Exploiting Semantic Knowledge in LTAG-based controlled indexing of technical data

Patrice Lopez
LORIA Labs

David Roussel
EADS Suresnes

1. Introduction

The work presented in this abstract follows the first experiments presented in (Lopez and Roussel, 2000) on the robust modeling of terms in the LTAG framework to index spoken annotation transcriptions. We continue to experiment the LTAG workbench (Lopez, 2000), and integrate it with on the shelf tools (term extractor, taggers, terminological model) that embed and manage different kind of linguistic resource. The key advantages of using the LTAG formalism in this context is a precise linguistic modeling useful for the representation of term variants, the exploitation of semantic constraints and the ability to specialize the terminological resources to several specific tasks. To illustrate the last point, we first present another application that motivates this work, the exploitation of technical documentation. In this particular application, the semantic disambiguisation can help to improve the accuracy of the documents and their reuse to design checking procedures. We then present more precisely the LTAG modeling and the implemented system TERESA based on a POS tagger, finite-state transducers encoding LTAG trees and a semantic tagger.

2. Application

When documentation is an important part of a company activity, there are always some existing resources which formalize semantic information available for technical words. For example, currently, in the EADS context, the design of an ontology that gives the semantic categories of specific terms is considered as an important starting point. During the document life cycle within a project, an ontology facilitates also intra-operation between different kind of document and so, a mandatory part of the work being done by the project community is to standardize the terms, acronyms and abbreviations. This task is an EADS directive and procedure.

Since these terms are already defined, their identification for the purpose of classifying and accessing documents is called *controlled indexation* in opposition to free indexation where the index terms are automatically defined. Controlled indexation allows us to exploit existing resource to achieve a better precision in the indexation and to link old and new information in a more coherent and comprehensive way for documentalists.

The experiments in this work have been made with XML elements called WARNINGS extracted from an aircraft documentation. The correct identification of a particular term and its variants The use of controlled indexing on these elements is twofold : first, help the navigation into those elements in order ot control the coherence of the content of these element, second, to be able to disambiguate semantically sequence of words.

An expected enhancement of robust controlled indexing is to derive more easily a procedure from the description of the warning in the whole documentation, or at least to take more easily into account the important warning in the procedures. The identification of a particular operation benefit from a disambiguation of certain sequence of words.

For instance, engine operation concern the motor intervention (table 1) or the system intervention (table 1). To avoid engine damaged, it is sometime necessary to access both the cockpit and the motor. One interest of semantic knowledge exploitation in controlled indexing is to extract directly the sentences that concern one type of engine intervention.

You must not operate the engine with the fan cowl doors open.
During engine operation, the bleed valve can open.
Operation of the engine can cause fuel to go overboard.
Ear protection must be worn by all person who operate the engine while engine operates.

Figure 1: Example of motor intervention.

The engine must operate 9 hours at idle with the lubrication system filled.
Do not motor, start or operate engine unless a positive fuel inlet pressure is indicated.
The exhaust gas is hot when the engines operates at idle or higher power.
To maintain satisfactory engine oil temperature do an engine start and operate the engine at idle.

Figure 2: Example of engine intervention that need a control from the cockpit.

In the first case (motor intervention, table 1), the operation implies the filtering of sentences that gather a person as an agent or implicit agent. A syntactic analysis is enough to disambiguate this case from the next case (table 2), that implies that the engine is the subject of the operation. However, different elementary trees are concern, and don't provide an easy interface to the integration of the syntactic analysis of the terms within an application.

To consider a unique semantic feature instead of different kind of derived tree, we extend the syntactic categories of elementary trees with semantic constraints and compile them as FST. This allows us to keep abstraction in the description of linguistic resource and to benefit from other linguitic tool, namely the semantic tagger Tropes in order to study the dependance between semantic classes in a corpora. Tropes is a semantic analyser (Ghiglione et al., 1998). It embed morphosyntatic and semantic analyser that i) segment a text in linguistic proposition, ii) extract homogeneous category according to their thematic content, iii) export the result in a XML coding, iv) to count the frequence and the dependency between semantic classes. The Tropes environment facilitate the adaptation of the default semantic classes hierarchy in order to take into account specific semantic knowledge. A set of heuristics are applied to disambiguate the semenatic categorie of a lexical unit. They consist in finding isotopies of a same semantic classe and exploiting statistical coocurrences between complementary concepts inside a grammatical proposition.

