow [\text{B, C, ...}], for the emulation of ID/LP requirements of the conjoined effect of the ID Principle and CO Principle. Consequently, in order to cope with the remaining HPSG implicative constraints, the parser was slightly extended. This extension ensures that the Prolog clauses implementing these principles is evoked while a sign is built up by the application of a PS rule. As depicted below, the final format of the PS rules in our workbench involves a third part, the beginning of which is marked with symbol \text{<>}, and which appears at the right of the rewriting rule:

\[\text{(6)}\]

\[
\begin{align*}
\text{lc}(\text{SubPhrase, SuperPhrase}) \rightarrow & \ \{\text{Phrase} \longrightarrow [\text{SubPhrase}|\text{Rest}] \text{<}> \text{List}\}, \\
& \ \text{parse}_\text{rest}(\text{Rest}), \\
& \ \text{cstrt}(\text{List}), \\
& \ \text{lc}(\text{Phrase, SuperPhrase}).
\end{align*}
\]
Computational Implementation

\[\text{cstrt}([]) \rightarrow \text{[]}\.
\]
\[\text{cstrt}([X|\text{List}]) \rightarrow \text{call1}(X), \text{cstrt}(\text{List})\.
\]
\[\text{call1}(X) \rightarrow \text{[]}, \{X\}\.
\]

Note that in the first clause, the predicate \text{cstrt/1} is evoked after the predicate \text{parse_rest/1} in order to avoid termination problems once some of the constraints, e.g. the \text{SUBCAT Principle}, succeed only if they can get hold of information from every daughter node of the relevant constituent.

7.2 Implemented Grammar

The whole workbench, including the binding constraints implemented in it, is a program with circa 3,000 lines of ProFIT code (vd. Annex I). Besides the parser, it includes a grammar fragment comprising a sort hierarchy declaration, a set of principles, a set of PS rules, a set of lexical entries, and a set of relational functions. Its development started in 1997, at the Language Technology Laboratory of the DFKI-German Research Center for Artificial Intelligence, Saarbrücken, Germany, where the majority of the workbench was developed.

Grammar fragment

The core grammar included in the workbench, except for the aspects concerning relative clauses and control, result from the implementation of the grammar specification provided in the Appendix of Pollard and Sag, 1994, with parameterization for Portuguese where required. Accordingly, in the implemented grammar, taking aside ID Principle and CO Principle, the presentation proper comprises the following principles: Head Feature Principle, Subcategorization
7.2 Implemented Grammar


Additionally, the implemented grammar includes the principles relevant for handling binding, namely the Binding Domains Principle and the COSUBCAT Principle:

(7)

\[
\text{principles}(X) \leftarrow \text{universal_principles}(X), \\
\text{language_specific_principles}(X).
\]

\[
\text{universal_principles}(X) \leftarrow \text{head_feature_principle}(X), \\
\text{subcat_principle}(X), \\
\text{spec_principle}(X), \\
\text{marking_principle}(X), \\
\text{non_local_feature_principle}(X), \\
\text{semantics_principle}(X), \\
\text{cosubcat_principle}(X), \\
\text{binding_domains_principle}(X).
\]

Given that we opted for a PS-based implementation system, the ID Principle and the CO Principle had to be greatly recast in terms of phrases structure rules, as discussed above. For the sake of illustration, an example of a PS rule is given below, where the rule handling Schema 1 of ID Principle is stated:

(8)

\[
\text{Mother}\& <\text{phrase}>\text{synsem!}>>\text{subcat!}[]& \\
\text{dtrs!}(<\text{hd_subj} & \\
\text{hd_dtr!HeadDtr}& \\
\text{subj_dtr!SubjDtr}) \\
--->
[(\text{SubjDtr}\& <\text{phrase}), \\
(\text{HeadDtr}\& <\text{sign} & \\
\text{synsem!}\{\text{loc!}!(><\text{inv!}<\text{minus}) & \\
\text{nonloc!}\text{udc!}(\text{to_bind!}\{\text{slash![}]})\}))))] \]
<>
[\text{principles}(\text{Mother})].
\]

Another change in Pollard and Sag's grammar specification concerned the module for semantics. The semantic representation system originally proposed was replaced with the one advocated by Frank and Reyle (1995), as discussed in Section 6.1.

The lexicon was reduced to a representative sample of entries with the significant syntactic behavior to be worked out. An example of a lexical entry is
given below, with the atomic clause for the Portuguese verb form viu, English saw (note that exhaustiveness of lexical description of word forms was not intended):

\[(9)\]

\[
\text{word(viu,} \\
\text{<word&} \\
\text{phon![viu]} & \\
\text{synsem!([loc!}[\text{cat!}(\text{head!}(<}\text{verb&} \\
\text{prd!<plus&} \\
\text{mod!<none&} \\
\text{vform!<fin&} \\
\text{aux!<minus&} \\
\text{inv!<minus&} \\
\text{subcat![@np(X), @np(Y)]&} \\
\text{marking!<unmarked&} \\
\text{cont!([non_nominal&} \\
\text{ls!([dist_lbs&} \\
\text{l_min!L2)&} \\
\text{subord[[]&} \\
\text{conds![((preds&} \\
\text{label!L2&} \\
\text{rel!ver&} \\
\text{argR!Ev&} \\
\text{arg1!X&} \\
\text{arg2!Y)])&} \\
\text{@nonloc_emptyA([X,Y])}))}. \\
\]

Unbounded dependency constructions and the associated thread-based technique were represented in the grammar with the implementation of the basics for the topicalization phenomenon.

No morphological module or set of lexical rules is included.

**Control**

The development of a computational grammar that could support our study on binding constraints was not a central goal but rather an instrumental step in our research. Accordingly, we mainly favored rapid prototyping of the grammar in preference to careful concern with performance. This was another major reason for opting for a highly efficient implementation formalism such as ProFiT, one that could sustain this type of demand and support an implementation which could be, to a considerable extent, careless as regards optimization issues.
Our concern with a modular and compact implementation of linguistic principles and with the fact that its specification could be easily read off from their implementation overrode thus concerns regarding the efficiency of the implemented grammar. Among other things, a different implementation of the HPSG principles associated with each PS rule would represent a significant increase in efficiency, namely implementation associating the relevant clauses of each principle with the PS rules responsible for the corresponding type of sign or structure to which the different clauses apply.

Performance

Even though performance was not a central concern of our research, we collected some data on the efficiency of the implemented grammar. The table below presents the cpu times for the parsing of some sample sentences of different length by the grammar, on a Power Mac G3, 300 MHz, 64MB platform with SICStus Prolog 3.6:

(10)

<table>
<thead>
<tr>
<th>INPUT STRING</th>
<th>WORDS</th>
<th>ANAPHORS</th>
<th>CPU TIME (MSEC)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Binding off</td>
</tr>
<tr>
<td>ctx1 ele sai</td>
<td>3</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>ctx1 o pedro sai</td>
<td>4</td>
<td>1</td>
<td>38</td>
</tr>
<tr>
<td>ctx1 ele gosta de ele</td>
<td>5</td>
<td>2</td>
<td>243</td>
</tr>
<tr>
<td>ctx1 ele disse que ele sai</td>
<td>6</td>
<td>2</td>
<td>190</td>
</tr>
<tr>
<td>ctx1 o pedro conversou com ele acerca_de ele</td>
<td>7</td>
<td>3</td>
<td>1,272</td>
</tr>
<tr>
<td>ctx1 o pedro conversou com ele acerca_de ele</td>
<td>8</td>
<td>3</td>
<td>10,170</td>
</tr>
<tr>
<td>ctx1 ele disse que ele disse que ele sai</td>
<td>9</td>
<td>3</td>
<td>6,421</td>
</tr>
<tr>
<td>ctx1 ele disse que ele conversou com ele acerca_de ele</td>
<td>10</td>
<td>4</td>
<td>35,600</td>
</tr>
</tbody>
</table>
It is interesting to compare the data in the last two columns, indicating the parsing times with binding constraints switched on and off. These figures show that the processing overload in this particular grammar due to binding constraints as they are implemented (vd. next section) is an average of 2.1 times the time required for parsing without such constraints, for an input of between 3 and 10 words in length. Moreover, the sample above seems also to indicate that the length of the input string and the number of anaphors to be handled are not the sole factors for increasing such overload, given that the extent to which the activation of binding constraints increases processing time also seems to greatly depend on the specific structure of the input sentence to be parsed (compare sentences of 8 and 9 words in length).

7.3 Implementation of Binding Constraints

Once the workbench was implemented, it was used as a tool to help develop our successive working hypotheses on the formal and computational modeling of binding constraints. Its last version, in Annex I, includes an implementation of the HPSG specification for binding presented in the previous chapter.

Signature

The signature extension described in Section 6.3 received an implementation as part of the data type declaration of the ProFIT program. In this declaration, the left-hand side of symbol \( > \) holds the supersort, and the list in the right hand side has the corresponding subsorts, while the list preceded by the keyword \( \text{intro} \) contains the appropriate attribute-value pairs for the supersort (vd. Erbach, 1995b, for further details). The declaration for specific non-local features for binding is as follows:
7.3 Implementation of Binding Constraints

(11)
\[
\begin{align*}
\text{nonloc} & > [\] \\
& \quad \text{intro} [\text{udc: } \text{udc, binding: } \text{binding}]. \\
\text{binding} & > [\] \\
& \quad \text{intro} [\text{list}_A, \text{list}_Z, \text{list}_U, \text{list}_\text{protoU}].
\end{align*}
\]

And the feature declaration for the relevant part included in the semantic representation is given below:

(12)
\[
\begin{align*}
\text{udrs} & > [\text{nominal, } \\
& \quad \text{non}_\text{nominal}] \\
& \quad \text{intro} [\text{ls: } \text{dist}_\text{lbs, subord, } \text{conds}]. \\
\text{nominal} & > [\text{nom}_pA, \text{nom}_pB, \\
& \quad \text{nom}_pC, \text{nom}_pZ] \\
& \quad \text{intro} [\text{anaphora: } \text{anaphora}]. \\
\text{anaphora} & > [\text{a}_\text{def, a}_\text{quant}] \\
& \quad \text{intro} [\text{refmark}]. \\
\text{a}_\text{def} & > [\] \\
& \quad \text{intro} [\text{antec}]. \\
\text{a}_\text{quant} & > [\] \\
& \quad \text{intro} [\text{var}].
\end{align*}
\]

Given the syntax of data type declaration in ProFIT, in both cases the correspondence between specification and implementation is quite direct.
Lexicon

To exemplify the implementation of lexical items, we will explore two illustrative cases, namely the coding of Portuguese pronoun ele (English: he), complying with principle B, and the definite article o (English: the [masc., sing.]), inducer of principle C effect. The next subsection addresses the long-distance reflexive ele próprio. Readers interested in other cases should refer to Annex I.

The clause implementing the lexical specification of the pronoun is stated below. Given that attributes (in upper case) in the specification are rendered as features in ProFIT (in lower case) preceding symbol !, and sorts are preceded by symbol <, the correspondence between implementation and specification of the lexical item is to a large extent self explanatory:

(13)

\begin{verbatim}
word(ele,
  <phrase&
   (phon![ele] &
    synsem!(loc!(cat!(head!(<noun
       cosubcat!Alist&
      extcosubcat!Zlist&
     prd!<minus&
    case!<nom&
   mod!<none>&
    subcat![[]&
   marking!<unmarked>&
    cont!(<nom_pB&
     ls!(<dist_lbs&
      1_max!Label&
      1_min!Label)&
     subord![[]&
    conds![(<anaph_np&
       label!Label&
      argR!Refmark)]&
    anaphora!(<a_def&
       Refmark!Refmark&
      antec!Antec))]&
    nonloc!(udc!(inher!(que![[]&
      rel![[]&
    slash![[]&
    to_bind!(que![[]&
      rel![[]&
    slash![[]&
    binding!(list_A!Alist&
      list_Z!Zlist&
      list_U!Ulist&


\end{verbatim}
7.3 Implementation of Binding Constraints

The head of the clause comprises the predicate word/1 whose argument is the ProFIT term corresponding to the lexical information directly associated with the pronoun, except with respect to the value of attribute ANTEC.

According to the specification, the value of ANTEC is the value returned by the relational constraint non-loc-ocommand. That constraint is implemented here in the body of the lexical clause by means of the predicate non_loc_ocommand/4. The semantics of this predicate are described in Section 6.3, and is implemented with the set of auxiliary clauses presented below. Its appearance in the scope of the delaying built-in freeze/2 predicate is justified to avoid termination problems or undesired outcomes of auxiliary predicates - namely append/3 - due to the fact that one of its arguments, corresponding to LIST-U value, can only be instantiated after the whole relevant complex sign is built up, and consequently after non_loc_ocommand/4 is evoked. It is defined as follows:

(14)

\[
\text{non} \_\text{loc} \_\text{ocommand}(Ulist, Alist, Refmark, Antec):- \\
\quad \text{ocommand}(\text{Refmark}, Alist, \text{Loc}_\text{ocommanders}), \\
\quad \text{append}(\text{Loc}_\text{ocommanders}, [\text{Refmark}], \text{Loc}), \\
\quad \text{diff}(\text{Ulist}, \text{Loc}, \text{Antec}).
\]

\[
\text{ocommand}(\_ , [], []). \\
\text{ocommand}(X, [Y|\text{Rest}], [Y|\text{Result}]) :- \\
\quad \text{var}(Y), \\
\quad \text{ocommand}(X, \text{Rest}, \text{Result}), !. \\
\quad \text{ocommand}(X, [X|\_], []) :- !. \\
\quad \text{ocommand}(X, [Y|\text{Rest}], [Y|\text{Result}]) :- \\
\quad \text{ocommand}(X, \text{Rest}, \text{Result}).
\]

The predicates append/3 and diff/3 above perform the expected operations on lists, namely the concatenation of lists, and the extraction of elements of the list in second argument from the list in first argument. The predicate ocommand/3 scans the list in its first argument until the second argument is found and returns a list with the predecessors of the latter.
In our implementation, we did not consider lexical entries with branching obliqueness hierarchies. The inclusion of lexical entries of this type, however, can be done quite straightforwardly by means of introducing a specific predicate to emulate sets of reference markers within obliqueness hierarchies (v.d. discussion in Section 3.3).

The second illustrative example, referring to the definite article, is given below:

(15)

```prolog
word(o, 
  <word& 
    phon![o]& 
    synsem!(loc!(cat!(head!(<det& 
      spec!nbar(Refmark, Label2))& 
      subcat![[]& 
      marking!<unmarked>& 
      cont!(<nom_pC& 
        ls!(<dist_lbs& 
          _max!Label1& 
          _min!Label2)& 
          subord![eq(Label1, Label2)]& 
          conds!((<art& 
            label!Label1& 
            argR!Refmark])& 
            anaphora!(<a_def& 
              refmark!Refmark& 
              antec!Antec)])& 
          nonloc!(udc!(inher!(que![]& 
            rel![[]& 
            slash![]])& 
            to_bind!(que![]& 
            rel![[]& 
            slash![]))& 
            binding!(list_A!& 
              list_Z!Zlist& 
              list_U!Ulist& 
              list_protoU![Refmark])))& 
  random(1, 999, Refmark), 
  freeze(Ulist, freeze(Zlist, non_command(Ulist, Zlist, Refmark, Antec)))).

non_command(Ulist,Zlist,Refmark, Antec):- 
  command(Refmark, Zlist, Ocommanders), 
  append(Ocommanders, [Refmark], Ocom), 
  diff(Ulist, Ocom, Antec).
```

The major differences compared to the previous example, as far as binding is concerned, mainly involves the anchoring of REFMARK value of the determiner
7.3 Implementation of Binding Constraints

as regards the ARG-R value of the corresponding nominal head, and the outcome of a different relational constraint, namely non-command, as ANTEC value.

The non-command constraint is implemented as predicate non_ocommand/4, whose auxiliary predicates are defined above in (14).

The anchoring referred to is ensured by variable sharing between the value of attribute refmark! and the first argument of @nbar/2, which is a ProFIT template abbreviating the description of an N’ structure, defined below:

\[
\text{nbar(}\mathbf{ArgR}, \mathbf{Lab}) := \text{loc(! (head!}<\text{noun}&\text{subcat!}[@\text{detp}]&\text{cont!} (\text{conds!} [(<\text{drscond}&\text{label!}Lab&\text{argR!}\mathbf{ArgR}|Rest])})\].
\]

Clitics and the ambiguity of Portuguese LD reflexive

The formal and computational modeling of Portuguese clitics is a highly non-trivial issue subject to intense research (for an HPSG account and comprehensive reference to papers in other frameworks vd. Crysmann, 1997, forth.). Given the complexity of this issue, it does not fall within the scope of the present dissertation to seriously attempt to propose implementing the morpho-syntax of clitics in their many different facets. However, given that lexical NPs in object position lead to their doubling by clitics, a handy solution had to be found so that our grammar could handle binding constraints when associated with NPs under such circumstances. Moreover, since binding properties of the otherwise long-distance anaphor ele próprio in object position are constrained by the type of clitic (reflexive or non-reflexive) co-occurring with this expression (cf. Section 4.2), this solution had to be designed in such a way that this constraining effect could be implemented.

In order to meet this desideratum, we first extended the data type declaration with subsorts of sort nominal, thus enlarging the range of types of possible CONT values. Each subsort corresponds to the semantic representation of a specific type of nominal according to its binding properties:
Computational Implementation

(17)

nominal > [nom_pA, nom_pB, nom_pC, nom_pZ]

intro [anaphora: anaphora].

nom_pA > [ ].
nom_pB > [ ].
nom_pC > [ ].
nom_pZ > [ ].

Second, in lexical entries of clitics, the SPEC value is used to specify the kind of NP that they can double. Accordingly, for a reflexive or non-reflexive clitic, the CONT value of the NP doubled by the clitic is of sort nom-pA or nom-pB respectively. As an example, the lexical entry for reflexive clitic se is stated below:

(18)

word(-se, <clitic&
  (phon!['-se']&
   synsem!(loc!(cat!(head!(<do_clit&
     spec!()>cont!(<nom_pA&
       ls!(<dist_lbs&
         l_max!L&
         l_min!L)
       conds!([(anaph_np&
         label!L&
         argR!X)])&
       anaphora!(<a_def&
         refmark!X))))&
       subcat![] &
       marking!<unmarked>)&
      cont!(<non_nominal&
       ls!(<dist_lbs)&
       subord![] &
       conds![[]] &
       @nonloc_empty))))).
Third, we implemented PS rules that build verbal signs from a verb and a clitic – this was designed in the spirit of Oliva, 1994, as this is intended to somehow emulate the effect of lexical rules in terms of relational constraints:

(19)

\[
(\langle \text{word} & \rangle \\
(\text{phon}!\text{PhonMother} & \\
\text{synsem}!(\text{loc}!(\text{cat}!(\text{head}!\text{Head} & \\
\text{subcat}!\text{SubcatMother} & \\
\text{marking}!\text{Marking} &) \\
\text{cont}!\text{Cont}) & \\
\text{nonloc}!\text{Nonloc})))
--->
(\langle \text{word} & \\
(\text{phon}!\text{PhonVerb} & \\
\text{synsem}!(\text{loc}!(\text{cat}!(\text{head}!(\langle \text{verb} & \text{Head} & \\
\text{subcat}![\text{Subj}, @\text{np}(X)] | \text{Rest} & \\
\text{marking}!\text{Marking} &) \\
\text{cont}!\text{Cont}) & \\
\text{nonloc}!\text{Nonloc})))

(\langle \text{clitic} & \\
(\text{phon}!\text{PhonClit} & \\
\text{synsem}!>>!(\text{head}!(\langle \text{do_clit} & \\
\text{spec}!>>!\text{cont}!A)))

<>

[\text{append}(\text{PhonVerb}, \text{PhonClit}, \text{PhonMother}), \\
\text{append}([\text{Subj}, \\
\text{loc}!(\text{cat}!(\text{head}!(\langle \text{prep} & \\
\text{pform}!a & \\
\text{subcat}![]) & \\
\text{cont}!A & \text{conds}![(\text{argR}!X) | _]])], \\
\text{Rest}, \\
\text{SubcatMother})].

