A Workbench for Grammatical Knowledge and Processing of Portuguese

António Horta Branco

Faculdade de Ciências de Lisboa, Dep. de Informática
GREMLIN - Grupo de Estudo sobre
Modelização Formal e Computacional das Línguas Naturais

Antonio.Branco@di.fc.ul.pt

0 Introduction

As an intermediate instrumental step in our current long term research on reference processing, we built up a grammar workbench which we called GwB. This workbench was primarily designed as a tool to help verifying the theoretical consistency and the empirical adequacy of our working hypotheses concerning the formal and computational modelling of Portuguese grammar, in general, and reference processing, in particular.

In this paper, we report on the underlying architecture of GwB and its functionality. First, we present the general language processing framework embodied in the grammar workbench. Second, we present a brief survey of the computational systems available for the implementation of that framework and briefly discuss the major aspects of the implementation of GwB in the chosen system.

1 The Language Processing Framework

Due to its interdisciplinary nature, the realm of natural language processing has been addressed by a community of researchers coming from diverse scientific disciplines and research traditions, using several different formal and computational tools, and aiming at various development purposes and research goals. Consequently, more often than not, it turns out to be quite hard to compare or combine results on the same or adjacent topics, which may be perfectly confluent, but whose methodological underpinnings may conjure up to give them a totally disparate appearance and impact. Nevertheless, this active and reciprocal exposition to different approaches has contributed also with an emerging trend towards integrative frameworks. A remarkable success in pursuing such desideratum has been achieved with the HPSG environment. Since its inception in late eighties, it has evolved to become, according to Uszkoreit, 1996, the "single most influential framework" in basic research in natural language processing. Accordingly, this is the language processing framework in the context of which our research on reference processing is taking place (Branco, 1996; Branco and Marrafa, 1997; Branco and Marrafa, 1998; Branco, 1998a,b).

HPSG is the acronym for Head-Driven Phrase Structure Grammar, which happens to be the title of the second of two seminal books by Ivan Sag and Carl Pollard, (Pollard and Sag, 1987) and (Pollard and Sag, 1994). This framework was set up on the basis of cardinal ideas and tools from linguistics and computer science. Its shape is the result of an ingenious blend of influences from data type theory, knowledge representation, unification-based formalisms, object-oriented programming, and several nonderivational research traditions in natural language syntax, such as categorial grammar, generalized phrase-structure grammar, arc-pair grammar and lexical-functional grammar (vd. Daelemans, De Smedt and Gazdar, 1992, and Pollard and Sag, 1994, Intr.). References of works on logical, computational, linguistic and cognitive issues in the HPSG framework are collected at the HPSG bibliography site, www.dfki.de/lt/HPSG, whose browsing may give an idea of the current intense research activity in this framework.

HPSG is a precise but flexible interleaving of proposals concerning the shape of different conceptual layers in the modeling of natural language grammars. Although these layers were designed to be consistently integrated into a single framework, they are quite autonomous. To a considerable extent, each one of those layers may suffer changes and be experimented with without compromising its inclusion into the whole framework. In what follows, we provide a description of HPSG by checking out the major features of those different layers.

1.1 Ontological setup

Like in other empirical theories, in HPSG the theory and the empirical phenomena it talks about are mediated by a mathematical structure that models the empirical domain over which the theory is supposed to unfold its predictions. That modeling structure is put in correspondence with the relevant observables so that entities and objects in the empirical domain are represented by entities and objects in the modeling structure. This provides a convenient set up for a rigorous and falsifiable theorizing as well as a suitable basis for improvement of results and progress of research. Therefore, under such tripartite ontological set up, the theory can be seen either as talking about the formal modeling domain, or as being interpreted in it.

In the research on natural language grammar the relevant *observables* are types of natural language expressions and their subparts. The observables are formally modeled by a system of sorted *feature structures*, which are graph theoretic entities. The *theory*, in turn, is a specification interpreted in that modeling domain. The constraints of the specification establish predictions in the sense that they define what objects from the domain are admissible as belonging to the natural language at stake and those that are not.

