High Precision Analysis of NPs with a Deep Processing Grammar

António Branco
Francisco Costa
Universidade de Lisboa (Portugal)
email: Antonio.Branco@di.fc.ul.pt

Abstract

In this paper we present LXGram, a general purpose grammar for the deep linguistic processing of Portuguese that aims at delivering detailed and high precision meaning representations. LXGram is grounded on the linguistic framework of Head-Driven Phrase Structure Grammar (HPSG). HPSG is a declarative formalism resorting to unification and a type system with multiple inheritance. The semantic representations that LXGram associates with linguistic expressions use the Minimal Recursion Semantics (MRS) format, which allows for the underspecification of scope effects. LXGram is developed in the Linguistic Knowledge Builder (LKB) system, a grammar development environment that provides debugging tools and efficient algorithms for parsing and generation. The implementation of LXGram has focused on the structure of Noun Phrases, and LXGram accounts for many NP related phenomena. Its coverage continues to be increased with new phenomena, and there is active work on extending the grammar's lexicon. We have already integrated, or plan to integrate, LXGram in a few applications, namely paraphrasing, treebanking and language variant detection. Grammar coverage has been tested on newspaper text.

1 Introduction

In this paper we present LXGram, a hand-built, general purpose computational grammar for the deep linguistic processing of Portuguese, specially geared to high precision processing of Noun Phrases. This grammar is based on the framework of Head-Driven Phrase Structure Grammar (HPSG; Pollard and Sag (1994)), one of the most prominent linguistic theories being used in natural language processing. Like several other computational HPSGs, LXGram uses Minimal Recursion Semantics (MRS; Copestake et al. (2005)) for the representation of meaning.

LXGram is developed in the Linguistic Knowledge Builder (LKB) system (Copestake, 2002), a development environment for constraint-based grammars. This environment provides a GUI, debugging tools and very efficient algorithms for parsing and generation with the grammars developed there (Malouf et al., 2000; Carroll et al., 1999).

Several broad-coverage grammars have been developed in the LKB. Currently, the largest ones are for English (Copestake and Flickinger, 2000), German (Müller and Kasper, 2000) and Japanese (Siegel and Bender, 2002). The grammars developed with the LKB are also supported by the PET parser (Callmeier, 2000), which allows for faster parsing times due to the fact that the grammars are compiled into a binary format in a first step. As the LKB grammars for other languages, LXGram is in active development, and it is intended to be a broad-coverage, open-domain grammar for Portuguese. At the same time, it produces detailed representations of meaning in tandem with syntactic structures, making it useful for a wide range of applications.

In Section 2, we describe the framework foundations of the grammar. The major design features of the grammar are introduced in Section 3. We talk about the coverage of LXGram in Section 4. Section 5 presents some of the phenomena treated within the NP domain and shows examples of implemented analyses relating to NP syntax and semantics. In Section 6, results on the performance of the grammar are reported, and in Section 7, we discuss applications where the grammar is or is being integrated. Finally, the paper closes with concluding remarks in Section 8.

2 Foundations

LXGram adopts the HPSG framework, a popular linguistic theory with a large body of literature covering many natural language phenomena. These insights can be directly incorporated in the implementation of a computational grammar.

2.1 HPSG

HPSG resorts to a declarative formalism to model linguistic data. It employs a type system (supporting multiple inheritance) and typed feature structures (recursive data structures defining "has-a" relations) in order to describe the properties of linguistic objects (words, phrases, rules). Unification of types and feature structures is central to HPSG, used to ensure that the various elements have compatible properties. For instance, the fact that a transitive verb takes an NP as its complement is captured in HPSG by defining a lexical type for transitive verbs, say *transitive-verb-lex(eme)*, with constraints like the following (among others), presented in the Attribute-Value Matrix (AVM) format widely employed in HPSG:

$$\begin{bmatrix} \textit{transitive-verb-lex} \\ \text{SYNSEM|LOCAL|CAT|VAL|COMPS} \left\langle \begin{bmatrix} \text{LOCAL|CAT} \begin{bmatrix} \text{HEAD} & \textit{noun} \\ \text{VAL} & \begin{bmatrix} \text{SPR} & \langle\rangle \\ \text{COMPS} & \langle\rangle \end{bmatrix} \end{bmatrix} \end{bmatrix} \right\rangle \end{bmatrix}$$

