Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT®
Theoretical Computer Science 313 (2004) 93 – 104

Theoretical Computer Science

www.elsevier.com/locate/tcs

Finite-state transducer cascades to extract named entities in texts

N. Friburger*, D. Maurel

Laboratoire d'Informatique de Tours, 64 Avenue Jean Portalis, Tours 37000, France

Abstract

A lot of Named Entity Extraction Systems were created in English thanks to the impulse of MUC conferences. This article describes a Finite-State Transducer Cascade for the extraction of named entities in French journalistic texts. Finite-State Cascades are widely used for Natural Language Processing: a cascade is a series of finite-state transducers applied to a text transforming it. Such transducer cascades allow implementation of syntactic analysis, translation memory and information extraction. We present our general system named *CasSys*: this system uses the INTEX natural language processing features to realize a transducer cascade. CasSys is not dedicated to the extraction of named entity; we use it for this task but thanks to Intex, it allows syntactic analyses, information extraction or other tasks.

Keywords: Finite-State Transducer; Finite-State Cascade; Named entity; Proper names; Pattern matching; MUC

1. Introduction

Named entities (and then proper names) have been widely studied in the field of information extraction. They have been studied in numerous works, from the Frump system [8] to the American programs Tipster[1] and MUC.[2] We think that they can also play a role in systems of Information Retrieval. The quantity of proper names (about 10% of a newspaper according to Coates-Stephens [6]) and their

* Corresponding author.
E-mail address: friburger@univ-tours.fr (N. Friburger).

[1] http://www.tipster.org

[2] http://www.muc.saic.com/

0304-3975/$ - see front matter © 2003 Published by Elsevier B.V.
doi:10.1016/j.tcs.2003.10.007

informative quality in journalistic-style texts make them relevant to an Information Retrieval use.

Finite-State Automata, and particularly transducers, are more and more used in natural language processing [23]. In this paper, we present the CasSys system. This system allows creating a cascade of transducers; we use it to extract named entities in French journalistic-style texts.

First of all, we present a brief overview of the named entity task at MUC conferences. Then, we present what is a Finite state transducer (FST) cascade and especially its different uses in natural language processing. Finally, we describe our work on persons' names through a linguistic analysis of texts to create the best cascade as possible. We give the results of the extraction of proper names on a 165,000-word text from the French newspaper *Le Monde*, and we discuss the main difficulties to be solved.

2. A brief overview of the named entities extraction task

The Named Entity Task is a particular task of MUC: this task aims at detecting and categorizing named entity (such as proper names) in texts. The Named Entity Task is classically defined for Information Extraction use.

Before MUC conferences, researches based on named entity extraction already existed in the English language but were not sufficient. For example, Coates-Stephens [6] describes his system named *Funes* along with a lot of syntactic and semantic rules describing and structuring the contexts of proper names. Many experimentations have been conducted since this work but it remains the most complete on a natural language point of view in the English language.

MUC structures the Named Entity Task and distinguishes three types of named entity: ENAMEX, TIMEX and NUMEX [5]. TIMEX contains time expressions and NUMEX contains numbers and percentages. We are only interested in the ENAMEX entities composed of proper names and acronyms categorized as follows:

- *Organization*: named corporate, governmental or other organizational entity such as *Boston Chicken Corp.* or *National Security Agency*,
- *Person*: named person or family, for example *Bill Clinton* or the Kennedy,
- *Location*: name of politically or geographically defined location (cities, provinces, countries, international regions, bodies of water, mountains, etc.) such as *Silicon Valley* or *Germany*.

The Named Entity Task has appeared with MUC-6 in 1995 [13]; it has obtained very good results from the first time it was described. Most of the systems get a 90% score in recall and precision, and the best score at MUC-7 is obtained by the LTG system [19]. In fact, results are very close to those of a human expert (97% of recall).

The Named Entities Extraction Systems use one of those three main ways to extract proper names:

- description of rules using lists of words, trigger words and linguistic clues,
- learning and
- a combination of learning and rules.

A description of the rules is the most frequent way used by the systems to locate and extract proper names.