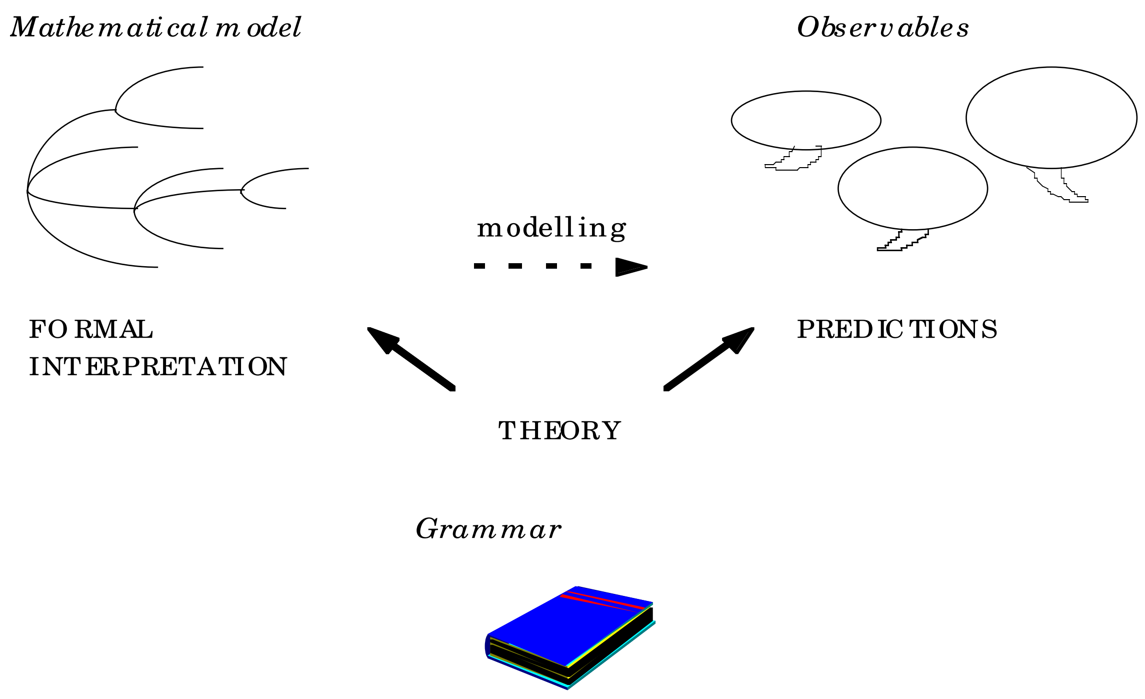

Figure 1 - The tripartite ontological setup

Mathematical model

A *feature structure* is a labeled acyclic graph such that nodes are tagged with sort labels and arcs with attribute labels. Accordingly, a given non atomic feature structure of sort *s* - a graph whose top node is labeled s - is said to have attributes - those with which the arcs getting out of node *s* are tagged. An attribute *a* in turn is said to have a value, which is another (atomic or not) feature structure to which the arc labeled with *a* is directed.

Given they should be total models of linguistic objects, feature structures are required to be (i) totally well-typed and (ii) sort resolved. In informal terms, this means that: (i) for each sort *s* of the graph, every arc/attribute that has *s* as its source, i.e., which is appropriate to "characterize" s, is actually present; (ii) every node is assigned a sort *s* that is most specific in the sort hierarchy in which *s* enters.