The NP complement of the verb is represented in this AVM as the value of the attribute COMPS. This attribute takes a list as its value (indicated by the angle brackets). In this case the sole element of this list describes an object with a HEAD feature of the type *noun* and empty complements (the attribute COMPS) and specifier (the feature SPR) (i.e. they have been saturated at the point where the verb combines with this element), which is the HPSG description of an NP.

2.2 MRS

Minimal Recursion Semantics (MRS) is used as the format of semantic representations that LXGram associates with expressions from Portuguese. MRS has several properties that are interesting for applications. A relevant one is the use of pointers to represent scope effects that are handled via recursion in traditional formal semantics. This use of pointers (called *handles*) allows for the underspecification of scope relations, which avoids listing all scope possibilities for ambiguous sentences (although they can still be computed on demand with the LKB machinery). This is a useful property: scope does not need to be resolved in all applications (e.g. machine translation does not require it), but at the same time scoped formulas can be obtained on demand if required (e.g. for automated inference).

We provide an example MRS representation derived for the sentence "todas as equipas podem vencer" (*all teams can win*) in Figure 1. This MRS describes the

$$\begin{bmatrix} \textit{mrs} \\ \text{LTOP} & \boxed{h1}\, h \\ \text{INDEX} & \boxed{e2}\, e \\ \text{RELS} & \left\langle \begin{bmatrix} \textit{\_todo\_q\_rel} \\ \text{LBL} & \boxed{h3}\, h \\ \text{ARG0} & \boxed{x6}\, x \\ \text{RSTR} & \boxed{h5}\, h \\ \text{BODY} & \boxed{h4}\, h \end{bmatrix}, \begin{bmatrix} \textit{\_equipa\_n\_rel} \\ \text{LBL} & \boxed{h7}\, h \\ \text{ARG0} & \boxed{x6} \end{bmatrix}, \begin{bmatrix} \textit{\_poder\_v\_rel} \\ \text{LBL} & \boxed{h8} \\ \text{ARG0} & \boxed{e2} \\ \text{ARG1} & \boxed{h9}\, h \end{bmatrix}, \begin{bmatrix} \textit{\_vencer\_v\_rel} \\ \text{LBL} & \boxed{h10} \\ \text{ARG0} & \boxed{e11} \\ \text{ARG1} & \boxed{x6} \end{bmatrix} \right\rangle \\ \text{HCONS} & \left\langle \begin{bmatrix} \textit{qeq} \\ \text{HARG} & \boxed{h1} \\ \text{LARG} & \boxed{h8} \end{bmatrix}, \begin{bmatrix} \textit{qeq} \\ \text{HARG} & \boxed{h5} \\ \text{LARG} & \boxed{h7} \end{bmatrix}, \begin{bmatrix} \textit{qeq} \\ \text{HARG} & \boxed{h9} \\ \text{LARG} & \boxed{h10} \end{bmatrix} \right\rangle \end{bmatrix}$$

Figure 1: MRS for the sentence "Todas as equipas podem vencer" (*all teams can win*)

two following scoped formulas, where the predicate *_todo_q* stands for a universal quantifier:

- $\_todo\_q(x_6, \_equipa\_n(x_6), \_poder\_v(e_2, \_vencer\_v(e_{11}, x_6)))$
- $\_poder\_v(e_2, \_todo\_q(x_6, \_equipa\_n(x_6), \_vencer\_v(e_{11}, x_6)))$

The first reading is the one that says that each team has a chance to win, while the second reading says that it is possible for there to be a situation in which all teams win (false assuming common sense knowledge). A single MRS representation is obtained for these two readings by instantiating the ARG1 feature of the relation *_poder_v* (*can*) with the handle h_9, which is related to the handle h_{10} labeling the relation *_vencer_v* (*win*) via a *qeq* relation (equality modulo intervening quantifiers). This is the way of saying that these two handles are the same (first reading) or that there is an intervening generalized quantifier relation (second reading).