3. Finite-state transducer cascades in natural language processing

3.1. The principles

Automata, and particularly, transducers are very much used in the automatic treatment of the languages. Roche and Schabes [22] said that the modelling by FST is necessary; it handles more powerful formalisms, such as context-free grammars. FSTs are the best formalisms to represent complex and precise linguistic phenomena; automata are fine for linguists, they are user-friendly and readable.

The linguistic description is easier to do than with regular expressions. The grammars are under a compact shape; the parts of grammars that are similar are represented only once. One can factorize, determinize and minimize transducers to generate more effective transducers [20].

A transducer is an automaton with an input and output alphabet. The input alphabet describes a pattern to be recognized, whereas the output alphabet transforms the input text.

Every transducer T_i is in fact a local grammar. Abney [2] defines a cascade of transducers as a sequence of strata. The cascade is based on a simple idea; apply transducers on the text in a precise order to transform the text or extract patterns from the text. A transducer does not allows to cover complete linguistic phenomena but every transducer participates in the coverage of a part of the linguistic phenomena studied. The recognition of simple patterns reduces the research space.

Then a cascade is robust because the decisions are taken locally. Looking for the easiest things first to find gives a high precision to the system.

Applications developing cascades of transducers take advantage of these three qualities; robustness, precision and speed brought by transducers.

A cascade is described as follows: the transducer T_i parses the text L_{i-1} and produces the text L_i, then a transducer T_{i+1} will transform the text L_i into L_{i+1} and so on. Transducers parse the text in a precise order to first track down the most certain patterns which Abney suggests naming "*islands of certainties*". The uncertain patterns are found next. Every transducer uses the results of the previous ones.

The system of cascade often uses the heuristic called *longest pattern matching*; between two patterns located by a transducer, one chooses the longest pattern found. In fact, this idea uses the fact that if the simple words are strongly ambiguous, a complex sequence of words is less ambiguous.

This heuristic is good except when several paths of the same length are possible in the graph and in the text; one of them will be chosen without control on the choice.

Numerous systems of cascades were developed, proving their major interest for the automatic treatment of the languages. We present here a state of the art of the various existing systems that are used in tasks of syntactic analysis, extraction of information and translation.

3.2. Syntactic analysis

Syntactic parsing requires rules describing the syntactical phenomena: transducers allow an effective description of these rules. Abney [2] created the Cass system (Cascaded Analysis of Syntactic Structure) that realizes the syntactic analysis of texts in English and in German. Resuming his ideas, a syntactic analyzer for Swedish was also created by Kokkinakis and Johansson-Kokkinakis [16]. Abney describes a philosophy of the syntactic analysis: one does the simplest and most sure things first. He first tracks down a syntactic basic group (called *chunk*) bounded by stopwords. The following work is based on these basic groups. The patterns are described by regular expressions and translated into a finite state automaton. All the automata of the same stratum are collected in only one deterministic automaton. Several strata of automata parse the text to obtain the syntactic analysis.

Another syntactic analyzer was developed by Xerox [3]; the IFSP system (incremental finite state parser) allows to realize syntactical analysis. The syntactic analysis of a text borrows two approaches that IFSP uses:

- The constructivist approach based on constraints added during the parsing [1,12].
- The reductionist approach allows to eliminate already supplied partial analysis [4].

IFSP permits to use a sequence of transducers on a text by using operators of replacement. Every transducer allows to add syntactical information. For input, the system uses a labelled text. The analyzer permits to refine the first decisions and does not systematically use the heuristics of the longest pattern matching. Gala-Pavia [11] uses the IFSP system with the aim of realizing a syntactic analysis for Spanish.

3.3. Information extraction

The FASTUS system (Finite State Automata-based Text Understanding) is a very famous system of Information Extraction in English and Japanese, created by Hobbs et al. [14]; it is developed since 1992 and sponsored by DARPA. This system analyses texts in wider and wider sequences to extract relevant data. It begins by tracking down the complex words; compound names, places, dates and proper names. Then FASTUS recognizes the simple nominal and verbal groups, and after, the complex nominal and verbal groups. The patterns previously found are used to discover all the events and the relations that connect them (coreference and inference) to fill templates (the templates answer questions such as *what to whom*? *When*? *And where*?). This system was presented to the MUC-6 evaluations, and obtained scores of 92% of recall and 96% of precision for the named entity task.

