A Fast, Memory Efficient, Scalable and Multilingual Dictionary Retriever

P. Fernandes, L. Lopes, C. A. Prolo, A. Sales, R. Vieira

Pontifícia Universidade Católica do Rio Grande do Sul
Avenida Ipiranga, 6681 – 90619-900 – Porto Alegre – RS – Brazil
{paulo.fernandes, lucelene.lopes, carlos.prolo, afonso.sales, renata.vieira}@pucrs.br

Abstract

This paper presents a novel approach to deal with dictionary retrieval. This new approach is based on a very efficient and scalable theoretical structure called Multi-Terminal Multi-valued Decision Diagrams (MTMDD). Such tool allows the definition of very large, even multilingual, dictionaries without significant increase in memory demands, and also with virtually no additional processing cost. Besides the general idea of the novel approach, this paper presents a description of the technologies involved, and their implementation in a software package called WAGGER. Finally, we also present some examples of usage and possible applications of this dictionary retriever.

Keywords: Dictionary retriever, Decision diagrams, Multi-valued Decision Diagrams

1. Introduction

Morphological analysis and Parts-Of-Speech tagging (POS-tagging) are practically solved problems in terms of accuracy. Nearly perfect scores (above 96%) can be achieved for many well-resourced languages like English (Schmid, 1994; Giménez and Màrquez, 2004). However, for many applications, as search engines and ubiquitous real time applications, time and space efficiency are at least as critical as accuracy.

According to a leading author in the natural language processing area: "This explosion of published information would be moot if the information could not be found, annotated and analyzed so that each user can quickly find information that is both relevant and comprehensive for their needs." (Manning et al., 2008). Moreover, the ability to add new words according to the individual form of speech used is also highly desirable in a such a volatile media as the web.

Therefore, we describe in this paper a novel way of storing dictionaries that can be used for morphological analysis focusing in efficiency and flexibility gains. Specifically, our proposal is to store a large number of words in a Multi-Terminal Multi-valued Decision Diagram (MTMDD) structure that can be used to classify any known word, discovering its word classes. The major benefits of such MTMDD approach are: (i) **memory efficiency** since, even for a set of several million words, the MTMDD structure can be stored with less than a hundred Mbytes; (ii) **flexibility** since new words can be efficiently added; and (iii) **fast recovery** since the time to classify words is negligible.

Such characteristics allow us to consider very large sets of words, as existing dictionaries for many languages altogether. This is a highly desirable feature considering multilingual language processing scenarios which are about to become more frequent in information systems. For example, a very thorough multilingual dictionary with English, French, Italian and Portuguese words would have around three million words, and could be stored in a nearly 70 Mbytes MTMDD structure capable of classifying 2 million words in less than 3 seconds in a laptop, *e.g.*, a MacBookPro with i7 2.2 GHz processor, 4 Gbytes memory.

From a practical point of view, the resource made available with this paper is a MTMDD-based word class retriever, *i.e.*, a software package called WAGGER that is capable of:

- Creating a new MTMDD-based word class retriever from a custom set of classes and an initial set of words with its corresponding classes (*e.g.*, from a dictionary of inflected words or from annotated corpora);
- Adding new words to an existing MTMDD-based word class retriever;
- Annotating raw text, according to an existing MTMDD-based word class retriever.

In this paper we briefly describe the MTMDD structures (Section 2). Section 3 presents general information about the WAGGER software package. Section 4 presents some numerical results in dealing with multi-lingual dictionaries using WAGGER software.

2. Multi-Terminal Multi-valued Decision Diagrams (MTMDD)

Multi-valued Decision Diagrams (MDDs) (Kam et al., 1998) are efficient, compact data-structures used to represent and manipulate extremely large sets of data. MDDs have been applied in the generation and manipulation of the reachable state space of structured models (Sales and Plateau, 2009; Ciardo et al., 2007). Besides low memory consumption, MDDs provide an efficient way of performing operations over large sets of data.

An MDD is a directed acyclic edge-labeled multi-graph with the following properties: (i) nodes are organized into N+1 levels, which are numbered by N to 0; (ii) *non-terminal nodes* are arranged at levels N to 1; (iii) level 0 has only two *terminal nodes*: 0 and 1; (iv) a non-terminal node p at level l contains arcs pointing to nodes at level l-1; (v) there are not duplicate nodes.

