Automatic Building of a Dictionary of Technical Terms and Collocations Based on AVL Tree

Davor Blažeković, Maja Matetić and Marija Brkić
Department of Informatics
University of Rijeka
Omladinska 14, 51000 Rijeka, Croatia
Phone: (+385) 91 503 1327 E-mail: davor.blazekovic@ri.t-com.hr, majam@inf.uniri.hr, mbrkic@inf.uniri.hr

Abstract - The aim of this work is automatic building of a dictionary of technical collocations. The input to the building procedure is a number of Croatian technical texts from a restricted domain. The dictionary is implemented as an AVL tree, a binary tree which ensures performance of operations such as insertion and retrieval in logarithmic time. Tree nodes contain words and their part of speech (POS) tags. POS tags are partly obtained using the Croatian Morphological Lexicon. POS information enables us to use syntactic filter in order to reduce noise in collocation retrieval.

I. INTRODUCTION

The subject of this paper is the process of building technical terms and collocations dictionary. Technical terms dictionary building for different constrained domains is a complex procedure that requires expert engagement and close collaboration with linguists.

An impressive amount of work was devoted over the past few decades to collocation extraction. The state of the art shows that there is interest in statistical [1, 2], unsupervised machine learning [3, 4], and syntactic approaches [5] to the problem. In our approach we use the syntactic preprocessing of texts in order to better identify candidate expressions.

We present the application we developed as a support in the task of automatic technical terms dictionary building for a restricted domain. Using a sample technical text from a restricted domain and text based POS tagged AVL dictionary, the induction of syntactic collocation patterns is made. We are interested in the extraction of collocations by specifying one of the words, and with respect to the POS tags of the surrounding words. One approach to this problem is described in [5].

Lexicon is the heart of any natural language processing system. Accurate words with grammatical and semantic attributes are essential or highly desirable for any application, whether it is machine translation, information extraction, various forms of tagging, or text mining. However, good quality lexicons are difficult to construct and require enormous amount of time and manpower [6].

Our dictionary is based on an AVL tree, a balanced binary tree structure named after its authors, G. M. Adelson-Velskii and E.M. Landis [7]. The heights of the subtrees in an AVL search tree must not differ by more than one. AVL trees are, therefore, known as height-balanced binary search trees. Imposing that limitation makes the search efficiency logarithmic, O ($\log_2 n$), in comparison to the linear lists, which have linear search efficiency O(n).

First we describe AVL trees and operations which are to be performed in order to balance them. An important step in collocations dictionary building is POS tagging which is introduced in the third paragraph. We follow with the description of the application we developed for automatic collocations dictionary building. At the end we give an example illustrating the application use.

II. DICTIONARY IMPLEMENTATION

The dictionary is implemented as a balanced binary tree structure – the AVL tree. AVL tree data structure is created by mathematicians Adelson-Velskii and Landis [7]. An AVL tree is a binary search tree in which the heights of the left and right subtree (T_L and T_R) differ by no more than one. Because AVL trees are balanced by changing their height, they are also known as height-balanced binary search trees. Fig. 1 shows an example of an AVL tree.

$$| HL - HR | <= 1 \qquad (1)$$

Node insertion and deletion operations can have O(N) worst-case efficiency, where N is the number of tree nodes. To execute more efficient operations, we have to minimize the tree height after each operation. A balanced AVL tree with N elements has height proportional to log N. The goal is to keep tree property after each operation.

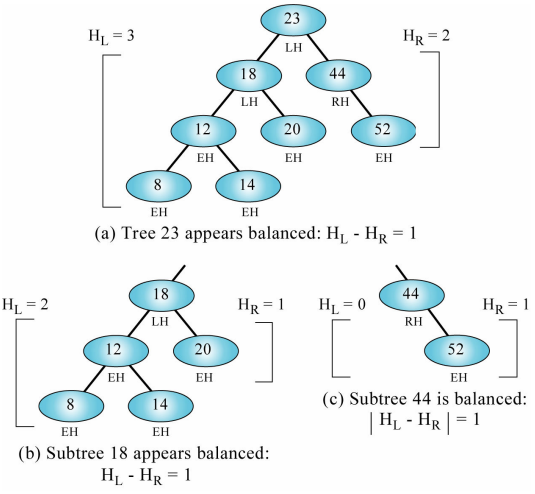

Fig. 1: AVL tree example [7].

Whenever we insert a node into a tree or delete a node from a tree, the resulting tree may be unbalanced. In that case we must rebalance the tree. Balancing AVL trees is done by rotating nodes either to the left or to the right.

