Finite-State Transducer Cascade to Extract Proper Names in Texts

Nathalie Friburger and Denis Maurel

Laboratoire d'Informatique de Tours
E3i, 64 avenue Jean Portalis, 37000 Tours
{friburger,maurel}@univ-tours.fr

Abstract. This article describes a finite-state cascade for the extraction of person names in texts in French. We extract these proper names in order to categorize and to cluster texts with them. After a finite-state pre-processing (division of the text in sentences, tagging with dictionaries, etc.), a series of finite-state transducers is applied one after the other to the text and locates left and right contexts that indicates the presence of a person name. An evaluation of the results of this extraction is presented.

1 Motivation

Finite-State Automata and particularly transducers are more and more used in natural languages processing [13]. In this article, we suggest the use of a finite-state transducer cascade to locate proper names in journalistic texts. In fact, we study names because of their numerous occurrences in newspapers (about 10 % of a newspaper) [3]. Proper names have already been studied in numerous works, from the Frump system [5] to the American programs Tipster[1] and MUC[2]. These two programs evaluate systems of information extraction in texts. The Named Entity Task is a particular task of MUC : this task aims to detect and categorize named entity (like proper names) in texts.

First of all, we present some known finite-state cascades used in natural language processing. Secondly we shall explain our finite-state pre-processing of texts (division of the text in sentences, tagging with dictionaries) and how to use transducers to extract patterns and categorize them. Then we shall describe our work through a linguistic analysis of texts to create the best cascade as ossible. Finally, we shall present the results of the extraction of proper names on a 165000-word text from the French newspaper *Le Monde*, and shall discuss the main difficulties and problems to be solved.

[1] www.tipster.org
[2] http://www.muc.saic.com/

B.W. Watson and D. Wood (Eds.): CIAA 2001, LNCS 2494, pp. 115–124, 2002.

2 Finite-State Transducer Cascades in Natural Language Processing

Finite-State Transducer Cascades have been developed for a few years to parse natural language. In this part, we quickly present three systems parsing with finite-state cascades. The advantages of transducers are their robustness, precision and speed.

Abney [1] presents a syntactic parser for texts in English or German language (Cass System). He describes the main principles of a cascade and defines a cascade as a "sequence of strata". The transducer T_i parses the text L_{i-1} and produces the text L_i. Abney says that reliable patterns are found first: he calls them "islands of certainty". The uncertain patterns are found next. In the same way [11] presents a finite-state cascade to parse Swedish, which is very close to Abney's one. The IFSP System (Incremental Finite State Parser [2], created at Xerox Research Center) is another system of cascade of transducers which has been used fo a syntaxic analysis of Spanish language [8].

Fastus [9] is a very famous system for information extraction from texts in English or Japanese, sponsored by DARPA: it is closer to the work we present here. This system parses texts into larger and larger phrases. It first finds compound nouns, locations, dates and proper names. Secondly it recognizes nominal or verbal groups, and particles. Thirdly complex noun phrases and complex verb phrases are found. The previous patterns are used to discover events and relations between them. This system was presented at the MUC evaluations for information extraction and it obtained good scores.

We present our own finite-state cascade, which finds proper names and their contexts in texts. We created this system in order to cluster journalistic texts.

3 Finite-State Pre-processing

We have chosen to use Intex system [14] to pre-process texts. Intex permits to capitalize on transducers on texts for the whole processing. Firstly we pre-process texts cutting them in sentences and tagging them with dictionaries. After that we use our own program, which completes Intex's possibilities and allows realizing a finite-state transducer cascade.

3.1 Sentences

Before applying the finite-state cascade to a text, we submit it to a finite-state pre-processing. Indeed, we cut the text into sentences [7]. A transducer describes possible ends of sentences and puts the mark $\{S\}$ between each sentence.

The main difficulties come from the dot which is a very ambiguous symbol when it is followed by an upper case letter: the dot can either be the final dot of a sentence or not. So we have found four types of ambiguities with the dot:

- In person names when they are preceded by an abbreviated form with the dot, as in "*M. Jean Dupont" (Mister Jean Dupont)*: the dot in *M.* is clearly not the end of the sentence.
- In person names too when they contain an abbreviated first name as in "*J. Dupont*".
- In abbreviations such as "*N.A.T.O*".
- In different abbreviated words as in "*Éd. Gallimard*" or in "*Chap. 5*" for example.