The rule returns a verbal sign whose SUBCAT value is identical to the SUBCAT value of the cliticized verb except for the direct object. The “new” direct object is specified as being introduced by the preposition a, and the corresponding NP as bearing a semantic representation of sort nom_pA, specified in the SPEC value of the clitic.

Fourth, the clitic doubling NPs are lexically specified as nom_pB (ele) – as can be seen in example (13) –, or nom_pA (si próprio). The long-distance reflexive ele próprio is however underspecified in the head of the lexical clause as nominal:
Computational Implementation

(20)

word(ele_pro1prio, 
  <phrase&
   (phon![ele_pro1prio]&
    synsem!(loc!(cat!(head!(<noun&
      cosubcat!Alist&
      extcosubcat!Zlist&
      prd!<minus&
      case!<nom&
      mod!<none>&
      subcat![[]&
      marking!<unmarked>&
      cont!(Cont& <nominal&
       ls!(<dist_lbs&
        l_max!Label&
        l_min!Label)&
       subord![[]&
       conds!({<anaph_np&
           label!Label&
           argR!Refmark}&
        anaphora!(<a_def&
          refmark!Refmark&
          antec!Antec)})&
       nonloc!(udc!(inher!(que![[]&
          rel![[]&
          slash![]]&
        to_bind!(que![[]&
          rel![[]&
          slash![]]&
        binding!(list_A!Alist&
          list_Z!Zlist&
          list_U!Ulist&
          list_protoU![Refmark])})&
       dtrs!<only_hd>):-
        random(1,999,Refmark),
        freeze(Ulist,
        freeze(Zlist,
        freeze(Alist,ambig(Cont,Refmark,Ulist,Zlist,Alist,Antec)))).

ambig(<nom_pZ,Refmark,Ulist,Zlist,Alist,Antec) :-
  ocommand(Refmark,Zlist,Antec), !.

ambig(<nom_pA,Refmark,Ulist,Zlist,Alist,Antec):-
  ocommand(Refmark,Alist,Antec), !.

ambig(<nom_pB,Refmark,Ulist,Zlist,Alist,Antec):-
  non_loc_ocommand(Ulist, Alist, Refmark, Antec).
In combination with the delaying mechanisms presented above, the predicate `ambig/6` is able to induce the convenient specification. During the parsing process, the CONT value of ele próprio is thus specified as long-distance anaphor (nom-pZ), short-distance anaphor (nom-pA) or pronoun (nom-pB).

If the NP ele próprio is not doubled by a clitic, no specification of the lexically assigned sort nominal occurs at the PS rule for the complex verb form+clitic (cf. (19)). As the first clause of `ambig/6` is evoked, it is responsible for assigning the sort nom-pZ and for calculating ANTEC value in compliance with principle Z.

If the NP is doubled by a reflexive clitic, due to the SPEC value of the clitic and the "lexical rule" in (19), its CONT value is constrained to be of sort nom-pA. Accordingly, the second clause of `ambig/6` applies and ANTEC receives a value that observes principle A.

Finally, if the NP is doubled by a non reflexive clitic, via the same mechanism, its CONT value is specified as nom-pB and the ANTEC value complies with principle B.

**Grammatical principles**

Coming now to the implementation of the two grammatical principles included in the specification for binding, their different complexity is reflected, as expected, in implementations likewise differing in complexity.

The COSUBCAT Principle is implemented with predicate `cosubcat_principle/1`, as described below:

(21)

```
cosubcat_principle(synsem!(loc!cat!(head!(cosubcat!Alist&
    extcosubcat!Zlist)&
    subcat![[])&
    nonloc!binding!(list_A!Alist&
    list_Z!Zlist))):-!.
```

```
cosubcat_principle(X) :- true.
```

The implementation of the Binding Domains Principle (BDP) is slightly more complex. It involves implementing each of its three clauses (for LIST-U and LIST-protoU, for LIST-A, and for LIST-Z). It would be too time-consuming to go
Computational Implementation

through all the details of implementing all the clauses. For the sake of illustration, we will briefly discuss the implementation of Clause I of BDP:

(22)

\[
\text{binding\_domains\_principle}(X) \Leftarrow \\
\text{clause2}(X), \\
\text{clause3}(X), \\
\text{clause1}(X).
\]

\[
\text{clause1}(X) \Leftarrow \\
\text{inher\_ListprotoU}(X), \\
\text{listprotoU\_listU}(X), \\
\text{inher\_ListU}(X).
\]

Clause I is implemented with predicate \text{clause1/1}, which has three auxiliary predicates.

Predicate \text{inher\_ListprotoU/1} is responsible for passing upwards the values of LIST-\text{protoU} values from daughter nodes and for compiling them in the corresponding mother nodes. It implements Clause I.(i) constraint:

(23)

\[
\text{inher\_ListprotoU}(<\text{sign}& \\
\text{(synsem}\!\text{nonloc}\!\text{binding}\!\text{list\_protoU}\!\text{MotherList} & \\
\text{dtrs}!(\text{hd}\_\text{dtr}\!\text{synsem}\!\text{nonloc}\!\text{Binding}\!\text{list\_protoU}\!\text{HeadList} & \\
\text{(spec\_dtr}\!\text{synsem}\!\text{nonloc}\!\text{binding}\!\text{list\_protoU}\!\text{DaughterList}) \\
\text{or} \\
\text{adj\_dtr}\!\text{synsem}\!\text{nonloc}\!\text{binding}\!\text{list\_protoU}\!\text{DaughterList}) \\
\text{or} \\
\text{mark\_dtr}\!\text{synsem}\!\text{nonloc}\!\text{binding}\!\text{list\_protoU}\!\text{DaughterList}) \\
\text{or} \\
\text{fil\_dtr}\!\text{synsem}\!\text{nonloc}\!\text{binding}\!\text{list\_protoU}\!\text{DaughterList}) \\
\text{or} \\
\text{subj\_dtr}\!\text{synsem}\!\text{nonloc}\!\text{binding}\!\text{list\_protoU}\!\text{DaughterList}) \\
\text{)}))) \Leftarrow \\
\text{append}(\text{HeadList}, \text{DaughterList}, \text{MotherList}),!.
\]

\[
\text{inher\_ListprotoU}(<\text{sign}& \\
\text{(synsem}\!\text{nonloc}\!\text{binding}\!\text{list\_protoU}\!\text{MotherList} & \\
\text{dtrs}!(\text{hd}\_\text{dtr}\!\text{synsem}\!\text{nonloc}\!\text{Binding}\!\text{list\_protoU}\!\text{HeadList} & \\
\text{cp\_dtrs}\!\text{CompDtrs})) \Leftarrow \\
\text{protoU\_of\_comps}(\text{CompDtrs}, \text{DaughterList}), \\
\text{append}(\text{HeadList}, \text{DaughterList}, \text{MotherList}),!.
\]

\[
\text{inher\_ListprotoU}(<\text{disc}& \\
\text{(synsem}\!\text{nonloc}\!\text{binding}\!\text{list\_protoU}\!\text{MotherList} & \\
\text{dtrs}!(\text{hd}\_\text{dtr}\!\text{synsem}\!\text{nonloc}\!\text{Binding}\!\text{list\_protoU}\!\text{HeadList} & \\
\text{cp\_dtrs}\!\text{CompDtrs})) \Leftarrow \\
\text{append}(\text{HeadList}, \text{DaughterList}, \text{MotherList}),!.
\]
dtrs!(hd_dtr!binding!list_protoU!HeadList&
text_dtr!Sents))) :-
    protoU_of_comps(Sents,DaughterList),
    append(HeadList,DaughterList,MotherList),!.

inher_ListprotoU(X):- true.

Daughters given in a list require the auxiliary predicate protoU_of_comps/2 to extract the LIST-protoU value from each of them:

(24)
    protoU_of_comps([],[]).
    protoU_of_comps([(synsem!nonloc!binding!list_protoU!ProtoU)
                     |Signs],
                     Result) :-
                     append(ProtoU,X,Result),protoU_of_comps(Signs,X).

The predicate listprotoU_listU/1, in turn, is responsible for passing the reference markers collected as the value of LIST-protoU to the LIST-U feature. It implements Clause I.(ii) constraint:

(25)
    listprotoU_listU(<disc&
                     synsem!nonloc!binding!(list_U!List&
                     list_protoU!List)):!.
    listprotoU_listU(X) :- true.

Finally, the inher_ListU/1 implements Clause I.(iii) and Clause I.(iv) constraints, the latter being implemented by the first clause below, and the former by the remaining clauses. This predicate is responsible for passing downwards the LIST-U value, a list containing all the markers of the relevant global context, from mother nodes to daughter nodes LIST-U value:

(26)
    inher_ListU(<sign&
                 (synsem!(loc!cont!anaphora!(refmark!Refmark&
                               var!Var)&
                               nonloc!binding!list_U!MotherList))&
                 dtrs!(hd_dtr!synsem!nonloc!binding!(list_A!NbarRefmarks&
                               list_U!NbarList)&
                 spec_dtr!synsem!nonloc!binding!list_U!SpecList)):-
                 freeze(MotherList,
Computational Implementation

diff(MotherList, [Refmark, Var], NbarList)),
freeze(MotherList,
diff(MotherList, NbarRefmarks, SpecList)), !.

inher_ListU(<sign&
    (synsem!(loc!cont!anaphora!refmark!MotherRefmark&
     nonloc!binding!list_U!MotherList))&
    dtrs!(hd_dtr!synsem!nonloc!binding!(list_A!NbarRefmarks&
     list_U!NbarList)&
    spec_dtr!synsem!nonloc!binding!list_U!SpecList)):-
    freeze(MotherList,
    diff(MotherList, [MotherRefmark],
    NbarList)),
    freeze(MotherList,
    diff(MotherList, NbarRefmarks, SpecList)), !.

inher_ListU(<sign&
    (synsem!nonloc!binding!list_U!MotherList&
    dtrs!(hd_dtr!synsem!nonloc!binding!list_U!MotherList&
    ((adj_dtr!synsem!nonloc!binding!list_U!MotherList) or
    (mark_dtr!synsem!nonloc!binding!list_U!MotherList) or
    (fil_dtr!synsem!nonloc!binding!list_U!MotherList) or
    (subj_dtr!synsem!nonloc!binding!list_U!MotherList))))):- !.

inher_ListU(<disc&
    (synsem!nonloc!binding!list_U!MotherList&
    dtrs!(hd_dtr!binding!list_U!MotherList&
    text_dtr!Sents))) :-
    pass_listU(MotherList, Sents), !.

inher_ListU(<sign&
    (synsem!nonloc!binding!list_U!MotherList&
    dtrs!(hd_dtr!synsem!nonloc!binding!list_U!MotherList&
    cp_dtrs!CompDtrs))) :-
    pass_listU(MotherList, CompDtrs), !.

inher_ListU(X) :- true.

pass_listU(List, []).

pass_listU(List, [synsem!nonloc!binding!list_U!List|Rest]) :-
    pass_listU(List, Rest).
Complexity issues

Having discussed the full specification of binding constraints in a constraint-based framework such as HPSG, and the complete implementation of that specification in a formalism of the logic programming family such as ProFIT, we can now turn to the algorithmic aspects of making a given grammatical representation comply with binding constraints. In particular, it will be specially worth comparing the order of complexity of the algorithm underlying our proposal vis-à-vis the complexity of the mainstream algorithm for binding constraints, due to Chomsky (1981) (although, as discussed in Section 4.4, one should bear in mind that the latter additionally ensures reciprocal verification and mediated antecedency effects, which do not fall within the scope of binding constraints proper).

In order to proceed, and disregarding inessential details, the basic structure of our algorithm underlying binding constraints satisfaction can be sketched as follows:

(27)

Part A: Creation and manipulation of four relevant lists of reference markers:

1. LIST-A

In a node k, LIST-A is the lexically provided LIST-A of the relevant local predicator.

2. LIST-Z

2.1 Setup points:

At the top of the grammatical representation, LIST-Z is LIST-A.

2.2 Incrementing points:

If k is a node of a sentential daughter, LIST-Z is the append of LIST-Z of the node dominating k with LIST-A;

If k is a node of a nominal predicator domain, LIST-Z is the append of LX with LIST-A, where LX is obtained by taking from LIST-Z of the node dominating k the (at most two: REFMARK and VAR) reference markers of the NP node immediately dominating k;

otherwise LIST-Z is LIST-Z of the node dominating k.
3. LIST-protoU
   In a node k, LIST-protoU is the append of LIST-protoUs of the daughters of k.

4. LIST-U
   In a node k, LIST-U is LIST-U of the node dominating k, except at the top of the grammatical representation, where LIST-U is LIST-protoU.

Part B: Operations on lists of NP nodes:

1. SD Anaphors
   Take from LIST-A the list of predecessors of a given element.
   Operation executed in our implementation by the predicate ocommand/3 (vd (12)).

2. LD Anaphors
   Take from LIST-Z the list of predecessors of a given element.
   Operation executed by the predicate ocommand/3.

3. Pronouns:
   Take from LIST-U the list of elements that are different from every element of list LX, which is obtained by taking from LIST-A the list of the predecessors of a given element a and A.
   Operation executed by the predicate non_loc_command/4.

   non_loc_command(+Ulist, +Alist, +Refmark, -Antec)
   non_loc_command(Ulist,Alist,Refmark,Antec):-
     ocommand(Refmark, Alist, Loc_ocommanders),
     append(Loc_ocommanders, [Refmark], Loc),
     diff(Ulist, Loc, Antec).

4. Definite descriptions
   Take from LIST-U the list of elements that are different from every element of list LX, which is obtained by taking from LIST-Z the list with the antecedents of a given element z and Z.
   Operation executed by the predicate non_command/4.

   non_command(+Ulist, +Zlist, +Refmark, -Antec)
   non_command(Ulist,Zlist,Refmark,Antec):-
For the sake of simplicity, let us assume that in the worst case, every leaf of the grammatical tree is a lexical NP node, so there will be as many reference markers around as the length \(n\) of the input sentence.

Considering Part A first, we can disregard parts A.1, A.2.1 and A.4 since they only involve value passing.

Parts A.2.2 and A.3 in turn involve the append of lists in the nodes of the tree. On the one hand, the complexity of the operations involved – namely appending of lists and extracting at most two elements from a list – is linear on \(n\). They involve traversing a list, which in the worst case is of length \(n\). On the other hand, there are as many paths in a grammatical tree between leaves and the top node as there are leaves, and as the depth of the tree is finite, the number of nodes in the tree is proportional to the length of the input sentence, which makes the traversing of the tree node by node an operation whose complexity is linear on \(n\). Accordingly, having in the worst case at most two append operations in each node – both A.2.2 and A.3 –, the order of complexity of completion of A.2.2 and A.3 is proportional to \(n^2\), making the complexity of Part A of the algorithm quadratic in the worst case.

Turning now to Part B, the operation of extracting the list of predecessors of an element in a list – as in B.1 and B.2 – is of linear complexity. Since in the worst case this operation may occur in every leaf, parts B.1 and B.2 of our algorithm involve quadratic complexity.

Parts B.3 and B.4 in turn involve operations whose complexity is proportional to \(n^2\). Besides the operations of extracting antecedents in a list and of appending lists, B.3 and B.4 integrate the operation of obtaining the complement of a list with respect to another list. For each element of one of the lists, this involves scanning the other list, thus making this procedure of quadratic complexity. As in the worst case every item of the input sentence may be a non-reflexive, this set of operations is executed \(n\) times, which in the worst case makes complexity of Part B proportional to \(n^3\).

Therefore, if we consider all the parts together, the complexity of our algorithm turns out to be polynomial to the order of \(n^3\).

As in our implementation unification involves at most lists of depth one without any occur check, this is also the expected order of magnitude for the complexity of our implementation of the algorithm in ProFIT. Note also that if branching obliqueness is taken into account, the essential aspects of results above remain unchanged. Empirically, each list in a relevant branching obliqueness hierarchy (LIST-A or LIST-Z) seems to have at most one set; and each relevant set has at most one list. Accordingly, given that a certain reference marker does not occur more
than once in any of these hierarchies, the complexity of the relevant operations with the o-command relation in such type of structure turns out to be similar to the complexity of these operations in a list of depth one as discussed above.

Finally, it is interesting to note that the algorithm above and the mainstream algorithm due to Chomsky (1981) do not solve exactly the same problem. In Section 4.4, we have shown that Chomsky's algorithm involves what we termed “reciprocal validation” and “mediated antecedency”. We also argued at length there that these two issues should not be counted as belonging to the realm of binding theory. Although mindful of this fact, it is interesting to note that the result now obtained shows that our proposal may be seen, in terms of binding constraints proper and from the point of view of complexity, as an improvement over the mainstream algorithm due to Chomsky (1981), which is known to be of exponential complexity (cf. discussion in Section 6.2).

### 7.4 Summary

As an instrumental step in our research on reference processing, we developed a grammar workbench to help verify and improve the theoretical consistency and empirical accuracy of our successive working hypotheses on the formal and computational modeling of binding constraints.

We began by reviewing the major features of implementation systems available for computationally implementing grammars specified in the HPSG framework.

Next we reported on the implementation of a core grammar of Portuguese in ProFIT, whose coverage corresponds roughly to the coverage of the grammar specified in the Annex of Pollard and Sag, 1994.

Finally, we reported on the integration of the specification of binding constraints developed in the previous chapter in this core grammar, and we discussed the complexity of the basic algorithm underlying the verification of binding constraints, which was shown to be polynomial to the order of $n^3$. 