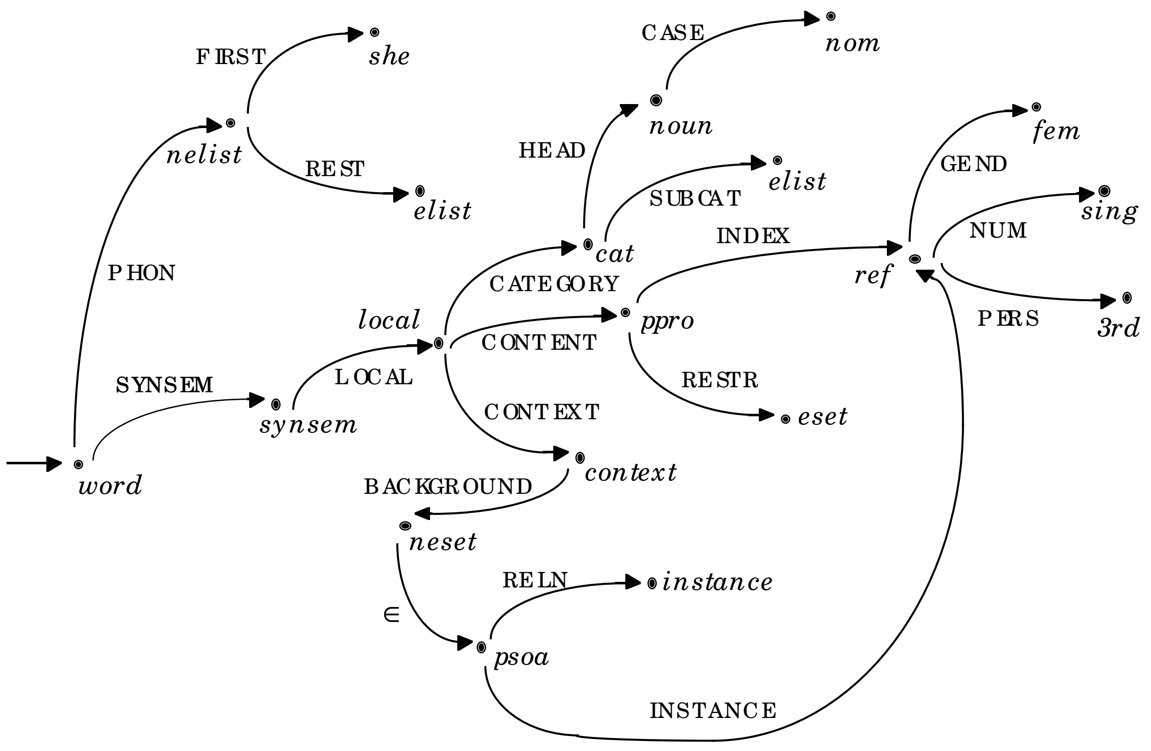

Figure 2 - The feature structure of 'she'

A crucial property of feature structures is that two distinct paths in the graph can lead to one and the same node. This means that structure sharing is allowed, i.e. that two paths in the graph can share the same structure as their common value.

Grammar

From an algebraic theoretic viewpoint, a theory/grammar in HPSG is a *specification* consisting of a signature and a presentation.

The *signature* defines what data types are allowed as possible pieces of structure encoding linguistic information. It includes a sort hierarchy and an appropriateness definition. The *sort hierarchy* is a partial order of sorts where the possible types of linguistic entities and their subtypes are fixed. The *appropriateness definition* in turn states what are the characteristics of each sort of the hierarchy. This is done by associating to sorts constraints that establish what are the appropriate features structures for the type of objects the sorts are in correspondence to. The sort hierarchy is an inheritance taxonomic tree such that a sort inherits the appropriateness constraints of its supersorts.

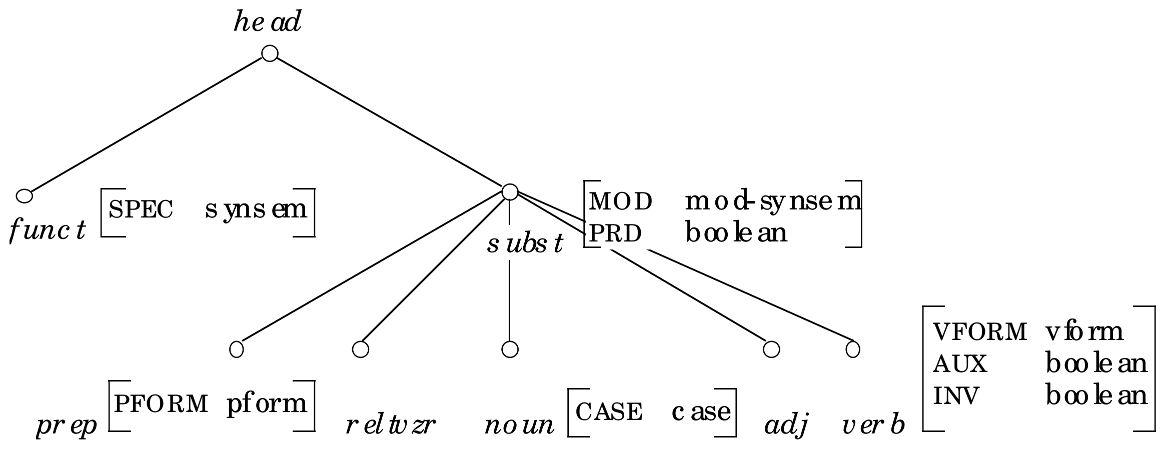

Figure 3 - The sort hierarchy for *head* and appropriatness conditions.