Semantic representations abstract from many grammatical and superficial details of language, like word order, syntactic structure and morphology. As such, they are very similar across different natural languages (modulo predicate names). This is also true of MRS. Furthermore, semantic representations hide grammar implementation. As such, they are the preferred grammar's interface for applications, that do not need any knowledge of the grammatical properties of Portuguese and may not need to look at syntactic analysis.

The MRS format is also used with several other computational HPSGs, for other languages. Several applications (e.g. Machine Translation) have been used with other HPSGs that communicate with these grammars via the MRSs (Bond et al., 2004). These applications can be easily integrated with grammars for different languages that also use MRS: they are almost completely language independent.

3 Design Features

Given the foundational options, LXGram adheres to a number of important design features.

Bidirectionality LXGram is bidirectional. The formalism employed is completely declarative. It can be used for parsing (yielding syntactic analyses and semantic representations from natural language input) and also for generation (yielding natural language from meaning representations). As such it can be useful for a wide range of applications.

Precision LXGram aims at high precision of linguistic processing. Modulo bugs, the grammar cannot parse ungrammatical input. Although this feature may have a negative impact on robustness, it is an important aspect of the grammar when it is used for generation, as it means it is not possible to generate ungrammatical strings.[1] It is indeed possible to mark some rules and some lexical entries to only be used for parsing and not for generation. In the configuration files for the LKB one can list these rules and lexical items. We are currently using this feature in order to be able to parse input that is not ungrammatical but is marked with respect to register, but preventing the grammar from generating such strings.

Importantly, the fact that it cannot parse ungrammatical input also means that the grammar will not produce impossible analyses for grammatical sentences.

Broad Coverage LXGram development is aimed at a broad coverage. We also seek to make LXGram neutral with respect to regional variation as much as possible. Currently, the grammar accommodates both European Portuguese and Brazilian

[1] We believe that dealing with ill-formed input is best done via other means (rather than let the grammar overgenerate so it can parse more), like partial parsing or the integration with/falling back to other tools.

Portuguese. Aspects of variation that are accounted for include lexical differences (merely affecting spelling or more substantial ones) as well as syntactic discrepancies (e.g. definite articles before possessives, word order between clitic pronouns and the verb).

Efficiency The processors on which LXGram runs (LKB, PET) are very efficient. In addition, there are grammar engineering techniques that improve efficiency (e.g. (Flickinger, 2000)) that are also exploited in our implementation.

Robustness The LKB and PET systems provide several ways to combine a grammar with the output of shallow tools, like part-of-speech taggers. Such integration can improve grammar coverage, as the grammar needs information about all words in the input, and some words may be missing in the grammar's lexicon. We have successfully combined LXGram with a part-of-speech tagger and a morphological analyzer (more in Section 6). The grammar code includes mappings from the input format (XML) to the feature structures that are manipulated by the grammar.

Availability A version of LXGram is publicly available at `http://nlxgroup.di.fc.ul.pt/lxgram`. LXGram can be used by applications without any knowledge of the grammar's implementation or internal workings. The LKB allows for applications to communicate with the grammar via sockets, accepting parser input in XML or raw text and returning semantic representations in XML, for which a DTD is available. It is also possible to automatically produce a list of all the predicates known by the grammar together with their arity and argument types (from the lexicon and syntax rules), that can be manually annotated with comments and examples. The predicates corresponding to lexical items are however quite transparent once the naming conventions that are used are explained.