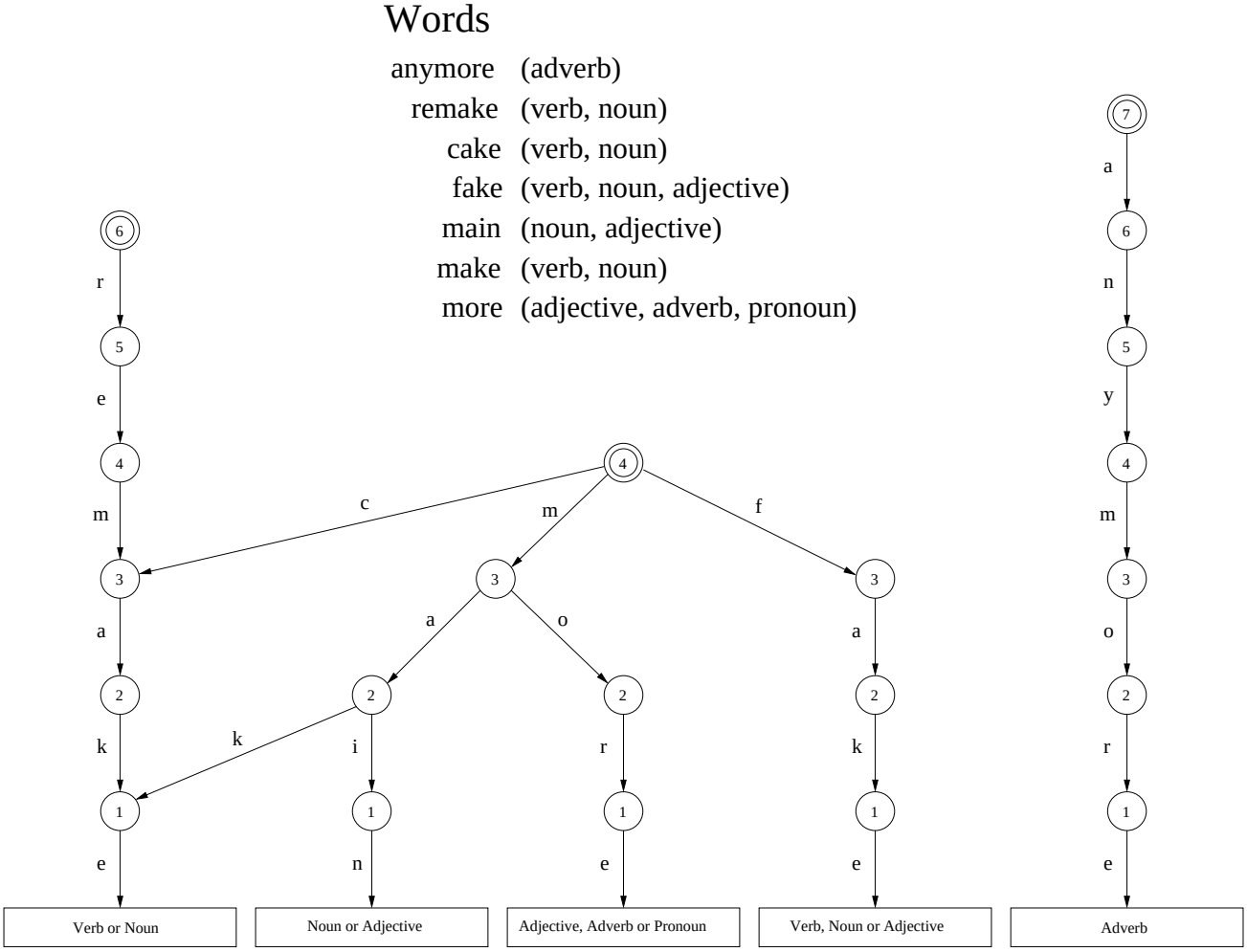

Figure 1: A set of words represented by a MTMDD

Multi-Terminal MDD (MTMDD) is an extension of MDD which have more than two terminal nodes, *i.e.*, it may has a set of *multi-terminal* nodes. In this paper we use an MTMDD structure to store a set of words, *i.e.*, a dictionary, where the terminal nodes represent the word classes associated to different words. Such dictionary representation using a MTMDD becomes the fast, efficient and scalable dictionary retriever.

Figure 1 illustrates an MTMDD representing a small set of words (in this case, seven words) associated to five word classes. In Figure 1, *non-terminal nodes* are depicted by *circles*, and *terminal nodes* are depicted by *squares*. *Double circled* non-terminal nodes represent the *root nodes* at each level of the MTMDD. Root nodes are used to access the words in a dictionary. The arc values between nodes correspond to letters of an alphabet. A given word is found in the dictionary represented by a N-level MTMDD *if and only if* the path through the MTMDD, starting at the root node corresponding to the level equal to this word's length, matching the arc values between nodes, *i.e.*, the letters which compose this word, reaches the terminal node that corresponds to the associated word class.

For instance, in Figure 1, the word "*make*" is *found* in the dictionary, since there is a *path* starting at the root node at level 4, which leads to the terminal node "*Verb or Noun*". On the other hand, in this example, the word "*math*" is *not found* in the dictionary, since there is no path starting at the root node at level 4, which reaches any terminal node at level 0.

