Pacific Symposium on Biocomputing 6:408-419 (2001)

EVENT EXTRACTION FROM BIOMEDICAL PAPERS USING A FULL PARSER

Akane Yakushiji, Yuka Tateisi, Yusuke Miyao
Department of Information Science, Graduate School of Science, University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo, 113-0033, Japan

Jun-ichi Tsujii
Department of Information Science, Graduate School of Science, University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo, 113-0033, Japan
CCL, UMIST, P.O.Box 88, Manchester, M60 1QD, England

We have designed and implemented an information extraction system using a full parser to investigate the plausibility of full analysis of text using general-purpose parser and grammar applied to biomedical domain. We partially solved the problems of full parsing of inefficiency, ambiguity, and low coverage by introducing the preprocessors, and proposed the use of modules that handles partial results of parsing for further improvement. Our approach makes it possible to modularize the system, so that the IE system as a whole becomes easy to be tuned to specific domains, and easy to be maintained and improved by incorporating various techniques of disambiguation, speed up, etc. In preliminary experiment, from 133 argument structures that should be extracted from 97 sentences, we obtained 23% uniquely and 24% with ambiguity. And 20% are extractable from not complete but partial results of full parsing.

1 Introduction

With the explosion of results in molecular-biology, there is an increasing need for automatic information extraction (IE) to support database building and to intelligently find novel knowledge in online journal collections. Extraction of substance names and other terms [1,2,3,4,5] has gained certain success, and now the focus of attention is shifting to extracting of interactions and the relationship between substances[6,7,8,9,10].

Many of the previous researchers extracted information on reactions by matching the sentences in the paper with hand-tailored patterns in regular expressions on some pre-defined set of verbs representing a certain type of reaction. However, as a fact can be represented in various forms in natural language text, many patterns of surface expressions need to be prepared for one event. Thus, although pattern-matching based information extraction can be effective and quick on limited types of events in a limited domain, the workload of preparing patterns for extraction would be too expensive if we expand our attentions to texts from other domains or to a wider scope of events.

We propose an alternative information extraction method based on full parsing with a large-scale, general-purpose grammar. In our system, a parser converts the variety of sentences that describe the same event into a canonical structure (*argument structure*) regarding the verb representing the event and its arguments such as (semantic) subject and object. Information extraction itself is done using pattern matching on the canonical structure. Since the variation of representation is absorbed by the parser, a relatively small number of patterns are required for extracting an event. Thus, our system can be applied to new domains more easily than the previous systems.

In this work, we implemented *Argument Structure Extractor* and did preliminary experiments. For 133 argument structures that should be extracted from 97 sentences, we obtained 23% uniquely and 24% with ambiguity. A further 20% are extractable from not complete but partial results of full parsing.

2 Problems in Information Extraction from Biomedical Papers

In biomedical papers, compared to the targets of traditional information extraction such as newspaper articles, the structure of a sentence tends to be more complicated. In this section, we address some characteristics of research papers in the biomedical domain that might be the cause of the problems in information extraction.

2.1 Variation of verbal forms

Reactions and relationships are represented in the form of phrases where verbs represent the reaction or relationship and the subject, object and other arguments of the verbs represent the substance, sources and other factors of the reaction or relationship. The relationship is not always represented in a simple subject–verb–object order. The verbal phrase may undergo various transformations such as passivization and nominalization. For example, the event that a protein *P1* activates *P2* may be represented in various forms, for example, "*P1* activates *P2*" (simple indicative form), "*P2* is activated by *P1*" (passive form), "*P1* activating *P2*" (gerundive form), "activation of *P2* by *P1*" (nominalized form), and "*P2* activation by *P1*" (nominalized form). Thus, if event information is extracted directly from surface linguistic expression as in traditional methods, the number of mapping rules from linguistic expressions to events becomes huge in practice.

In addition, the structure of a sentence expresses a sequence of events and other relationships between events. For example, the sentence

> An active phorbol ester must therefore, presumably by activation

> of protein kinase C, cause dissociation of a cytoplasmic complex of NF- kappa B and I kappa B by modifying I kappa B.

represents a sequence of three biological reactions i.e.,

1. an active phorbol ester activates protein kinase C
2. the active phorbol ester modifies I kappa B
3. the active phorbol ester dissociates a cytoplasmic complex of NF-kappa B and I kappa B.