8 Conclusions:
Summary and Outlook

8.1 Summary of the Dissertation
8.2 From the Angle of Antecedents
8.3 Binding Constraints in Discourse
    Inter-sentences
    Locality
    Recess
    Coordination
    Adverbial subordination
8.4 Gaps and Linguistic Universals
    Accessibility hierarchy
    Quantifier types
8.5 Generation of Anaphoric Expressions
8.6 Cognitive Underpinnings
    Polysemy and polymorphism
    Search optimization in cognitive space
    Looking for a different kind of cognitive underpinnings
    Quantification, semantic universals and cognition
Conclusions: Summary and Outlook

It is commonplace that any research theme will have a large number of related themes contiguous to it, and that circumscribing an issue to be investigated involves perhaps as many accidental factors as essential facets making it a prima facie self-contained topic. It is not controversial to state that isolating a research theme is a crucial step along the path of enhancing our scientific knowledge about it, although it may involve to a greater or lesser extent a risk of oversimplification.

This was certainly applicable for the research topics addressed in this dissertation. As expected, it may turn out that many of the research paths and analytical solutions adopted here may be extended to contiguous issues, and that the results achieved may induce new perspectives and progress in other not such contiguous issues. It may also turn out that pursuing research to other broader research areas will help to revise and enhance in a principled way the results now acquired.

In Section 8.1 of this chapter, we will first summarize the major results obtained within the research reported in this dissertation.

In the remainder of the chapter, we will briefly discuss how the research reported here can be continued. In contrast to preceding chapters, this discussion will be predominantly speculative. It will consider the seemingly important aspects of integrating binding constraints into successively broader domains, where larger sets of data, processing factors or practical requirements have to be taken into account.

After Section 8.1, we will address in a preliminary, exploratory fashion such diverse issues as the possible existence of constraints on anaphoric links imposed not by anaphors but plausibly by antecedents; the possibility that binding constraints may be extended to discourse, inasmuch as they may be found to be applicable for inter-sentential anaphoric links as well; the eventual relation of binding constraints to linguistic universals and a possible principled explanation for the fact that not all natural languages have items exhibiting every binding
constraint; the generation of anaphoric expressions; and the cognitive underpinnings that conjecturally underlie binding constraints.

8.1 Summary of the Dissertation

The research reported in this dissertation aimed to bridge the persistent gap between the conceptual elegance, empirical adequacy and practical relevance of binding constraints, on the one hand, and their integration into grammar and the course of semantic processing, on the other.

In order to pursue this goal, we sought to clarify how binding constraints should be fully integrated into semantic processing by discussing how to fully integrate them into a grammar that could be neatly interfaced with reference processing systems, and by clarifying how that could be achieved formally and computationally in a mainstream constraint-based framework such as HPSG.

We obtained the following major results:

While circumscribing the convenient theoretical and practical setup for the study of binding constraints, we carried out:

• an extensive overview of reference processing and anaphor resolution;
• an overview and critique of the adequacy of centering theory for the purpose of anaphor resolution and anaphora understanding.

When addressing problems and possible counterexamples to the generalizations underlying binding constraints recently raised, we ensured the preservation of the hypothesis about their universality by uncovering new parameterization factors, namely:

• linear vs. non-linear obliqueness hierarchy option for predicators;
Conclusions: Summary and Outlook

- possibility for binding constraints to apply on the basis of (i) derived, (ii) non-derived, or (iii) derived and non-derived obliqueness hierarchies, depending on specific languages.

Concentrating on long-distance reflexives, which have appeared in the literature as a particularly problematic class of anaphors, we argued that:

- Portuguese has long-distance reflexives;
- the widespread, mainstream prediction concerning the correlation between long-distance reflexives, on the one hand, and subject-orientedness and morphological simplicity, on the other, has no empirical ground;
- there is a fourth binding constraint for long-distance reflexives.

Building on the existence of a fourth binding constraint, we have shown that:

- binding constraints can be organized as corners of a logical square of opposition and new symmetries among them can be uncovered;
- binding constraints can be analyzed as the effect of phase quantifiers over obliqueness orders.

When addressing the proper place of binding constraints inside the architecture of grammar, we proposed:

- a thorough integration of phase quantification over obliqueness in grammar, either in the semantic representation or in the syntax-semantics interface component (using DRT);
- neat interface points between grammars with binding constraints and anaphor resolvers;
- a systematic classification of types of anaphora and a clarification of the coordination between semantic representation and reference processing.

When looking for accommodating binding constraints in a constraint-based setup for grammatical knowledge representation and processing, we provided:

- a specification of a grammar with binding constraints without resorting to special purpose description devices (using HPSG);
- a definition of a new polynomial algorithm for verifying binding constraints embedded in their constraint-based specification, as an alternative to the exponential, mainstream one.
Finally, we carried out:

• a demonstration by example of the practical viability of the specification developed by providing an implementation of a support grammar and specified binding constraints in that grammar (using ProFIT);

It is worth noting that these results form a sort of serendipitous chain in the sense that each result led to the discovery of the next, although they do not form an implicative chain.

Developing an HPSG/ProFIT grammar for binding constraints does not require a full-fledged DRT account of obliqueness quantification, although this account is highly useful in illuminating how to proceed when developing such a grammar. By the same token, having a DRT account of obliqueness quantification does not require the existence of a fourth binding principle, although the latter was very stimulating for hypothesizing an analysis of binding constraints in terms of phase quantification over obliqueness hierarchies. And for obtaining an empirical generalization that shows up as the fourth binding constraint, it is not indispensable for Portuguese to have a long-distance reflexive, which happens, nevertheless, to be particularly telling with respect to falsifying the hypothesis that long-distance reflexivity can be reduced to a cyclic effect of short-distance reflexivity.

As they are logically independent, all these major results are of chief relevance for enhancing the empirical, formal and computational plausibility of each other, and together they form of a set of coherent results.

8.2 From the Angle of Antecedents

As discussed throughout this dissertation, binding constraints are grammatical conditions on possible anaphoric links. It is then worth noting that, while the anaphoric links that are grammatically allowed result from considering the possible antecedents that are grammatically admitted for a given anaphor, binding
Conclusions: Summary and Outlook

constraints can be said to be conditions on anaphora from the viewpoint, so to speak, of anaphors.

A seemingly parallel issue would thus be to study possible constraints on anaphoric links, but this time from the angle of antecedents. Admittedly, this suggestion raises the issue that there might be a whole area of research, the due relevance of which has not been sufficiently acknowledged.

While discussing different types of anaphoric relations in chapter 5 Formal Semantics, we already hinted at a problem of this kind when addressing the issue of bound anaphora. As antecedents of anaphoric links, quantificational NPs seem to impose constraints that are not active when other sorts of expressions are taken as antecedents. In the case of so-called bound anaphora, a quantificational NP is required to act as an antecedent in this type of link if it o-commands the relevant anaphor.

Interestingly, there seems to be at least another class of nominals that, when taken as antecedents, also imposes specific restrictions on binding configurations between anaphors and antecedents. In fact, indefinites seem to exhibit a specific behavior in this respect, even when they are not entering the much debated “donkey” type of anaphora.

For instance, it is interesting to observe, as illustrated below, that indefinites are able to act as antecedents of short- or long-distance reflexives, and of pronouns:

(1) a. A studenti shaved himselfi.
    b. Um rapazi disse-me que a Maria gosta dele próprio.
       A boy told me that Mary likes him.
    c. The boy who bought a balli emptied iti.

Nevertheless, the same does not happen with respect to nonpronouns. The contrast below suggests that the impossibility of anaphoric link in a. should be attributed not to some failure in observing binding constraints - since examples b. and c. with other types of antecedent in the same circumstance are grammatical -, but perhaps to some extra requirement on the part of the indefinite that is acting as antecedent:

(2) a. # The student who saw a girli kissed John’s girlfriendi.
    b. The student who saw heri kissed John’s girlfriendi.
    c. The student who saw Maryi kissed John’s girlfriendi.
Special restrictions on possible anaphoric links with indefinite antecedents however seem not to be restricted to nonpronouns. Although somewhat differently, the anaphoric links between pronouns and indefinite antecedents also seem to observe specific restrictions:

(3) a. The boy who bought a balli emptied iti.
    b. # The boy who bought iti emptied a balli.

As it is shown by example a., where the indefinite linearly precedes the pronoun, an indefinite can serve as antecedent of a pronoun. However, in example b., where it is the pronoun that precedes the indefinite, the anaphoric link is not possible. Since in both cases no binding constraint is violated, it is plausible to hypothesize that an additional constraint should be assigned as a property of indefinites when acting as antecedents in anaphoric links.

The issue of other possible binding constraints considered from the point of view of antecedents yet to be uncovered, seems thus to be worth examining and to be a relevant path for systematic research.

8.3 Binding Constraints in Discourse

As discussed in chapter 2 Heuristics and Constraints, binding constraints appear under the status of filters on anaphoric links (in contradistinction to heuristic preferences). When envisaged against the set of different types of filters on anaphora, binding constraints have the distinguishing feature of typically being filters on intra-sentential anaphoric links.

An interesting suggestion was recently put forward, however, by Gundel (1998) with respect to the possible existence of binding constraints also on anaphoric links holding over inter-sentential boundaries in discourse.
Inter-sentences

In her paper, Gundel aimed crucially to argue that the assumption by Centering Theory of a single backward-center involves important problems in terms of empirical adequacy. In that context, she made a suggestion that, however simple it is, it may prove to be a considerable epistemic twist in envisaging a central piece of data typically assumed to give empirical support to Centering Theory.

Contrasts between discourses such as those below have been taken in the literature on Centering Theory as crucial evidence for the existence of a preference for a pronoun to be used in the last sentence of the discourse, in detriment of a non-pronoun:

(4) a. Susan gave Betsy a pet hamster.
    b. She reminded her that hamsters were quite shy.
    c. And then she laughed.

(5) a. Susan gave Betsy a pet hamster.
    b. She reminded her that hamsters were quite shy.
    c. # And then Susan laughed.

Gundel’s suggestion, however, put forward a new way of envisaging this type of contrasts. She proposes that instead of taking it as being a visible effect of a preference for a pronoun, this contrast may actually result from a filtering constraint. In particular, this author notices (Gundel, 1998, pp.195-6) that the contrast may be seen as lending empirical support to the view that, very much like what happens with respect to some binding constraints, there is a constraint preventing a non-pronoun from having an antecedent in this kind of structural configuration.

Locality

The possible resemblance of this restriction to a constraint such as principle B of Binding Theory can in fact be advanced further still. Although we are dealing here with anaphoric links across sentence boundaries, it appears to be possible to
find a requirement at discourse level which is analogous to the locality requirement impinging on intra-sentential links.

Changing the last sentence of the discourse above for another one so that the new sentence can be interpreted as beginning a new discourse segment makes it possible in last sentence for Susan now to be grammatically referred to by the nonpronoun (Gundel, 1998, ex.(19)):

(6) a. Susan \(i\) gave Betsy a pet hamster.
   b. She \(i\) reminded her that hamsters were quite shy.
   c. And then Susan \(i\) left.

In Gundel's words, "In (5)c., where it is easy to imagine a connection to the preceding sentence (e.g. Susan laughed because she knows that Betsy had bad experiences with aggressive pets), an unpronominalized reference to Susan is unacceptable. Susan does not have to be pronominalized in (6)c., however, since this sentence, unlike its counterpart in (5), is most naturally interpreted as beginning a new discourse segment." (p.196).

Consequently, these contrasts show that it is possible that discourse boundaries between subsequent discourse segments are crucial for defining possible locality domains at discourse level, where nonpronouns cannot take antecedents from preceding sentences inside that domain.

Recess

Once the hypothesis that binding constraints may extend over inter-sentential anaphoric links receives enough plausibility, a new line of inquiry as to its empirical adequacy is justified and further similarities with binding principles may become evident. For instance, it may happen that the data below should be interpreted as evidence for the importance of recesses in binding configuration also where binding constraints apply inter-sententially.

As happens intra-sententially with reflexives, which require their antecedents to \(\text{o}\)-command them, in the following anaphoric links the pronoun cannot seemingly take a nominal as antecedent which is not in the main predication scheme of the previous sentence, but "nested" inside an argument of that predication scheme:
(7) [A mãe d\[o Pedro]\] já chegou. Ele[\_] está a falar com o Director.

the mother of the Pedro already arrived. he is at talking with the Director

Pedro’s mother already arrived. He is talking with the Director.

This suggests that, while from an inter-sentential point of view, nonpronouns seem to be ruled by a discourse analogous of principle B, pronouns in turn are likely to be ruled by a discourse analogous of principle A.

### Coordination

Under this hypothesis, we may well be able to re-explore well known puzzles for binding theory. This seems to be the case of anaphoric links holding between sentences that are coordinated.

The contrast below is attributed to Bosch (1983), in van Hoek, 1997, p.87, and shows an example where a nonpronoun cannot have a nominal as antecedent that does not o-command it, i.e. a case where although the nonpronoun is o-free, the anaphoric link is not acceptable, contra (intra-sentential) principle C:

(8) a. He[\_] lied to me and he[\_] betrayed me.

b. # He[\_] lied to me and John[\_] betrayed me.

Given data such as this, it is tempting to hypothesize that, for the sake of verifying binding constraints, the coordination of sentences creates contexts that are like sentences concatenated as units of multi-sentence discourse. Accordingly, the reason why a nonpronoun is not acceptable in (5)c. may be the same reason why it is not acceptable in (8), i.e roughly speaking a non-pronoun cannot take an antecedent which is inside the same discourse segment and which enters a main predication scheme of one of the sentential units of that discourse segment.

Interestingly, it seems that the contrast between (5)/(6) can also be repeated with respect to coordination contexts. In fact, in these contexts as well, the hypothesized locality requirement for inter-sentential anaphoric links seems to hold. Replacing the second coordinated sentence with a new one such that it may be understood as beginning a new discourse segment in (9)b., the nonpronoun can now take an antecedent from the sentence preceding it (van Hoek, 1997, p.87):

(9) a. # He[\_] lied to me and John[\_] betrayed me.

b. He[\_] lied to me, and John[\_] was my friend.
Adverbial subordination

Note that research on binding constraints has mostly focused on anaphoric links established over phrase boundaries of clauses which are concatenated under the subordination relation. When extending research to anaphoric links established in other types of clausal and sentential concatenation as well, it may be possible to analyze more effectively puzzles concerning not only coordination of sentences or sentences in discourse, but also adverbial subordination.

This may be the case of well known puzzles such as the ones depicted below. In the two examples in (10) the anaphoric links are as expected according to principles B and C. As the nonpronoun does not locally o-command the pronoun, nor does the pronoun o-command the non-pronoun, it does not matter which one linearly precedes the other. In both possibilities, the anaphoric links are acceptable:

(10) a. John\textsubscript{1} will do it if he\textsubscript{2} is around.
    b. If he\textsubscript{2} is around, John\textsubscript{1} will do it.

However, when a reversed distribution of anaphoric expressions is considered, i.e. when the pronoun occurs in the main clause and the nonpronoun in the adverbial clause, something unexpected in the light of binding theory occurs. Following, Kuno (1987, p.31), among others, only when the non-pronoun linearly precedes the pronoun is the anaphoric link grammatical:

(11) a. If John\textsubscript{1} is around, he\textsubscript{2} will do it.
    b. # He\textsubscript{2} will do it if John\textsubscript{1} is around.

This type of contrasts is highly reminiscent of contrasts such as (5)/(6) and (9), and plausibly support the research suggestion that some similarity may exist between them. Moreover, this similarity may even relate to a general pattern of similarity between coordination/discourse and adverbial subordination in as much as the contrast above is not restricted to if clauses, it also covers other adverbial clauses, as in contrast (12)/(13) (Garnham, 1989, p.367):

(12) a. John\textsubscript{1} stormed out of the room, after he\textsubscript{2} got up.
    b. After he\textsubscript{2} got up, John\textsubscript{1} stormed out of the room.

(13) a. After John\textsubscript{1} got up, he\textsubscript{2} stormed out of the room.
    b. # He\textsubscript{2} stormed out of the room, after John\textsubscript{1} got up.
It may well extend to other adverbial modifiers (van Hoek, 1997, p.85):

(14) a. Kathleen Turner falls in Love with Tom Cruise in her latest movie.

b. # She falls in Love with Tom Cruise in Kathleen Turner's latest movie.

Given this exploratory discussion, the study of binding constraints in discourse appears to be a very promising path of research.

8.4 Gaps and Linguistic Universals

An important observation that remains to be consistently addressed in the literature concerns the fact that not all binding constraints seem to be active in all languages.

English is a well-known case with respect to principle Z, insofar as it seems that this language does not have anaphoric expressions complying with that principle. Languages without anaphoric expressions exhibiting the behavior of reflexives, i.e. expressions ruled by principle A, are discussed in Levinson, 1991, pp.133-140. Furthermore, Vietnamese does not seem to have expressions complying with principle C (vd. Section 3.1).

Accessibility hierarchy

As suggested in Section 3.1, it is tempting to draw a parallel between these binding gaps and the gaps observed cross-linguistically with respect to other linguistic phenomena, in particular, those observed in Keenan and Comrie's (1977) study.

In this work, the authors carried out inter alia a comparative study of relativization in different languages. They reached the conclusion that also with
respect to relativization, it is not true that relativization is possible in all languages from positions with all grammatical functions. In particular, there seems to be a kind of hierarchy which the authors called Accessibility Hierarchy.

The hypothesis of such hierarchy results from the observation that in a given language, if relativization is not possible for a position with grammatical function $F$, then relativization is not possible for positions with more oblique grammatical functions than $F$.

As discussed in Section 3.1, this sort of hierarchy between gaps concerning other sorts of linguistic phenomena seems thus to be a non-negligible clue worth pursuing in order to understand binding gaps.

**Quantifier types**

A different sort of hint that might also be promising in this respect can be found in Loebner's (1987, pp.62ff) classification of quantifiers and his remarks concerning the possible relation, for a given language, between types of quantifiers and gaps in their lexicalization. Given the new conception of binding constraints as obliqueness quantifiers, these hints may thus be relevant for understanding gaps among binding principles for specific languages as well.

In order to establish a correspondence between semantically similar quantifiers from different squares of duality, Loebner assumes a classification of quantifiers from type 1 to type 4. He then tries to formally characterize each class of quantifiers by finding out which formal properties individuated each class.

Commenting on the several squares of duality he found, Loebner notices that there is "a clear asymmetry among the four types of quantifiers as to their lexicalization. Type 1 is lexicalized throughout and so is type 2, but there are many languages which exhibit considerable gaps in the lexicalization of type 2. Japanese and Chinese, for example, use complex expressions in most cases of universal quantification [type 2]. Type 3 [negated existential quantifiers] is synthesized in some cases of English. With respect to type 3, Indo-European languages seem to be exceptional in that they possess proper lexical units such as no, never, none, neither, nothing, etc. and even in these cases the respective words are historically compounds containing a negative prefix. Type 4 [negated universal quantifiers] is lexicalized with a single word in four examples out of twenty [duality squares found for English]." (Loebner(87):62-3).
Which one of these two suggestions – possible universal hierarchy of binding constraints vs. lexicalization of quantifier types – leads to a correct understanding of binding gaps, and if any, what new insights can be found in understanding binding theory as a whole are some interesting issues that offer themselves as worth pursuing.

