Recycling Terms into a Partial Parser

Christian Jacquemin
Institut de Recherche en Informatique de Nantes (IRIN)
IUT de Nantes
3, rue du Maréchal Joffre
F-44041 NANTES Cedex 01 – FRANCE
jacquemin@irin.iut-nantes.univ-nantes.fr

Abstract

Both full-text information retrieval and large scale parsing require text preprocessing to identify strong lexical associations in textual databases. In order to associate linguistic felicity with computational efficiency, we have conceived *FASTR* a unification-based parser supporting large textual and grammatical databases. The grammar is composed of term rules obtained by tagging and lemmatizing term lists with an on-line dictionary. Through *FASTR*, large terminological data can be recycled for text processing purposes. Great stress is placed on the handling of term variations through metarules which relate basic terms to their semantically close morphosyntactic variants.

The quality of terminological extraction and the computational efficiency of *FASTR* are evaluated through a joint experiment with an industrial documentation center. The processing of two large technical corpora shows that the application is scalable to such industrial data and that accounting for term variants results in an increase of recall by 20%.

Although automatic indexing is the most straightforward application of *FASTR*, it can be extended fruitfully to terminological acquisition and compound interpretation.

Introduction

Large terminological databases are now available and can be used as lexicons in Natural Language Processing (NLP) systems aimed at terminology extraction. In *FASTR* term lists are transformed into large lexicalized grammars and are parsed with a robust and computationally tractable unification-based parser. Our method contrasts with pattern-matching techniques by offering an expressive and convenient descriptive framework. It also differs from a general multipurpose parser by an ability to recycle linguistic knowledge embodied in terminological data. Higher quality in terminological extraction is achieved thanks to a description of term variations.

Areas of application using such a tool for terminology extraction include automatic indexing through an assignment of text pointers to thesaurus entries, knowledge acquisition form textual databases, noun phrase structural disambiguation and machine translation with a specific concern for the translation of idioms, compounds and terms.

When designing any NLP system with large linguistic resources, there is tension between tractability and descriptive power. Finite state automata are efficient tools for lexical extraction. But their lack of convenience for information description makes the testing of different methodological choices difficult. Such a limitation is specifically problematic during the development stage. Symmetrically, unification-based parsers offer rich and conceptually tractable formalisms, but their computational cost is very high. The approach taken in *FASTR* is to use a convenient grammatical description stemming from *PATR-II* (Shieber 1986) associated with an optimized computational engine. Efficiency and constraint-based grammar formalism have motivated the acronym of the application (*FAST + PATR-II*) that stands for *FAST TERM RECOGNIZER*.

When terminology extraction is applied to automatic indexing, two measures are important : *recall* and *precision*. Precision is crucial for applications using acquisition methods which are subject to an excessive recall, blurring terminological entries with syntactic recurrences or semantic preferences. Conversely, in a knowledge-based method like *FASTR*, recall is a decisive evaluation of the coverage of the extraction. The recall rate mainly depends on the ability of the processor to extract term occurrences which differ from their description in the terminological base. With the purpose of enhancing the recall rate, *FASTR* includes a metagrammar used to generate term variant rules from term rules. Such an addition of robustness does not entail a degradation of precision because variations are restricted to a "safe" window bordered by the term components.

The formalism of *FASTR* is organized into three levels : a single word lexicon, a terminological grammar and a metagrammar for term variations. The initialization of *FASTR* consists of the description of the inflectional system of the language under study, the generation of a lexicon and a grammar from a list of terms with an on-line lexicon and the handcrafted creation of a set of paradigmatic metarules (about a hundred) which are refined according to the experimental results.

Processing in *FASTR* starts with segmentation and stemming. During stemming, a few term rules are activated through a bottom-up filtering. Then, metarules are applied to these rules and yield transformed rules used to extract terms and their variants. For example, from the preceding sentence and from a grammar including *term variant*, the sequence *terms and their variants* would be extracted as a variation of *term variant*. The data required by *FASTR* consist of a declension file, an initial terminological database and an on-line dictionary to transform terms into compilable linguistic data. As far as human time is concerned, only a slight experimental tuning of the metarules is necessary.

A Three-tier Formalism

The formalism of *FASTR* stems from *PATR-II* (Shieber 1986). Rules are composed of a context-free portion describing the concatenation of the constituents and a set of equations constraining information of these constituents.