The *presentation* is a set of implicative constraints which are interpreted as being true or false in the model domain. They can also be seen as descriptions that constrain the set of feature structures that, being in accordance to the data types established in the signature, are predicted as being admissible by the theory as well formed expression of the language at stake.

Usually, the constraints stated in the presentation are also termed, depemding on the type of linguistic information they convey, as grammatical principles and lexical entries. The grammar of a specific language includes a set of principles that are claimed to belong also to the grammar of any other natural language. These are known as Universal Grammar (UG) principles. Besides those principles, each grammar for a specific language has principles which hold only for that language or to a subset of languages to which that language belongs. All these principles are conjunctive constraints as they altogether enter in the grammar as a conjunction of constraints to which any well formed expression must comply with.

On a par with the set of conjunctive principles, the constraints corresponding to lexical entries form the set of disjunctive principles. Disjunction may also be introduced by certain grammatical principles which are formed by a set of disjunctive constraints which are know as schemata.

$UG = P_1 \wedge ... \wedge P_n$

Language **L** $= UG \wedge P_{n+1} \wedge ... \wedge P_{n+m} \wedge ((S_1 \vee ... \vee S_p) \vee L_1 \vee .... \vee L_q)$

Where:

$P_1 \wedge ... \wedge P_n$	are universal principles
$P_{n+1} \wedge ... \wedge P_{n+m}$	are language **L** specific principles
$(S_1 \vee ... \vee S_p)$	are schemata grouped in language **L** specific principles
$L_1 \vee .... \vee L_q$	are lexical signs of **L** (basic or the output of lexical rules)

Figure 4 - The set of implicative constraints

Finally, the set of constraints known as lexical rules includes constraints with specific properties, used to express generalizations over lexical signs. A *lexical rule* is a pair of two “meta-descriptions”, since it relates lexical signs (constraints) and not the objects related by ordinary relations or constraints. Once the lexical rule applies, the constraint specified in the left-hand side picks out the lexical signs whose structure comply with it and those signs are supplied with the additional constraints specified in the right-hand side of the rule.

Description language

The constraints of the theory/grammar are stated using a special purpose language whose expressions are known as attribute-value matrix (AVM) diagrams. In a rough presentation of its syntax, one could say that non atomic AVMs consist of two column matrices, where the first column presents the attributes and the second column the corresponding values. The values of attributes are AVMs. Matrices receive a left subscript which indicates what is its sort. An atomic AVM consists just of an atomic sort.

There is a special symbol for stating structure sharing. When two attributes exhibit the same tag, the same boxed numeral, that means that they have the same value, i.e. their values are token identical. Tags may be followed by AVMs, which state the non atomic value which the tag stands for.

$$ {}_{\mathit{sort}}\begin{bmatrix} \text{ATTRIBUTE 1} & \text{AVM 1} \\ \cdots & \cdots \\ \text{ATTRIBUTE n} & \text{AVM n} \end{bmatrix} $$

Figure 5 - The syntactic schema of AVMs

Descriptions of sets are given within curly braces and descriptions of lists are abbreviated by using the angle-bracket notation.

It worth noting that since AVMs express constraints on the model domain, i.e. they are descriptions of admissible (total) model objects, they may be partial descriptions of the object to which characterization they contribute.

1.2 Linguistic configuration

With the ontological set up and description formalism in place, we can now address the general linguistic configuration of the grammars for particular languages. By linguistic configuration, we mean the set of formal options concerning linguistic generalizations involving issues common to all languages and to most of the different linguistic phenomena and constructions.

In HPSG, a *sign* based approach to linguistic constraints was adopted, where the different aspects of the linguistic objects at stake, e.g. phonological, syntactic, semantic and pragmatic aspects, are described in a single representation. This makes of HPSG a monostratal linguistic theory. No grammatical principle has precedence over any other in terms of constraint satisfaction, and a linguistic object is described by one single expression of the description language which integrates all different analytical levels of linguistic theorizing.