### 8.5 Generation of Anaphoric Expressions

According to Dale and Reiter (1996, p.16), “it has become commonplace to view language generation as encompassing two kinds of concerns: deciding what to say, and deciding how to say it”. Both aspects involve non-trivial tasks on natural language processing, which raises the issue of how to refer to them, once the relevant entities have been chosen, i.e. which referring expressions to generate.

The issue of generating referring expression has been typically divided into two subtasks, namely the task of building a referring expression responsible in discourse for the first mention of a given entity, and the task of choosing suitable anaphoric expressions that subsequently refer to that entity.

Although non-pronouns exhibit anaphoric dependencies, this property has been overlooked when this type of expression is generated. As non-pronouns have richer semantic content, their design has been mostly addressed from the point of view of the “discriminatory power” (cf. Dale, 1989, p.71) of the information they convey for the purpose of identifying a given entity from a set of entities when it is first mentioned (see also Reiter and Dale, 1992 for further details). On the other hand, research on the generation of anaphoric expressions has concentrated mostly on pronouns, specially in non-reflexives given the usual biased concern with inter-sentential anaphoric links (vd. Dale, 1989, p.70).

Nevertheless, if one addresses the issue of generating anaphoric expressions in its full generality, the problem as to which type of anaphoric expression to choose, either in inter or intra-sentential anaphoric links, becomes a relevant one in the generation of natural language. For instance, if one has (15)a. as the input expression for generation, one would like to generate the sentence in (15)b., and exclude constructions such as c. or d.:
8.5 Generation of Anaphoric Expressions

(15) a. \( \exists e, e', s. \text{say}(e, s, e') \land \text{shave}(e', s, s) \)

b. The blond student said that he shaved himself.

c. The blond student said that the blond student shaved the blond student.

d. The blond student said that he shaved him.

Naturally, a declarative grammar integrating binding constraints such as the one we developed offers good perspectives that the grammatical knowledge encoded therein can be used to handle this issue. With this setup, given a reversible implemented grammar and the suitable grammatical representation, generating the grammatically correct anaphoric expressions appears not to present special problems.

In a language such as Portuguese with four types of anaphoric expressions, in such a simple case as (15) above, one could obtain almost as many as \( 64 = 4^3 \) possibilities, if binding constraints were not taken into account (and no concern with first mention was considered). This is illustrative of the importance of counting on binding constraints for the generation of sentences where multiple reference to the same entity occur.

However, the tricky point here is the very assumption that a suitable grammatical representation exists a priori or can be provided for the purpose of language generation.

The specification of an input language for generation is a highly non-trivial issue, requiring among other things that enough information (and no more than necessary) is present. Accordingly, possibly relevant questions relating to the issue of generating anaphoric expressions will then involve studying the grammatical information needed in the specification language for generation. An interesting research issue therefore seems to be to ascertain how much of the grammatical information we used in full grammatical analysis of binding constraints may be dispensed with for the purpose of generation if any. This issue may even be more perspicuously formulated: How much grammatical information should be included in the abstract input representation – and more crucially how can it be produced – for the purpose of applying binding constraints in generation.
8.6 Cognitive Underpinnings

Research on reference processing and anaphor resolution yielded a significant volume of work in the psycholinguistics literature (vd. Dopkins and Nordlie, 1995 for a general overview, and van der Lely and Stollwerck, 1997 and references cited therein for an overview of research on binding constraints). As may happen with other linguistic phenomena, the search for possible cognitive underpinnings of anaphora appears to be a natural endeavor for the scientific effort to uncover possibly more fundamental explanations for the facts at stake.

Polysemy and polymorphism

In order to present what can be taken as the rationale for the understanding of anaphora from a cognitive perspective, it may be interesting to start with the first sentence of Gundel et al., 1993: “One of the more interesting facts about human language is that we can use different forms to refer to the same thing, and the same form can be used to refer to many different things” (p.276). This equation helps to emphasize that one is typically tempted to see anaphora simply as an eloquent case of one of these two facets, i.e. “different forms refer to the same thing”. In fact, anaphoric links are established precisely between different expressions - different forms - that either refer to the same entity or are interpretively, closely and essentially intertwined in sustaining reference to a given entity.

The interesting point to make is that, although this facet is certainly the most noticeable one, a more comprehensive picture of anaphora in cognitive terms can be drawn when the other facet of the pair polysemy vs. polymorphism is also taken into account.
Because “the same form can be used to refer to many different things”, there certainly exists a non-negligible interpretive effort to assign a polysemic expression in a specific occurrence of the entity actually referred to from the entities that that expression could be used to refer to. Consequently, in order to minimize interpretive effort, when an entity already referred to is referred to again (as such, in part, as part of, etc.), anaphors are used to avoid going through that laborious interpretive process again. Anaphora should thus be understood not just as a singular manifestation of the circumstance that “different forms may refer the same thing”, but as a phenomenon emerging at the juncture of polysemy and polymorphism.

Polysemy appears as a necessary feature of a representational system that has finite resources to represent a virtually infinite number of entities: As there might be an undetermined number of students with yellow t-shirts, it would be unbearable to a finite mind to have a specific representational device for each such student rather than the polysemic form the student with a yellow t-shirt.

Against this background, anaphoric polymorphism appears thus as a convenient feature of a system that has to handle polysemic potential in real time: As speakers keep referring, say, to the same student with yellow t-shirt, they can avoid going through the whole interpretive process of deciding which one of the possibly infinite number of referents of the student with a yellow t-shirt should be picked out. This can be done by using a different form which is interpretively parasitic and simply signals that the same entity is often being referred to.

Setting the analysis of anaphora under this wider perspective allows one to understand the correlation, frequently mentioned in the literature, between anaphors and expressions which have weaker semantic content. But it also allows one to meet the basic rationale of proposals for a cognitive grounding of anaphora.

In this connection, it can be understood that cognitive-driven analyses of anaphora have typically seen, in one way or other, this phenomenon as a case where a given task is reformulated in terms of a simpler task. Given that some entities may be recurrently referred to, in those cases of recurrent reference the general process of polysemy reduction is taken over by specific subprocesses of anaphoric interpretation. In these cases, the cognitive search in the larger long-term or semantic memory is avoided by means of a search in the shorter working or short-term memory. Anaphora is thus seen in its cognitive roots as a phenomenon of search optimization, emerging from a drastic but possible reduction in the cognitive search space.
Search optimization in cognitive space

Having presented the cognitive rationale for anaphora, we can now turn to the discussion of the possible cognitive underpinnings of binding constraints.

Building further on the computational metaphor of cognitive life, authors such as Givón (1992) have proposed to see anaphors as “mental processing instructions” in “mental storage space” (p.22, 43). Furthermore, advancing the rationale for anaphora as search optimization, different types of anaphors are thus assumed to pick referents from different “sections” of the relevant search space. While search overhead due to polysemy is minimized by means of anaphoric polymorphism, helping to restrain some of the semantic searches to a smaller search space, anaphoric search is thus also minimized by a “divide to conquer” strategy. The search space for anaphor is “sectioned” and each section is reserved to be searched by different types of anaphors.

From this perspective, the natural metrics for search “distance” is typically assigned to attentional prominence. Given the items possibly in relevant working memory, anaphors of a given type can pick up items with a certain attentional prominence, while anaphors of another type pick items with some other degree of attentional prominence in the relevant cognitive search space (cf. Gundel et al., 1993)

Now, the obvious question is whether this general framework for the cognitive rooting of anaphora helps in uncovering possible cognitive underpinnings of binding constraints.

To be consistent with the essential tenets of this framework, in particular to maximally optimize the search for referents, one would then expect that different types of anaphors – whose antecedent entity is to be found in different “sections” of memory, ordered according to attentional prominence of the relevant items – have different sets of admissible antecedents. The strong prediction would thus be that anaphors of different types have different, disjoint sets of antecedents.

Another, weaker but still plausible prediction in this connection would be that, if the sets of antecedents turn out not to be disjoint, they would at least be expected to be successively included within each other (this is in fact one of the basic claims of Gundel et al. (1993)). If one admitted that an anaphor is of a given type such that it searches or is sensitive to items with a certain degree of attentional prominence, this would mean that anaphor is also sensitive to items with a higher degree of prominence.
Overlapping of sets of possible antecedents

Disappointingly, none of these consequences seems to hold when we look at anaphors with different binding constraints. Taking a suitable non-exempt syntactic position in a grammatical construction, one can successively fill that position with anaphors of different types with respect to the binding constraints at stake. By doing so, one can then gather the different sets of possible antecedents for each type of anaphor and observe the relationship between them.

It is then easy to see that the possible antecedents of short-distance reflexives are possible antecedents for long-distance reflexives; some possible antecedents of long-distance reflexives are possible antecedents of pronouns; and the possible antecedents of non-pronouns are possible antecedents of pronouns:

(16) a. Venn diagram of sets of possible antecedents per anaphor of different binding type wrt to a non-exempt syntactic position:

From another perspective, for a given possible antecedent, it can be said there are always at least two different types of anaphors that can take it. If one considers instead an exempt syntactic position, then even reflexives have possible antecedents that are possible antecedents of pronouns.

In any case, what is crucial to note is that the sets of admissible antecedents per anaphor type are not mutually disjoint (contra strong prediction), nor are they successively included within each other (contra weak prediction).

The paradigm of search optimization therefore does not seem to offer an added value for a cognitive explanation of binding constraints. The use of an anaphor of a specific type – i.e. with a specific binding potential – does not strictly bear
discriminative power with respect to isolating specific antecedents. In fact, for a given anaphor interpreted against a given antecedent, that anaphor can always be replaced at least by another one of a different binding type that can take the same antecedent.

Attentional prominence, recency and recess

These negative results are strengthened when other natural expectations stemming from the search optimization paradigm are also addressed. For example, attentional prominence is crucial to establish the topology of the cognitive search space. Therefore, factors influencing attentional prominence should manifest themselves in the way antecedent candidates enter different sets of possible antecedents. Among those factors, recency of mention should be expected to be the most relevant and visible one, as discussed at length in the literature (vd. among others O'Brien et al., 1997).

Nevertheless, recency seems to play no direct role in the selection of the set of possible antecedents for an anaphor of a given binding type. As illustrated below, although Carlos has been more recently mentioned than Pedro when the long-distance reflexive ele próprio is used or interpreted, it cannot be taken as possible referent of the reflexive. In contrast, the mention of Pedro is more distant in time, but Pedro is a possible antecedent of ele próprio:

(17) O Pedro disse à irmã de [o Carlos] que a Maria gosta dele próprio.

Pedro said to Carlos’s sister that Mary likes him.

The above sentence is a typical example illustrating the relevance of o-command, or in less formal terms, what we call the effect of recesses in the grammatical geometry for interpreting anaphors. A cognitive model for anaphora where attention plays a central role in setting “distances” in the search space is at odds to handle the fact that such recesses exist and that there is no immediate impact of recency of mention in the shape of the sets of possible antecedents for a given anaphor.

An attempt to overcome this difficulty can be found in works such as van Hoek, 1997. This author tries to rescue a cognitive approach for binding constraints by assuming, to a considerable extent, that the shape of the cognitive search space associated with a given grammatical construction is eventually isomorphic to the shape of its obliqueness hierarchy. While this move makes it possible to provide the empirically correct predictions, it nevertheless removes any added value from the
theoretical apparatus of the cognitive framework for the understanding of a possible specific cognitive justification for binding constraints.

Search for nonexistent items in the search space

Examining further consequences of the search optimization model, one would definitely expect it not to be possible to launch searches in the relevant cognitive search space when the item or items that are sought still do not exist in that search space. Keeping open a search procedure for something that will not exist until it is later introduced in the search space is certainly wasteful and not in the interests of optimization.

In linguistic terms this means, for instance, that pronouns linearly preceding their antecedents should not be possible, or at least should be an awkward possibility if acceptable at all. In other words, backwards anaphora would be highly problematic in this model if allowed.

Now the point is precisely that backwards anaphora is not only possible, as it apparently does not drastically involve more processing overhead than the corresponding reversed anaphoric configurations:

(18) a. John\textsubscript{i} likes himself\textsubscript{i}.
    a'. Himself\textsubscript{i}, John\textsubscript{i} likes.

    b. John\textsubscript{i} will take a course on Metaphysics in case he\textsubscript{i} gets a fellowship.
    b'. In case he\textsubscript{i} gets a fellowship, John\textsubscript{i} will take a course on Metaphysics.

    c. O Pedro\textsubscript{i} disse que a Joana acha que a Maria gosta de\textsubscript{ELE PRÛPRIO}
       the Pedro said that the Joana thinks that the Maria likes of
       Pedro said Joana thinks Maria likes him.
    c'. De\textsubscript{ELE PRÛPRIO}, a Maria disse que a Joana acha que o Pedro\textsubscript{i} gosta.
       of, the Pedro said that the Joana thinks that the Maria likes
Looking for a different kind of cognitive underpinnings

The sort of problems indicated above raises serious doubts as to the explicative and heuristic value of models for anaphora based on search optimization in cognitive space for the understanding of binding constraints. This should not be taken, however, as expressing the view that cognitively rooted factors (such as attentional prominence and recency of mention) do not play an important role in anaphor resolution, at least as preference mechanisms. Nor should it be taken as implying that binding constraints have been proved not to have any cognitive justification.

Instead, these negative results should be taken as showing that cognitive underpinnings of binding constraints seem not to be found in general or more immediate assumptions about cognitive life, but are perhaps more deeply entangled in other possibly non-trivial aspects of cognition.

We do not have an alternative paradigm for cognitive life where binding constraints have a natural justification, nor could this have been expected within the scope of the present dissertation. Nevertheless, we think that the results reported in this dissertation may provide some interesting alternative hints as to the possible cognitive grounding of binding constraints.

The rationale behind this conviction is quite simple to state in broad terms. Given the quantificational structure underlying them, binding constraints are a manifestation of quantification, a universal semantic module of natural language, which may very well be rooted in some cognitive invariant.

Quantification, semantic universals and cognition

In order to lend relevant exploratory plausibility to this suggestion, we should review a couple of successive abstraction steps.

The first relevant step is the one that takes into account the abstraction involved in grouping together different linguistic phenomena into the single semantic class of quantification. Following Loebner's words, quantification involves "a seemingly very comprehensive range of phenomena which are syntactically and
grammatically rather diverse but semantically closely enough related to form a class of their own” (Loebner, 1987, p.53).

As we discussed in chapter 4 Symmetries and Duality, the fact that some natural language terms may express quantification had already been identified in the literature as the fact that they can express a one place second order predicate. It was however with Loebner that the common characteristic of natural language quantifiers was abstracted and it was noticed that “duality is a fundamental concept in connection with quantification, but has been neglected almost completely in the relevant linguistic literature” (p.54).

The regular pattern of duality stressed by Loebner, and displayed below, had the heuristic value of allowing one to unite within the same semantic analysis a set of apparently disparate phenomena. This heuristic value was stressed in the context of a reflection on semantic universals by van Benthem (1991, p.23), who observed that the duality pattern “serves one further function [...]: it suggests a systematic viewpoint from which to search for comparative facts”; a suggestion we pursued when claiming that one should take binding constraints as one of the natural language quantification phenomena:

(19)

\[
\begin{array}{c}
Q \\
\text{inner negation}
\end{array}
\begin{array}{c}
Q\sim \\
\text{outer negation}
\end{array}
\begin{array}{c}
\sim Q \\
\text{inner negation}
\end{array}
\begin{array}{c}
\sim Q\sim \\
\text{outer negation}
\end{array}
\]

The second relevant abstraction step is to perform a new abstraction over the first one. In this case one should abstract from the different squares of duality involving natural language quantifiers, and notice the regular pattern of the relations between the corners of the square.

The different corners of duality squares, i.e. the different quantifiers, are related to each other by two third order operators, namely internal and external negation. The important point to note is that under an operation of function composition these operators give rise to an algebraic structure with very well known properties ( – stands for identity, – for external negation, ~ for internal negation, and # for \(\sim\)):
Conclusions: Summary and Outlook

(20)

\[
\begin{array}{c|cccc}
\circ & \neg & \sim & \# \\
\hline
\neg & \neg & \sim & \# \\
\sim & \sim & \# & \neg \\
\# & \# & \sim & \neg \\
\end{array}
\]

An algebraic structure of this type is known as a Klein four-group. It is a group (i.e. the relevant binary operation is associative and there is an identity element), which is commutative and has two generators (i.e. every element of the group is the result of iterating the operation over the generators).

Again, as happened with the abstraction relating to duality, this may not merely be a matter of vacuous formalization. This new abstraction degree may also have a critical heuristic impact. It may eventually allow one to recognize that natural language quantification, in general, and binding constraints, in particular, are just one manifestation of a broader class of phenomena, where other phenomena are found to pattern according to the same abstract structure.

Significantly, van Benthem signals that a similar structure was also found in cognitive psychology. In the fourth and last stage of children’s cognitive development hypothesized by Piagetian development psychology (vd. Piatelli-Palmarini, 1979), typically attained by fourteen-year olds, the so called stage of formal operational thought (vd. Gray, 1990 for an overview of formal operational thought) involved the maturing of the cognitive operations of identity, negation and converse. Apparently, there is a “Piaget’s often repeated observation that [these operations] give rise to a Klein four-group” (van Benthem, 1986, p.206).

Naturally, this similarity of the structure underlying natural language quantification in general, and binding constraints, in particular, on the one hand, and a certain class of cognitive operations, on the other, is too interesting for new research paths to pass unnoticed. According to a suggestion by van Benthem (1991), “the reason why this is a natural scheme is that further iteration serve no purpose” (p.31). Nevertheless, as this author himself notices in another paper, as interesting as this similarity certainly is, “nobody has ever been able to fit this into some significant theory” (van Benthem, 1986, p.206).