The description of a single word includes minimally the string of the word stem, a part-of-speech category and its inflection number. These three values are used to dynamically rebuild the different inflections of the word. They are automatically extracted from an on-line dictionary with morphological information. We currently use the DELAS dictionary of LADL laboratory (University of Paris 7). For example, rule (1) describes the noun *ray*, plural *rays*.

(1) Word : 'ray'
<cat> = 'N'
<inflection> = 1.

Terms are described by grammar rules. The formalism of *PATR-II* has been extended to support such additional facilities as rules with an extended domain of locality, structure disjunction and negative atomic values. Rule (2) represents the term *[X ray] diffraction*. This rule localizes the embedded structure *X ray*. Lexical anchors, indicated by the value of the feature lexicalization, are used prior to the parsing phase for a selective bottom-up activation of the rules. For example, rule (2) is anchored to *diffraction* and is activated when this word is encountered in the input sentence.

(2) Rule : $N_1 \rightarrow (N_2 \rightarrow N_3\ N_4)\ N_5$
<N_1 label> = 'XRD'
<N_1 metaLabel> = 'XX'
<N_1 lexicalization> = 'N_5'
<N_3 lemma> = 'X'
<N_3 inflection> = 7
<N_4 lemma> = 'ray'
<N_4 inflection> = 1
<N_5 lemma> = 'diffraction'
<N_5 inflection> = 1.

The third level of the formalism consists of a metagrammar. Metarules are composed of two context-free descriptions : the source and the target and a set of equations constraining them. Information shared by the source and the target is embodied by identical symbols. For example, metarule (3) describes a coordination of a two-constituent term inserting a conjunction (except *but*) and a word (except a definite or indefinite determiner) between both constituents. When applied to rule (2), it outputs a novel rule which accepts *X ray or neutron diffraction* as a variant of *X ray diffraction*.

(3) Metarule : Coor($X_1 \rightarrow X_2\ X_3$)
$= X_1 \rightarrow X_2\ C_3\ X_4\ X_3$ "'C' = conjunction "
<X_1 metaLabel> = 'XX"'
<C_3 lemma> ! 'but' "' ! denotes inequality"'
<X_4 cat> ! 'Dd' "'Dd'= definite determiner"
<X_4 cat> ! 'Di'. "'Di'= indefinite determiner"

Parsing

Morphology

FASTR has been applied to the French and the English languages and can be easily extended to any language without word agglutination thanks to an external description of morphology. The suffix stripping operation precedes syntactic analysis and requires a dictionary of lemmas and a declension file (Savoy 1993). Each entry in the dictionary has a basic stem and words with an irregular inflectional morphology such as *mouse/mice* have one or more auxiliary stems. Derivational links such as *synapse/synaptic* can also be accounted for through multi-valued part-of-speech categories such as noun-adjective. The declension file is illustrated by formulae (4) and (5). A set of features is provided for each inflectional case of each inflected category (e.g. (4) for nouns). A list of suffixes corresponds to each declension class (e.g. (5) for the first two classes of nouns). ?1 indicates the first auxiliary stem. The inflection class of a word is denoted by the value of the feature inflection in word rule (1) and term rule (2).

" The two cases of nouns "
(4) N[1] <number> = 'singular'.
N[2] <number> = 'plural'.

" *dog/dog-s* (stem *dog*) "
(5) N[1] 0 s
" *mouse/mice* (stem *mouse*, aux. stem *mice*) "
N[2] 0 ?1

In order to prepare suffix stripping, a generalized lexicographic tree is built form the whole set of the reversed suffixes of the current language. Each inflected word is also reversed and all its endings corresponding to an actual suffix are removed. The corresponding stems are looked for in the dictionary. If one of their inflections is equal to the current inflected word, the features associated with the declension case are unified with the features of the lemma and attached to the inflected word. Thus, the morphological stemmer associates all its homographic inflections to an inflected word.

The term rules whose lexical anchor is equal to one of the lemmas in the input are activated and processed by a top-down algorithm. In order to ensure short parsing times, unification is delayed until rewriting is achieved. Whenever a rule fails to be parsed, it is repeatedly tried again on its variants generated by metarules.

Term syntax and local syntax

Metarules can be straightforwardly described by embedding the formalism into a logical framework where rule generation by metarules is calculated through unification. With this aim in mind, the definitions of term rules and metarules given in the preceding part can be transformed into logical (in)equations by using the formulae of Kasper and Rounds (1986). As in (Vijay-Shanker 1992), type variables whose denotations are sets of structures derived from non-terminals can be replaced by monoadic predicates. Individual variables that stand for individual feature structures are used to capture reentrance. For example, rule (2) is translated into formula (6). A monoadic predicate arity is added to restrict the application of metarules.