4 Coverage

4.1 Lexical Coverage

When one is using a lexicalist framework like HPSG, lexical coverage is a key issue because all tokens in the input should be known by the grammar in order for the grammar to produce a parse. Furthermore, the amount of information included in the lexicon that is used by an HPSG is very large. Part-of-speech and morphological information is not sufficient. For the correct assignment of semantic representations, subcategorization frames as well as other information pertaining to semantics must be correctly associated with every lexical item, something that cannot be known with sufficient quality by just using shallower tools, like part-of-speech taggers.

In LXGram a hand-crafted lexicon containing several hundreds of nouns, adjectives and verbs was developed. However, the manual creation of lexica with this amount of information is time consuming and error prone. We are exploring methods to alleviate this problem. An option is to combine the grammar with shallower tools in order to have access to some of the information needed and assume default values for the information that cannot be obtained this way. We have already integrated the grammar with a set of shallow tools (a part-of-speech tagger, a lemmatizer and a morphological analyzer) in order to guess information about unknown words. Preliminary results indicate an increase in coverage on unrestricted newspaper text from 2% to 13%. Although this approach cannot guarantee correct semantic representations

(or even syntactic trees, since subcategorization frames constrain syntactic structure), it can be useful in applications that only require some restricted amount of linguistic information.

4.2 Overall Grammatical Coverage

In order to get a quantitative overview of the grammar, it can be characterized as follows:

- 24,484 lines of code, including comments and excluding the lexicon;
- 53 syntax rules;
- 40 lexical rules, mostly inflectional;
- 3,154 total types;
- 414 types for lexical items;
- 2,718 hand-built lexical entries.

For a qualitative overview, these are the linguistic phenomena covered so far:

- **Declarative sentences**
- **Yes-no questions** e.g.: "Portanto o Estado tem um gosto?" (So does the State have preferences?)
- **Imperative sentences** e.g.: "Dá-me um desses bolos." (Give me one of those cakes)
- **Some subcategorization frames of verbs, nouns and adjectives** e.g.: "a Polónia *empatou* com a França" (Poland *tied* with France); "eu já *disse* que *pode ser* um dos mais baratos" (I *told* you already that it *can be* one of the cheapest); "*filho* de um professor dos *arredores* de Viena" (*son* of a teacher from the *outskirts* of Vienna)
- **Comparative constructions** e.g.: "a vida é maior *do que* o cinema" (life is larger *than* cinema)
- **Noun phrase structure, including determiners, possessives, cardinal and ordinal numerals, prepositional phrases, adjectives, etc.** (examples in the next section)
- **Modification of verbal projections by prepositional and adverbial phrases** e.g.: "*No CAPC* termina *hoje* a exposição" (the exhibit ends *today at CAPC*);
- **Relative clauses** e.g.: "sete outros suspeitos *que a polícia ainda procura*" (seven other suspects *that the police are still looking for*)
- **Null subjects and objects** e.g.: "Saímos depois do jantar." ((We) left after dinner); "Podemos comer lá perto." (We can eat near there)

- **Floated Quantifiers** e.g.: "os índices subiram *todos*" (the indices have *all* gone up)

The development of the grammar is going on and this grammar is getting its coverage increased with important phenomena that are missing. In particular, for the near future, we are working towards including more subcategorization frames for verbs, nouns and adjectives, and implementing wh-questions, coordination and adverbial subordination.