Such a representation of a sequence of reactions is not easy to find by pattern-matching based methods, because an individual reaction is extracted independently using the patterns on the verb that represents the reaction. In the above example, the individual event of activation, modification, and dissociation can be extracted using the patterns for verbs *"activate"*, *"modify"*, and *"dissociate"*, respectively, but the relation of the events, which can only be extracted by looking at the structure of the whole sentence, cannot be extracted. Moreover, since the activation event and the modifying event are expressed using a by-phrase and there is no explicit subject, the activator and modifier (both "active phorbol ester") cannot be easily determined by traditional pattern-based methods.

2.2 Embedding

Another problem is that the representation of reactions and relationships is often embedded in another sentence representing the authors' view of the event. For example, in

> In this report we show that lipopolysaccharide (LPS)-induced activation of B or pre-B cells results in loss of I kappa B alpha from NF-kappa B complexes in vivo.

the authors conclude with the fact that is represented in the *that*-clause in the sentence, but in

> Surprisingly, even p65, but not c-rel, was phosphorylated after induction in vivo, suggesting that TNF-alpha selectively activates only specific NF-kappa B heteromers and that modifications regulate not only I kappa B molecules but also NF-kappa B molecules.

the activation described in the *that*-clause is rather tentative observation induced by the fact of phosphorylation. Another example may be

> We have shown previously that both octamer binding transcription factors, namely the ubiquitous Oct-1 and the B cell-specific Oct-2A protein, can be enhanced in transcriptional activity by their association with the B cell-specific coactivator protein Bob1, also called OBF-1 or OCA-B.

where the fact represented by the *that-clause* is an old information already published elsewhere, and thus should not be extracted from the current paper.

Such information cannot be extracted just by looking at the verbs representing the biological reactions so that pattern matching on such verbs fails to capture the information carried by the verbs in main clause such as show and suggest.

3 Use of a Full Parser for Information Extraction

We try to overcome these problems by using a full parser that analyzes the whole structure of sentences. We first convert the surface form of sentences to an intermediate canonical form called an *argument structure* using a general-purpose, domain independent parser. Event information is then extracted by domain-specific mapping rules from argument structures to frame representations. The workload of defining mapping rules can be significantly reduced since the surface variation is absorbed in the few argument structures. Using a general-purpose grammar for syntactic analysis makes it possible to modularize the system, so that the IE system as a whole becomes easy to be tuned to specific domains, and easy to be maintained and improved.

The use of a parser also enables the IE system to find the dependency between all the phrases in the sentence. We can extract the sequence of other relationships between events expressed in one sentence, or the authors' view represented by embedding, in a more straightforward way.

However, A full parsing approach has not been used in practical applications on the basis of the following three reasons. First, full parsers in general tend to be slower, and need a large memory than shallow analysis such as pattern matching because they handle the full possible structure of whole sentences even when the full structure is not necessary. Second, it is often argued that the results of full parsers have more ambiguity than that of pattern matchers because full parsers produce the full structure of a sentence whereas pattern matching methods produces a partial structure by ignoring the part of the sentences that do not match the pattern. Third, full parsers have lower coverage than shallower analyzes because of the complexity of process.

In the current work, we introduce two preprocessors that resolves the local ambiguities in sentences to improve the efficiency. One of the preprocessor is

a term recognizer[4,5] that glues the words in a noun phrase into one chunk so that the parser can handle them as if it is one word. The other is a shallow parser that reduces the lexical ambiguity.

A term recognizer identifies technical terms and classify them into semantic classes such as protein names and source names. The internal structure in the phrase can be ignored, and unnecessary processing can be avoided. The effect of reducing this kind of unnecessary analysis is more significant in technical papers in biomedical domain, because there are long and complex names such as 'B-cell specific transcription factor' and 'signal transducer and activator of transcription (STAT) protein'. The term recognizer also solves the coverage problem caused by technical terms that are not properly recognized due to the lack of entries in the lexicon of a general–purpose parsers. A virtual lexical entry can be added in the dictionary corresponding to each class of terms produced by the recognizer. Then, the parser can use those lexical entries to interpret the terms.