(6) $\mathrm{XRD}(x) \Leftrightarrow \mathrm{cat}(x) \approx \text{'N'} \wedge \mathrm{arity}(x) \approx 2$
$\wedge\ \mathrm{lexicalization}(x) \approx x_4 \wedge \mathrm{metaLabel}(x) \approx \text{'XX'}$
$\wedge\ 1(x) \approx x_1 \wedge \mathrm{cat}(x_1) \approx \text{'N'} \wedge \mathrm{arity}(x_1) \approx 2$
$\wedge\ 1(x_1) \approx x_2 \wedge 2(x_1) \approx x_3 \wedge \mathrm{cat}(x_2) \approx \text{'N'}$
$\wedge\ \mathrm{lemma}(x_2) \approx \text{'X'} \wedge \mathrm{inflection}(x_2) \approx 1$
$\wedge\ \mathrm{cat}(x_3) \approx \text{'N'} \wedge \mathrm{lemma}(x_3) \approx \text{'ray'}$
$\wedge\ \mathrm{inflection}(x_3) \approx 1 \wedge 2(x) \approx x_4 \wedge \mathrm{cat}(x_4) \approx \text{'N'}$
$\wedge\ \mathrm{lemma}(x_4) \approx \text{'diffraction'} \wedge \mathrm{inflection}(x_4) \approx 1$

Standard fixed-point semantics is associated to this syntax which is used to calculate the interpretation of such formulae. The denotation of a formula is an automaton calculated through an inductive interpretation of the terms it contains (Rounds and Manaster-Ramer 1987). As a consequence of this mathematical formulation, the metarules are expressed as couples of monoadic predicates with shared variables. For example, the metarule of coordination (3) is described by formula (7). The syntax of both sides of the metarule is identical to the syntax of rules except for the monoadic rule predicate p which is a variable. ¬ stands for negation.

(7) $\mathrm{Coor}(p(y) \Leftrightarrow \mathrm{arity}(y) \approx 2 \wedge 1(y) \approx y_1 \wedge 2(y) \approx y_2)$
$= (\mathrm{Coor}(p)\ (y) \Leftrightarrow \mathrm{arity}(y) \approx 4 \wedge 1(y) \approx y_1$
$\wedge\ 2(y) \approx y_3 \wedge 3(y) \approx y_4 \wedge 4(y) \approx y_2$
$\wedge\ \mathrm{cat}(y_3) \approx \text{'C'} \wedge \neg(\mathrm{lemma}(y_4) \approx \text{'but'})$
$\wedge\ \neg(\mathrm{cat}(y_4) \approx \text{'Dt'}) \wedge \neg(\mathrm{cat}(y_4) \approx \text{'Dd'}))$

The result of the application of a metarule to a rule is calculated in two steps. Firstly, the left-hand-side of the metarule is unified with the rule. If unification fails, no output rule is generated. Otherwise, let σ be the substitution providing the unification. Then, the formula of the transformed rule is equal to the right-hand-side of the metarule, where the variables are substituted according to σ. The computational implementation is straightforwardly derived from this calculus. For example, metarule (7) applies to rule (6) with the substitution σ (8) and yields the transformed rule (9) whose *PATR-II* expression is (10).

(8) $\sigma \equiv [y = x, \mathrm{XRD}/p, x_1 = y_1, x_4 = y_2]$

(9) $\mathrm{Coor(XRD)}(x) \Leftrightarrow \mathrm{cat}(x) \approx \text{'N'} \wedge \mathrm{arity}(x) \approx 4$
$\wedge\ \mathrm{lexicalization}(x) \approx x_4 \wedge \mathrm{metaLabel}(x) \approx \text{'XX'}$
$\wedge\ 1(x) \approx x_1 \wedge \mathrm{cat}(x_1) \approx \text{'N'} \wedge \mathrm{arity}(x_1) \approx 2$
$\wedge\ 1(x_1) \approx x_2 \wedge 2(x_1) \approx x_3 \wedge \mathrm{cat}(x_2) \approx \text{'N'}$
$\wedge\ \mathrm{lemma}(x_2) \approx \text{'X'} \wedge \mathrm{inflection}(x_2) \approx 1$
$\wedge\ \mathrm{cat}(x_3) \approx \text{'N'} \wedge \mathrm{lemma}(x_3) \approx \text{'ray'}$
$\wedge\ \mathrm{inflection}(x_3) \approx 1 \wedge 4(x) \approx x_4 \wedge \mathrm{cat}(x_4) \approx \text{'N'}$
$\wedge\ \mathrm{lemma}(x_4) \approx \text{'diffraction'} \wedge \mathrm{inflection}(x_4) \approx 1$
$\wedge\ 2(y) \approx y_3 \wedge 3(y) \approx y_4 \wedge \mathrm{cat}(y_3) \approx \text{'C'}$