5 Noun Phrases

A special design feature of LXGram is that it includes a comprehensive implementation of Portuguese Noun Phrase structure, covering:

- **Bare noun phrases** (i.e. NPs lacking a determiner) e.g.: "boa gestão" (good management); "imagens da bancada" (images of the seats)
- **Determiners and predeterminers** e.g.: "*esta* sua última produção" (*this* last production of his); "*todos estes* problemas" (*all these* problems); "*todos os* partidos políticos" (*all the* political parties), "*aquele* tempo *todo*" (*all that* time)
- **Word order constraints among NP elements** e.g.: "as duas primeiras instituições" (the two first institutions); "os primeiros sete meses deste ano" (the first seven months of this year); "sete outros suspeitos que a polícia ainda procura" (seven other suspects that the police are still looking for); "os outros três membros do conselho" (the other three members of the council); "os seus dois primeiros anos polémicos na Casa Branca" (his first two polemic years in the White House); "o primeiro grande conflito que aportava em Belém" (the first great conflict that reached Berlin); "outro lugar qualquer" (any other place); "um lugar qualquer" (any place); "qualquer outra solução" (any other solution)
- **Prenominal and postnominal possessives** e.g.: "o *seu* terceiro maior parceiro comercial" (*its* third major commercial partner); "um adjunto *seu* que atendeu ali o telefonema" (an assessor *of his* who answered the phone call there)
- **Modification of adjectives** e.g.: "os escritores *mais* importantes" (the *most* important writers); "o discurso *razoavelmente* optimista" (the *reasonably* optimistic speech)
- **Missing nouns** e.g.: "dois que são gémeos" (two who are twins)
- **Word order between adjectives and complements of nouns** e.g.: "o conhecimento *essencial* das pessoas" (the *essential* knowledge about people)
- **Adjectives with the semantics of arguments of nouns** e.g.: "o veto *americano* à renovação do mandato" (the *American* veto to the renewal of the position)

Precision was given a lot of attention. For instance, many items are constrained not to appear more than once in a given NP (determiners, possessives, cardinals, ordinals, etc.). Scope phenomena are also handled (motivated by semantics), as well as order constraints. Agreement is enforced.

We present some examples of phenomena for which LXGram provides interesting semantic representations and that we have not found in the literature pertaining to implemented grammars.

5.1 Floated Quantifiers

The first example relates to floated quantifiers. For all the sentences in (1), which are all grammatical in Portuguese, LXGram provides the MRS equivalent of $all(x, price(x), will(go\_up(x)))$:

(1) a. Todos os preços vão subir.
all the prices will go up

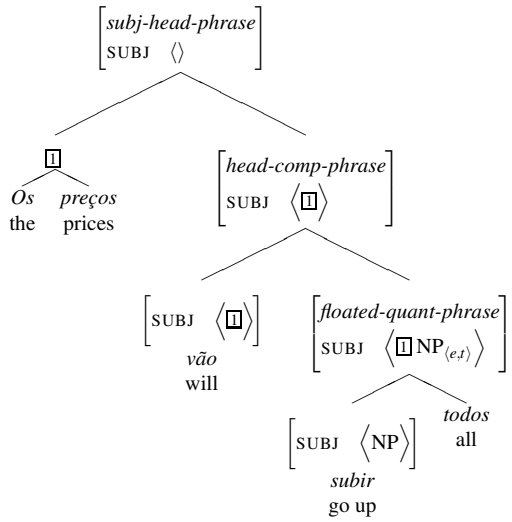
b. Os preços todos vão subir.
the prices all will go up

c. Os preços vão todos subir.
the prices will all go up

d. Os preços vão subir todos.
the prices will go up all

In all of these cases we associate empty semantics to the definite article ("os"). Semantic information is percolated around the syntactic trees so that the universal quantifier, which can be realized at several different places, ends up being linked to the semantics of the NP subject in the semantic representations for all these sentences. We also make sure that definite articles always carry quantifier semantics when no floated quantifier is present.