A shallow parser reduces the lexical ambiguity by using local constraints to discard more unlikely part of speech and try to make dependency structure that can be constructed locally. Using this information, we can restrict the lexical entries used by the full parser to those within the local constraints by the shallow parser. This is similar to using a part–of–speech tagger, but a shallow parser can reduce lexical ambiguity more effectively by imposing stronger constraints.

The remaining problems can be solved by introducing postprocessor that handles the result of the parser. For ambiguity resolution, disambiguation heuristics such as taking the longest match can be also applied to the results of full parsing, but more general disambiguation methods, such as a statistical disambiguation method [11], can also be incorporated. We can also introduce a postprocessor that extracts information from the partial results of (failed) parsing to improve the coverage.

4 Implementation

Figure 1 shows the design of an IE system using a full parser. The surface form of sentences are converted to an intermediate canonical form called an *argument structure* by the parser and then frame structures for the events are constructed from the argument structure. The user of the IE system, usually a domain specialist, provides domain-specific knowledge as set of the target verbs and mapping rules from the argument structures of the verbs to the frame structures of events as shown in Figure 2. The argument structure, shown in the left hand side of Figure 2, consists of attributes REL, ARGS, and ADJ,

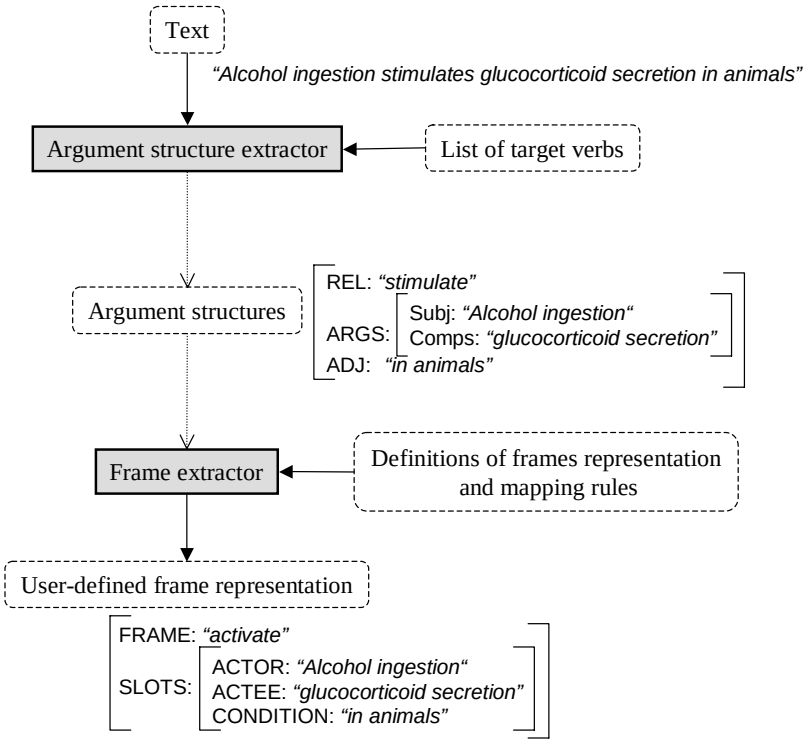

Figure 1: Overview of our system

and their values: The value of REL is the verb in base form; ARGS consists of SUBJ (subject) and COMPS (complement) subfeatures; The value of ADJ consists of the adjuncts, markers and other modifiers. The frame structure on the right hand side of Figure 2 represents of the frame of the event. Given this knowledge, the system outputs extracted information in user defined frame representation from natural language texts.

The system consists of *Argument Structure Extractor* and *Frame Extractor*. Argument Structure Extractor applies full parsing to input texts and extracts argument structures. First, a term recognizer and a shallow parser preprocess the texts to decrease the workload of a full parser. Then the full parser analyzes the syntax of the whole sentences and gains argument structures. Frame Extractor constructs user-defined frame representations from the argument structures, according to the mapping rule (Figure 2). In the map-

Target Verbs:

"bind", *"make (complex with)"*

Mapping Rules:

$$\begin{bmatrix} \text{REL} : \textit{"bind"} \\ \text{ARGS} : \begin{bmatrix} \text{SUBJ} : [1] \\ \text{COMPS} : [2] \end{bmatrix} \end{bmatrix} \rightarrow \begin{bmatrix} \text{FRAME} : \textit{"bind"} \\ \text{SLOTS} : \begin{bmatrix} \text{BINDER} : [1] \\ \text{BINDEE} : [2] \end{bmatrix} \end{bmatrix}$$

$$\begin{bmatrix} \text{REL} : \textit{"make"} \\ \text{ARGS} : \begin{bmatrix} \text{SUBJ} : [1] \\ \text{COMPS} : \textit{"complex"} \end{bmatrix} \\ \text{ADJ} : \textit{"with [2]"} \end{bmatrix} \rightarrow \begin{bmatrix} \text{FRAME} : \textit{"bind"} \\ \text{SLOTS} : \begin{bmatrix} \text{BINDER} : [1] \\ \text{BINDEE} : [2] \end{bmatrix} \end{bmatrix}$$

Figure 2: Users' input of target verbs and mapping rules based on domain-specific knowledge

ping rules, users can specify synonyms (lexical variation[a]) to output in the same frame form For example, according to the knowledge in Figure 2, argument structures of "make complex with " and "interact" are abstracted in the same frame form.

We implemented a preliminary system of Argument Structure Extractor and applied it to abstracts on MEDLINE database. An HPSG-based parsing system XHPSG [12] is used as a full parser. The XHPSG grammar is for general-purpose natural language processing but we did not tune the grammar or the lexicon for biomedical domain. As a shallow parser, we adopt ENGCG [13] because it retains the ambiguity in the output if it cannot resolve the ambiguity by local constraints.

The term recognizer and the postprocessor module for using of partial parsing results and applying general disambiguation technique are not implemented yet. In the following experiments, we use a corpus which is annotated with named-entities by biologists[14] to simulate the effect of the term recognizer, and use a simple heuristics of taking the results with the shortest arguments and the ones with the longest arguments. All modules are implemented in Perl and a feature structure manipulation language LiLFeS [15]. The experiments are performed on Pentium III Xeon 550MHz with 4GB of memory.

5 Experimental Results and Discussions

First, we estimated the effect of preprocessors (ENGCG and manual annotation of named-entities). We parsed 179 sentences from the annotated corpus from

[a] Lexical variation sometimes involves variation of argument structures. e.g. "bind X" and "make complex with X"

Table 1: Result of extraction of argument structures

	# of arg. structures
(1) extracted uniquely	31
(2) extracted with ambiguity	32
(3) not extracted	70
total	133

"<PROTEIN_10> incorporation into cells was also observed when the cells were incubated with <PROTEIN_2> or with <PROTEIN_3>.

$$\left[\begin{array}{l} \text{REL} : \textit{"observe"} \\ \text{ARGS} : \left[\ \text{Comps} : \textit{"<PROTEIN\_10> incorporation into cells"}\ \right] \\ \text{ADJ} : \textit{"also", "when the cells were incubated",} \\ \qquad\qquad \textit{"with <PROTEIN\_2> or with <PROTEIN\_3>"} \end{array}\right]$$

$$\left[\begin{array}{l} \text{REL} : \textit{"incubate"} \\ \text{ARGS} : \left[\ \text{Comps} : \textit{"the cells"}\ \right] \end{array}\right]$$

Figure 3: Example of input sentence and output argument structures

MEDLINE abstracts[b] with XHPSG parser with and without the preprocessors. Both parsers output the correct results (parse trees) for 66 sentences. For these sentences, the number of generated constituents (roughly showing memory usage) was reduced from 2597 to 786 and parsing time from 19.5 seconds to 2.7 seconds. For the parsing speed, we can expect more efficiency by introducing faster algorithms for parsing HPSG. For example, TNT parser[16], which is a high-speed HPSG parser with filtering techniques, achieves 50 times faster parsing speed compared with the parser used in this experiment. Thus, the current results support the practical usability of full parsers for IE systems.

Next, we evaluated the recall of extracting argument structures for first 97 out of the 179 sentences. These 97 sentences include 133 argument structures (some sentences include more than one argument structures) to be extracted. We ignored the differences in the structure inside ADJ in the argument structures, i.e., two argument structures are considered equevalent if the boundaries of the strings that are the elements of ADJ values are equal. Among 133 argument structures, our system extracted (1) 31 (23%) argument structures uniquely (modulo the substructure of ADJ), (2) 32 (24%) with ambiguity, and could not extract (3) 70 (53%) as shown in Table 1. Figure 3 shows an example

[b]Because of memory limit, currently we split sentences at some of subordinating conjunction.