(10) Rule : $N_1 \rightarrow (N_2 \rightarrow N_3\ N_4)\ C_6\ N_7\ N_5$
$\langle N_1$ label$\rangle$ = 'Coor(XRD)'
$\langle N_1$ metaLabel$\rangle$ = 'XX'
$\langle N_1$ lexicalization$\rangle$ = 'N_5'
$\langle N_3$ lemma$\rangle$ = 'X'
$\langle N_3$ inflection$\rangle$ = 1
$\langle N_4$ lemma$\rangle$ = 'ray'
$\langle N_4$ inflection$\rangle$ = 1
$\langle N_5$ lemma$\rangle$ = 'diffraction'
$\langle N_5$ inflection$\rangle$ = 1.

The mapping performed by the metarules in *FASTR* differs from the definition of metarules in GPSG (Gazdar *et al.* 1985) on the following points :

- The *matching* of the input rule and the source is replaced by their unification. The *correspondence* between source and target is achieved by identical variables shared by both sides of the metarule.
- In GPSG, when input rule and target disagree about the value of some feature, the target always wins. In *FASTR*, the target wins if its value for this feature is independent of its source. Conversely, if source and target share this value, the unification of the source and the rule fails and no output is provided.
- The metavariable W used in GPSG and standing for a set of categories is not available in *FASTR*. However, an empty category in the context-free skeleton can stand for any subtree of the original rule. Thus, variable y_1 from metarule (7), associated to X_2 in formula (3), stands for the subterm *X ray* when applied to rule (6).

When implementing metarules in a grammar parser, there are two possibilities for the time to apply the metarules to a rule. The *compile-time* application calculates all the images of all the rules in the grammar prior to parsing. In the *run-time* approach, metarules are dynamically applied to the active rules during parsing. Weisweber and Preuß (1992) demonstrate that there is no difference in complexity between both approaches. Moreover, in the compile-time approach, metarules generate a huge set of transformed rules which may make the parsing process totally inefficient. Due to the very large size of our grammar, we have opted for the dynamic approach. The computational performances of the application reported in (Jacquemin 1994a) indicate that the parser only spends 10% of its time in generating metarules and fully justify the run-time approach.

Computational Lexicalization

The keystone of the computational tractability is lexicalization which allows for a bottom-up filtering of the rules before parsing. It is completed by fast mechanisms for data access such as a B-Tree (for the disk resident lexicon of single words) and a Hash-Code table (for the memory resident stop words).

The formalism of *FASTR* is lexicalized in the sense of Schabes and Joshi (1990) because it is composed of rules associated with each lexical item which is the anchor of the corresponding rules. The parsing algorithm for lexicalized grammars takes advantage of lexicalization through a two-step strategy. The first step is a selection of the rules linked to the lexical items in the input. The second step parses the input with a grammar restricted to the filtered rules. In case of rules with multiple lexical items such as the rules representing multi-word terms, the anchor can be any of the lexical items. For example, the term *aortic disease* can be anchored either to *aortic* or to *disease*. In Jacquemin (1994b), an algorithm for optimizing the determination of computational anchors is described. It yields a uniform distribution of the rules on to the lexical items with respect to a given weighting function. A comparison between the "natural" lexicalization on the head nouns and the optimized one has been made with *FASTR*. It shows that the rules filtered by the optimized lexicalization represent only 57% of the rules selected by the *natural* lexicalization and ensure a 2.6-time higher parsing speed.

The computational performances of parsing with *FASTR* mainly depend on the size of the grammar (see Figure 1). The parsing speed with a 71,623-rule terminological grammar, a 38,536-word lexicon and 110 metarules is 2,562 words/minute on a Sparc 2 workstation (real time). As 71,623 terms is a reasonable size for a real-word multi-domain list of terms (for example *WordNet* currently includes 35,155 synonyms sets), a workstation is well-suited for processing large corpora with such terminological databases.