3. LTAG-based Terminological Processing

3.1. LTAG representation of a terminology

A given term can be represented as a partial parsing tree, i.e. a derived tree, as represented figure 3. After removing all lexical information in this tree, we obtain a elementary tree schema in the sense of (Candito, 1996) that can be used to represent syntactically similar terms.

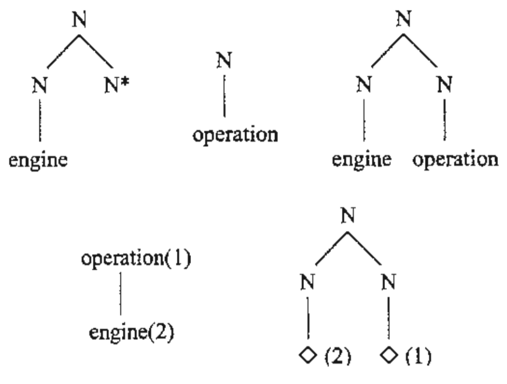


Figure 3: Anchored elementary trees, derived and derivation tree and the corresponding term schema for N-N terms.

This principle can be used to represent a complete terminology by parsing the list of terms with a LTAG grammar which coverage is limited or with existing term trees. For each term we obtain one or several derived and derivation trees. We have used the LTAG Workbench presented in (Lopez, 2000) for this purpose. Practically for English and French, a LTAG grammar covering only NP structures and basic verb and relatives is enough to cover

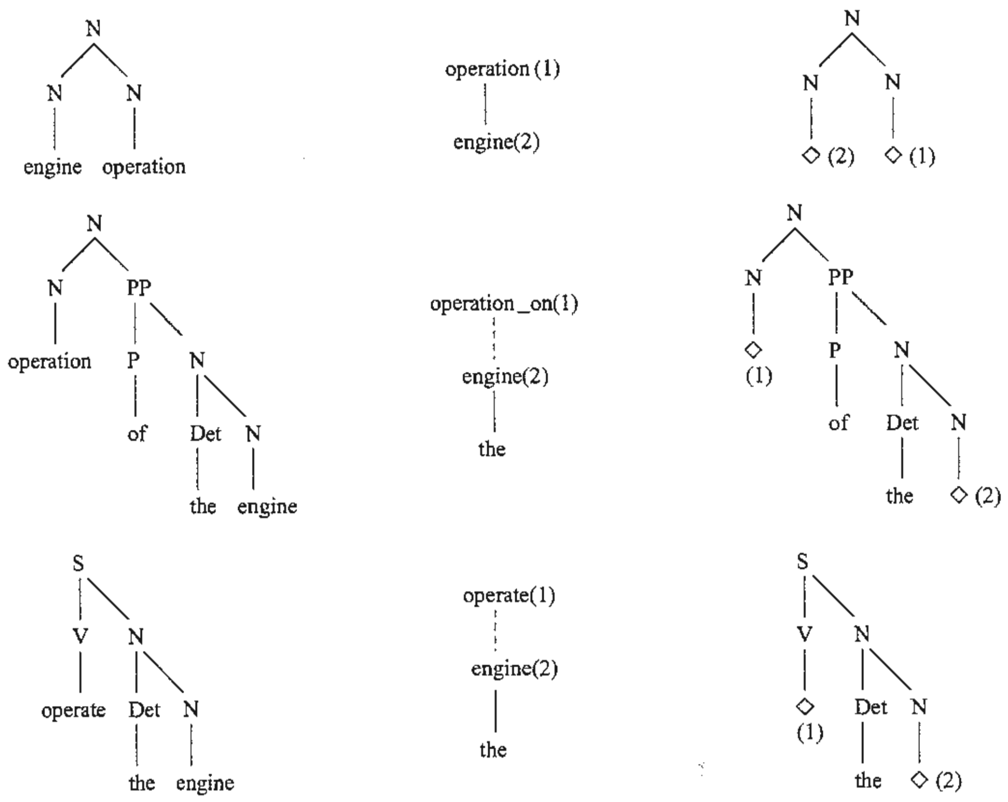


Figure 4: A basic term aligned with two of its variants.