All unbalanced trees fall into one of these four cases (Fig. 2):

1) Left of left, when a subtree of a tree that is left high has also become left high
2) Right of right, when a subtree of a tree that is right high has also become right high
3) Right of left, when a subtree of a tree that is left high has become right high
4) Left of right, when a subtree of a tree that is right high has become left high.

Fig. 3 contains a simple left rotation [7]. Although the subtree 18 is balanced, the root is not. We therefore rotate the root to the left, making it the left subtree of the new root, 18. The case in Fig. 3(b) is more complex. It shows a right high root with two right high subtrees. This creates a right-of-right out-of-balance condition. To correct the balance, we rotate the root to the left, making the right subtree, 20, the new root. In this process, 20's left subtree is connected as the old root's right subtree, preserving the order of the search tree.

A part of the program implementation of simple left rotation in C++ is given:

```
static Cvor* SingleRotateWithLeft( Cvor* K2 )
        {
            Cvor* K1;
            K1 = K2->L;
            K2->L = K1->R;
            K1->R = K2;
```

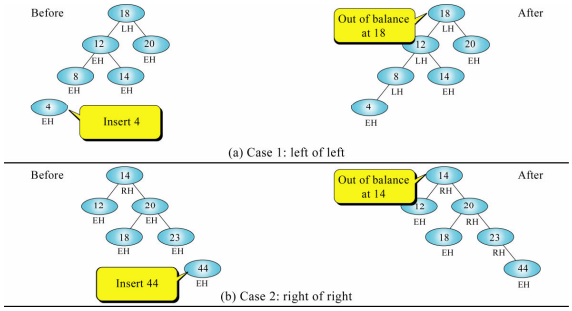

Fig 2: Out of balance AVL trees [7].

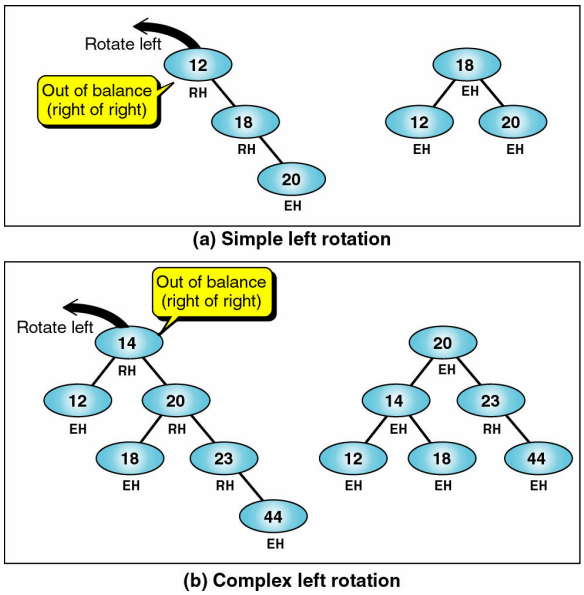

Fig 3: Right of right – single rotation left [7].

```
K2->Height = Max( Height( K2->L ), Height( K2->R
) ) + 1;
            K1->Height = Max( Height( K1->L ),
K2->Height ) + 1;
            return K1;

        }
```

III. POS TAGGING

POS tagger is a necessary module in any natural language understanding subsystem. It is a program that annotates each word in the input by specifying its grammatical properties, such as part of speech, number, person, etc. Part of speech information about each word in a sentence helps to determine its syntactic structure. Since the existence of a tagged corpus is crucial for training a POS tagger and creating more advanced language processing tools, we tagged our dictionary with the help of the Croatian Morphological Lexicon described in [8].

Tree nodes contain words and their part of speech (POS) tags. POS information enables us to use syntactic filter in order to reduce noise in collocation retrieval.

Problem arises as a consequence of Croatian language complexity and specificity regarding language inflexions and free word order [9, 10, 11, 12, 13]. There are only few applications and methods developed for machine processing of Croatian language texts. The input to the Croatian Morphological Lexicon is a limited number of words and the output is a list of these words with their POS tags. The tagging efficiency is rather unsatisfactory because only 20-30 percent of input words are successfully tagged, and the input is constrained to nineteen words per one tagging session. Assuming that Morphological Lexicon is error free, the accuracy of the automatically derived POS tags is 100%. The untagged words are then manually processed.

We deal with only major categories defined in the MULTEXT-East morphosyntactic specifications and exclude their attribute-value pairs to reduce the number of tags [14]. The major POS categories comprise nouns (N),

verbs (V), adjectives (A), pronouns (P), adverbs (R), adpositions (S), conjunctions (C), numerals (N), particles (Q), interjections (I), abbreviations (Y), and residuals (X).