In the example presented in Figure 1, where only seven words were represented, the gains of memory consumption may not be so visible. However, as the number of words represented in the MTMDD increases, the reuse of nodes leads to significant memory gains. In addition, it is important to remark that the seek time of a word in this structure is linear with the word length. In fact, that means there is no overhead beyond reading the word, and, obviously, the amount of words represented in the MTMDD structure does not impact at all on the seek time of a word.

3. WAGGER Software Package

In order to test in practice the MTMDD tool applied a dictionary retriever, an experimental software package was implemented. This package is called WAGGER – Word clAss taGGER using MTMDD-based dictionaries, and it is a command line application that can be integrated to any third party software. This software was implemented using C++ language and versions compiled to Linux and Mac OS X are available at the address http://www.inf.pucrs.br/afonso.sales/wagger. In that page

```
$ ./wagger -cre -dic=<dictionary> -out=<mtmdd>

$ ./wagger -add -mdd=<mtmdd> -dic=<words> -out=<mtmdd>

$ ./wagger -tag -mdd=<mtmdd> -txt=<text> -out=<tagged> [ -unk=<unknown> ]
```

Figure 2: WAGGER functionalities

the reader may find all examples and files used in this paper.
From a practical perspective, WAGGER software has three functionalities available to the user:

- creation of a new dictionary;
- addition of new words to an existing dictionary;
- annotation of a set words according to an existing dictionary.

The usage of these functionalities is described in Figure 2. In this figure there are three different types of calls to WAGGER. The first one creates (`-cre`) a new MTMDD structure (`-out`), holding all the words from the input dictionary (`-dic`). The second call adds (`-add`) to an existing MTMDD structure (`-mdd`) the words within dictionary (`-dic`), creating a new MTMDD structure (`-out`). The third call annotates all words within a text (`-txt=text`) according to a dictionary stored in a MTMDD structure (`-mdd`) delivering the possible word classes for each input word in an output file (`-out`), and optionally, in separate file (`-unk`), all words that were not found in the dictionary. The dictionary format (see example in Figure 3) is composed of four sections:

- The maximum length of a word, an integer (this value is needed to define the maximum depth of the MTMDD structure);
- The alphabet, *i.e.*, the valid letters for the dictionary (this list is needed to define the maximum out degree of MTMDD structure nodes);
- The word classes for this dictionary (this list is needed to define the number of terminal nodes in the MTMDD structure);
- The list of words of the dictionary with its classes.

In Figure 3, the dictionary corresponding to the MTMDD structure presented in Figure 1 is written. It has the maximum word length equal to 7 and the traditional latin alphabet without diacritics. There are 5 word classes for this example are: Verb, Noun, Adjective, Adverb and Pronoun. The words considered in this dictionary are represented, indexing its class according to the order of classes defined. Note, for example, that the word “remake” can be either a Verb (indexed as 1) or a Noun (indexed as 2).

```
#MAXLENGTHWORD
7
#ALPHABET
abcdefghijklmnopqrstuvwxyz
#CLASSES
Verb
Noun
Adjective
Adverb
Pronoun
#WORDS
fake 1
fake 2
fake 3
make 1
make 2
cake 1
cake 2
remake 1
remake 2
main 2
main 3
more 3
more 4
more 5
anymore 4
```

Figure 3: WAGGER dictionary file example

4. Numerical Results Evaluation

In order to illustrate the advantages of our proposal in terms of memory and time saving we run an experiment which is described below.
Using the proposed approach, we initially loaded three dictionaries with the following sets of words:

- 87,039 English words;
- 833,256 correctly spelled Portuguese words (using diacritics, *e.g.*, “ç”, “ã”);
- 786,842 Portuguese words without diacritics[1] (as usually written in informal text exchanges, or in limited keyboards).

[1]The smaller number of words without diacritics is due to the collapsing of words that differ just because of diacritics, *e.g.*, the word “maçã” (“apple” in English) will be considered as the same as the word “maca” (“stretcher” in English).

Dictionary	Number of words	Memory used	Time to create	Time to load
English	87,039	13.88 MB	5.63 s	0.47 s
Portuguese (w/ diacritics)	833,256	21.35 MB	100.11 s	0.68 s
Portuguese (w/o diacritics)	786,842	20.12 MB	92.70 s	0.67 s
Both Portuguese (w/ and w/o diacritics)	1,002,756	22.46 MB	172.10 s	0.72 s
Both Portuguese + English	1,086,580	34.51 MB	178.08 s	1.12 s

Table 1: Memory consumption and time to create and load the dictionaries

The first rows in Table 1 shows the memory used to store its corresponding MTMDD structure, the time to create it from a set of words, and to load it in memory considering the WAGGER software running over a machine with Intel Core i7 quad-core, which runs at 2.2 GHz under Mac OS X 10.6.8, with 4 GB of main memory. The time to seek the word class of a word is negligible, so it is not represented in this table. It is important to notice that, once generated the MTMDD structure, it is saved in a file and loaded later in order to seek for word classes.