3.4. Translation memory

Translation memory is an original way to use Finite State Cascades. The systems with translation memory are translators based on examples. Vogel and Ney [25] proposes a system with a translation memory based on a transducer cascade for German and English. It works on a bilingual corpus in which every translation memory contains a sentence in input and its translation in output. Several translations are authorized for the same sentence with a certain score. To translate a text, one must parse the texts with all the translation transducers.

3.5. CasSys: our own FST Cascade System

We have designed **CasSys**, our tool to create transducer cascades. We use the FST Toolbox of Intex system [24] to implement the system.[3] The Intex system allows us to create a system of cascade having the capacity to manage both constructivist and reductionist approaches. CasSys is independent of the use which one wants to do, syntactic analysis, information extraction or other tasks can be performed thanks to the possibilities of the Intex natural language processor. The output of the transducers can be added to the pattern found in the text, or the recognized patterns can be extracted from the texts and replaced by a label. The labels of already found patterns can be used in the following transducers.

We present here the use of CasSys for named entities extraction. Our tool consists in two stages to extract proper names, consisting in two FST Cascade performed by CasSys on each texts. The first cascade consists in a list of rules describing the local grammar of proper names, using linguistic clues (for example, contexts and dictionaries of first names, place names, occupation nouns to extract person names). The input alphabet of the transducers describes patterns of proper names; the output alphabet contains information on the type of the proper name described in the input.

The second cascade uses proper names already found in the text by the first cascade; it allows finding the remaining proper names with the proper names found by the first stage. The drawback of such a second stage is that the errors made in the first stage remain in the second; so we try to build a cascade as precise as possible.

4. Named Entity Extraction: the Example of Person Names

Newspapers contain a lot of proper names that provide important information on the contents of texts. Most of the names are unknown [17] and cannot be stored in dictionaries because of their quantity and because proper names belong to opened classes of words. Mikheev et al. [19] studied the use and the impact of gazetteers and showed that locations need a dictionary to be extracted but persons and organizations can be found without such sources. Coates-Stephens [6] studied proper names in English; he conducted a syntactic and semantic study on the appearance of names apposition, compound proper names, etc. and he created the Funes system to extract them using the rules he described. Numerous works use rules to describe proper names occurrences.

We noticed that the three main types of proper names are not equal in terms of occurrence, context of appearance and amount in newspaper articles. Person names represent 39.8% of proper names in our texts, whereas Organization represents 16.3%,

[3] It is important to precise that Intex allows to treat any language, and consequently our CasSys system too.

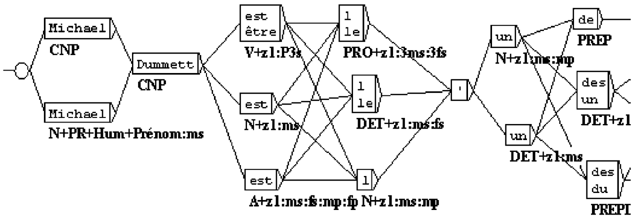

Fig. 1. A tagged sentence with Intex System.

which is less than persons and locations. Location names represent 43.9% of proper names. Moreover, the various kinds of proper names cannot be extracted in a same manner: discovering location names needs a dictionary, person names and organization names are found, thanks to their lexical contexts.

4.1. Finite-state pre-processing

The Intex system [24] permits to pre-process texts. It allows capitalizing on transducers on texts for the whole processing. Firstly we pre-process texts, cutting them in sentences with a transducer that adds the symbol {S} at the end of the sentence. Then we tag the text from a morpho-syntactic point of view; we use dictionaries that link words with information: lemmas, grammatical categories (noun, verb, etc.) and semantic features (concrete, place names, first names, abbreviations, etc.).

The advantages of these dictionaries are:

- Every word is given with its lemmatized form, which avoids describing all the flexions of a word in the transducers that discover them.
- The used dictionaries contain syntactical information that can help locating patterns for proper names.