Be that as it may, the highly non-trivial parallelism between underlying abstract structures of so called cognitive formal operations and negation operations involved in linguistic quantification is promising enough, in the light of the results of the present dissertation, for it to be seriously taken as an important stimulus for research into the possible cognitive underpinnings of binding constraints.
Annex I  Implemented Grammar
parse(Phrase) --> leaf(Subphrase),
                lc(Subphrase, Phrase).

leaf(Cat) --> [Word], {word(Word, Cat)}.

leaf(Phrase) --> (Phrase ---> []).

lc(Phrase, Phrase) --> [].

lc(SubPhrase, SuperPhrase) --> [Phrase ---> [SubPhrase|Rest] <> List],
                                parse_rest(Rest),
                                cstrt(List),
                                lc(Phrase, SuperPhrase).

cstrt([]) --> [].

cstrt([X|List]) --> call1(X), cstrt(List).

call1(X) --> [], {X}.
Annex I

parse_rest([]) --> [].
parse_rest([Phrase|Phrases]) --> parse(Phrase),
parse_rest(Phrases).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Portuguese HPSG
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%============================================================================
%     Sort Hierarchy and Feature Declaration
%============================================================================
top > [bool, case, cat, con_str, ctx, gend, head, loc, marking, modsyns, nonloc, nonloc1, num, per, pform, sign, trace, vform, udrs, dist_lbs, anaphora, drscond, udc, binding].

bool > [minus, plus].
  minus > [].
  plus > [].

case > [nom, acc].
  nom > [].
  acc > [].

cat > []
  intro [subcat,
    head:   head,
    marking: marking].
ctx> []
  intro [cont: udrs, binding].

sign > [word,
       phrase,
       clitic,
       disc]
  intro [phon, synsem: synsem].

  word > [].
  phrase> []
    intro [dtrs: con_str].
  clitic> []
  disc> []
    intro [dtrs: con_str].

con_str>[head_str, coord_str].

  head_str > [only_hd,
              hd_subj, hd_subj_cps, hd_cps, hd_adj, hd_mark, hd_spr, hd_filler, hd_text].

    only_hd> [].
    hd_subj > []
      intro [hd_dtr: sign, subj_dtr:]

phrase].

    hd_subj_cps > []
      intro [hd_dtr: word, cp_dtrs].
    hd_cps > []
      intro [hd_dtr: word, cp_dtrs].
    hd_adj > []
      intro [hd_dtr: phrase, adj_dtr:]

phrase].

    hd_mark > []
      intro [hd_dtr: phrase, mark_dtr:]

word].

    hd_spr > []
      intro [hd_dtr: sign, spec_dtr:]

word].

    hd_filler > []
      intro [hd_dtr: phrase, fil_dtr:]

phrase].

    hd_text > []
      intro [hd_dtr: ctx, text_dtr].

coord_str> []
  intro [conj_hd: word, con_dtrs].
Annex I

gend > [fem, 
masc].
  fem > [].
  masc > [].

head > [func, 
  subst].
  func > [det, 
      mark, 
      clit]
  intro [spec: synsem].
  det > [].
  mark > [].
  clit > [do_clit, 
      io_clit].
  do_clit > [].
  io_clit > [].

subst > [adj, 
  noun, 
  prep, 
  reltvrzr, 
  verb, 
  punct, 
  discl]
  intro [prd: bool, 
      mod: modsyns].
  adj > [].
  noun > []
  intro [case: case, 
      cosubcat, 
      extcosubcat].
  prep > []
  intro [pform: pform, 
      cosubcat, 
      extcosubcat].
  reltvrzr[].
  verb > []
  intro [aux: bool, 
      inv: bool, 
      vform: vform].
  punct> [].
  discl> [].

loc > []
  intro [cat: cat,
      cont: udrs].

marking > [marked, 
  unmarked].
  marked>[comp,
      conj].
  comp > [quando,
      que].
  quando > [].
  que > [].

312
conj > [].
unmarked > [].

modsyns > [synsem,
none].
none > [].

nonloc1 > []
intro [que,
rel,
slash].

udc > []
intro [inhernonloc1,
to_bind: nonloc1].

binding > []
intro [list_A,
list_Z,
list_U,
list_protoU].

nonloc > []
intro [udc: udc,
binding: binding].

num > [plur,
sing].
plur > [].
sing > [].

per > [first,
second,
third].
first > []
second> []
third > [].

pform > [a, de, acerca_de, com, para, por].
a > []
de > []
acerca_de > []
com > []
para > []
por > [].

synsem> []
intro [loc: loc,
nonloc: nonloc].

trace> []
intro [phon,
Annex I

synsem: synsem].

vform> [bse, 
   fin, 
   ger, 
   inf, 
   pas, 
   prp].
   
   bse > [].
   fin > [].
   ger > [].
   inf > [].
   pas > [].
   prp > [].

dist_lbs> []
   intro [l_max, 
           l_min].

anaphora > [a_def, 
            a_quant]
   intro [refmark].
   a_def > []
   a_quant > []
   intro [antec].

udrs > [nominal, 
         non_nominal]
   intro [ls: 
          dist_lbs, 
          subord, 
          consds].
   non_nominal> [].

% Convention for the subord values: eq,= ; leq,=< ; geq,>= ; les,< ; gr,>
% Sorts nom_pA and nom_pB introduced for the sake of correct constraining
% of clitic doubling phrase

nominal > [nom_pA, 
           nom_pB, 
           nom_pC, 
           nom_pZ]
   intro [anaphora: anaphora].
   nom_pA> [].
   nom_pB> [].
   nom_pC> [].
   nom_pZ> [].

drscond > [refer, 
           genquant]
   intro [label].

refer > [preds, 
         anaph_np, 
         art]
Implemented Grammar

intro [argR].

preds > []
intro [rel, arg1, arg2, arg3].

genquant > []
intro [rel, res, scope].

art > [].

anaph_np > [].

%============================================================================
% Abbreviatory Conventions
%============================================================================

s(Comp, ArgR) := loc!(cat!(head!<verb&
    subcat![]&
    marking!Comp)&
    cont!(conds![(<drscond&
                  argR!ArgR)|Rest
                 ]))

np(ArgR) := loc!(cat!(head!<noun&
    subcat![]&
    cont!(conds![(<drscond&argR!ArgR)|Rest]
                ))

nbar(ArgR, Lab) := loc!(cat!(head!<noun&
    subcat![@detp]&
    cont!(conds![(<drscond&
                  label!Lab&
                  argR!ArgR)|Rest
                 ]))

detp := loc!cat!(head!<det&
    subcat![]).

pp(Pform, X) := loc!(cat!(head!<prep&
    pform!Pform&
    subcat![]&
    cont!conds![(argR!X)|Rest]).

nonloc_empty := nonloc!(udc!(inher!(que![]&
    rel![]&
    slash![])&
    to_bind!(que![]&
    rel![]&
    slash![]))&
    binding!(list_A![]&
    ...)
Annex I

list_Z!&
list_U!&
list_protoU![]}).

nonloc_emptyA(ListA) := nonloc!(udc!(inher!(que[])&rel[]&slash[])&to_bind!(que[])&rel[]&slash[]))&

binding!(list_A!ListA& list_Z!&
list_U!&
list_protoU![]}).

%============================================================================
% Principles
%============================================================================

principles(X):- universal_principles(X),
language_specific_principles(X).

universal_principles(X):-
  head_feature_principle(X),
  subcat_principle(X),
  spec_principle(X),
  marking_principle(X),
  non_local_feature_principle(X),
  cosubcat_principle(X),
  binding_domains_principle(X).

language_specific_principles(X):-
  constituent_order_principle(X).

%============================================================================
% The Head Feature Principle
%============================================================================

head_feature_principle( Mother& <phrase&
dtrs!(<hd_mark&
  hd_dtr!HeadDtr&
  mark_dtr!(synsem!(<quando))
}) :-
  head_feature_princ1(Mother, HeadDtr), !.

head_feature_principle( Mother& <phrase&
dtrs!(<head_str&
  hd_dtr!HeadDtr)
}) :-
  head_feature_princ(Mother, HeadDtr), !.

head_feature_principle( <phrase&
dtrs!<coord_str
}) :- true, !.
implemented grammar

head_feature_princ(synsem!>>>head!Head_value,
synsem!>>>head!Head_value).

head_feature_princl(synsem!>>>(head!(prd!Pred &
mod!Mod &
vform!Vform &
aux!Aux &
inv!Inv)),
synsem!>>>(head!(prd!Pred &
vform!Vform &
aux!Aux &
inv!Inv))).

head_feature_principle(X) :- true.

%----------------------------------------------------------------------------
% The Subcategorization Principle
%----------------------------------------------------------------------------

subcat_principle(Mother& <phrase&
dtrs!(hd_dtr!HeadDtr&
{(subj_dtr!(synsem!Synsem)) or
{spec_dtr!(synsem!Synsem)})
):-
subcat_princ(Mother,HeadDtr,[Synsem]), !.

subcat_principle(Mother& <phrase&
dtrs!(hd_dtr!HeadDtr&
cp_dtrs!CompsList)
):-
synsems_of_comps(CompsList, SynsemsList),
subcat_princ(Mother,HeadDtr,SynsemsList), !.

synsems_of_comps([],[]).

synsems_of_comps([(synsem!Synsem1)|Signs], [Synsem1|Synsems]) :-
synsems_of_comps(Signs, Synsems).

subcat_princ(synsem!>>>subcat!Mother,
synsem!>>>subcat!Head,
CompsSynsems)
:- append(Mother, CompsSynsems, Head).

subcat_principle(<disc or
{<phrase&
dtrs!(<hd_mark or
<hd_adj or
<hd_filler or
<coord_str)})
):- true.
Annex I

%----------------------------------------------------------------------------
% The SPEC Principle
%----------------------------------------------------------------------------

spec_principle(
<phrase&
  dtrs!(hd_dtr!HeadDtr&
    ((spec_dtr!Synsem) or
     (mark_dtr!Synsem)))
):-
  spec_princ(Synsem, HeadDtr), !.

spec_princ(synsem!(>>>spec!X),
           synsem!X).

spec_principle(X) :- true.

%----------------------------------------------------------------------------
% The Marking Principle
%----------------------------------------------------------------------------

marking_principle(
  Mother& <phrase&
    dtrs!(<head_str&
      mark_dtr!MarkerDtr)
  ):-
    marking_princ(Mother,MarkerDtr), !.

marking_principle(
  Mother& <phrase&
    dtrs!(<head_str&
      hd_dtr!HeadDtr)
  ):-
    marking_princ(Mother,HeadDtr), !.

marking_princ(synsem!(>>>marking!M), synsem!(>>>marking!M)).

marking_principle(X) :- true.

%----------------------------------------------------------------------------
% The Trace Principle
%----------------------------------------------------------------------------

% REM: This grammar fragment implements a restrict account of unbounded
dependency constructions (topicalization). The effect of the
trace principle is obtained by categorial restrictions on the
nodes of the phrase structure rules (vd. SCHZ).

%----------------------------------------------------------------------------

The NONLOCAL Feature Principle

Guideline: P&S(94) appendix has a problem: The NONLOCAL Feature Principle imposes
that all sentences have 'to_bind!(slash![])'; but SCH6 only
accepts sentences with 'to_bind!(slash![X])', where X is the
LOC value of the filler.

non_local_feature_principle(<disc>) :- true, !.

non_local_feature_principle(<phrase&) (synsem!(nonloc!udc!(inher!(que!QiM&
rel!RiM&
slash!SiM)&
to_bind!(que!QbM&
rel!RbM&
slash![[]]))&
dtrs!(hd_dtr!(synsem!(nonloc!udc!(inher!(que!QiH&
rel!RiH&
slash!SiH)&
to_bind!(que!QbH&
rel!RbH&
slash!SiH[]))&
fil_dtr!(synsem!(nonloc!udc!(inher!(que!QiF&
rel!RiF&
slash!SiF)&
to_bind!(que!QbF&
rel!RbF&
slash!SiF[])))
loc!FillerLoc))))
) :-
append(QiH, QiF, QiM),
append(QbH, QbF, QbM),
append(RiH, RiF, RiM),
append(RbH, RbF, RbM),
diff(SiH, [FillerLoc], SiM), !.

non_local_feature_principle(<phrase&) (synsem!(nonloc!udc!(inher!(que!QiM&
rel!RiM&
slash!SiM)&
to_bind!(que!QbM&
rel!RbM&
slash!SiM[]))&
dtrs!(hd_dtr!(synsem!(nonloc!udc!(inher!(que!QiH&
rel!RiH&
slash!SiH)&
to_bind!(que!QbH&
rel!RbH&
slash!SiH[]))&
cp_dtrs!(synsem!(nonloc!udc!(inher!(que!QiNH&
rel!RiNH&
slash!SiNH)&
to_bind!(que!QbNH&
rel!RbNH&
slash!SiNH[]))&
(synsem!(nonloc!udc!(inher!(que!QiNH1&
rel!RiNH1&
slash!SiNH1)&
to_bind!(que!QbNH1&
rel!RbNH1&
slash!SiNH1[])))))) :-


Annex I

append(QiNH, QiNH1, QiNH2), append(Qih, QiNH2, QiM),
append(QbNH, QbNH1, QbNH2), append(QbH, QbNH2, QbM),
append(RiNH, RiNH1, RiNH2), append(RiH, RiNH2, RiM),
append(RbNH, RbNH1, RbNH2), append(RbH, RbNH2, RbM),
append(SiNH, SiNH1, SiNH2), append(SiH, SiNH2, SiM),
append(SbNH, SbNH1, SbNH2), append(SbH, SbNH2, SbM), !.

non_local_feature_principle(<phrase&
  {synsem!(nonloc!udc!(inher!(que!QiM&
    rel!RiM&
    slash!SiM)&
   to_bind!(que!QiH&
    rel!RiH&
    slash!SiH&
    slash!SbH))&
   dtrs!(hd_dtr!(synsem!(nonloc!udc!(inher!(que!QiH&
    rel!RiH&
    slash!SiH&
    slash!SbH))&
   to_bind!(que!QiM&
    rel!RiM&
    slash!SiM))))&
   spec_dtr!synsem!(nonloc!udc!(inher!(que!QiNH&
    rel!RiNH&
    slash!SiNH&
    slash!SbNH))&
   to_bind!(que!QiNH&
    rel!RiNH&
    slash!SiNH&
    slash!SbNH))&
   cp_dtrs![(synsem!(nonloc!udc!(inher!(que!QiNH&
    rel!RiNH&
    slash!SiNH&
    slash!SbNH))&
   to_bind!(que!QiNH&
    rel!RiNH&
    slash!SiNH&
    slash!SbNH))]))) :-
  append(QiH, QiNH, QiM),
append(QbH, QbNH, QbM),
append(RiH, RiNH, RiM),
append(RbH, RbNH, RbM),
append(SiH, SiNH, SiM),
append(SbH, SbNH, SbM), !.

non_local_feature_principle(<phrase&
  {synsem!(nonloc!udc!(inher!(que!QiM&
    rel!RiM&
    slash!SiM)&
   to_bind!(que!QbM&
    rel!RbM&
    slash!SbM))&
   dtrs!(hd_dtr!(synsem!(nonloc!udc!(inher!(que!QiH&
    rel!RiH&
    slash!SiH&
    to_bind!(que!QbH&
      rel!RbH&
      slash!SbH)))&
   (cp_dtrs![(synsem!(nonloc!udc!(inher!(que!QiNH&
      rel!RiNH&
      slash!SiNH&
      to_bind!(que!QbNH&
        rel!RbNH&
        slash!SbNH))])))})
   spec_dtr!synsem!(nonloc!udc!(inher!(que!QiNH&
      rel!RiNH&
      slash!SiNH&
      to_bind!(que!QbNH&
        rel!RbNH&
        slash!SbNH)))))
   to_bind!(que!QiNH&
      rel!RiNH&
      slash!SiNH&
      slash!SbNH))&
   cp_dtrs![(synsem!(nonloc!udc!(inher!(que!QiNH&
      rel!RiNH&
      slash!SiNH&
      slash!SbNH))&
   to_bind!(que!QiNH&
      rel!RiNH&
      slash!SiNH&
      slash!SbNH))])))
  :-
  append(QiNH, QiH, QiM),
append(QbNH, QbH, QbM),
append(RiNH, RiH, RiM),
append(RbNH, RbH, RbM),
append(SiNH, SiH, SiM),
append(SbNH, SbH, SbM), !.

% REM: It is possible to use '>>>' only as shorthand to paths which
% can be rebuilt from the feature declaration. As 'cps_dtr'
% attribute accepts any value (in fact, a list), the path
% form 'dtrs!' to the 'nonloc!' attribute of any daughter cannot
% be state simply by using 'nonloc!'. This leads to state
% a prolog clause for each kind of daughter, hampering
% simplification of the grammar.

non_local_feature_principle(<phrase
  (synsem!(nonloc!udc!(inher!(que!QiM&
    rel!RiM&
    slash!SiM&
    to_bind!(que!QbM&
      rel!RbM&
      slash!SbM))))&
  dtrs!(hd_dtr!(synsem!(nonloc!udc!(inher!(que!QiH&
    rel!RiH&
    slash!SiH&
    to_bind!(que!QbH&
      rel!RbH&
      slash!SbH))))&
  (subj_dtr!synsem!(nonloc!udc!(inher!(que!QiNH&
    rel!RiNH&
    slash!SiNH&
    to_bind!(que!QbNH&
      rel!RbNH&
      slash!SbNH))))))))

:-
  append(QiNH,QiH,QiM),
  append(QbNH,QbH,QbM),
  append(RiNH,RiH,RiM),
  append(RbNH,RbH,RbM),
  append(SiNH,SiH,SiM),
  append(SbNH,SbH,SbM), !.

non_local_feature_principle(<phrase
  (synsem!(nonloc!udc!(inher!(que!QiM&
    rel!RiM&
    slash!SiM&
    to_bind!(que!QbM&
      rel!RbM&
      slash!SbM))))&
  dtrs!(hd_dtr!(synsem!(nonloc!udc!(inher!(que!QiH&
    rel!RiH&
    slash!SiH&
    to_bind!(que!QbH&
      rel!RbH&
      slash!SbH))))&
  (mark_dtr!synsem!(nonloc!udc!(inher!(que!QiNH&
    rel!RiNH&
    slash!SiNH&
    to_bind!(que!QbNH&
      rel!RbNH&
      slash!SbNH))))))))

:-
  append(QiNH,QiH,QiM),
  append(QbNH,QbH,QbM),
  append(RiNH,RiH,RiM),
  append(RbNH,RbH,RbM),
  append(SiNH,SiH,SiM),
  append(SbNH,SbH,SbM), !.

non_local_feature_principle(<phrase
  (synsem!(nonloc!udc!(inher!(que!QiM&
    rel!RiM&
    slash!SiM&
    to_bind!(que!QbM&
      rel!RbM&
      slash!SbM))))&
Annex I

dtrs!(hd_dtr!(synsem!(nonloc!udc!(inher!(que!QiH&
rel!RiH&
slash!SiH)&
to_bind!(que!QbH&
rel!RbH&
slash!SbH))))&
(adj_dtr!synsem!(nonloc!udc!(inher!(que!QiNH&
rel!RiNH&
slash!SiNH)&
to_bind!(que!QbNH&
rel!RbNH&
slash!SbNH))}))

:-
append(QiH,QiNH,QiM),
append(QbH,QbNH,QbM),
append(RiH,RiNH,RiM),
append(RbH,RbNH,RbM),
append(SiH,SiNH,SiM),
append(SbH,SbNH,SbM), !.

%----------------------------------------------------------------------------
%       The Semantics Principle
%----------------------------------------------------------------------------
semantics_principle(<phrase&
synsem!loc!(cat!head!<prep&
cont!ContMother)&
dtrs!(<hd_subj_cps&
cp_dtrs!][<phrase&
synsem!>>>cont!ContDaugther)))
    :- ContMother=ContDaugther, !.
semantics_principle(<phrase&
synsem!loc!cont!ContMother&
dtrs!(<hd_mark&
hd_dtr!synsem!>>>cont!ContHead&
mark_dtr!synsem!>>>marking!<que))
    :- ContMother=ContHead, !.
semantics_principle(X) :- inherit_conds(X),
inhsub_and_closformI(X),
project_ls(X),
closformII(X),
inhanaph(X).