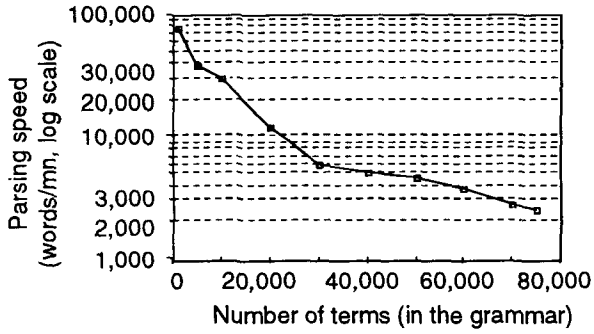

Figure 1. Parsing speed of *FASTR* (Sparc 2, real time)

Application to Automatic Indexing

A list of 71,623 multi-domain terms and two corpora of scientific abstracts have been provided by the documentation center *INIST/CNRS* : a 118,563-word corpus on metallurgy [METAL] and a 1.5-million word medical corpus [MEDIC]. The laboratory of *INIST/CNRS* has achieved tagging and lemmatization of terms and has evaluated the results of the indexing provided by *FASTR*.

In this experiment, the metagrammar consists of positive paradigmatic metarules (e.g. (11)) and filtering negative metarules rejecting the spurious variations extracted by the positive ones (e.g. (12)). Examples of variations from [MEDIC] accepted by (11) or rejected by (12) are shown in Figure 2.

(11) Metarule Coor($X_1 \rightarrow X_2\ X_3$)
$= X_1 \rightarrow X_2\ C_3\ X_4\ X_3$
<X_1 metaLabel> = 'XX'.

(12) Metarule NegCoor($X_1 \rightarrow X_2\ X_3$)
$= X_1 \rightarrow X_2\ C_3\ X_4\ X_3$
<X_1 metaLabel> = 'XX'
<X_4 cat> = 'P' "'P' = preposition"
<X_4 cat> = 'Dd'
<X_4 cat> = 'Di'.

Variations accepted by (11)
mechanical and enzymatic methods
Down and Williams syndromes
amplitude and frequency modulations
Northern and Western blotting

Variations rejected by (12)
relaxation and the time
satellite and whole chromosome
cells or after culture
tissue or a factor

Figure 2. Antagonist description of variations

Negative metarules are used instead of negative constraints such as the ones stated in (3) to keep a trace of the rejected variations. More details about this description are reported in (Jacquemin and Royauté 1994). An evaluation of terminology extraction on corpus [METAL] indicates that term variations represent 16.7% of multi-word term occurrences extracted by *FASTR* (an account for term variants increases recall by 20%). The three kinds of variants retrieved through metarules are coordinations (2%), modifier insertions (8.3%) and permutations (6.4%). See Figure 3 for examples. Elisions such as *Kerrr magnetooptical effect* → *Kerr effect* are not accounted for because our local approach to variation is not appropriate to elliptic references. In this framework, *FASTR* retrieves 74.9% of the term variants with a precision of 86.7%. These results confirm the substantial gain in recall obtained by accounting for term variants in automatic indexing. A

better precision could be reached through a more accurate description of permutation. An improvement in term variant recall requires the handling of elision.

Related Work

Firstly, our formalism is inspired by two fields of lexicalized and logical tree formalisms. The first one is the general framework of *Lexicalized Tree Adjoining Grammar* (LTAG) which has shown to be fruitful for the description of idioms (Abeillé and Schabes 1989). The second one is the important extension of *Tree Adjoining Grammar* (TAG) to a logical framework (Vijay-Shanker 1992) which contrasts with the traditional approach that operations in a TAG combine trees. From these works, we have adopted the constraint of LTAG which states that rules must have at least one lexical frontier node together with the logical representation of Vijay-Shanker (1992) where rules are not restricted to immediate dependency. The lexicalized tree grammar is motivated by the domain to be described : terms mainly consist of compounds with an internal structure and lexical constituents. The logical formalism provides us with a straightforward extension to metarules.