The implementation revolves around allowing floated quantifiers to attach to verbs, resulting in verb-headed nodes that combine with NP subjects lacking quantifier semantics. Raising verbs, like the form "vão" in this example, constrain their subject according to the constraints of the subject of their VP complement. For instance, the last example (1d) receives a syntactic analysis like the one described by the following tree:

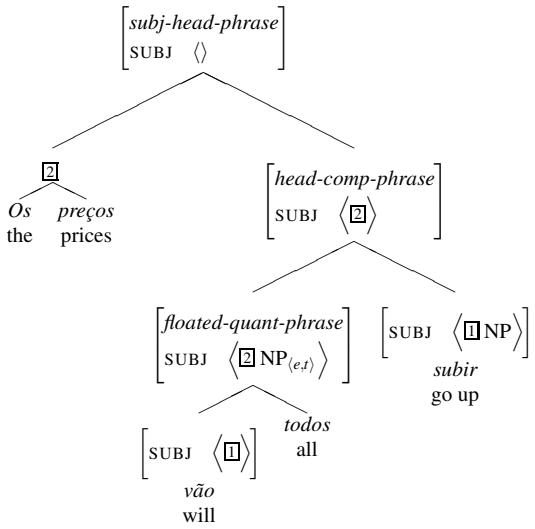


Here, NP abbreviates a feature structure that describes a noun phrase (of the semantic type $\langle\langle e,t\rangle,t\rangle$), and $\text{NP}_{\langle e,t\rangle}$ abbreviates the constraints that describe an NP introduced by a determiner lacking quantifier semantics (i.e. an NP with an MRS representation that is similar to that of constituents with the the semantic type $\langle e,t\rangle$).

In HPSG, the SUBJ feature encodes the constraints on the subject that a constituent selects. We use a dedicated syntax rule to combine "subir" and "todos" (*floated-quant-phrase*), that creates a node requiring an NP with the semantic type $\langle e,t\rangle$ as its subject. The verb form "vão" is treated as a raising verb: in HPSG the syntactic requirements on the subject of a raising verb are the same as the requirements on the subject of the VP complement that that verb selects for. This is denoted by the boxed integers in this tree (which represent unification).

In this example, the VP complement of "vão" is the phrase "subir todos", as these two constituents are combined via the *head-comp-phrase* rule (selection of complements is represented in a way similar to the selection of subjects, but via the feature COMPS instead of the feature SUBJ). The subject of the *head-comp-phrase* is the subject of its head daughter ("vão"). The topmost node is the result of applying a syntactic rule to project subjects to the left of their head (*subj-head-phrase*).

The example in (1c) is processed in a similar fashion:



In this example, the complement of "vão" is the node spanning "subir", which selects for a quantified subject. The subject of the raising verb is accordingly also a quantified NP. Here, the rule to project a floated quantifier applies lower than the construction that projects complements, creating a node that requires a non-quantified subject. The SUBJ feature of head-complement constructions comes from the head daughter (the *floated-quant-phrase* node in this example). Therefore, the node produced by the *head-comp-phrase* rule also requires a non-quantified subject.

Note that the composition of semantics with MRS in based on the concatenation of the RELS and HCONS lists associated to the various constituents and passing around

the values of the features LTOP and INDEX (see Figure 1). It is not based on function application. The composition of semantics with MRS is quite flexible.

5.2 Scope of Adjectives and Relative Clauses

The second example that we show here relates to the semantic scope between different elements of noun phrases. In particular, we can see a distinction in the interpretation of the two following examples:

(2) a. um possível médico chinês
a possible doctor Chinese
a possible Chinese doctor

b. um possível médico que é chinês
a possible doctor who is Chinese
a possible doctor who is Chinese

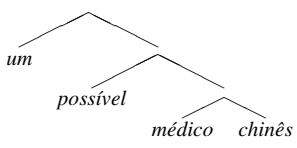
In the first NP an entity is described as possibly being a Chinese doctor. The second NP describes an entity as possibly being a doctor and certainly being Chinese. Accordingly, LXGram delivers slightly different semantic representations for these two NPs. The first case produces something similar to