Each word is tagged with all its occurrences in dictionaries. Fig. 1 shows the transducer for the beginning of the sentence "*Michael Dummett est l'un des plus grands philosophes britanniques d'aujourd'hui*" ("*Michael Dummett is one of the most famous contemporary British philosophers*"). This sentence has been tagged with Intex using the dictionaries provided by Intex and our own dictionaries:[4] the inputs are in boxes (the second line being the lemma of the word), the outputs are in bold face and contain syntactic information (N = noun, V = Verb, etc.) and semantic information (Hum = Human).

After this pre-processing, we use our program CasSys to extract the proper names.

[4] Intex provides Delaf dictionaries of simple words and their inflected forms [7], and we use our own Prolintex dictionary of place-names realized within the Prolex project [21], Prenom-prolex dictionary of first names (more than 6500 entries), acronym-prolex dictionary of abbreviations with their extensions (about 3300 entries) and finally occupation names dictionary [9].

4.2. A linguistic study of person-names

Before creating the cascade, we have studied the different internal and external evidences of person names in newspaper articles. Indeed the contexts help in tracking down proper names.

McDonald [18] describes the notions of internal and external evidence. In fact, most proper names extraction tools naturally use these evidences without naming them so. The internal evidence is inside the proper name itself: this is a word belonging to the name and helping in locating proper names and categorizing them if possible.

Internal evidences examples (in bold face) are as follows:

- *Microsoft* **Inc.**
- *Wall Street* **Journal**
- **George W.** *Bush*

The external evidence is the context of appearance of the proper names in the sentence. In his discourse, especially in a journalistic one, the author gives information indicating the type of the proper name. The external evidence is on the right or on the left context of a proper name in the sentence.

External evidences examples are as follows:

- **the city of** *Paris*
- **judge** *Van Ruymbeke*
- *Blair*, **the British Prime Minister**

The combination of internal and external evidence are possible: **French president Jacques** *Chirac* (where *French president* is an external evidence and *Jacques* an internal evidence). In the following work, we talk about left context and right context for the external evidence.

First of all, we noticed that the left context allows discovering more than 90% of person names in journalistic texts: this is certainly due to stylistic imperatives appropriate to this type of texts that should be objective and should describe facts. A study of an extract from *Le Monde* newspaper (about 165,000 words) allowed us to determine the categories of the most frequent contexts. The different cases encountered are described below:

Case 1: 25.9% of person names are preceded by a context containing a title or an occupation name, followed by an internal evidence, the first name, and by the patronymic name. Ex: **M. Alain** *Juppé*, *le* **président John** *Kennedy*.

Case 2: 19.1% of person names preceded by a context containing a title or an occupation name are followed by a simple patronymic name, or by an unknown first name (that is not in our dictionary of first names) and a patronymic name. Ex: *le* **président** *Chadli*.

Case 3: 43.4% of person names have no describable contexts but have a first name (known thanks to our dictionary) and followed by the name of the person. Ex: **Pierre** *Bourdieu*.

Case 4: 5.2% of forms are located thanks to a verb referring only to human actions (*to say*, *to explain*, *etc.*). For example, "*Wieviorka est décédé le 28 décembre*" (*Wieviorka died on December 28*) or "*Jelev a dit …*" (*Jelev said…*). Here we counted appositions too, such as in "*Jospin*, *premier Ministre …*" (*Jospin*, *Prime Minister…*).

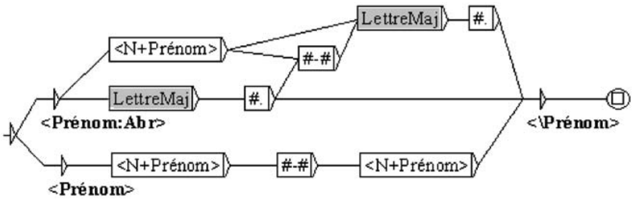

Fig. 2. A transducer describing compound first names (ex: *John Fitzgerald*) or abbreviated first names (ex: *J.F.*).