To illustrate the computational gains achieved by increasing the size of dictionaries, we merged both Portuguese ones resulting in a nearly million word one and it does not represent a significant increase, neither in amount of memory, nor the time to load it, but the time to create this merged dictionary is so advantageous, since it takes only a little less (172.10 sec.) than the time to create both Portuguese ones (100.11 + 92.70 sec.). Nevertheless, the creation the MTMDD structure is likely to happen only once.

Pushing further our experiment, we added the English dictionary to the merged Portuguese, resulting in a slightly larger one. Unfortunately, in this case the lack of similar words between English and Portuguese results in an almost as large structure as the sum of the original dictionaries.

5. Conclusion

The MTMDD-based word class retriever is intended to be the basis of a tool that provides efficient and accurate POS-tagger with the MTMDD operations providing the flexibility required to accommodate the incremental character of open and evolving dictionaries. Such application opens the possibility to interpret online sources of written texts that are set aside by the existing POS-tagging tools, due to the lack of efficiency.

In fact, to the authors' best knowledge, the MTBDD proposed approach is superior to other existing solutions, since it combines the storage efficiency of Directed Acyclic Word Graphs (DAWGs (Lucchesi and Kowaltowski, 1993)) with an efficiently implemented set of operations to manipulate the underlying data structure.

A prototype of the proposed software tool, called WAGGER, as well as examples for some dictionaries, are freely available at http://www.inf.pucrs.br/afonso.sales/wagger, where there is also explanation about the usage of our tool.

Future works to the tool presented in this paper are the development of front end tools to perform some useful tasks in text processing. One of the examples is an automatic tool that receives sentences and identifies in which language the sentences were written. Such tool would be composed of a set of MTMDD-based dictionaries with the possible languages (one for each language) and according to the number of words founded in each dictionary would decide the language of the sentence. Notice that this same approach can be adapted to text categorization if, instead of language dictionaries, there is a set of MTMDD-based specific domain jargon dictionaries.

Another possible future work is to integrate the WAGGER software package as input POS-tagger of other software tools (*e.g.*, ExATO lp (Lopes et al., 2009)). However, the usage possibilities of our multilingual dictionary retriever is not only restricted to the POS-tagging applications.

We hope many computational intensive text processing applications based on dictionaries may become feasible by eliminating the efficiency bottlenecks through the use of our fast and memory efficient software tool.

6. Acknowledgements

The authors want to express their gratitude to Vinicius Emmanuel Wobeto, undergraduate student at the PUCRS University, for assembling French, Italian and Latin dictionaries. The authors would also wish to thank Luis Otvio Colla Furquim for providing us the Portuguese dictionary used in the experiments. Paulo Fernandes is partially funded by CNPq grant PQ 307284/2010-7. Lucelene Lopes is funded by CAPES grant PNPD 02388/09-0. Renata Vieira is partially funded by CNPq grant PQ 306832/2009-7. The order of authors is merely alphabetical.

7. References

Gianfranco Ciardo, Gerald Lüttgen, and Andrew S. Miner. 2007. Exploiting interleaving semantics in symbolic state-space generation. *Formal Methods in System Design*, 31(1):63–100.

Jesús Giménez and Lluís Màrquez. 2004. Svmtool: A general pos tagger generator based on support vector machines. In *Proceedings of the 4th LREC*.

Timothy Kam, Tiziano Villa, Robert K. Bryaton, and Alberto Sangiovanni-Vincentelli. 1998. Multi-valued decision diagrams: theory and applications. *Multiplie-Valued Logic*, 4(1-2):9–62.

Lucelene Lopes, Paulo Fernandes, Renata Vieira, and Guilherme Fedrizzi. 2009. ExATO lp – An Automatic Tool

for Term Extraction from Portuguese Language Corpora. In *Proceedings of the 4th Language & Technology Conference: Human Language Technologies as a Challenge for Computer Science and Linguistics (LTC '09)*, pages 427–431, Poznan, Poland, November. Adam Mickiewicz University.

Cláudio L. Lucchesi and Tomasz Kowaltowski. 1993. Applications of finite automata representing large vocabularies. *Software: Practice and Experience*, 23(1):15–30.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, Cambridge.

Afonso Sales and Brigitte Plateau. 2009. Reachable state space generation for structured models which use functional transitions. In *Int. Conf. on the Quantitative Evaluation of Systems (QEST'09)*, pages 269–278, Budapest, Hungary, September.

Helmut Schmid. 1994. Probabilistic Part-of-Speech Tagging Using Decision Trees. In *Proceedings of International Conference on New Methods in Language Processing*, September.