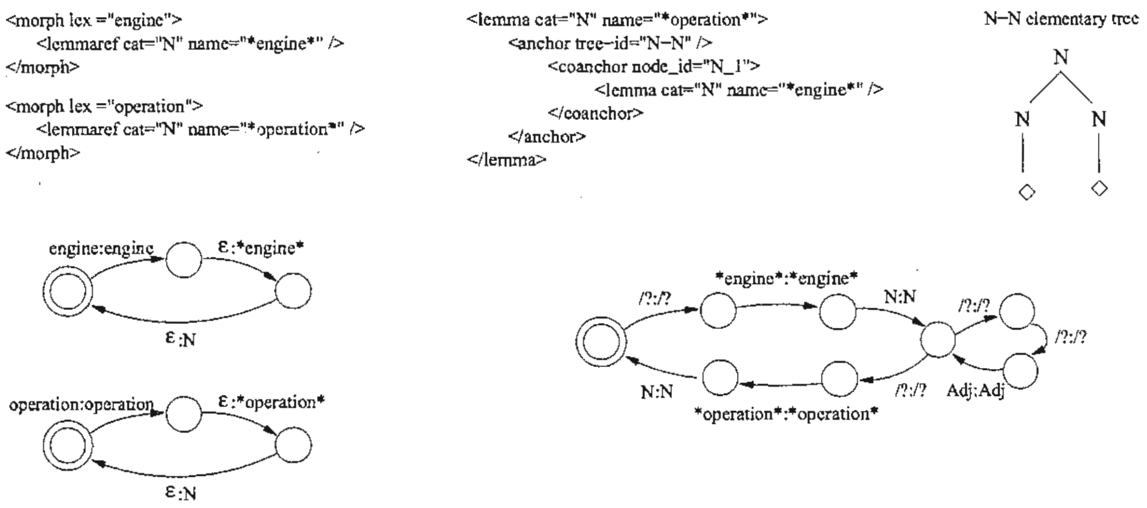
nearly all terms and their variants. The resulting set of elementary tree schema can be reused and be anchored by new terminologies. This linguistic representation allows us to extend very easily the coverage of a list of terms to their variants without the use of specific complex meta-rules specifically developed for the terminological purpose as in (Jacquemin and Tzoukermann, 1999).

3.2. Term variance

Term variants are very important in terminology analysis and extraction. Variance is can be caused by inflection, morphological derivation, adjectival modification, optional or variable preposition, ellipsis, coordination or the use of copulative or support verbs. Experiments have shown that usually approximately 30% of terms occur as term variants of basic terms (Jacquemin and Tzoukermann, 1999). In our model a basic term template and its variants are gathered in a tree family, i.e. the possible variance is a linguistic knowledge encoded in a family. The lexical information are then removed from these trees templates and their correspondances in term of *morphological root* are directly annotated as shown in figure 4. Classically candidate terms are validated by an occurence in a corpus.

3.3. Finite-state Compilation

Using a classical LTAG parser would be too expensive and too powerful since the identification of a term is limited to the lexical anchoring of a LTAG tree taking into account possible variances. Consequently we compile the LTAG model into another structures more relevant for computational processing. Finite-State Transducers-based processing is particularly well suited for processing large scale and lexicalized grammars.



```
<morph lex ="engine">
    <lemmaref cat="N" name="*engine*" />
</morph>

<morph lex ="operation">
    <lemmaref cat="N" name="*operation*" />
</morph>
```

```
<lemma cat="N" name="*operation*">
    <anchor tree-id="N-N" />
        <coanchor node_id="N_1">
            <lemma cat="N" name="*engine*" />
        </coanchor>
    </anchor>
</lemma>
```

Figure 5: A term represented as an elementary tree schema encoded in TAGML and its compilation in Layered FST (without the morpho-syntactic features and the morphology root for more readability).