IV. AUTOMATIC GENERATION OF DICTIONARY

Lexicon is the heart of any natural language processing system. Accurate words with grammatical and semantic attributes are essential or highly desirable for any application, whether it is machine translation, information extraction, various forms of tagging, or text mining. However, good quality lexicons are difficult to construct and require enormous amount of time and manpower. The authors in [6] have addressed this problem by showing how the WordNet can be used to construct a document specific dictionary.

Such a problem is relevant, for example, in machine translation context. If the document specific dictionary is available apriori, the generation of a target language document from a source language document essentially reduces to syntax planning and morphology processing for the pair of languages involved.

V. APPLICATION DESCRIPTION

The tool being developed for text processing incorporates a fusion of linguistic knowledge and corpora methods. The Web offers a huge repository of documents written in a multitude of languages and dialects, of different categories, and constantly changing over time. It is, therefore, well suited for collocation related tasks [6].

Collocation acquisition from corpora is usually based on statistical significance test on the words that occur close to each other. Syntactic parsing methods allow linguistically-informed methods of extraction nowadays. POS tagging as a text preprocessing method supports the identification of collocation candidates.

User interface with application controls can be seen in Fig. 4. Explanation of the functionality of main application controls follows below.

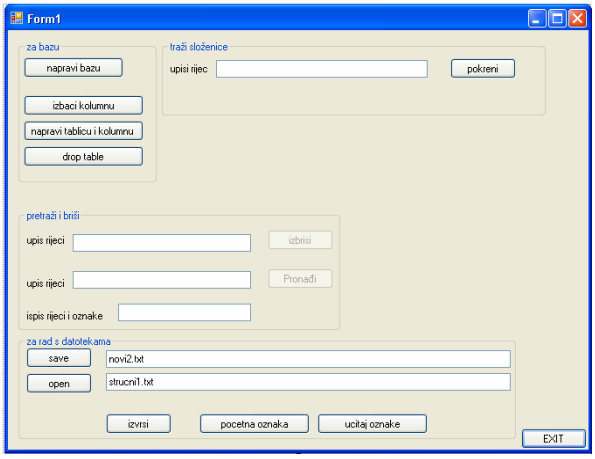

Fig 4: Application interface.

Izvrši (Execute) – generates a dictionary of alphabetically sorted words on the base of a technical text, using AVL tree. Sorting five hundred text pages takes about ten seconds.

Početna oznaka (Initial tag) – initializes dictionary words to null tags. An example showing a part of a tagged dictionary is given below:

kavi -0
koja -0
koji -0
kojima -0
kompaktan -0
korisnik -0
korisničko -0
koristiti -0
kvalitetno -0
lako -0

Null tags mark where manually inserted tags or tags from the Croatian Morphological Lexicon are to be placed. Generated dictionary is the input for the Croatian Morphological Lexicon. Output is partly POS tagged dictionary.

Učitaj oznake (Load tags) – built tagged dictionary can be used for tagging new texts. Missing tags can be inserted in the dictionary if they are found in another specified dictionary. Here is an example showing a part of the described procedure results:

Kod -0	Kod -N
Loša -0	Loša -A
Prednosti -0	Prednosti -N
Takav -0	Takav -P
Također -C	Također -C
To -0	To -P
Vrlo -0	Vrlo -A
akvizicijska -0	akvizicijska -A
ali -0	ali -C
automatizirane -0	automatizirane -A
bi -0	bi -V
biti -V	biti -V
broj -N	broj -N
dobije -0	dobije -V
dobiti -0	dobiti -V
dobiveni -A	dobiveni -A
dodatno -0	dodatno -A

Pokreni (Start) – button which activates technical terms and collocations search. Collocation search is constrained with a word which should be one of the collocates.

A. An example: Building a Dictionary of Technical Terms Based on AVL Tree

By using a sample technical text from our corpus and the corpus-based POS tagged AVL dictionary, the induction of syntactic collocation patterns is made. We are interested in the extraction of collocations by specifying one of the words, and with respect to the POS tags of the surrounding words.