"phorbol esters may induce posttranslational modifications of cellular transcription factors that alter their DNA-binding characteristics"

$$\left[\begin{array}{ll} \text{REL} & : \textit{"induce"} \\ \text{ARGS} & : \left[\begin{array}{ll} \text{SUBJ} & : \textit{"phorbol esters"} \\ \text{COMPS} & : \textit{"posttranslational modifications"} \end{array}\right] \\ \text{ADJ} & : \textit{"of cellular transcription factors that alter their DNA-binding characteristics"} \end{array}\right]$$

$$\left[\begin{array}{ll} \text{REL} & : \textit{"induce"} \\ \text{ARGS} & : \left[\begin{array}{ll} \text{SUBJ} & : \textit{"phorbol esters"} \\ \text{COMPS} & : \textit{"posttranslational modifications of cellular transcription factors that alter their DNA-binding characteristics"} \end{array}\right] \end{array}\right]$$

Figure 4: Example of ambiguous output

of input and output of the parser. The tags <PROTEIN_2>, <PROTEIN_3>, and <PROTEIN_10> are generated by the preprocessor (or, in the current experiment, human biologists) to replace the long names of proteins.

The 31 argument structures extracted uniquely were extracted from 28 sentences. The number of the parse results for these sentence are large: only 12 sentence had less than 10 parse trees. There were 3 sentences with about 10 parse trees, 8 sentence with about 100 trees, 3 sentence with about 10,000 trees, and 2 sentence had more than 1,000,000 parse trees. Still, the system was able to extract the unique argument structures. This result claims that we can ignore the ambiguity caused by phrases unrelevant to desired information, such as modifiers, and can extract the information with much less ambiguity without any disambiguation methods.

Among the ambiguous results ((2) in Table 1), the most problematic case of ambiguity was that of PP attachment, i.e., whether an NP and a PP in 'NP+PP' construction should be separated to two slots or not. This problem was found in 21 of the 32 ambiguous results.

PP attachment is a general problem that occurs in other domain as well. Our group is investigating a domain-independent disambiguation method by a stochastic model on feature structures, which can be integrated with our system in the future. For example, Figure 4 shows an example of ambiguous argument structures caused by a PP-attachment problem. In such cases, simple disambiguation methods, such as one by shortest/longest matching with regular expressions, cannot correctly disambiguate them, but the disambiguation requires information of individual words such as cooccurrence. In the

Table 2: Cause of (3) failure of extraction

	# of arg. structures	
failure of parsing		53
extractable from partial results	26	
not extractable	27	
failure for memory limit		7
other miss		10
total		70

example, we can expect to see the "modification"–"of"–"factor" relation more often than the "induce"–"of"–"factor" relation, so the structure that includes the "of"-phrase in the COMPS should be selected.

Table 2 shows the cause of parsing failure. Further investigations revealed that most of the parsing failure was concerned with punctuations and coordinations. This indicates that the sentences in this domain have much more complicated coordination structure than those in newspaper articles for which the grammar was originally constructed and that the grammar must be enhanced in this aspect.

We further investigate the possibility of the use of partial results by manual investigation of the partial results of sentences for which no argument structure was obtained because of the failure of parsing. Among 53 argument structures, 26 were extracted manually from partial results, as shown in Table 2. The argument structures obtained from partial results generally lacked the information on modifiers in the ADJ slot, but the basic ACTOR–ACTEE relationship was obtained. This tentative result suggests that even when the information from an entire sentence cannot be extracted, an IE system can make use of the information obtained from some local part of a sentence. These partial result can be used as the equivalent of those information obtained by traditional pattern matching methods that extract the information by looking at the phrases locally. Thus, with a processor that can handle the partial results of parsing, we can expect an IE system that can make use of the global information from the entire sentence when available and the partial information from the local information from a part of the sentence when full parsing results are not available. Note that the disambiguation module and the processor of partial results can be constructed in domain independent way. Thus, the problem of ambiguity and low coverage can be solved more systematically with a full parsing method in contrast to one-by-one adding of surface string patterns.