Case 5: The remaining 6.4% of person names have no context whatsoever that can distinguish them from other proper names. However we noticed that 49% of the remaining person names can yet be detected. Indeed, person names without contexts are mostly well-known persons for whom the author considers it unnecessary to specify the first name, the title or the profession. It is necessary to realize a second analysis to find the patronymic name, that as one has already been discovered somewhere else in the text, that reduces to 3.3% the number of undetectable forms. This percentage can further be reduced by a dictionary of celebrity names. For example, "*Brandissant un portrait de Lénine*, *ou de Staline*, ..." (*Brandishing a portrait of Lenin*, *or Stalin*, ...).

4.3. The different person names forms

We also studied the different forms of person names. First names followed by a patronymic name or patronymic names alone are most often found. As noticed by Kim and Evens [15], the author of a newspaper article generally first gives the complete form of the person name, then abbreviated forms; this is why the majority of person names are found with their first name and their last name.

We have described all first name (*N+Prenom*) forms in transducers (using dictionary tags of the text and morphologic clues). Fig. 2 is one of the numerous transducers describing person names. [5]

First names unknown to the dictionary are not tagged as first names, but they are included as an integral part of the person name as in *< person > < ctxt:titmilit > général < \ctxt > < nom > Blagoje Adzic < \nom > < \Person >* (the person name is *Blagoje Adzic* but we have not distinguished the first name from the patronymic name).

Different patronymic forms are also described using morphology (word beginning with an upper case).

At last, most contexts are left contexts that simply are civilities (ex: *Monsieur*, *Madame*, *etc.*), or titles: politics (ex: *ministre*, *président*, *etc.*), nobility titles (ex: *roi* (*king*), *duchesse*, *baron*, *etc.*), military titles (ex: *général*, *lieutenant*, *etc.*), religious titles (ex: *cardinal*, *Père*, *etc.*), administration staff (ex: *inspecteur*, *agent*, *etc.*)

[5] LettreMaj is an automaton listing upper case letters.

as well as occupation names (ex: *juge*, *architecte*, *etc.*). The occupation names are the least frequent terms in contexts. The place-name dictionary allows tracking down the adjectives of nationalities in expressions such as "*le président américain Clinton*", "*l'allemand Helmut Kohl*" (*the German Helmut Kohl*).

4.4. Description of the finite-state cascade to extract person names

According to our various observations on the study of person names and their contexts [10], we have defined the cascade and given priority to the longest patterns to track down the whole names.

For example, if we apply a transducer that recognizes "*Monsieur*" followed by a word beginning with an upper case letter before the transducer that recognizes "*Monsieur*" followed by a first name (*<prenom>*) then by a name (*<nom>*), in a text containing the sequence "*Monsieur Jean Dupont*", we discover the pattern:

<person> <ctxt:civ>Monsieur<\ctxt> <nom>Jean<\nom> <\person>

instead of the pattern

<person><ctxt>Monsieur<\ctxt><prenom>Jean<\prenom><nom>Dupont <\nom> <\person>

The first result of parsing is an error, while the second parse is the good one.

Then we have designed about thirty transducers to obtain the best results. The longest patterns are in the first transducers to be applied.

The following section presents an example of the results obtained by our cascade on a sample of texts and presents an evaluation of the extraction system.

4.5. Evaluation

Here is an example of the results obtained on an article from *Le Monde*. An extract of the original text reads:

Le **président haïtien Aristide** *accepte la candidature de* **M. Théodore** *au poste de premier ministre* (...) *Avant leur départ pour Caracas, les présidents du Sénat et de la Chambre des députés*, **M. Déjean Bélizaire** *et* **M. Duly Brutus**, *avaient obtenu du "président provisoire" installé par les militaires*, **M. Joseph Nérette**, *l'assurance qu'il démissionnerait si les négociations débouchaient sur la nomination d'un nouveau premier ministre.*{S} (...) *Pendant la campagne*, **M. Théodore** *avait concentré ses attaques contre le* **Père Aristide**, *et n'avait cessé de le critiquer après sa triomphale élection.*{S}