All the transducers used in this work are Layered Finite State Tranducers (LFST). LFST have been described in (Adant, 2000). LFST extends usual FST by constraining each sequence of transition to be divided in layers. The alphabets associated to these layers can be different. Traditionally the character /? is the default symbol, and ϵ the symbole for empty string. LFST allow the combination of different levels of information while keeping an important sharing of states. Figure 5 gives an exemple of two morphosyntatic transducers and a syntactic transducer compiled from a LTAG grammar initially in the TAGML format. The convertion algorithm extends certain transition by possible categories which can be introduced by modifiers.

All the resulting transducers are then combined into a morphosyntactic transducer and a *terminological* transducer which are both determinized and minimized as possible thanks to standard FST algorithms (Mohri, 1997). The lexicalization step consists of representing the text as an identity transducer that is combined first with the *morpho-syntactic* transducer and then with the *terminological* transducer, resulting in a transducer where all possible terms have been identified.

3.4. Adding Semantic Constraints

A basic assumption concerning the use of semantic knowledge in NLP applications is that it improves the customization of the final results. On the other hand, the amount of ambiguities the application have to deal with grows up and perturb the result interpretation.

The idea is to add semantic class categories in the node label of LTAG trees similarly as presented in (Lopez and Roussel, 2000). The semantic consistency principle is exploited in order to localize the semantic constraints of the predicate represented by the term and the tree. When compiled into a LFST, the semantic category introduce a new layer as shown in figure 6.

Practically the semantic class are provided by the Tropes semantic tagger based on a training corpus.

4. The TERESA System

4.1. Bootstrapping the system resources

The resources for a given terminological domain are obtain thanks to two training corpora. The first one validates term variants allowed given a list of terms as explained in section 3.2. The second one is used to obtain a list of relevant semantic class thanks to the Tropes semantic tagger. For instance, in the application presented in

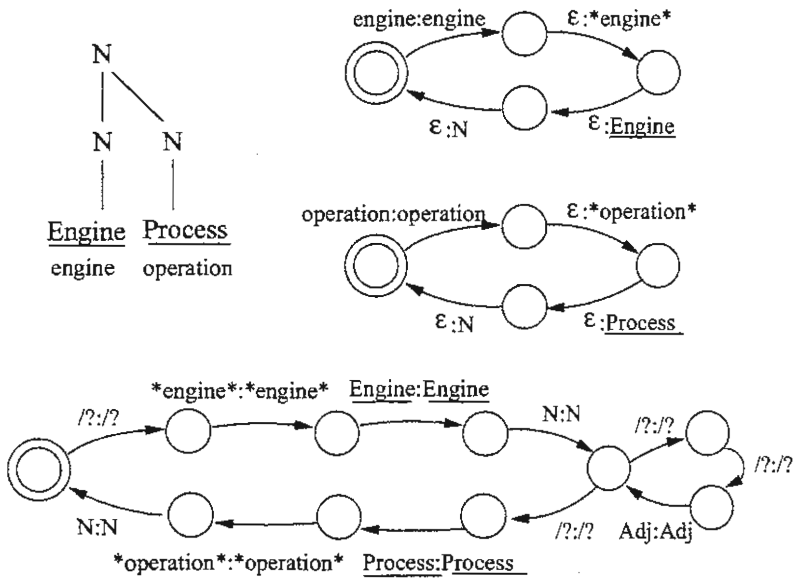


Figure 6: Semantic information integrated in elementary tree schema and in its compiled LFST.

section 2, the semantic tagging is based on 125 collocations that have extracted from a short extract of 350 caution documents. The semantic tag must match the semantic categories given by the term hypothesis. If the semantic tag differs, the corresponding term hypothesis is pruned.

4.2. Term analysis with TERESA

The TERESA system (TERminological Extraction and Statistical Analysis) allows us to analyse or extract terms in textual or semi-structured documents. Textual data is first tokenized and the morphology is processed thanks to the combination of the input string represented as a FST and a morpho-syntactic FST. The result is a FST that encoded all possible lexical analysis of the text.