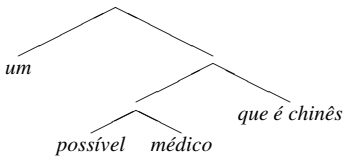
$$\lambda P.\ a(x, possible(doctor(x) \wedge chinese(x)), P(x)).$$

The second NP is treated along the lines of

$$\lambda P.\ a(x, possible(doctor(x)) \wedge chinese(x), P(x)).$$

These two different readings are derived simply by constraining the relative syntactic scope of adjectives and relative clauses. Namely, LXGram forces prenominal adjectives to attach higher than postnominal adjectives (2a) but lower than relative clauses (2b). In this case, the scope differences in the semantic representations are simply derived from the differences in syntactic scope:





Of course, the following examples receive equivalent semantics:

(3) a. um médico chinês
a doctor chinese
a Chinese doctor

b. um médico que é chinês
a doctor who is Chinese
a doctor who is Chinese

6 Evaluation

Some evaluation experiments were conducted to test LXGram's coverage. In one of them, a corpus with newspaper text (self-reference) was used, with 145 sentences. For this experiment, we used a part-of-speech tagger and a morphological analyzer (self-reference) in order to guess some information about out-of-vocabulary words. A default value was assumed for the missing subcategorization information (all unknown verbs were treated as transitive verbs). The average sentence length was 22 words. In this experiment, 13.1% of all sentences received at least one parse by the grammar.[2] On the same test corpus, the average time it took for a sentence to parse was 1.1 seconds on a P4 machine at 3GHz. The average amount of memory required to analyze a sentence was 145.5MB.

In another experiment, with 180,000 short sentences (5 to 9 words) selected randomly from two newspaper corpora (CETEMPúblico and CETENFolha), LXGram had achieved 26% coverage, using a similar approach to handle unknown words (self-reference).

During the development of LXGram we maintain several test suites, consisting of example sentences for the implemented phenomena. The test suites use a controlled vocabulary. Also, several examples attest several phenomena, in order to test the interaction of the different modules. They are very useful to test the syntax rules of the grammar and the semantics that LXGram produces, and for regression testing. The test suite for NPs contains 851 sentences (429 of which are negative examples, that the grammar should not parse). The average sentence length is 5.3 words (2–16). On this test suite LXGram has 100% coverage and 0% overgeneration. The average time needed to analyze a sentence is 0.11 seconds, with an average memory requirement of 15.5MB. Plotting parse time by sentence length, we see an approximately linear increase in parse time with this test suite.

7 Applications and Further Work

We have used LXGram to automatically discriminate between texts written in European Portuguese and Brazilian Portuguese, with encouraging results, which match the results obtained with other dialect detection methodologies. (self-reference).[3]

Additionally, we are working towards integrating it with an existing question answering system (self-reference).[4] This is in part the reason for the special focus on NPs, as these constituents are often short answers to factoid questions.

Because the grammar is entirely bidirectional, a paraphraser is gained for free from the implementation of LXGram: the grammar can simply be used to generate from the semantic representations that it derives from an input sentence, thus producing paraphrases of the textual input. We are also working to integrate the grammar, running under this functionality, into the QA system.

[2]Note that one of the HPSGs with the broadest coverage at the moment, the ERG, covers 17% of the British National Corpus. The main cause of parse failure is out-of-vocabulary words.

[3]In particular, the results obtained with LXGram were quite similar to the results obtained with the standard methods (based on character n-grams) that are used to identify the language in which a given text is written, when used for this purpose.

[4]See (Bobrow et al., 2007) for a similar approach, where an LFG is employed in a question answering system aiming at high precision.