Therefore the resolution of these ambiguities induces errors to be taken into account. For example, dots after a symbol (money, physical and chemical symbols, etc.) as in "*Ce livre coûte 20 F. Le vendeur me l'a dit.*" (*This book costs 20 F. The salesman said it to me.*) or dots after a compound word as in "*Cet aliment contient de la vitamine A. Le docteur conseille d'en manger.*" (*This food contains vitamin A. The doctor advises to eat it.*) really notify the end of a sentence.

Figure 1 presents the transducer that inserts the output $\{S\}$ between each sentence. The various cases are handled respectively in sub-graphs (grey boxes) *cas2* (person names and abbreviation patterns),*cas3* (symbols and compound words) and *cas4* (for abbreviated words).

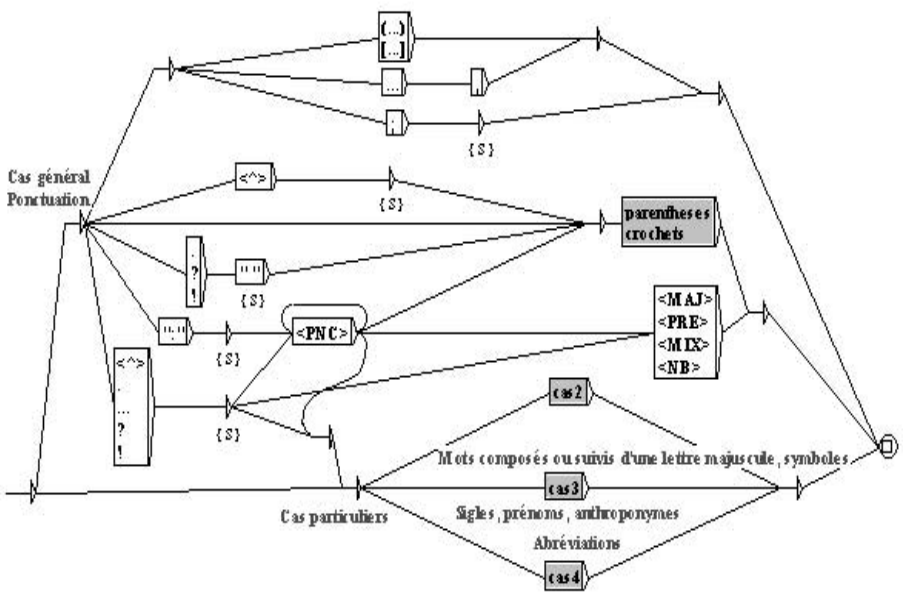

Fig. 1. Transducer describing end of sentences and ambiguous dot patterns

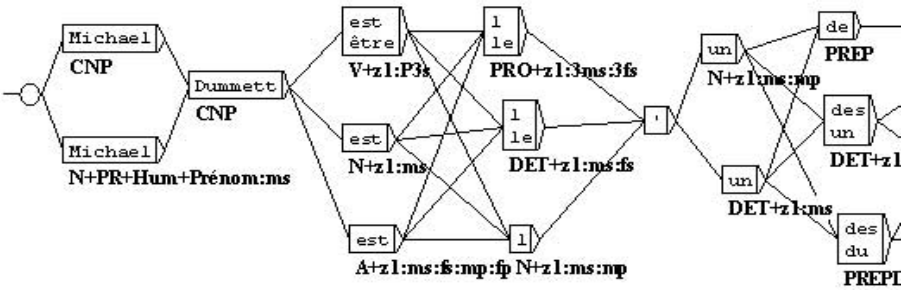

Fig. 2. A tagged sentence with Intex System

3.2 Tagging

Now we tag the text from a morpho-syntactic point of view. Thus we use dictionaries that link words with information: lemmas, grammatical categories (noun, verb, etc.) and semantic features (concrete, place-names, first names, abbreviations, etc.)[3]. The advantage to these dictionaries is double:

- Every word is given with its lemmatized form, which avoid to describe 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. Figure 2 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 is tagged with Intex and our dictionaries: 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).

4 Finite-State Transducer Cascade: The Example for Extracting Person's Names

4.1 Transducers

Transducers are finite-state machines with an input alphabet and an output alphabet: this property can be used to extract patterns and categorize them.

[3] Delaf dictionaries of simple words and their inflected forms [4], Prolintex dictionary of place-names realized within the Prolex project [12], 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 [6].

The input alphabet contains patterns to be recognized in texts whereas the output alphabet, in our case, contains in our case information marked out in a language inspired by XML. The patterns we are looking for are proper names and eventually their contexts (if we can locate and exploit them).