Phrase constituency is factored out in immediate dominance and linear precedence relations in view of a natural account of languages with and languages without free word order.

Subcategorization information is fully lexicalized in the relevant predicator and the subcategorizing of syntactic arguments occurs via argument cancellation in the style of categorial grammar.

Valence alternations, like in passive constructions, are given a lexicalized account where lexical rules are responsible for stating the relevant generalization between lexical items.

As to *unbounded syntactic dependencies*, the relationship between the syntactic gap and its filler is seen as a matter of structure sharing, which relies on a thread-based approach to such non local dependencies.

The importance of *structure sharing*, however, is not restricted to the account of unbounded dependencies. Following Pollard and Sag 1994, p.19, it is worth noting that “it is not going too far to say that in HPSG structure sharing is the central explanatory mechanism... Indeed, the relationships between fillers and traces, between ‘understood’ subjects and their controllers, between pronouns and their antecedents, between ‘agreement sources’ and ‘agreement targets’, and between the category of a word and the category of its phrasal projections are all analyzed as instances of structure sharing”.

Finally, tough the original *semantic component* was designed by Pollard and Sag in the spirit of situation semantics, Frank and Reyle, 1995 have shown that an underspecified, principle based semantics inspired on DRT could be better integrated in the overall sign-based philosophy of HPSG grammars.

Language specific theories

Given this ontological and linguistic setup, Pollard and Sag in their second book built a grammar for a substantial fragment of English. That grammar covers core phenomena like phrase structure, complementation, agreement and interpretation, as well as some constructions that are central in the linguistic debate and in some sort are a benchmark for checking the adequacy and explanatory potential of linguistic theories, like relative clauses, unbounded dependency constructions, complement control and binding. Given the high level of descriptive economy and formal rigor provided by the HPSG environment, it was possible to state the whole grammar for that fragment of English in the nine pages of the Appendix of (Pollard and Sag, 1994), while the remaining four hundred pages of the main text in the book were used to justify the thoroughness of the approach and to check the empirical adequacy of the proposals about specific linguistic phenomena and constructions.

Given the grammar architecture described above, involving language specific as well as universal principles, the specific grammar designed by Pollard and Sag embodies a number of principles which, with convenient parametric adaptation for each language, may be taken as accounting for the core aspects of natural languages. It is not feasible to present, even in an abridged formulation, the core of that grammar within the limits of this paper. Nevertheless, for the sake of getting a glimpse of the basic structuring of linguistic information proposed in HPSG, we present below the AVM of an expression and then comment on its subparts.

$$
{}_{\textit{word}}\begin{bmatrix}
\text{PHON} & <\text{ʃi:}> \\
\text{SYNSEM} & {}_{\textit{synsem}}\begin{bmatrix}
\text{LOC} & {}_{\textit{local}}\begin{bmatrix}
\text{CATEGORY} & {}_{\textit{cat}}\begin{bmatrix} \text{HEAD} & {}_{\textit{noun}}\begin{bmatrix}\text{CASE } \textit{nom}\end{bmatrix} \\ \text{SUBCAT} & <> \end{bmatrix} \\
\text{CONTENT} & {}_{\textit{ppro}}\begin{bmatrix} \text{INDEX} & {}_{\textit{ref}}[\ldots] \\ \text{RESTR} & \{\ \} \end{bmatrix} \\
\text{CONTEXT} & {}_{\textit{context}}\begin{bmatrix} \text{BACKGR} & \left\{ {}_{\textit{psoa}}[\ldots] \right\} \end{bmatrix}
\end{bmatrix}
\end{bmatrix}
\end{bmatrix}
$$

Figure 6 - The lexical sign of 'she'

The PHON value encodes a phonological representation of 'she'. The SYNSEM value of a sign, whose subsort in this example is *word*, encodes the information which can be subcategorized by a predicator. The structure of the LOC value describes the information shared between a trace and its filler. CAT value renders information on syntactic category and subcategorization frame. CONTENT and CONTEXT are reserved, respectively, for semantic and pragmatic information. Finally, the DTRS value keeps information on the constituent structure of the sign.

$$_{phrase}[\text{DTRS} \quad headed-struc] \;\rightarrow\; \begin{bmatrix} \text{SYNSEM|LOC|CAT|HEAD} & \boxed{1} \\ \text{DTRS|HEAD-DTR|SYNSEM|LOC|CAT|HEAD} & \boxed{1} \end{bmatrix}$$

Figure 7 - An example of implicative constraint: the Head Feature Principle

As an example of an implicative constraint, Figure 7 depicts the Head Feature Principle, which states that, informally, if a phrase is the projection of a head, then the value of the CAT | HEAD feature of the phrase is identical of the CAT | HEAD feature of its head.