Secondly, our approach to text processing is a form of partial parsing. A current trend in large scale NLP system (Jacobs 1992) refuses to consider parsing as an exhaustive derivation of a very large grammar which would process any encountered sentence. To alleviate these problems parsing should be planned as the cooperation of several methods such as text preprocessing, parsing by chunks, multiple-step partial parsing, shallow parsing... etc. The scope of the preprocessing task is *"abstract[ing] idiosyncrasies, highlight[ing] regularities, and, in general feed[ing] digested text into the unification parser"* (Zernik 1992). With this aim in mind *FASTR* brings forth occurrences of complex lexical entries and their local variations. It is adapted to integration in a multi-step parsing strategy. It takes as input a raw corpus and yields chunks corresponding to partial parses. This output can be fed into a following module or reprocessed with more precise metarules.

Thirdly, our research on term extraction places great stress on term variations. The most direct precursors of the use of term variation in information retrieval are Sparck Jones and Tait (1984). These authors advocate the systematic generation of syntactic term variants in query processing. Their approach, however, makes the assumption that only semantic equivalent variant should be generated and that each of the words in a variant should be given instead of allowing paradigmatic places. They only account for restricted associations such as *information retrieval/retrieval of information*. Strzalkowski and Vauthey (1992) follow the way suggested by Sparck Jones and Tait (1984) at the end of their paper. Instead of generating term variants in a query, they look for different term occurrences in text documents analyzed by a general multipurpose parser. Their parse trees are composed of head/modifier relations of four categories. These four classes account for most of the syntactic variants of two-word terms into pairs with compatible semantic content such as *information retrieval/information retrieval system/retrieval of information from databases...* We think however that most of these variants can be extracted without parsing the whole sentence. They can be detected safely through a local parse with a noun-phrase micro-syntax.

Extensions and Conclusion

Although applied straightforwardly to automatic indexing, *FASTR* can be extended to terminology acquisition through a bootstrapping method where new terms are acquired by observing the variations of controlled terms in corpora. Figure 3 reports four occurrences of term variants retrieved through three metarules belonging to three different families. Each of these occurrences yields a novel candidate term which either already belongs to the terminology or can be added after validation.

A second extension of *FASTR* concerns acquisition of noun phrase interpretation from a corpus. Observation of variation is an opportunity to find objective linguistic clues which denote the semantic relation between both words of a binominal compound. For example, *cell into a metastatic tumor* is a permutation of *tumor cell* involving the preposition *into*. Figure 4 lists four *N cell* terms for which more than four permutations *cell Prep X N* have been encountered in corpus [MEDIC]. The prepositions found in more than one permutation are followed by their number of occurrences. For example, the prepositions encountered in the permutations of *blood cell* are *from*, *in*, *into* and *on*. These four prepositions denote a relation of spatial inclusion of a trajector *cell* into a landmark *blood* (Langacker 1987).

Term	Variation	Candidate term
water absorption	*water and sodium absorption* (coordination)	*sodium absorption*
Central Africa	*Central and West Africa* (coordination)	*West Africa*
controlled delivery	*controlled drug delivery* (insertion)	*drug delivery*
magnetic coupling	*magnetic transcutaneous coupling* (insertion)	*transcutaneous coupling*
information access	*access to lexical information* (permutation)	*lexical information*
wave effect	*effect of short wave* (permutation)	*short wave*

Figure 3. Acquisition of candidate terms through variation

Term	Prepositions
Membrane cell	*in* [4], *into, to*
Myeloid cell	*of* [3], *from*
Blood cell	*from* [8], *in* [13], *into, on*
Tumor cell	*in* [3], *from* [4], *into, with, of*

Figure 4. Noun phrase interpretation through variation

Although initially devised for automatic indexing, *FASTR* can play a crucial role in other text-based intelligent tasks. This part has sketched out a picture of incremental terminological acquisition and noun-phrase understanding through the analysis of term variants.

As Resnik (1993) points out, large-scale knowledge sources can be used as a source of lexical information. Similarly, our approach to corpus linguistics makes a extensive use of terminological data and investigates systematically and precisely the variations of terms in technical corpora. The next natural step in term and compound processing is to provide *FASTR* with a learning ability. With this aim in mind, we are currently investigating two novel research directions : firstly, a hybridisation of *FASTR* with a connectionist model dedicated to nominal composition (Jacquemin 1993) and, secondly, a cooperation between *FASTR* and *LEXTER* (Bourigault 1993) a tool for term acquisition through the filtering of part-of-speech patterns.