On a par with the above lines of research, we are intensively using the grammar to semi-automatically produce a treebank that contains syntactic representations and semantic descriptions of the sentences in a newspaper corpus. LXGram has also served in the past to implement and experiment with novel linguistic analyses of interesting phenomena (self-reference). By making it freely available, we intend to encourage this sort of experimentation also by other researchers. One can reap important benefits from computationally implementing linguistic analyses: the debugging tools allow for fast checking of correctness; the impact on other analyses that are already implemented can be immediately assessed via regression testing, making it possible to test the interaction between linguistic analyses for different phenomena; it is possible to automatically compare different competing analyses for efficiency, based on test suites or corpora.

8 Conclusions

In this paper we presented LXGram, a computational grammar for the deep linguistic processing of Portuguese. LXGram is implemented in a declarative formalism. It can be used for analysis as well as generation. It produces high precision syntactic analyses and semantic representations. LXGram supports the two main varieties of Portuguese: European and Brazilian Portuguese. It is not dependent on a particular domain or genre.

So far the focus of the implementation was on noun phrases and basic sentence structure, a coverage that is being extended in ongoing work. The outcome of different evaluation experiments shows scores that are in line with those obtained with similar grammars for other languages.

References

Bobrow, D. G., B. Cheslow, C. Condoravdi, L. Karttunen, T. H. King, R. Nairn, V. de Paiva, C. Price, and A. Zaenen (2007). PARC's bridge and question answering system. In T. H. King and E. M. Bender (Eds.), *Proceedings of the GEAF07 Workshop*, Stanford, CA, pp. 46–66. CSLI Publications.

Bond, F., S. Fujita, C. Hashimoto, K. Kasahara, S. Nariyama, E. Nichols, A. Ohtani, T. Tanaka, and S. Amano (2004). The Hinoki treebank: Working toward text understanding. In S. Hansen-Schirra, S. Oepen, and H. Uszkoreit (Eds.), *COLING 2004 5th International Workshop on Linguistically Interpreted Corpora*, Geneva, Switzerland, pp. 7–10. COLING.

Callmeier, U. (2000). PET — A platform for experimentation with efficient HPSG processing techniques. *Natural Language Engineering 6*(1), 99–108. (Special Issue on Efficient Processing with HPSG).

Carroll, J., A. Copestake, D. Flickinger, and V. Poznański (1999). An efficient chart generator for (semi-)lexicalist grammars. In *Proceedings of the 7th European Workshop on Natural Language Generation (EWNLG'99)*, Toulouse, pp. 86–95.

Copestake, A. (2002). *Implementing Typed Feature Structure Grammars*. Stanford: CSLI Publications.

Copestake, A. and D. Flickinger (2000). An open-source grammar development environment and broad-coverage English grammar using HPSG. In *Proceedings of the Second conference on Language Resources and Evaluation (LREC-2000)*, Athens, Greece.

Copestake, A., D. Flickinger, I. A. Sag, and C. Pollard (2005). Minimal Recursion Semantics: An introduction. *Journal of Research on Language and Computation 3*(2–3), 281–332.

Flickinger, D. (2000). On building a more efficient grammar by exploiting types. *Natural Language Engineering 6*(1), 15–28. (Special Issue on Efficient Processing with HPSG).

Malouf, R., J. Carrol, and A. Copestake (2000). Efficient feature structure operations without compilation. *Natural Language Engineering 6*(1), 29–46. (Special Issue on Efficient Processing with HPSG).

Müller, S. and W. Kasper (2000). HPSG analysis of German. In W. Wahlster (Ed.), *Verbmobil: Foundations of Speech-to-Speech Translation* (Artificial Intelligence ed.)., pp. 238–253. Berlin Heidelberg New York: Springer-Verlag.

Pollard, C. and I. Sag (1994). *Head-Driven Phrase Structure Grammar*. Stanford: Chicago University Press and CSLI Publications.

Siegel, M. and E. M. Bender (2002). Efficient deep processing of Japanese. In *Proceedings of the 3rd Workshop on Asian Language Resources and International Standardization. Coling 2002 Post-Conference Workshop*, Taipei, Taiwan, pp. 31–38.