2 The Implementation

Several different implementation systems for natural language grammars have been developed in the last few years that can be used to implement grammars developed in the framework of HPSG. From those, the ones whose development and improvement have been consistently pursued along the years were documented and comparatively assessed in surveys like (Backofen *et al.*, 1996) or (Bolc *et al.*, 1996). These reports, together with direct experimentation with some of the systems at the Language Technology Laboratory of the DFKI-German Research Center for Artificial Intelligence, Saarbrücken, Germany, provided us the basis for opting for a specific system where GwB could be implemented.

AUTHORSHIP	Authors	Site
ALE Attribute Logic Engine	Bob Carpenter and Gerald Penn	Carnegie Mellon Univ.
ALEP Advanced Linguistic Engineering Platform	BIM, Belgium	Cray Systems, Luxembourg

CL-ONE	RGR project	Univ. of Saarland, Saarbrücken, and Univ. of Edinburgh
ConTroll	Seminar für Sprachwissenschaft	Univ. of Tübingen
CUF Comprehensive Unification Formalism	DYANA project	IMS, Univ. of Stuttgart
PAGE/TDL Platform for Advanced Grammar Engineering	DISCO project	DFKI, Saarbrücken
ProFIT Prolog with Features, Inheritance and Templates	Gregor Erbach	Univ. of Saarland, Saarbrücken
TFS Typed Feature Structure	Martin Emele and Rémi Zajac	IMS, Univ. of Stuttgart (Polygloss project)

Signature

If we look at the formalisms from the viewpoint of their implementational capabilities regarding the signature of the grammar, a first clear distinction appears between two sets of systems. On the one hand we have ALEP, whose limited type system and absence of multiple inheritance mechanisms does not allow a convenient implementation of HPSG sort hierarchy and corresponding appropriateness conditions. On the other hand, we find all the remaining systems, with rich enough type systems allowing a thorough implementation of HPSG grammars' signature. The authors of (Genabith *et al.*, 1994), who have extensively experimented with the ALEP system in several EU funded projects, reported that "one can write [in ALEP] not HPSG grammars but 'HPSG-inspired' grammars".

PS vs. SH-based parsing

A second major trait that differentiates the implementation systems listed above has to do with the gist of the strategy adopted for the parsing algorithm. The systems can be seen falling apart into those whose parsing algorithm is centered around phrase structure (PS-based) and those whose algorithm is centered around the sort hierarchy (SH-based). This distinction underpins important differences both in terms of expressive capacity of the systems, and in terms of their efficiency.

Given that PS-based parsing has been exhaustively studied in the last decades, PS-based systems show to be clearly superior to SH-based ones in terms of efficiency. Nevertheless, the adoption of a PS-based parsing algorithm implies some adjustments to the HPSG description language as well as to the linguistic configuration assumed for the accounting of constituency structure.

FORMALISM	**ALE**	**ALEP**	**CL-ONE**	**ConTroll**	**CUF**	**TDL**	**ProFIT**	**TFS**
Type system								
multiple inheritance	yes	no	yes	yes	yes	yes	yes	yes
lists	specific	specific	prolog	specific	specific	specific and lisp	prolog	specific
sets	no	no	yes	no	no	no	no	no
Description language								
LP-constraints	no	no	yes	no	no	no	no	no
functional constraints	yes	no	yes	yes	yes	yes	yes	yes
lexical rules	yes	yes	no	yes	no	no	no	no