6 Conclusion and Future Works

We have designed and implemented an information extraction system using a full parser to investigate the plausibility of full analysis of text using general-purpose parser and grammar applied to biomedical domain. We partially solved the problems of full parsing of inefficiency, ambiguity, and low coverage by introducing the preprocessors. The preliminary experiment showed that, among 133 argument structures to be extracted, 31 (23%) were extracted uniquely, and 32 (24%) were extracted with ambiguity. With a disambiguation module and the postprocessor that enables the system to use the partial results, 99 (74%) argument structures in total are expected to be extracted. From the result, we can conclude that the full parsing method is plausible in application to IE system in this domain.

The design and implementation of postprocessors is the most important aspect of our future work, along with the enhancement of a grammar. For constructing a practical IE system, the user interface for defining the frame structure and showing the result will also be an important issue.

References

1. Y. Ohta, Y. Yamamoto, T. Okazaki, and T. Takagi. Automatic construction of knowledge base from biological papers. In *Proc. 5th International Conference on Intelligent Systems for Molocular Biology*, pages 218–225, 1997.
2. K. Fukuda, T. Tsunoda, A. Tamura, and T. Takagi. Towards information extraction: Identifying protein names from biological papers. In *Proc. 3rd Pacific Symoisium of Biocomputing*, pages 707–718, 1998.
3. D. Proux, F. Rechenmann, L. Julliard, V. Pillet, and B. Jacq. Detecting gene symbols and names in biological texts: A first step toward pertinent information extraction. In *Genome Informatics 1998*, pages 72–80. Universal Academy Press, 1998.
4. N. Collier, C. Nobata, and J. Tsujii. Extracting the Names of Genes and Gene Products with a Hidden Markov Model. In *Proc. of COLING 2000*, pages 201–207, 2000.
5. C. Nobata, N. Collier, and J. Tsujii. Automatic Term Identification and Classification in Biology Texts. In *Proc. NLPRS*, 1999.
6. T. Sekimizu, H. S. Park, and J. Tsujii. Identifying the interaction between genes and gene products based on frequently seen verbs in MEDLINE abstracts. In *Genome Informatics*, pages 62–71. Universal Academy Press 1998, 1998.

7. K. Humphreys, G. Demetriou, and R. Gaizauskas. Two applications of information extraction to biological science journal articles: Enzyme interactions and protein structures. In *Proc. 5th Pacific Symoisium of Biocomputing*, pages 72–80, 2000.
8. J. Thomas, D. Milward, C. Ouzounis, S. Pulman, and M. Carroll. Automatic extraction of protein interactions from scientific abstracts. In *Proc. 5th Pacific Symposium on Biocomputing*, pages 538–549, 2000.
9. T. C. Rindflesch, L. Tanabe, J. N. Weinstein, and L. Hunter. Edgar: Extraction of drugs, genes and relations from the biomedical literature. In *Proc. 5th Pacific Symposium on Biocomputing*, pages 514–525, 2000.
10. T. Ono, H. Hishigaki, A. Tanigami, and T. Takagi. Automatic extraction of information on protein-protein interaction from scientific literature. In *Genome Informatics 1999*, pages 296–297. Universal Academy Press, 1999.
11. J. Carroll and G. Minnen. Can Subcategorisation Probabilities Help a Statistical Parser? In *Proc. WVLC-6*, pages 118–126, 1998.
12. Y. Tateisi, K. Torisawa, Y. Miyao, and Jun'ichi Tsujii. Translating the XTAG english grammar to HPSG. In *Proceedings of TAG+4 workshop*, 1998.
13. A. Voutilainen. Designing a (finite-state) parsing grammar. In Emmanuel Roche and Yves Schabes, editors, *Finite-State Language Processing*. A Bradford Book, The MIT Press, 1996.
14. Y. Tateisi, T. Ohta, N. Collier, C. Nobata, and J.Tsujii. Building an annotated corpus in the molecular biology domain. pages 28–34, 2000.
15. T. Makino, M. Yoshida, K. Torisawa, and J. Tsujii. LiLFeS — towards a practical HPSG parser. In *Proc. COLING–ACL,'98*, pages 807–811, 1998.
16. K. Torisawa, K. Nishida, Y. Miyao, and J. Tsujii. An HPSG parser with CFG filtering. *to appear in journal of Natural Language Engineering Special Issue — Efficient Processing with HPSG: Methods, Systems, Evaluation*, 2000.