% %
% Clause I : Inheritance of UDRS-Conditions
% %
%----------------------------------------------------------------------------

inherit_conds(<phrase&
synsem!>>>(conds!CondsMother)&
dtrs!(<head_str&
(hd_dtr!synsem!>>>(conds!CondsHead)&
((subj_dtr!synsem!>>>(conds!Conds2) or
(mark_dtr!synsem!>>>(conds!Conds2) or
(fill_dtr!synsem!>>>(conds!Conds2) or
(spec_dtr!synsem!>>>(conds!Conds2))))

322
implemented Grammar

\textbf{Clause II: Inheritance of subordination restrictions}
\textbf{Clause IV: Closed Formula Principle (I)}

\begin{verbatim}
}\texttt{\texttt{- append(CondsHead,Conds2,CondsMother).}

\texttt{\texttt{\textbf{inherit} conds(<phrase>&
\texttt{\texttt{\texttt{synsem!{}}}\texttt{>>>conds!CondsMother}}&
\texttt{\texttt{\texttt{dtrs!{(}}<\texttt{hd_cps}} \texttt{or} \texttt{<\texttt{hd_subj_cps}}\texttt{)}&
\texttt{\texttt{\texttt{(}}\texttt{hd_dtr!synsem!{}}}\texttt{>>>conds!CondsHead}}&
\texttt{\texttt{\texttt{(}}\texttt{cp_dtrs!SynsemList}}\texttt{))
\texttt{\texttt{)}}\texttt{:- conds_of_comps(SynsemList,CondsList),
\texttt{\texttt{append(CondsHead,CondsList,CondsMother).}}

\texttt{conds_of_comps([], []).}

\texttt{cons_of_comps([([(<phrase or <word>&synsem!{}}}\texttt{>>>conds!Cond)|Rest], CondsList)
\texttt{:- append(Cond, Result, CondsList),
\texttt{cons_of_comps(Rest, Result).}}

\texttt{cons_of_comps([([trace&synsem!loc!{}}}\texttt{>>>conds!Cond)|Rest], CondsList)
\texttt{:- append([Cond], Result, CondsList),
\texttt{cons_of_comps(Rest, Result).}}

\texttt{\textbf{inherit} conds(<disc>&
\texttt{\texttt{\texttt{synsem!{}}}\texttt{>>>conds!CondsMother}}&
\texttt{\texttt{\texttt{dtrs!{(}}<\texttt{hd_text}}&
\texttt{\texttt{\texttt{\texttt{(}}hd_dtr!synsem!{}}}\texttt{>>>conds!CondsHead}}&
\texttt{\texttt{\texttt{\texttt{(}}text_dtr!List)}\texttt{))
\texttt{\texttt{)}}\texttt{:- cons_of_comps(List,CondsList),
\texttt{\texttt{append(CondsHead,CondsList,CondsMother).}}

\% \%
\% Clause II : Inheritance of subordination restrictions
\% \%
\% Clause IV : Closed Formula Principle (I)
\% \%------------------------------------------------------------------------------------------------------

\texttt{inhsub_and_closformI(<phrase>&
\texttt{\texttt{\texttt{synsem!{}}}\texttt{>>>subord!RestrMother}}&
\texttt{\texttt{\texttt{dtrs!{(}}<\texttt{head_str}}&
\texttt{\texttt{\texttt{\texttt{(}}hd_dtr!synsem!{}}}\texttt{>>>subord!RestrHead}}&
\texttt{\texttt{\texttt{\texttt{(}}adj_dtr!synsem!{}}}\texttt{>>>subord!Restr2} \texttt{or}
\texttt{\texttt{\texttt{\texttt{(}}fil_dtr!synsem!{}}}\texttt{>>>subord!Restr2} \texttt{or}
\texttt{\texttt{\texttt{\texttt{(}}mark_dtr!synsem!{}}}\texttt{>>>subord!Restr2} \texttt{or}
\texttt{\texttt{\texttt{\texttt{(}}spec_dtr!synsem!{}}}\texttt{>>>subord!Restr2))))
\texttt{\texttt{)}}\texttt{:- append(RestrHead,Restr2,RestrMother).}

\texttt{inhsub_and_closformI(<phrase>&
\texttt{\texttt{\texttt{synsem!{}}}\texttt{>>>subord!RestrMother}}&
\texttt{\texttt{\texttt{dtrs!{(}}<\texttt{head_str}}&
\texttt{\texttt{\texttt{\texttt{(}}hd_dtr!synsem!{}}}\texttt{>>>cont!((ls!l_min!LminHead}}&
\texttt{\texttt{\texttt{\texttt{(}}subord!RestrHead))} &
\texttt{\texttt{\texttt{\texttt{(}}sub!dtr!synsem!{}}}\texttt{>>>cont!((ls!l_min!LminSubj}} &
\texttt{\texttt{\texttt{\texttt{(}}subord!RestrSubj))))})
\texttt{\texttt{)}}\texttt{:- append(RestrHead,RestrSubj,Restr2),
\texttt{\texttt{append(Restr2,[geq(LminSubj,LminHead)],RestrMother).}}

\texttt{inhsub_and_closformI(<phrase>&
\texttt{\texttt{\texttt{synsem!{}}}\texttt{>>>subord!RestrMother}}&
\texttt{\texttt{\texttt{dtrs!{(}}<\texttt{hd_cps}} \texttt{or} \texttt{<\texttt{hd_subj_cps}}\texttt{)}}
\texttt{\texttt{)}}
\end{verbatim}
Annex I

subord_of_comps([], []).

subord_of_comps([((<phrase or <word>&synsem!>>>subord!Cond)|Rest], CondsList) :-
   append(Cond, Result, CondsList),
   subord_of_comps(Rest, CondsList).

subord_of_comps([<trace|Rest], CondsList) :-
   subord_of_comps(Rest, CondsList).

superior_constr(LminHead, SynsemList, List) :-
   lmin_of_comps(SynsemList, LminList),
   geq_constr(LminHead, LminList, List).

lmin_of_comps([], []).

lmin_of_comps([((<phrase&synsem!>>>l_min!Cond)|Rest], CondsList) :-
   append([Cond], Result, CondsList),
   lmin_of_comps(Rest, CondsList).

lmin_of_comps([(<trace&synsem!loc!>>>l_min!Cond)|Rest], CondsList) :-
   append([Cond], Result, CondsList),
   lmin_of_comps(Rest, CondsList).

geq_constr(_, [], []).
append([Cond], Result, CondsList),
max_of_comps(Rest, Result).

geq_constr1(_, [], []).

geq_constr1(LmaxCtx, [Lmax|Rest], List) :-
  append([geq(LmaxCtx, Lmax)], Result, List),
geq_constr1(LmaxCtx, Rest, Result).

% Clause III : Projection of the distinguished labels
%----------------------------------------------------------------------------
project_ls(<phrase&
synsem!>>>ls!Ls&
dtrs!hd_dtr!synsem!>>>ls!Ls).

project_ls(<disc&
synsem!>>>ls!Ls&
dtrs!hd_dtr!cont!ls!Ls).

% Clause IV : Closed Formula Principle (II)
%----------------------------------------------------------------------------
closformII(<phrase&
(synsem!>>>loc!(cat!(head!<verb&
subcat![[]]&) &
cont!ls!l_max!Lmax) &
dtrs!hd_dtr!synsem!>>>l_max!Lmax)
) :- !.

closformII(X) :- true.

% Clause VI : Inheritance of anaphora constraints
%----------------------------------------------------------------------------
inhanaph(<phrase&
synsem!>>>anaphora!Anaphora&
dtrs!(<hd_spr&
spec_dtr!synsem!>>>anaphora!Anaphora)).
inhanaph(_) :- true, !.

%---------------------------------------------------------------
% The COSUBCAT Principle
%----------------------------------------------------------------------------
cosubcat_principle(" synsem!(loc!cat!(head!(cosubcat!Alist&
extcosubcat!Zlist)&

Annex I

subcat([])
nonloc(binding((list_A!Alists
list_Z!Zlist)) :- !.

cosubcat_principle(X) :- true.

%----------------------------------------------------------------------------
% The Binding Domains Principle
%----------------------------------------------------------------------------

binding_domains_principle(X) :- clause2(X),
clause3(X),
clause1(X).

% Clause I : Inheritance of LIST-U and LIST,protoU
%----------------------------------------------------------------------------

clause1(X) :- inher_ListprotoU(X),
listprotoU_listU(X),
inher_ListU(X).

inher_ListprotoU(<sign&
(synsem!nonloc!binding!list_protoU!MotherList&
dtrs!(hd_dtr!synsem!nonloc!binding!list_protoU!HeadList&
(spec_dtr!synsem!nonloc!binding!list_protoU!DaughterList) or
(adj_dtr!synsem!nonloc!binding!list_protoU!HeadList&
(dtrs!(hd_dtr!synsem!nonloc!binding!list_protoU!DaughterList)) or
(mark_dtr!synsem!nonloc!binding!list_protoU!DaughterList) or
(fil_dtr!synsem!nonloc!binding!list_protoU!DaughterList) or
(subj_dtr!synsem!nonloc!binding!list_protoU!DaughterList))))
) :- append(HeadList,DaughterList,MotherList), !.

inher_ListprotoU(<sign&
(synsem!nonloc!binding!list_protoU!MotherList&
dtrs!(hd_dtr!synsem!nonloc!binding!list_protoU!HeadList&
(cp_dtrs!CompDtrs))
) :- protoU_of_comps(CompDtrs,DaughterList),
append(HeadList,DaughterList,MotherList), !.

inher_ListprotoU(<disc&
(synsem!nonloc!binding!list_protoU!MotherList&
dtrs!(hd_dtr!binding!list_protoU!HeadList&
text_dtr!Sents))
) :- protoU_of_comps(Sents,DaughterList),
append(HeadList,DaughterList,MotherList), !.

inher_ListprotoU(X) :- true.

protoU_of_comps([],[]).

protoU_of_comps([|S|, Result) :-
append(ProtoU, X, Result), protoU_of_comps(S, X).
Implemented Grammar

\%-------------------------------------------------------------------------
listprotoU_listU(<disc
  synsem!nonloc!binding!(list_U!List&
  list_protoU!List)) :- !.

listprotoU_listU(X) :- true.

\%-------------------------------------------------------------------------
inher_ListU(<sign&
  (synsem!(loc!cont!anaphora!(refmark!Refmark&
    var!Var)&
    nonloc!binding!list_U!MotherList))&
  dtrs!(hd_dtr!synsem!nonloc!binding!(list_A!NbarRefmarks&
    list_U!NbarList)&
    spec_dtr!synsem!nonloc!binding!list_U!SpecList)
) :- freeze(MotherList, diff(MotherList, [Refmark,Var],NbarList)),
    freeze(MotherList, diff(MotherList, NbarRefmarks,SpecList)), !.

inher_ListU(<sign&
  (synsem!(loc!cont!anaphora!refmark!MotherRefmark&
    nonloc!binding!list_U!MotherList))&
  dtrs!(hd_dtr!synsem!nonloc!binding!(list_A!NbarRefmarks&
    list_U!NbarList)&
    spec_dtr!synsem!nonloc!binding!list_U!SpecList)
) :- freeze(MotherList, diff(MotherList, [MotherRefmark],NbarList)),
    freeze(MotherList, diff(MotherList, NbarRefmarks,SpecList)), !.

inher_ListU(<sign&
  (synsem!nonloc!binding!list_U!MotherList&
    dtrs!(hd_dtr!synsem!nonloc!binding!list_U!MotherList&
      ((adj_dtr!synsem!nonloc!binding!list_U!MotherList) or
      (mark_dtr!synsem!nonloc!binding!list_U!MotherList) or
      (fil_dtr!synsem!nonloc!binding!list_U!MotherList) or
      (subj_dtr!synsem!nonloc!binding!list_U!MotherList))))) :- !.

inher_ListU(<disc&
  (synsem!nonloc!binding!list_U!MotherList&
    dtrs!(hd_dtr!synsem!nonloc!binding!list_U!MotherList&
      text_dtr!Sents))
) :- pass_listU(MotherList,Sents), !.

inher_ListU(<sign&
  (synsem!nonloc!binding!list_U!MotherList&
    dtrs!(hd_dtr!synsem!nonloc!binding!list_U!MotherList&
      cp_dtrs!CompDtrs))
) :- pass_listU(MotherList,CompDtrs), !.

inher_ListU(X) :- true.

pass_listU(List, []).

pass_listU(List,[synsem!nonloc!binding!list_U!List | Rest]) :-
    pass_listU(List, Rest).
Clause II : Inheritance of LIST-A

clause2(X):- proj_listA(X),
               inherit_listA(X).

proj_listA(<phrase&
   (synsem!nonloc!binding!list_A!RefmarksList&
    dtrs!{hd_dtr!synsem!(loc!cat!(head!<noun&
                       subcat![[]])&
                       nonloc!binding!list_A!RefmarksList})
  or
    (subj_dtr!synsem!nonloc!binding!list_A!RefmarksList))).

proj_listA(<phrase&
   (synsem!nonloc!binding!list_A!RefmarksList)&
   dtrs!cp_dtrs!CpDtrs) :- pass_listA(RefmarksList, CpDtrs).

proj_listA(<phrase&
   (synsem!nonloc!binding!list_A!MotherList&
    dtrs!{hd_dtr!synsem!loc!cat!head!<noun&
               spec_dtr!synsem!nonloc!binding!list_A!MotherList})
  or
    (cp_dtrs!list_A!List)).

proj_listA(<phrase&
   cp_dtrs!List) :- get_at_pps(List), !.

proj_listA(X) :- true.

get_at_pps([]).

get_at_pps([synsem!(loc!cat!head!<prep&
   nonloc!binding!list_A!RefmarksList)&
   dtrs!cp_dtrs![synsem!(loc!cat!head!<noun&
                 nonloc!binding!list_A!RefmarksList)]|Rest]) :-
   get_at_pps(Rest).

pass_listA(List, []):- !.

pass_listA(List, [synsem!loc!cat!head!(<verb or <adj) | Rest]):- pass_listA(List, Rest), !.

pass_listA(A,B) :- pass_listA1(A,B).

pass_listA1(List,[synsem!nonloc!binding!list_A!List | Rest]) :-
   pass_listA(List, Rest).

inherit_listA(<phrase&
Implemented Grammar

\[
\text{implement}{\text{ed}}\ \text{Grammar}
\]

\[
\begin{align*}
\text{synsem}(\text{loc}!\text{cat}!\text{head}!(\text{verb} \text{or} \text{adj})& \\
& \text{nonloc}!\text{binding}!\text{list}_A!\text{RefmarksList})& \\
& \text{dtrs}!\text{hd_dtr}!\text{synsem}!\text{nonloc}!\text{binding}!\text{list}_A!\text{RefmarksList}).
\end{align*}
\]

\[
\text{inherit_listA}\{<\text{phrase}> \\
& \text{synsem}(\text{loc}!\text{cat}!\text{head}!<\text{noun}> \\
& \text{subcat}![[]]![]![]& \\
& \text{nonloc}!\text{binding}!\text{list}_A!\text{RefmarksList})& \\
& \text{dtrs}!\text{hd_dtr}!\text{synsem}!\text{nonloc}!\text{binding}!\text{list}_A!\text{RefmarksList}).
\}
\]

\[
\text{inherit_listA}\{<\text{disc}> \\
& \text{synsem}(\text{loc}!\text{cat}!\text{head}!<\text{disc}1& \\
& \text{nonloc}!\text{binding}!\text{list}_A!\text{RefmarksList})& \\
& \text{dtrs}!\text{hd_dtr}!\text{binding}!\text{list}_A!\text{RefmarksList}).
\}
\]

\[
\text{inherit_listA}\{<\text{phrase}> \\
& \text{dtrs}!\text{adj_dtr}!(\text{synsem}(\text{loc}!\text{cat}!\text{head}!<\text{prep}& \\
& \text{nonloc}!\text{binding}!\text{list}_A!\text{MotherList})& \\
& \text{dtrs}!\text{hd_dtr}!\text{synsem}!\text{nonloc}!\text{binding}!\text{list}_A!\text{MotherList})).
\}
\]

\[
\text{inherit_listA}(X) \isspace \text{true}.
\]

\[
\%
\%
\text{Clause III : Inheritance of LIST-Z}
\%
\%
\text{----------------------------------------------------------------------------}
\]

\[
\text{clause3}(X) \itime \text{project_listZ}(X), \\
\text{inherit_listZ}(X).
\]

\[
\text{project_listZ}\{<\text{disc}> \\
& (\text{dtrs}!\text{text_dtr}!\text{TextDtrs})& \\
& \} \itime \text{project_listZ1}(\text{TextDtrs}), !.
\]

\[
\text{project_listZ1}([]).
\]

\[
\text{project_listZ1}\{[\text{synsem}!\text{nonloc}!\text{binding}!\text{list}_A!\text{List} & \\
& \text{list}_Z!\text{List}|\text{TextDtrs}]& \\
& \} \itime \text{project_listZ1}(\text{TextDtrs}).
\]

\[
\text{project_listZ2}\{<\text{phrase}> \\
& \text{synsem}!\text{nonloc}!\text{binding}!\text{list}_Z!\text{UpList} & \\
& \text{list}_Z!\text{UpList} & \\
& \text{dtrs}!\text{adj_dtr}!\text{synsem}!\text{nonloc}!\text{binding}!\text{list}_A!\text{Alist} & \\
& \text{list}_Z!\text{Zlist})& \\
& \text{freeze}(\text{UpList}, \text{append}(\text{UpList}, \text{Alist}, \text{Zlist})).& \\
& \}
\]

\[
\text{project_listZ2}\{<\text{phrase}> \\
& \text{synsem}!\text{nonloc}!\text{binding}!\text{list}_Z!\text{List2} & \\
& \text{dtrs}!\text{cp_dtrs}!\text{CpDtrs}& \\
& \}
\]

\[
\text{project_listZ2}(\text{UpList}, [\text{synsem}!(\text{loc}!\text{cat}!\text{head}!\text{verb}& \\
& \text{nonloc}!\text{binding}!(\text{list}_A!\text{Alist} & \\
& \text{list}_Z!\text{List}))|\text{CpDtrs})
\]