Computation	**ALE**	**ALEP**	**CL-ONE**	**ConTroll**	**CUF**	**TDL**	**ProFIT**	**TFS**
Base language	Prolog	Prolog	Prolog	Prolog	Prolog	Lisp	Prolog	Prolog
Algorithmic issues								
parsing	PS-based	PS-based	PS-based	SH-based	SH-based	SH-based	PS-based	SH-based
unification	specific	specific	Prolog	specific	specific	specific	Prolog	specific
Control issues								
delays	no	no	yes	yes	yes	yes	no	no
calls	yes	no	yes	no	yes	yes	yes	no
Efficiency	good	good	good	low	low	good	very good	very low

In PS-based systems the ID/LP format for constituency structure must be converted to the PS rules format. This implies that the disjunctive schemata of the Immediate Dominance Principle together with the Constituent Ordering Principle are removed from the set of implicative constraints and their constraining effect is taken over by a convenient design of the PS rules. Also, the format of the implicative constraints have to be reshaped. Their principled relational account in terms of definite clauses in PS-based systems imposes that not only the case covering the constraint expressed in the antecedent of the implicative statement but also the case covering the negation of that constraint have to be explicitly accommodated in the grammar.

These drawbacks have though to be put in contrast with the fact that in configurational languages like English or Portuguese, this adaptation from ID/LP format to PS format turns out to be quite straightforward. On the other hand, from the above listed SH-based systems, both CUF and TFS are not maintained any more and ConTROLL was not publicly available in 1997. Also, the only SH-based system that may approximately match PS-systems in terms of efficiency, TDL, is embodied in a complex multicomponent platform which greatly reduces the portability of any workbench developed in it. Given this, our set of options was restricted to ALE, ProFIT and CL-ONE.

From this group of three systems, CL-ONE have to be discarded given that, in its stage of development, it does not present an adequately stable behavior (Gregor Erbach, p.c.). ProFIT was then the chosen system, in detriment of 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.

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, 1995).

The basic rationale of ProFIT is 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 aiming at taking the maximum benefit from the improvements in processing efficiency and in expressive conspicousness obtained respectively in each of those two 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 allows that the built-in Prolog term unification can be used, which makes ProFIT "5 to 10 times faster than systems which implements a unification algorithm on top of Prolog." (Erbach, 1995, p.186).

The set of ProFIT programs is thus a superset of Prolog programs. While a Prolog program consists only of definite clauses, a ProFIT program consists of a datatype declaration (sort hierarchy plus appropriateness conditions on the features of the sorts) and definite clauses. Sorted feature terms can then be used together with Prolog terms, and an NLP 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 straightforwardly to improve the performance of grammars written in ProFIT.

```
PFT:=  <Sort             [1.Term of a sort Sort                              ]
     | Attribute!PFT     [2.Attribute-Value Pair                             ]
     | PFT & PFT         [3.Conjunction of terms                             ]
     | PROLOGTERM        [4.Any Prolog term                                  ]
     | FinDomTerm        [5.Boolean combinations of atoms                    ]
     | @Template         [6.Template call                                    ]
     | 'PFT              [7.Quoted term, is not translated                   ]
     | ''PFT             [8.Double-quoted, main functor not translated       ]
     | >>>Attribute!PFT  [9.Search for an attribute                          ]
     | Sort>>>Attribute!PFT   [10. Short for >Sort&>>>Attribute!PFT          ]
     | PFT or PFT        [11.Disjunction: expands to multiple terms          ]
```

Figure 8 - BNF of ProFIT terms

Linguistic Functionality

At the time of writing this paper, GwB is a program with 3 500 lines of code written in ProFIT. It includes a left-corner bottom-up parser, 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 developement started in 1997, at the Language Technology Laboratory of the DFKI-German Research Center for Artificial Intelligence, Saarbrücken, Germany.

The workbench was designed so that it could serve either as a test bed for research on the processing of the syntax and semantics of natural languages, or as the basis for the development of wide coverage application-oriented grammars.

As a test bed, it may be used to help checking the formal and empirical adequacy of hypothesis concerning the modelling of specific linguistic modules and phenomena. Given the separation between the encoding of linguistic knowledge and the encoding of the parsing device, it may be also used to experiment with the efficiency and performance of different parsing algorithms and their possibly different implementation in the framework of logic grammars.

The grammar included in the workbench, except for those 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 consists of the following principles: Head Feature Principle, Subcategorization Principle, SPEC Principle, Marking Principle, NONLOCAL Principle, and Semantics Principle.