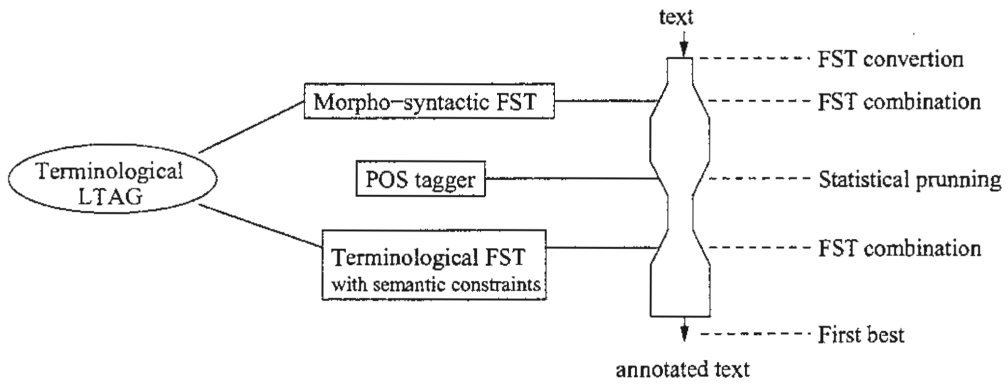


Figure 7: Overview of the TERESA system.

We apply then on this FST a POS tagger specially developed for this purpose. While the vast majority of POS tagger are limited to a linear tagging of text and a fully disambiguated tag assignation. The tagger used for this experiment is able to process efficiently word graphs coming from an Automatic Speech Recognizer for instance, and to give, if necessary, the list of ambiguous tags with their corresponding probabilities. This tagger is based on a classical trigram model with a viterbi search algorithm, it uses the linear interpolation algorithm for sparse data, implements a suffix based statistical models for unknown words. Classically a beam is used to speed up significantly the viterbi search with a negligible impact on the accuracy result. This pure statistical process is combined with a deterministic step based on the application of negative rules. This rules are compiled into a FST that is combined to the input represented as a FST, preliminarly to the statistical process, similarly to (Tzoukermann and Radev, 1999). The ability to deliver ambiguous results is very important since we know that accuracy of POS taggers is limited.

The terminological LTAG model that encodes semantic class category constraints is then combined to the resulting ambiguous tagged LFST. After this step all possible terms are identified according to the morphosyntactic, the POS tagging and a semantic match.

Finally only the best path of the final structure is considered thanks to a classical Dijkstra shortest path algorithm implementation.

5. Conclusion

We have presented a LTAG-based terminological system able to identify very precisely a given list of terms in text. The same LTAG-based terminology can be specialized to spoken application and can exploit other relevant parsing techniques. This specialization illustrates the benefit of using a linguistically motivated formalism as a generic resource. The interest of LTAG for our indexing application is the ability to exploit semantic knowledge in this process thanks to the precise semantic interface and the use of a semantic tagger.

This work fit into a serie of experiments using LTAG formalism in applications in order to :

- manage grammars because it's easier to control and design one lexicalized grammar than several small grammars
- design of a robust LTAG parser that cope with the analysis of a graph of speech recognition hypothesis.
- to detect certain ambiguities in the procedures and prevent misunderstandings.

A major feature is the possible integration with existing NLP tool thanks to the XML framework adopted.

References

Arnaud Adant. 2000. Study and Implementation of a weighted finite-state library - application to speech synthesis. M.sc., Faculté Polytechnique de Mons.
Marie-Hélène Candito. 1996. A principle-based hierarchical representation of LTAGs. In *COLING'96*, Copenhagen, Denmark.
R. Gbiglione, A. Landré, M. Bromberg, and P. Molette. 1998. *L'analyse automatique des contenus*. Dunod, Paris.
C. Jacquemin and E. Tzoukermann. 1999. NLP for Term Variant Extraction: A Synergy of Morphology, Lexicon and Syntax. In T. Strzalkowski, editor, *Natural Language Information Retrieval*. Kluwer, Boston, MA.
Patrice Lopez and David Roussel. 2000. Predicative LTAG grammars for Term Analysis. In *TAG+5*, Paris, France.
Patrice Lopez. 2000. LTAG Workbench: A General Framework for LTAG. In *TAG+5*, Paris, France.
Mehryar Mohri. 1997. Finite-State Transducers in Language and Speech Processing. *Computational Linguistics*, 23:269–312.
Evelyne Tzoukermann and Dragomir Radev. 1999. Use of weighted finite state transducers in part of speech tagging. In Andras Kornai, editor, *Extended Finite State Models of Language*. Cambridge University Press.