This is a part of the AVL dictionary word list with its POS tags:

Inverzno -A
Kod -N
Loša -A
PC -N

Prednosti -N
SetTimer -N
Takav -P
Također -C
To -P
Vrlo -A
Windowsi -N
akvizicijska -A
ali -C
automatizirane -A
bi -V
biti -V
broj -N
debugati -V
dobije -V
dobiti -V
dobiveni –A
dodatno -A
dotičnim -A
dovoljno -A

The syntactic patterns for English proposed by different authors are shown in the following list [6]:

1. Lexical collocations in BBI dictionary:
V-N, N-A, N-V, N-P-N, A-Adv, V-Adv;
2. Hausmann's collocation definition:
N-A, N-V, V-N, V-Adv, A-Adv, N-[P]-N;
3. Xtract collocation extraction system:
N-A, N-V, V-N, V-P, V-Adv, V-V, N-P, N-D;
4. WordSketch concordance system:
N-A, N-N, N-P-N, N-V, V-N, V-P, V-A, N-Conj-N, A-P;
5. FipsCo system:
N-A, N-N, N-P-N, N-V, V-N, V-P, V-P-N.

We are interested in the extraction of collocations by specifying one of the words which forms one part of a technical term. That would allow us to compile a dictionary of technical terms for any domain, giving an overview of regular and irregular usage of such terms. As an illustration, the result of the collocation extraction is given below. One of the collocates is the word 'sustav' (system) and the chosen syntactic patterns are A-N and P-N. The user interface is shown in Fig. 5.

Collocations containing word 'sustav' in the sample text are:

nezavisan –A sustav –N

Takav –P sustav –N

regulacijski –A sustav –N

VI. CONCLUSION

Our application for automated collocations dictionary generation is based on the dictionary of words and their tags. The quality of the collocation extraction depends on the size of the tagged dictionary. The absence of a complete morphological lexicon, which should be a basic resource for text tagging, is a significant obstacle in Croatian language texts processing. Language resources development requires mutual adjustment and compliance with the international standards.

Fig. 5. The interface of the collocation extraction system.

REFERENCES

[1] P. Pecina, "A Machine Learning Approach to Multiword Expression Extraction", In Proceedings of the LREC 2008 Workshop Towards a Shared Task for Multiword Expressions (MWE 2008), Marrakech, Morocco, May 2008.

[2] H. Wang, "Extraction of Word Collocations in Singapore Mandarin Chinese", in Proceedings of the International Conference on Asian Language Processing 2008, Thailand, November 12-14, 2008.

[3] A-C. N. Ngomo, "Knowledge-free discovery of domain-specific multiword units", in Proceedings of the 2008 ACM symposium on Applied computing, pages 1561-1565, Fortaleza, Ceara, Brazil, 2008.

[4] A-C. N. Ngomo, *SIGNUM: A Graph Algorithm for Terminology Extraction, Computational Linguistics and Intelligent Text Processing*, Lecture Notes in Computer Science, pages 85-95, 2008.

[5] V. Seretan, L. Nerima, and E. Wehrli, "Using the Web as a corpus for the syntactic-based collocation identification", in Proceedings of International Conference on Language Resources and Evaluation (LREC 2004), pages 1871–1874, Lisbon, Portugal, 2004.

[6] N.Verma and P. Bhattacharyya, Automatic Lexicon Generation through WordNet, The Second Global Wordnet Conference, Brno, Czech Republic, 2004

[7] R. F. Gilberg, B. A. Forouzan: *Data Structures: A Pseudocode Approach With C,* Course Technology Press, 1998.

[8] M. Tadić, "Croatian lemmatization server," in Proc. of the 5th Formal approaches to South Slavic and Balkan languages Conference, Sofija, 2006, pp. 140-146.

[9] T. Žubrinić, "Mogućnosti strojnoga označavanja i lematiziranja korpusa tekstova hrvatskoga jezika", magistarski rad, Filozofski fakultet Sveučilišta u Zagrebu, 1995, http://www.fi.muni.cz/gwc2004/proc/81.pdf

[10] M. Tadić, "Building the Croatian-English Parallel Corpus", LREC2000 zbornik, Atena, 31. svibnja-2. lipnja 2000, ELRA, Pariz-Atena 2000, Vol. I, str. 523-530.

[11] M. Tadić, "Računalna obradba hrvatskih korpusa: povijest, stanje i perspektive", Suvremena lingvistika 43-44, str. 387-394, 1997.

[12] M. Tadić, "Raspon, opseg i sastav korpusa suvremenoga hrvatskoga jezika", Filologija 30-31, str. 337-347, 1998.

[13] M. Brkić, M. Matetić: "Preparation for POS tagging of Croatian weather forecast domain", Proceedings of the 31st International Convention MIPRO 2008, Vol. III, CTS and CIS, Opatija, 2008, pp. 228-232

[14] Multext-East Resources, version 3. Available: http://nl.ijs.si/ME/V3/.