The thread-based technique for unbounded dependency constructions was represented in the grammar with the implementation of the basics for topicalization. Given that we opted for a PS-based implementation system, the Immediate Dominance Principle and the Constituent Ordering Principle had to be recasted in terms of phrases structure rules, as mentioned above. Another change in the original Pollard and Sag's specification concerned the module for semantics. The semantic representation system proposed there was replaced by the one advocated by Frank and Reyle, 1995, where underspecification is ensured in the context of a DRT set up.

The lexicon was reduced to a representative sample of entries with the most significative syntactic behavior. No morphological module or set of lexical rules is included.

References

Branco, António. 1996. Branching Split Obliqueness at the Syntax-Semantics Interface. *Proceedings, COLING-96, 16th International Conference on Computational Linguistics*. Copenhagen: CST, pp. 149-156.

Branco, António and Marrafa, Palmira. 1997. Subject-oriented and non Subject-oriented Long distance Anaphora: an Integrated Approach. *Proceedings, PACLIC'96, Selected Papers from the 11th Pacific Asia Conference on Language, Information and Computation*. Seoul: Kyung Hee University, pp. 21-31.

Branco, António and Marrafa, Palmira. 1998. Long-distance Reflexives and the Binding Square of Opposition. In Webelhuth, Koening and Kathol eds.. *Lexical and Constructional Aspects of Linguistic Explanation*. Stanford: CSLI, Chap. 11.

Branco, António. 1998a. The Logical Structure of Binding. *Proceedings, COLING-ACL'98, 17th International Conference on Computational Linguistics and 36th Annual Meeting of the Association for Computational Linguistics*, Montreal: Univ. of Montreal, pp. 181-186.

Branco, António. 199b. Towards a Lean Constraint Based Implementation of Binding Theory. *Proceedings, DAARC'98Discourse, Anaphora and Resolution Colloquium*, Lancaster: Lancaster Univ.

Backofen, Rolf; Becker, Tilman; Calder, Jo; Capstick, Joanne; Dini, Luca; Dörre, Jochen; Erbach, Gregor; Estival, Dominique; Manandhar, Suresh; Mineur, Anne-Marie; van Noord, Gertjan; Oepen, Stephan; Uszkoreit, Hans. 1996. *Final Report of the EAGLES Formalisms Working Group*. Expert Advisory Group on Language Engineering Standards. Luxemburg: CEC.

Bolc, Leonard; Krysztof, Czuba; Kup´s´c, Anna; Marciniak, Malgorzata; Mykowiecka, Agnieszka; Przepiórkowski, Adam. 1996. *A Survey of Systems for Implementing HPSG Grammars*. Technical Report 814, Institute of Computer Science, Polish Academy of Sciences.

Daelemans, Walter; De Smedt, Koenraad; and Gazdar, Gerald. 1992. Inheritance in Natural Language Processing. *Computational Linguistics*, vol. 18, n.2.

Erbach, Gregor. 1995. ProFIT, Prolog with Features, Inheritance and Templates. In *Proceedings, 7th Conference of the European Chapter of the Association for Computational Linguistics*. ACL.

Frank, Anette; Reyle, Uwe. 1995. Principle Based semantics for HPSG. In *Proceedings, 7th Conference of the European Chapter of the Association for Computational Linguistics*. ACL.

Genabith, Joseph; Markantonatou, Stella; Sadler, Louisa; and Verhagen, Mark. 1994. English HPSG in ALEP. In Markantonatou, Stella and Sadler, Louisa (eds.) (1994), pp.91-116.

Markantonatou, Stella and Sadler, Louisa. eds. 1994. *Grammatical Formalisms: Issues in Migration and Expressivity*. Luxemburg: CEC.

Meurers, Walt. *On Implementing an HPSG Theory*. Abeitspapiere des SFB Nr. 58, University of Tübingen.

Pollard, Carl and Sag, Ivan. 1987. *Information-Based Syntax and Semantics*. Stanford: CSLI.

Pollard, Carl and Sag, Ivan. 1994. *Head-Driven Phrase Structure Grammar*. Stanford: CSLI.

Uszkoreit, Hans. 1996. “A Note on the Essence of HPSG and its Role in Computational Linguistics”. In *Hans Uskoreit’s homepage*. http://www.coli.uni-sb.de/~hansu/