Here is an example of a person name found in a text and marked out by the transducer cascade:

Le juge Renaud Van Ruymbeke (the judge Renaud Van Ruymbeke) ⇒
<person><ctxt:profession> juge < \ctxt> <prenom> Renaud < \prenom> <nom> Van Ruymbeke < \nom> < \person>.

The cascade is based on a simple idea: to apply transducers on the text in a precise order to transform or extract patterns from the text. Every pattern discovered is replaced in the text by an indexed label. We eliminate the simplest patterns of the text to avoid that a later transducer extracts them as well.

4.2 A Linguistic Study of Person's Name

Before creating the cascade, we have studied right and left contexts of person names in newspaper articles. Indeed the contexts help to track down proper names. We noticed that the left context allows to discover more than 90 % of the person names in journalistic texts: this is certainly due to stylistic imperatives appropriate to this type of texts which should be objective and should describe facts. A study of an extract from *Le Monde* newspaper (about 165000 words) allowed us to determine the categories of the most frequent contexts.

- Case 1: 25.9 % of person names are preceded by a context containing a title or an occupation name, followed by the first name and by the patronymic name. Ex: *M. Alain Juppé, le président John Kennedy (president John Kenne-dy).*
- Case 2: 19.1 % of person names preceded by a context containing a title or an occupation name followed by a single patronymic, or by an unknown first name (which is not in our dictionary of first names) and finally of a patronymic name. Ex : *le président Chadli.*
- Case 3: 43.4 % of person names with no describable contexts but with a first name (known thanks to our dictionary) and followed by the name of the person. Ex : *Pierre Bourdieu.*
- Case 4: 5.2 % of the forms are located thanks to a verb refering 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...*)
- 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 persons' names can yet be detected. Indeed, person names without contexts are mainly very known persons for whom the author considers unnecessary to specify the first name, the title or the profession. It is necessary to realize a second analysis to find the patronymic name, which

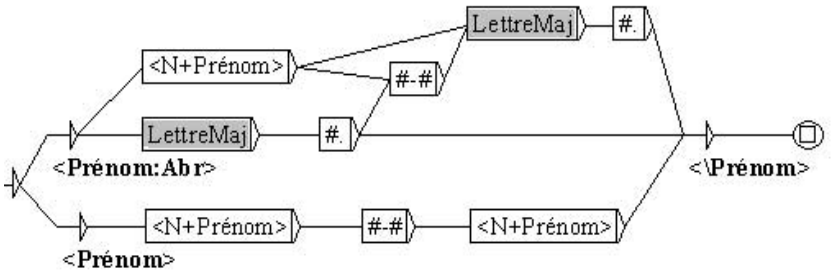

Fig. 3. A transducer describing compound (ex: *John Fitzgerald*) or abbreviated (ex: *J.F.*) first names

one has already discovered in another place of the text; which reduces to 3.3 % the number of undetectable forms. This percentage can still be reduced by a dictionary of celebrity names. Ex : " *Brandissant un portrait de Lénine, ou de Staline, ...*" (*Brandishing a portrait of Lenin, or Stalin, ...*).

4.3 Different Person Name Forms

We also studied different forms of person names. First names followed by a patronymic name or patronymic names alone are most often found. As noticed by [10], the author of a news paper generally gives first the complete form of the person name, then abbreviated forms; that is why the majority of person names are often found with their first name and their last name.

We have described all first name forms (Figure 3[4]) in transducers (using dictionary tags of the text and morphologic clues).

First names unknown to the dictionary are not tagged as first names, but they are included as an integral part of the person's 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 contexts are a majority of left contexts which are simply civilities (ex: *Monsieur, Madame, etc.*), 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.*) as well as occupation names (ex: *juge, architecte, etc.*). The occupation names are the least frequent terms in contexts. The place-name dictionary allows to track down the adjectives of nationalities in expressions

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

such as "*le président américain Clinton*", "*l'allemand Helmut Kohl*" (*the German Helmut Kohl*).

4.4 Finite-State Cascade Description

According to our various observations on the study of person names and their contexts, 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*>), and that we have 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*>

This is an error because the best parsing is the second.

Then we have designed about thirty transducers to obtain the best results. They are generally constituted of one context part (left or right), a first name part and a patronymic part. But some are only first name and patronymic part, or context and patronymic name. The longest patterns are in the first transducers to be applied.