329
Annex I

\[ \text{project\_listZ} (\text{UpList}, \text{CpDtrs}) \leftarrow \text{freeze(UpList, append(UpList, Alist, Zlist))}, \]
\[ \text{project\_listZ} (\text{UpList}, \text{CpDtrs}). \]

\[ \text{project\_listZ}\{\langle\text{phrase}& \text{synsem!loc!cat!\{head!<noun& subcat![]\}& cont!anaphora!\{refmark!Refmark& var!Var\}}\& nonloc!binding!list_Z!Zlist\& dtrs!hd_dtr!synsem!nonloc!binding!(\text{list_A!Alist}& list_Z!ZlistN) \} :- \]
\[ \text{freeze(Zlist, o\_command(Var,Zlist,Z1))}, \]
\[ \text{freeze(Zlist, diff(Zlist,[Refmark,Var],Z1))}, \]
\[ \text{freeze(Z1, append(Z1,Alist,ZlistN))}, !. \]

\[ \text{project\_listZ}\{\langle\text{phrase}& \text{synsem!loc!cat!\{head!<noun& subcat![]\}& cont!anaphora!refmark!Refmark}& nonloc!binding!list_Z!Zlist\& dtrs!hd_dtr!synsem!nonloc!binding!(\text{list_A!Alist}& list_Z!ZlistN) \} :- \]
\[ \text{freeze(Zlist, o\_command(Refmark,Zlist,Z1))}, \]
\[ \text{freeze(Zlist, diff(Zlist,[Refmark],Z1))}, \]
\[ \text{freeze(Z1, append(Z1,Alist,ZlistN))}, !. \]

\[ \text{o\_command}(_, [], []). \]
\[ \text{o\_command}(X, [Y|Rest], [Y|Result]) \leftarrow \text{var(Y), o\_command(X, Rest, Result)}, !. \]
\[ \text{o\_command}(X, [X|_], []) \leftarrow !. \]
\[ \text{o\_command}(X, [Y|Rest], [Y|Result]) \leftarrow \text{o\_command(X, Rest, Result)}. \]

\[ \text{project\_listZ}(X):-true. \]

\[ \text{-------------------------------------------------------------------------} \]
\[ \text{inher\_listZ}(X) \leftarrow \text{inher\_nonhead(X)}, \]
\[ \text{inher\_head(X)}. \]

\[ \text{inher\_nonhead}(\text{synsem!nonloc!binding!list_Z!Zlist}\& dtrs!((\text{subj\_dtr!synsem!nonloc!binding!list_Z!Zlist} \text{or} (\text{mark\_dtr!synsem!nonloc!binding!list_Z!Zlist} \text{or} (\text{spec\_dtr!synsem!nonloc!binding!list_Z!Zlist))))):-!. \]

\[ \text{inher\_nonhead}(X):-! . \]
\[ \text{inher\_head}\{\langle\text{phrase}& \text{synsem!loc!cat!\{head!<noun&} \]

330
Implemented Grammar

\[
\text{subcat[]}():= !.
\]

\[
\text{inher_head}([\text{synsem!nonloc!binding!list}_2\text{list}])\]
\[
\text{dtrs!hd_dtr}((\text{synsem!nonloc!binding!list}_2\text{list} \text{or}
\text{binding!list}_2\text{list})):-!.
\]

\[
\text{inher_head}(X):-\text{true}.
\]

%----------------------------------------------------------------------------
% The Constituent Order Principle
%----------------------------------------------------------------------------
% REM: As we are using context free rules, there is no strict separation
% between immediate dominance and precedence constraints. Henceforth,
% "constituent_order_principle" just reduplicates the constraints
% on precedence imposed by the rules in computing the correct
% order of the elements of the PHON list.

\[
\text{constituent_order_principle}(\text{Mother& <phrase& dtrs!hd_dtr!(phon!HeadDtr)& ((subj_dtr!Left) or (spec_dtr!Left) or (mark_dtr!Left) or (fil_dtr!Left)})):-\text{constituent_order_princ}(\text{Mother, Left, HeadDtr}).
\]

\[
\text{constituent_order_principle}(\text{Mother& <phrase& dtrs!(hd_dtr!HeadDtr& cp_dtrs!CompsList)}):-\text{phons_of_comps}(\text{CompsList, PhonsList}), \text{constituent_order_princ}(\text{Mother, HeadDtr, PhonsList}).
\]

\[
\text{phons_of_comps}([],[]).
\]

\[
\text{phons_of_comps}([\text{phon!Phon}|\text{Signs}], \text{Result}) :- \text{append}(\text{Phon, X, Result}), \text{phons_of_comps}(\text{Signs, X}).
\]

\[
\text{constituent_order_principle}(\text{Mother& <phrase& dtrs!(<hd_adj& hd_dtr!HeadDtr& adj_dtr!(phon!AdjDtr& synsem!(>>>marking!<quando>))})):-\text{constituent_order_princ}(\text{Mother, HeadDtr, AdjDtr}).
\]

\[
\text{constituent_order_princ}(\text{phon!Mother, phon!PhonDtr1, PhonDtr2}) :- \text{append}(\text{PhonDtr1, PhonDtr2, Mother}).
\]

\[
\text{constituent_order_principle}(\text{Mother& <disc& phon!PhonsList& dtrs!text_dtr!TList}):-
\]

331
Annex I

phons_of_comps(TList, PhonsList).

---

The ID Principle

---

Schema 0

dtrs! <hd_text

---

{Mother$ <disc$
  {synsem!(loc!{cat!(head!<disc1&
    subcat[]&
    marking!<unmarked>&
    nonloc!(udc!(inher!(que![[]&
      rel![[]&
      slash![[]&
      to_bind!(que![[]&
      rel![[]&
      slash![[]]))})&
    dtrs!(<hd_text&
      hd_dtr!HeadDtr&
      text_dtr![TextDtr])
  })

  --->

  [HeadDtr$ <ctx

  ,

  TextDtr$ (<phrase or <word)

  ]

  <->

  [principles(Mother)
  ].

{Mother$ <disc$
  {synsem!(loc!{cat!(head!<disc1&
    subcat[]&
    marking!<unmarked>&
    nonloc!(udc!(inher!(que![[]&
      rel![[]&
      slash![[]&
      to_bind!(que![[]&
      rel![[]&
      slash![[]]))})&
    dtrs!(<hd_text&
      hd_dtr!HeadDtr&
      text_dtr![TextDtr1,TextDtr2,TextDtr3])
  )

332
--->

```
[ 
  HeadDtr& <ctx 
} 
, 
  TextDtr1& (<phrase> 
} 
, 
  TextDtr2& <word& synsem!>>head!<punct 
} 
, 
  TextDtr3& (<phrase> 
} 
  <> 
[ principles(Mother) 
].
```

%----------------------------------------------------------------------------
%
% Schema 1
% dtrs! <hd_subj
%----------------------------------------------------------------------------
% REM:The presence of INV minus in schema 1 and 2, as required by P&S(94) for
% languages like English and Portuguese, will hamper these
% schemata to be used for coping with NPs.

```
{ 
   Mother& <phrase& synsem!>>subcat[]&
   dtrs!(<hd_subj&
     hd_dtr?HeadDtr&
     subj_dtr?SubjDtr)
}
```
Annex I

% Schema 2
% 
% \texttt{dtrs! <hd_cps}
% 
% \%------------------------------------------------------------------------
\%
\{
Mother\& \texttt{<phrase\&<synsem!>>subcat![SubjSynsem]}&
\texttt{dtrs!(<hd_cps&}
hd\texttt{_dtr!HeadDtr&}
\texttt{cp\_dtrs![CompDtr])}
\}\rightarrow
\[
\{
\texttt{HeadDtr& <word&<synsem!(>>subcat![SubjSynsem[CompsSynsems]])}
\},
\{
\texttt{CompDtr\& (<phrase or <trace)}
\}\}
\}
\}
principles(Mother)
\}.
\%
\%
\%

\{
Mother\& \texttt{<phrase\&<synsem!>>subcat![SubjSynsem]}&
\texttt{dtrs!(<hd_cps&}
hd\texttt{_dtr!HeadDtr&}
\texttt{cp\_dtrs![CompDtr1,CompDtr2])}
\}\rightarrow
\[
\{
\texttt{HeadDtr& <word&<synsem!(>>subcat![SubjSynsem[CompsSynsems]])}
\},
\{
\texttt{CompDtr1\& (<phrase)}
\},
\{
\texttt{CompDtr2\& (<phrase&
<synsem!(>>head!<prep}))}
\}
\}
\}
principles(Mother)
\}.
\%
\%
\%

%------------------------------------------------------------------------
% % Schema 3
% % 
% \texttt{dtrs! <hd_subj_cps}
% 
% \%------------------------------------------------------------------------
% % REM:The P&S(94):Ch.9 split of the SUBCAT list allows to say that

334
% prepositions does not subcategorize for subjects. However, since
% schema 2 imposes that the mother node SUBCAT value is non
% empty, PP cannot be analyzed by means of that schema, and there is no
% schema that renders a “natural” account of PPs: schema 1 and 3 would
% say in the sort of its DTRS attribute that the preposition complement
% is a subject. The following rule is an adaptation of schema 3 to circumvent
% this problem.

\[
\begin{array}{c}
\text{Mother} \rightarrow \text{HeadDtr} \rightarrow \text{CompDtr} \\
\text{principles(Mother)}
\end{array}
\]
Annex I

\%----------------------------------------------------------------------------
\% Schema 5
\% dtrs! <hd_adj
\%----------------------------------------------------------------------------

{Mother& <phrase&
  synsem!>>>subcat[]&
  dtrs!{<hd_adj&
    hd_dtr!HeadDtr&
    adj_dtr!AdjDtr}
}
--->
[HeadDtr& <phrase &
  synsem!(HeadSynsem &<synsem&
    @s{<unmarked,X}&
    nonloc!udc!(to_bind!(slash[])))
  ],
[AdjDtr& <phrase&
  synsem!{(>>mod!HeadSynsem&R
    >>conds!Conds})
  ]
]<>
[principles(Mother),
invert(Conds,[(arg1!X)|_])
].

invert([],[]).

invert([X|Rest],InvertedList) :- invert(Rest,Result),
  append(Result,[X],InvertedList).

\%---------------------------------------------------------------------------
\% Schema 6
\% dtrs! <hd_filler
\%---------------------------------------------------------------------------

{Mother& <phrase&
  synsem!>>>subcat[]&
  dtrs!{<hd_filler&
    hd_dtr!HeadDtr&
    fil_dtr!FillerDtr}
}
--->
[
{FillerDtr& <phrase&
synsem!(loc!FillerLoc&
    >>>head!(<noun or <prep}))
}
,
(HeadDtr& <phrase&
    synsem!(:>subcat![]&
    >>>head!(<verb&
        vform!<fin>&
        nonloc!udc!(inher!{slash!HeadSlash})})
    }
    )
]<>
[member1(HeadSlash, FillerLoc),
principles(Mother)
].

%---------------------------------------------------------------------------
%                         Schema 7
% dtrs! <hd_spr
%---------------------------------------------------------------------------
(
    Mother& <phrase&
        synsem!(:>subcat![]&
        dtrs!(<hd_spr&
            hd_dtr!HeadDtr&
            spec_dtr!SpecDtr)
    }
    )
    )
[ SpecDtr& <word
    ]
,
(HeadDtr& <sign&
    synsem!(:>subcat![@detp]
    )
    )
]<>
[principles(Mother)
].

%============================================================================
%                      Lexical Rules
%============================================================================

% REP: Lexical rules is taken here in the sense of Oliva(94): schemata for morphological structures

%----------------------------------------------------------------------------
% Clitic Doubling

337
Annex I

% Direct Object Clitics

{<word>
  (phon!PhonMother&
   synsem!{loc!{(cat!{head!Head&
      subcat!SubcatMother&
      marking!Marking}&
      cont!Cont}&
     nonloc!Nonloc})
  --->
  [<word>(phon!PhonVerb&
    synsem!{loc!{(cat!{(head!(<verb & Head)&
        subcat![Subj, & np(X) | Rest]&
        marking!Marking}&
        cont!Cont}&
       nonloc!Nonloc}))
  ]
  ,
  [<clitic>(phon!PhonClit&
    synsem!{head!{(do_clit&
        spec!>>>cont!A})
  ])
  }
  <>
  append(PhonVerb, PhonClit, PhonMother),
  append([Sub],
  loc!{(cat!{(head!{(prep&
    pform!<a>&
    subcat!!} &
    cont!(A&conds![(argR!X) | _]))
  ],
  Rest, SubcatMother})
}.

% Indirect Object Clitics

{<word>
  (phon!PhonMother&
   synsem!{loc!{(cat!{head!Head&
      subcat!SubcatMother&
      marking!Marking}&
      cont!Cont}&
     nonloc!Nonloc})
  }
\textbf{Implemented Grammar}

$$\{\text{---}>\}
\begin{array}{l}
\{ \\
\text{word}\&
\begin{array}{l}
\text{synsem!}(\text{loc!}(\text{cat!}(\text{head!}(\text{<verb \& Head})\
\text{subcat!}[\text{Subj,OD,0pp(<a,X)|Rest}\&
\text{marking!Marking})\&
\text{cont!Cont})\&
\text{nonloc!Nonloc}))
\end{array}
\}
\}
\{ \\
\text{clitic}\&
\begin{array}{l}
\text{synsem!}>>\text{head!}(\text{<io_clit&}
\text{spec!}>>\text{cont!A})
\end{array}
\}
\}
\}
\}
\}
\end{array}
\end{array}
\equiv
\begin{array}{l}
\text{append(PhonVerb, PhonClit, PhonMother),}
\text{append([Subj, OD, loc!(cat!}(\text{head!}(\text{<prep&}
\text{pform!<a>&
\text{subcat![]})&
\text{cont!(A&conds![(argR!X)|_]}))
\text{], Rest, SubcatMother})
\}.}
\begin{array}{l}
\%============================================================================
\%
\%                       Relational constraints
\%
\%============================================================================
\end{array}
\begin{array}{l}
\text{append([],X,X).}
\text{append([X|Y],Z,[X|W]) :- append(Y,Z,W).}
\text{diff([],_,[]).}
\text{diff([X|R],[],[X|R]).}
\text{diff([X|R],[Y|S],[X|R]) :- member1([Y|S],X), !, diff(R,[Y|S],D).}
\text{diff([X|R],[Y|S],[X|D]) :- diff(R,[Y|S],D).}
\text{member1([X|_],X).}
\text{member1([_|R],X) :- member1(R,X).}
\end{array}
\begin{array}{l}
\%============================================================================
\end{array}
\end{array}
Annex I

Lexicon

---

Determiners

---

word(o,
<word&
  (phon![o]&
    synsem!(loc!(cat!(head!(<det&
      spec!@nbar(Refmark,Label2))&
      subcat![[]&
      marking!<unmarked)&
    cont!{(nom_pC&
      ls!(<dist_lbs&
      l_max!Label1&
      l_min!(Label2)&
      subord!eq(Label1,Label2))&
    conds!{
      (cart&
      label!Label1&
      argR!Refmark)
    &
      anaphora!(<a_def&
        refmark!Refmark&
        antec!Antec))}&
    nonloc!(udec!(inh!(que![[]&
      rel![]&
      slash![])&
    to_bind!(que![[]&
      rel![]&
      slash![]))&
    binding!(list_A!&
      list_Z!list&
      list_U!list&
      list_protoU!(Refmark)))
  )
}):
  random(1, 999, Refmark),
  freeze(Ulist, freeze(Zlist, non_ocommand(Ulist, Zlist, Refmark, Antec)))
).

non_ocommand(Ulist, Zlist, Refmark, Antec):
  ocommand(Refmark, Zlist, Ocommanders),
  append(Ocommanders, [Refmark], Ocom),
  diff(Ulist, Ocom, Antec).

% REM: It may happen that the index randomly assigned to a reference
% marker is identical to another index also randomly assigned.
% A way to improve this point would be build on the predicate randseg/3.
% The assignment of individual indices would thus have to be "coordinated".
% Another way is to prefix the indices randomly generated with
% a unique value assigned in the corresponding rule.

word(a,
<word&
  (phon![a]&
    synsem!(loc!(cat!(head!(<det&
random(1, 999, Refmark),
freeze(Ulist, freeze(Zlist, non_ocommand(Ulist, Zlist, Refmark, Antec)))
).

word(as, <word
  (phon!as)&
  synsem!(loc!(@cat!(@head!(@det&
    spec!@nbar(Refmark,Label2))&
    subcat![]&
    marking!<unmarked>&
    cont!(<nom_pC&
      ls!(<dist_lbs&
        l_max!Label1&
        l_min!Label2)&
      subord!eq(Label1,Label2)&
      conds![
        <art&
          label!Label1&
          argR!Refmark)&
        anaphora!(<a_def&
          refmark!Refmark&
          antec!Antec)))&
      nonloc!(udc!(inher!{que![]}&
        rel![]&
        slash![{}])&
      to_bind!(que![]}&
        rel![]&
      slash![{}]))&
    binding!(list_A!&
      list_Z!Zlist&
      list_U!Ulist&
      list_protoU![Refmark]))
  )
  )
}:-
random(1, 999, Refmark),
freeze(Ulist, freeze(Zlist, non_ocommand(Ulist, Zlist, Refmark, Antec)))
.

word(as, <word
  (phon!as)&
  synsem!(loc!(@cat!(@head!(@det&
    spec!@nbar(Refmark,Label2))&
    subcat![]&
    marking!<unmarked>&
    cont!(<nom_pC&
      ls!(<dist_lbs&
        l_max!Label1&
        l_min!Label2)&
      subord!eq(Label1,Label2)&
      conds![
        <art&
          label!Label1&
          argR!Refmark)&
        anaphora!(<a_def&
          refmark!Refmark&
          antec!Antec)))&
      nonloc!(udc!(inher!{que![]}&
        rel![]&
        slash![{}])&
      to_bind!(que![]}&
        rel![]&
      slash![{}]))&
    binding!(list_A!&
      list_Z!Zlist&
      list_U!Ulist&
      list_protoU![Refmark]))
  )
  )
}:-
random(1, 999, Refmark),
freeze(Ulist, freeze(Zlist, non_ocommand(Ulist, Zlist, Refmark, Antec)))
.

341
Annex I

word(um,
  <word&
    (phon![um]
      synsem!(loc!(cat!(head!(<det&
        spec@nbar(Refmark,Label2))&
        subcat[])
      marking!<unmarked>)&
      cont!(<nominal&
        ls!(<dist_lbs&
          _max!Label1&
          _min!Label2)&
        subord![eq(Label1,Label2)]&
        cons![
          (<art&
            label!Label1&
            argR!(Refmark))
          &
          anaphora!(<anaphora&
            refmark!(Refmark))]&
        nonloc!(udc!(inher!(que[]])&
          rel![[])&
          slash![[]])&
        to_bind!(que[]]&
          rel![[])&
          slash![[]])&
        binding!(list_A!_&
          list_Z!_&
          list_U!_&
          list_protoU![Refmark]))
      )
    ) :-
      random(1,999,Refmark)
      .

word(cada,
  <word&
    (phon![cada]
      synsem!(loc!(cat!(head!(<det&
        spec@nbar(Var,Label11))&
        subcat[])
      marking!<unmarked>)&
      cont!(<nominal&
        ls!(<dist_lbs&
          _max!Label1&
          _min!Label12)&
        subord![gr(Label1,Label11),gr(Label1,Label12)]&
        cons![
          (<art&
            label!Label11&
            argR!Var),
          (<art&
            label!Label12&
            argR!Var2),
          (<genquant&
            label!Label1&
            res!Label11&
            scope!Label12),
          (<preds&
            label!Letype&
            rel!abstraction&
            argR!Refmark&
            arg1!Var&
            arg2!Var2)
          &
          anaphora!(<a_quant&
          binding!(list_A!_&
            list_Z!_&
            list_U!_&
            list_protoU![Refmark]))
      )
    ) :-
      random(1,999,Refmark)
      .