We finally obtained those extracted patterns:

<person><ctxt:titpolit> **président** *<\ctxt><ctxt:nation>* **haïtien** *<\ctxt> <nom>* **Aristide** *<\nom><\person>*

<person><ctxt:civ> **M.** *<\ctxt><nom>* **Duly Brutus** *<\nom><\person>*

Table 1
Results obtained on an extract of *Le Monde* (%)

	Case 1	Case 2	Case 3	Case 4	Case 5	Total
Recall	95.7%	99.4%	96.6%	60%	48.7%	91.9%
Precision	98.7%	99.5%	99.2%	94.9%	99.3%	98.9%

< person >< ctxt:civ > **M.** *< \ctxt >< nom >* **Déjean Bélizaire** *< \nom >< \person >*

< person >< ctxt:civ > **M.** *< \ctxt >< prenom >* **Joseph** *< \prenom >< nom >* **Nérette** *< \nom >< \person >*

< person >< ctxt:civ > **M.** *< \ctxt >< nom >* **Théodore** *< \nom >< \person >*

< person >< ctxt:titeglise > **Père** *< \ctxt >< nom >* **Aristide** *< \nom >< \person >*

To evaluate our Finite State cascade, we have verified the results on a part (about 80,000 words) of our corpus of the newspaper *Le Monde*[6] (Table 1). We used the recall and precision measures.

$$Recall = \frac{number\ of\ person\ names\ correctly\ found\ by\ the\ system}{number\ of\ person\ names\ correct\ and\ incorrect\ found\ by\ the\ system},$$

$$Precision = \frac{number\ of\ names\ correctly\ found\ by\ the\ system}{number\ of\ person\ names\ really\ present\ in\ the\ text}.$$

The results obtained in the first four categories of patterns for person names are very good. We obtained more than 96.9% of recall and more than 99.1% of precision on the person names preceded by a context and/or by a first name.

We notice that in cases 4 and 5 the results are bad. In case 4, the patterns that surround the person names are very ambiguous, as in the example: "*Microsoft declared*": the verb *to declare* can be associated with a human being but also with a company. Cases 4 and 5 can be improved during the search of the other names.

5. Conclusion

CasSys is our FST cascade system, built with the Intex toolbox. This tool permits to generate cascades for syntactic analysis or information extraction uses. The principle of the FST cascade is rather simple and effective; transducers are applied one after the other to the texts to locate pattern, to transform the text etc. The transducer cascades are very often used in the field of natural language processing.

We have described a cascade to extract person names in texts, but also location and organization names; our system is actually the most complete on French texts and obtains the best results. The description of the patterns to find person names turns out to be boring if one wants to obtain the best possible result. Combinations and possible

[6] These resources are available at Elda (www.elda.fr).

interactions in the cascade are complex. The results are promising on *Le Monde*: it is a newspaper of international readership with journalists who respect classic standards and have a concern for precision and details (especially when quoting people and proper names). The results will be worse with other newspapers mainly because of the authors' approximate style.

The other proper names (place names, names of organizations, etc.) are more difficult to track down because they have few contexts or internal evidences, and when they have a context, it is much more varied.

References

[1] S. Abney, Parsing by chunks, in: R.C. Berwick, S.P. Abney, C. Tenny (Eds.), Principle-Based Parsing: Computation and Psycholinguistics, Kluwer Academic Publishers, Boston, 1991, pp. 257–278.

[2] S. Abney, Partial parsing via finite-state cascades, in: Workshop on Robust Parsing, 8th European Summer School in Logic, Language and Information, Prague, Czech Republic, 1996, pp. 8–15.

[3] S. Ait-Mokhtar, J. Chanod, Incremental finite state parsing, in ANLP'97, Washington D.C., USA, 1997.

[4] T. Chanod, A robust finite-state parser for French, in: Proc. Workshop on Robust Parsing, Prague, Czech, 1996, pp. 16–25.

[5] N. Chinchor, Muc-7 named entity task definition, in http://www.itl.nist.gov/iaui/894.02/related_projects/muc/proceedings/muc_7_toc.html#appendices, 1997.