4.5 Evaluation

Here is an example of 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 démis-sion-ne-rait si les né-go-cia-tions débou-chaient sur la no-mi-na-tion 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*>

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

<*person*> <*ctxt:civ*> **M.** <*ctxt*> <*prenom*> **Joseph** <*prenom*> <*nom*> **Nérette** < *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*> **Théodore** <*nom*><*person*>
<*person*> <*ctxt:titeglise*> **Père** < *ctxt*> <*nom*> **Aristide** < *nom*> < *person*>

To verify that the results obtained after this finite-state cascade were correct, we verified a part (about 80000 words) of our corpus of the newspaper *Le Monde*[5] (Table 1). We used the recall and precision measures.

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

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

The results obtained on 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 case 5 the results are bad. In case 4, the patterns that surround the person names are very ambiguous ex :"*Microsoft declared*": the verb *to declare* can be associated to a human being but also to a company as in the example. In case 5, the names are found because they have been found in the text in another context. For example, a text contains the following sentence : "*Ce livre contient des critiques directes au président Mitterrand*" (*This book contains direct criticisms of president Mitterrand*) where the context "*president*" permits to know that *Mitterrand* is a person name. In the same text, we have the sentence "*M. Léotard interpelle Mitterrand sur ...*" (*Mr Léotard calls to Mitterrand on ...*). Thanks to the pattern found before, we know that *Mitterrand* is a person name in this text. Cases 4 and case 5 can be improved during the search of the other names.

5 Conclusion

We present finite-state machines to pre-process a text and locate proper names. The principle of the cascade of transducers is rather simple and effective; on the other hand the description of the patterns to be found turns out to be

[5] Ressources obtained at Elda (`www.elda.fr`)

boring if one wants to obtain the best possible result. Combinations and possible interactions in the cascade are complex. The other proper names (place-names, names of organizations, etc.) are more difficult to track down because their contexts are much more varied.

The results are promising: *Le Monde* is a newspaper of international readership whose journalists 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 more approximate style of authors.

Beyond extraction of patterns, bare patterns can serve in numerous domains. One can thus imagine the creation of a system to write XML semi-automatically or to semi-automatically append names to electronic dictionaries.

References

[1] Abney, S. (1996). *Partial parsing via finite-state cascades*, In Workshop on Robust Parsing, 8th European Summer School in Logic, Language and Information, Prague, Czech Republic, pp. 8-15. 116

[2] Ait-Mokhtar, S., Chanod, J. (1997) *Incremental finite state parsing*, in ANLP'97. 116

[3] Coates-Stephens, S. (1993). The Analysis and Acquisition of Proper Names for the Understanding of Free Text, in *Computers and the Humanities*, 26 (5-6), pp. 441-456. 115

[4] Courtois, B., Silberztein, M. (1990). *Dictionnaire électronique des mots simples du français*, Paris, Larousse. 118

[5] Dejong, G. (1982). An Overview of the frump System, in *W. B. Lehnert et M. H. Ringle éd., Strategies for Natural Language Processing*, ErlBaum, pp. 149-176. 115

[6] Fairon, C. (2000). *Structures non-connexes. Grammaire des incises en français : description linguistique et outils informatiques*, Thèse de doctorat en informatique, Université Paris 7. 118

[7] Friburger, N., Dister, A., Maurel, D. (2000). *Améliorer le découpage des phrases sous INTEX*, in Actes des journées Intex 2000, RISSH, Lièges, Belgique, to appear. 116

[8] Gala-Pavia, N. (1999). *Using the Incremental Finite-State Architecture to create a Spanish Shallow Parser*, in Proceedings of XV Congres of SEPLN, Lleida, Spain. 116

[9] Hobbs, J. R., Appelt, D. E., Bear, J., Israel, D., Kameyama, M., Stickel, M., Tyson, M. (1996). 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. 116

[10] Kim, J. S., Evens, M. W. (1996). *Efficient Coreference Resolution for Proper Names in the Wall Street Journal Text*, in online proceedings of MAICS'96, Bloomington. 120

[11] Kokkinakis, D. and Johansson-Kokkinakis, S. (1999). *A Cascaded Finite-State Parser for Syntactic Analysis of Swedish*. In Proceedings of the 9th EACL. Bergen, Norway. 116

[12] Piton, O., Maurel, D. (1997). 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, pp. 321-328. 118

[13] Roche, E., Schabes, Y. (1997). *Finite-State Language Processing*, Cambridge, Massachussets, MIT Press. 115

[14] Silberztein, M. (1998). *"INTEX: a Finite-State Transducer toolbox"*, in Proceedings of the 2nd International Workshop on Implementing Automata (WIA'97), Springer Verlag. 116