342
refmark!Refmark& % e-type
var!Var)))&
nonloc!(udc!(inher!(que![])&
rel![[]])&
slash![[]])&
to_bind!(que![]&
rel![[]]&
slash![[]])&

binding!(list_A&
list_Z&
list_U&
list_protoU![Refmark,Var])
)
}
:- random(1,999,Refmark),
random(1,999,Var).

word(pedro,
<word&
(phon![pedro]&
synsem!(loc!(cat!(head!){noun&
prd!<minus&
case!<nom&
mod!<none>&
subcat![@detp]&
marking!<unmarked>&
cont!{<nominal&
ls!{<dist_lbs&
l_min!L2}&
subord![[]&
conds!{(<preds&
label!L2&
rel!name&
argR!ArgR&
arg1!name_pedro)
})&
}@nonloc_emptyA([[]))}
).

word(carlos,
<word&
(phon![carlos]&
synsem!(loc!(cat!(head!){noun&
prd!<minus&
case!<nom&
mod!<none>&
subcat![@detp]&
marking!<unmarked>&
cont!{<nominal&
ls!{<dist_lbs&
l_min!L2}&
subord![[]&
conds!{(<preds&
label!L2&
rel!name&
argR!ArgR&
arg1!name_carlos)
Annex I

```
word(maria, <word
  (phon[maria] &
   synsem:loc!(cat!{head!{<noun
     prd!<minus
     case!<nom
     mod!<none} &
     subcat![@detp] &
     marking!<unmarked} &
     cont!{<nominal
       ls!{<dist_lbs
         l_min!L2} &
       subord![]} &
     conds![
       <preds
         label!L2 &
         rel!name &
         argR!ArgR &
         arg1!name_maria
       ]}) &
  @nonloc_emptyA([]}))
).

word(estudante, <word
  (phon[estudante] &
   synsem:loc!(cat!{head!{<noun
     prd!<minus
     case!<nom
     mod!<none} &
     subcat![@detp] &
     marking!<unmarked} &
     cont!{<nominal
       ls!{<dist_lbs
         l_min!L2} &
       subord![]} &
     conds![
       <preds
         label!L2 &
         rel!estudante &
         argR!ArgR
       ]}) &
  @nonloc_emptyA([]))
).

word(fotos, <word
  (phon[fotos] &
   synsem:loc!(cat!{head!{<noun
     prd!<minus
     case!<nom
     mod!<none} &
     subcat![@detp, @pp{<de,X}] &
     marking!<unmarked} &
     cont!{<nominal
       ls!{<dist_lbs
         l_min!L2} &
       subord![]} &
     conds![
       <preds
         label!L2 &
Implemented Grammar

```plaintext
word{descric5a3o, word{
    phon![descric5a3o]&
    synsem!(loc!(cat!(head!(<noun{
        prd!<minus&
        case!<nom&
        mod!<none>&
        subcat![@detp, @pp(<de,X), @pp(<por,Y)>]&
        marking!<unmarked>&
        cont![@nominal&
            ls![@dist_lbs&
                l_min!L2]&
            subord![&
                conds![
                    (<preds&
                        l_min!L2&
                        rel!descrever&
                        argR!Event&
                        arg1!X&
                        arg2!Y)
                    ]]&
            @nonloc_emptyA({X,Y}))]
        ]}&
        @nonloc_emptyA({X}))
    }}&

%----------------------------------------------------------------------------
%
% Noun Phrases
%
%----------------------------------------------------------------------------

word{ele, <phrase{
    phon![ele]&
    synsem!(loc!(cat!(head!(<noun{
        cosubcat!Alist&
        extcosubcat!Zlist&
        prd!<minus&
        case!<nom&
        mod!<none>&
        subcat![&
            marking!<unmarked>&
            cont![@nom_pB&
                ls![@dist_lbs&
                    l_max!Label&
                    l_min!Label]&
                subord![&
                    conds![[@anaph_np&
                        label!Label&
                        argR!Reflabel&
                        rel![]&
                        anaphora![@a_def&
                            refmark!Reflabel&
                            antec!Antec])&
                        nonloc!(udc!(inher!(que![]&
                            rel![]&
                            slash![]))&
                        to_bind!(que![]&
                            rel![]&
```

345
annex I

\[ \text{binding!}\{\text{list}_\text{A}\text{Alist}\&\text{list}_\text{Z}\text{Zlist}\&\text{list}_\text{U}\text{Ulist}\&\text{list}_\text{protoU}\{\text{Refmark}\}\}\&\text{dtrs!}\langle\text{only}_\text{hd}\rangle\] =
random(1, 999, \text{Refmark}),
freeze(\text{Ulist}, \text{non}\_\text{loc}_\text{ocommand}(\text{Ulist}, \text{Alist, Refmark, Antec})).

\text{non}\_\text{loc}_\text{ocommand}(\text{Ulist, Alist, Refmark, Antec}): -
ocommand(\text{Refmark}, \text{Alist, Loc}_\text{ocommanders}),
append(Loc_\text{ocommanders}, [\text{Refmark}], \text{Loc}),
diff(Ulist, Loc, Antec).

\text{word}\{\text{ele}_\text{prolprio}, <\text{phrase}\&</\text{phon}\{\text{ele}_\text{prolprio}\&</\text{synsem}\{</\text{loc}\langle</\text{cat}\langle</\text{head}\langle</\text{nour}\&</\text{cosubcat}\text{Alist}\&</\text{extcosubcat}\text{Zlist}\&</\text{prd}\langle</\text{minus}\&</\text{case}\langle</\text{mod}\langle</\text{none}\&</\text{subcat}\langle</\text{marking}\langle</\text{cont}\langle</\text{ls}\langle</\text{l_max}\langle</\text{l_min}\langle</\text{subord}\langle</\text{conds}\langle</\text{anaphora}\langle</\text{a_def}\&</\text{refmark}\{</\text{antec}\langle</\text{Antec}\}\rangle</\text{nonloc}\langle</\text{udc}\langle</\text{rel}\langle</\text{slash}\langle</\text{to_bind}\langle</\text{dtrs}\langle</\text{only}\_\text{hd}\rangle\] =
\text{random}(1, 999, \text{Refmark}),
freeze(\text{Ulist, Alist, ambig(Cont, Refmark, Ulist, Zlist, Alist, Antec)})))

\text{ambig}(\text{Cont, Refmark, Ulist, Zlist, Alist, Antec}) :-
\text{Cont} = <\text{nom}_\text{pZ},
ocommand(\text{Refmark}, \text{Zlist, Antec}), !.

\text{ambig}<\text{nom}_\text{pA}, \text{Refmark, Ulist, Zlist, Alist, Antec}): -
ocommand(\text{Refmark, Alist, Antec}), !.

\text{ambig}<\text{nom}_\text{pB}, \text{Refmark, Ulist, Zlist, Alist, Antec}):-
non_loc_ocommand(Ulist, Alist, Refmark, Antec).

word(si_pro1prio,
   <phrase&
   (phon![si_pro1prio]&
   synsem!(loc!(cat!(head!(<noun&
      cosubcat!Alist&
      extcosubcat!Zlist&
      prd!<minus&
      case!<nom&
      mod!<none&
      subcat[]&
      marking!<unmarked>&
      cont!(<nom_p&
         ls!<<dist_lbs&
         l_max!Label&
         l_min!Label&
      subord[]&
      conds![{anaph_np&
         label!Label&
         argR!Refmark)
   &
      anaphora!(<a_def&
         refmark!Refmark&
         antec!Antec))&
   nonloc!(udc!(inher!(que![&
      rel[]&
      slash[]&
      to_bind!(que![&
      rel[]&
      slash[]&
      binding!(list_A!Alist&
      list_Z!Zlist&
      list_U[_&
      list_protoU![Refmark])&
   dtrs!<only_hd)
}) :- random(1,999,Refmark),
   freeze(Alist,ocommand(Refmark,Alist,Antec)).

ocommand(_, [], []).

ocommand(X, [Y|Rest], [Y|Result]) :- var(Y), ocommand(X, Rest, Result), !.

ocommand(X, [X[_], []]) :- !.

ocommand(X, [Y|Rest], [Y|Result]) :- ocommand(X, Rest, Result).

%---------------------------------------------------------------------------
% Verbs
%---------------------------------------------------------------------------

word(saiu,
   <word&
   (phon![saiu]&
   synsem!(loc!(cat!(head!(<verb&
      prd!<plus&
      mod!<none&
   s
Annex I

```plaintext
word(entrar,
  <word
    (phon![entrar]
      synsem!loc!(cat!(head!(<verb
        prd!<plus
        mod!<none
        vform!<fin
        aux!<minus
        inv!<minus
        subcat![@np(X)]
        marking!<unmarked>
        cont!(@non_nominal
          ls!(<dist_lbs
            l_min!L2)
          subord![]
          cnds![
            (<preds
              label!L2
              rel!sair
              argR!Ev
              arg1!Ev
              argl!X)
          ])
        )@nonloc_emptyA([X])))
  )
).

word(viu,
  <word
    (phon![viu]
      synsem!loc!(cat!(head!(<verb
        prd!<plus
        mod!<none
        vform!<fin
        aux!<minus
        inv!<minus
        subcat![@np(X), @np(Y)]
        marking!<unmarked>
        cont!(@non_nominal
          ls!(<dist_lbs
            l_min!L2)
          subord![]
          cnds![
            (<preds
              label!L2
              rel!entrar
              argR!Ev
              arg1!Ev
              argl!X)
          ])
        )@nonloc_emptyA([X])))
  )
).
```

348
Implement*ed Grammar

\[
\text{word}(\text{ouviu}, \text{arg1}!X& \text{arg2}!Y) ) \quad @\text{nonloc_emptyA}([X, Y]))
\]

\[
\text{word}(\text{barbeou}, \text{arg1}!X& \text{arg2}!Y) ) \quad @\text{nonloc_emptyA}([X, Y]))
\]

\[
\text{word}(\text{gosta}, \text{arg1}!X& \text{arg2}!Y) ) \quad @\text{nonloc_emptyA}([X, Y]))
\]

349
word(disse,
<word&
(phon![disse]
 synsem!(loc!(cat!(head!(<verb&
 prd!<plus&
 mod!<none&
 vform!<fin&
 aux!<minus&
 inv!<minus&
 subcat!(@np(X), @s(<que,EvSub)>
 marking!<unmarked>&
 cont!(<non_nominal&
 ls!(<dist_1bs&
 l_min!L2)&
 subord![]&
 conds![
 (<preds&
 label!L2&
 rel!dizer&
 argR!Ev&
 arg1!X&
 arg2!EvSub)
 ]))&
 @nonloc_emptyA([[X]])
 )
 )
)
).

word(convenceu,
<words
 (phon![convenceu]&
 synsem!(loc!(cat!(head!(<verb&
 prd!<plus&
 mod!<none&
 vform!<fin&
 aux!<minus&
 inv!<minus&
 subcat!(@np(X), @np(Y), @pp(<de,EvSub)>
 marking!<unmarked>&
 cont!(<non_nominal&
 ls!(<dist_1bs&
 l_min!L2)&
 subord![]&
 conds![
 (<preds&
 label!L2&
 rel!convencer&
 argR!Ev&
 arg1!X&
 arg2!Y&
 arg3!EvSub)
 ]))&
 @nonloc_emptyA([[X,Y]])
 )
)
Annex I

%----------------------------------------------------------------------------
% Prepositions
%----------------------------------------------------------------------------

word(de, <word& (phon![de]& synsem!{loc!{(cat!{head!{<prep& prd!<plus& mod!<none& pform!<de>& subcat![<np(X)>] or @s(<que,_)]& marking!<unmarked>& cont!{<non_nominal& lsl!<dist_lbs& l_min!L2]& subord![]& conds![[]])& @nonloc_emptyA([X])})
})
.

word(a, <word& (phon![a]& synsem!{loc!{(cat!{head!{<prep& prd!<plus& mod!<none& pform!<a>& subcat![<np(X)>]& marking!<unmarked>& cont!{<non_nominal& lsl!<dist_lbs& l_min!L2]& subord![]& conds![[]])& @nonloc_emptyA([X])})
})
.

word(com, <word& (phon![com]& synsem!{loc!{(cat!{head!{<prep& prd!<plus& mod!<none& pform!<com>& subcat![<np(X)>]& marking!<unmarked>& cont!{<non_nominal& lsl!<dist_lbs& l_min!L2]& subord![]& conds![[]])& @nonloc_emptyA([X])})
})
.

352
word {acerca_de, <words
  [phon! [acerca_de] &
   synsem! (loc! (cat! (head! (<preps
      prd! <plus &
      mod! <none &
      pform! [acerca_de] &
      subcat! @np(X) &
      marking! <unmarked> &
      cont! (<non_nominal &
        ls! (<dist_lbs &
        l_min! L2) &
        subord! [ &
        conds! [ ] ) &
      @nonloc_emptyA ([X] ) ) ) ) ) )
  }

word {por, <words
  [phon! [por] &
   synsem! (loc! (cat! (head! (<preps
      prd! <plus &
      mod! <none &
      pform! [por] &
      subcat! @np(X) &
      marking! <unmarked> &
      cont! (<non_nominal &
        ls! (<dist_lbs &
        l_min! L2) &
        subord! [ &
        conds! [ ] ) &
      @nonloc_emptyA ([X] ) ) ) ) )
  }

%----------------------------------------------------------------------------
% Markers
%----------------------------------------------------------------------------

word {que, <words
  [phon! [que] &
   synsem! (loc! (cat! (head! (<marks
      spec! @s (<unmarked, _)) &
     subcat! [ &
     marking! <que> &
     cont! (<non_nominal &
       ls! (<dist_lbs &
       l_min! L2) &
       subord! [ &
       conds! [ ] ) &
     @nonloc_empty))
  )

word {quando,
Annex I

(word(
  phon!['quando']&
  synsem!{loc!{cat!{head!(<mark&
    spec!@s(<unmarked,EvSub})&
    subcat[]&
    marking!'<quando>'&
    cont!{<non_nominal&
      ls!(<dist_lbs&
        _min!L2)&
      subord[]&
      conds![
        <preds&
          label!L2&
          rel!simult&
          arg1!X&
          arg2!EvSub}
      ]}&
    @nonloc_empty})
  })

%----------------------------------------------------------------------------
%
% Clitics
%
%----------------------------------------------------------------------------

word(-se, <clitic&
  (phon!['-se']&
   synsem!{loc!{cat!{head!{<do_clit&
     spec!>cont!{<nom_pA&
       ls!(<dist_lbs&
         _max!L&
         _min!L2)&
       conds![
         <anaph_np&
           label!L&
           argR!X}]]&
       anaphora!{<a_def&
         refmark!X}}})
   subcat[]&
   marking!<unmarked>&
   conds!{<non_nominal&
     ls!(<dist_lbs&
       subord[]&
       conds[]})&
   @nonloc_empty})

).

word(-o, <clitic&
  (phon!['-o']&
   synsem!{loc!{cat!{head!{<do_clit&
     spec!>cont!{<nom_pB&
       ls!(<dist_lbs&
         _max!L&
         _min!L)&
       conds!{<anaph_np&
         label!L&
         argR!X}]]&
       anaphora!{<a_def&
         refmark!X}}})
   @nonloc_empty})

).

354
Implemented Grammar

```
(word(-1he, <clitic&
    (phon!'-1he')&
    synsem!loc!(cat!(head!={<io_clit&
        spec!>(cont!={<nom_pB&
            ls!={<dist_lbs&
                1_max!1&
                1_min!1&
            conds!={<anaph_np&
                label!X&
                argR!X})&
            anaphora!={<a_def&
                refmark!X})}&
        subcat![]&
        marking!<unmarked)&
        cont!={<non_nominal&
            ls!={<dist_lbs>&
            subord![]&
            conds![]})&
        @nonloc_empty)})
    )
    )
    )
)

--- > [].
```

%----------------------------------------------------------------------------
%       Trace
%----------------------------------------------------------------------------

```
Annex I

% Punctuation
%-------------------------------------------------------------------
word('.', <word& (phon!['.']& synsem!{loc!{cat!{head!<punct& subcat![[]& marking!<unmarked}& cont!{(non_nominal& ls!{(dist_lbs& l_max!L& l_min!L}& subord![]& conds![]}& @nonloc_empty)})& @nonloc_empty)})&
}
.

%-------------------------------------------------------------------
% Contexts
%-------------------------------------------------------------------
word(ctx0, <ctx& cont!{(non_nominal& ls!{(dist_lbs& l_max!lmax_ctx& l_min!lmin_ctx)& subord![]& conds![]}& binding!(list_A![[]& list_Z![[]& list_U![]& list_protoU![{}])
}
.

word(ctx1, <ctx& cont!{(non_nominal& ls!{(dist_lbs& l_max!lmax_ctx& l_min!lmin_ctx)& subord![]& conds![](<preds& label!L2& rel!name& argR!Ind& argI!name_pedro)

})& binding!(list_A![[]& list_Z![[]& list_U![]& list_protoU![Ind])
}):- random(1,999,Ind)
.

356
Example 1

<table>
<thead>
<tr>
<th>Phrase</th>
<th>Literal translation</th>
<th>Translation</th>
<th>Input string</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ele</td>
<td>he</td>
<td>He</td>
<td>ele</td>
</tr>
</tbody>
</table>
Output in ProFIT notation:

```
phrase&
  phon![ele]&
  synsem!<synsem&
  loc!<loc&
  cat!<cat&
  subcat![]&
  head!<noun&
    case!<nom&
    cosubcat!_R&
    extcosubcat!_I&
    prd!<minus&
    mod!<none&
    marking!<unmarked&
  cont!<nom_pB&
    ls!<dist_lbs&
      l_max!_Q&
      l_min!_Q&
  subord![]&
    conds![_anaph_np&
      label!_Q&
      argR!659
    ]&
  anaphora!<a_def&
    refmark!659&
  nonloc!<nonloc&
  udc!<udc&
    inher!<nonloc1&
      que!![]&
      rel!![]&
      slash!![]&
    to_bind!<nonloc1&
      que!![]&
      rel!![]&
      slash!![]&
  binding!<binding&
    list_A!_R&
    list_Z!_I&
    list_protoU![659]&
  dtrs!<only_hd
```
Output in AVM notation:
Example 2

O Pedro entrou. Ele barbeou-se a ele próprio.
the Pedro entered. he shaved-clit to ELE PRÔPRIO
Pedro came in. He shaved himself.

ctx1, o, pedro, entrou, ., ele, barbeou, -se, a, ele_próprio