[6] S. Coates-Stephens, The analysis and acquisition of Proper names for the understanding of free text, in: Computers and the Humanities, Vol. 26, Nos. 5–6, Kluwer, Hingham, 1993, pp. 441–456.

[7] B. Courtois, M. Silberztein, Dictionnaire électronique des mots simples du franais, Paris, Larousse, 1990.

[8] G. Dejong, An overview of the frump system, in: W.B. Lehnert, M.H. Ringle (Eds.), Strategies for Natural Language Processing, Lawrence Erlbaum Associates, Hillsdale, NJ, 1982, pp. 149–176.

[9] C. Fairon, Structures non-connexes. Grammaire des incises en français: description linguistique et outils informatiques, Thèse de doctorat en informatique, Université Paris 7, 2000.

[10] N. Friburger, D. Maurel, Elaboration d'une cascade de transducteurs pour l'extraction de motifs: l'exemple des noms de personnes, in: Actes de TALN 2001: 8eme confrence annuelle sur le traitement informatique des langues naturelles, Tours, France, 2001, pp. 183–192.

[11] N. Gala-Pavia, Using the incremental finite-state architecture to create a Spanish shallow parser, in: Proc. XV Cong. of SEPLN, Lleida, Spain, 1999.

[12] G. Grefenstette, Light parsing as finite state filtering, in: Workshop on Extended Finite State Models of Language (ECAI'96), Budapest, Hungary, August 11—12, 1996.

[13] R. Grishman, B. Sundheim, Message Understanding Conference—6: a brief history, in: Proc. The 6th Message Understanding Conf., Morgan Kaufmann, Los Altos, CA, 1996.

[14] J.R. Hobbs, D.E. Appelt, J. Bear, D. Israel, M. Kameyama, M. Stickel, M. Tyson, FASTUS: a cascaded finite-state transducer for extracting information from natural-language text, in: Finite-State Devices for Natural Language Processing, MIT Press, Cambridge, MA, 1996.

[15] J.S. Kim, M.W. Evens, Efficient coreference resolution for proper names in the Wall Street Journal Text, in: Online Proc. of MAICS'96, Bloomington, 1996.

[16] D. Kokkinakis, S. Johansson-Kokkinakis, A cascaded finite-state parser for syntactic analysis of Swedish, in: Proc. 9th EACL, Bergen, Norway, 1999.

[17] I. Mani, R.T. MacMillan, Identifying unknown proper names in newswire text, in: Corpus Processing for Lexical Acquisition, MIT Press, Cambridge, MA, 1996, pp. 41–59.

[18] McDonald, Internal and external evidence in the identification and semantic categorisation of proper names, in: Boguraev, Pustejavsky (Eds.), Corpus Processing for Lexical Acquisition, MIT Press, Cambridge, 1996, pp. 32–43.

[19] A. Mikheev, M. Moens, C. Grover, Named entity recognition without gazetteers, in: EACL'99 Ed., Bergen, Norway, 1999, pp. 1–8.

[20] M. Mohri, Minimization of sequential transducers, in: Lecture Notes in Computer Series, Vol. 807, Springer, Berlin, 1994.

[21] O. Piton, D. Maurel, Le traitement informatique de la géographie politique internationale, in: Colloque Franche-Comté Traitement automatique des langues (FRACTAL 97), Besançon, 10–12 décembre, Bulag, numéro spécial, 1997, pp. 321–328.

[22] E. Roche, Y. Schabes, Deterministic part-of-speech tagging with finite state transducers, in: Computational Linguistics, Vol. 21, Mitsubishi Electric Research Laboratories, Cambridge Research Center, 1995, pp. 227–253.

[23] E. Roche, Y. Schabes, Finite-State Language Processing, MIT Press, Cambridge, MA, 1997.

[24] M. Silberztein, INTEX: an FST toolbox, Theoret. Comput. Sci. 234 (2000) 33–46.

[25] S. Vogel, H. Ney, Translation with cascaded finite-state transducers, in: Proc. ACL Conf. (Association for Computer Linguistics), Hongkong, 2000, pp. 23–30.