Acknowledgement

I would like to thank Jean Royauté from INIST/CNRS for his helpful and friendly collaboration on this project. Many thanks also to Benoît Habert from ENS Fontenay for numerous constructive discussions.

References

Abeillé, Anne and Yves Schabes. 1989. Parsing Idioms in Lexicalized Tags. In *Proceedings, 4th Conference of the European Chapter of the Association for Computational Linguistics (EACL'89)*, Manchester, June 1989, 1–9.

Bourigault, Didier. 1993. An Endogeneous Corpus-Based Method for Structural Noun Phrase Disambiguation. In *Proceedings, 6th European Chapter of the Association for Computational Linguistics (EACL'93)*, Utrecht, June 1993.

Gazdar, Gerald, Ewan Klein, Geoffrey Pullum, Ivan Sag. 1985. *Generalized Phrase Structure Grammar*, Oxford : Blackwell.

Jacobs, Paul S. (edt). 1992. *Text-based Intelligent systems, Current Research and Practice in Information Extraction and Retrieval.* Hillsdale : Lawrence Erlbaum.

Jacquemin, Christian. 1993. A Coincidence Detection Network for Spatio-Temporal Coding : Application to Nominal Composition. In *Proceedings, 13th International Joint Conference on Artificial Intelligence (IJCAI'93)*, Chambéry, August 1993, 1346–1351.

Jacquemin, Christian. 1994a. *FASTR* : A unification grammar and a parser for terminology extraction from large corpora. In *Proceedings, IA-94*, Paris, June 1994.

Jacquemin, Christian. 1994b. Optimizing the computational lexicalization of large grammars. In *Proceedings, 32nd Annual Meeting of the Association for Computational Linguistics*, Las Cruces, June 1994.

Jacquemin, Christian and Jean Royauté. 1994. Retrieving terms and their variants in a lexicalized unification-based framework. In *Proceedings, 17th Annual International ACM SIGIR Conference (SIGIR'94)*, Dublin, July 1994.

Kasper, Robert T. and William C. Rounds. 1986. A logical semantics for feature structures. In *Proceedings, 24th Annual Meeting of the Association for Computational Linguistics*, NY, June 1986, 257–266.

Langacker Ronald W. 1987. *Foundations of Cognitive Grammar. Vol I. Theoretical Prerequisites.* Stanford : Stanford University Press.

Resnik, Philip S. 1993. *Selection and Information : A Class-Based Approach to Lexical Relationships.* Ph D diss in Computer Science, University of Pennsylvania.

Rounds, William C. and Alexis Manaster-Ramer. 1987. A logical version of functional grammar. In *Proceedings, 24th Annual Meeting of the Association for Computational Linguistics*, Stanford CA, July 1987, 257–266.

Savoy, Jacques. 1993. Stemming of French words based on grammatical categories. 1993. *Journal of the American Society for Information Science*, Vol. 44, No 1, January 1993, 1–10.

Schabes, Yves and Aravind K. Joshi. 1990. Parsing with Lexicalized Tree Adjoining Grammar. In *Current Issues in Parsing Technologies*, Masaru Tomita (edt), Dordrecht : Kluwer Academic Publishers.

Shieber, Stuart N. 1986. *An Introduction to Unification-Based Approaches to Grammar.* CSLI Lecture Notes 4, Stanford, CA : CSLI.

Sparck Jones, Karen and J. I. Tait. 1984. Automatic Search Term Variant Generation. *Journal of Documentation*, Vol. 40, No. 1, March 1984, 50–66.

Strzalkowski, Tomek and Barbara Vauthey. 1992. Information Retrieval Using Robust Natural Language Processing. In *Proceedings, 30th Annual Meeting of the Association for Computational Linguistics (ACL'92)*, Newark, DE, June 1992, 104–111.

Vijay-Shanker, K. 1992. Using Description of Trees in a Tree Adjoining Grammar. *Computational Linguistics*, Vol. 18, No. 4, December 1992, 481–518.

Weisweber, Wilhelm and Susanne Preuß. 1992. Direct Parsing with Metarules. In *Proceedings, 14th International Conference on Computational Linguistics (COLING'92)*, Nantes, July 1992, 1111–1115.

Zernik, Uri. 1992. Shipping Departments vs. Shipping Pacemakers : Using Thematic Analysis to Improve Tagging Accuracy. In *Proceedings, Annual Meeting of the American Association for Artificial Intelligence (AAAI-92)*, 335–342.