INFORMATICA, 2011, Vol. 22, No. 2, 203–224

Semi-Automatic Bilingual Corpus Creation with Zero Entropy Alignments

Algirdas LAUKAITIS[1], Olegas VASILECAS[1], Ricardas LAUKAITIS[2], Darius PLIKYNAS[2]

[1]*Fundamental Sciences Faculty, Vilnius Gediminas Technical University*
Saulėtėkio al. 11, LT-10223 Vilnius, Lithuania
[2]*Academy of Business and Management, Research Centre*
Basanavičiaus 29A, LT-03109 Vilnius, Lithuania
e-mail: algirdas.laukaitis@fm.vgtu.lt, olegas@fm.vgtu.lt, ricardas.laukaitis@vva.lt, darius.plikynas@vva.lt

Received: October 2010; accepted: April 2011

Abstract. In this paper, we describe a model for aligning books and documents from bilingual corpus with a goal to create "perfectly" aligned bilingual corpus on word-to-word level. Presented algorithms differ from existing algorithms in consideration of the presence of human translator which usage we are trying to minimize. We treat human translator as an oracle who knows exact alignments and the goal of the system is to optimize (minimize) the use of this oracle. The effectiveness of the oracle is measured by the speed at which he can create "perfectly" aligned bilingual corpus. By "Perfectly" aligned corpus we mean zero entropy corpus because oracle can make alignments without any probabilistic interpretation, i.e., with 100% confidence. Sentence level alignments and word-to-word alignments, although treated separately in this paper, are integrated in a single framework. For sentence level alignments we provide a dynamic programming algorithm which achieves low precision and recall error rate. For word-to-word level alignments Expectation Maximization algorithm that integrates linguistic dictionaries is suggested as the main tool for the oracle to build "perfectly" aligned bilingual corpus. We show empirically that suggested pre-aligned corpus requires little interaction from the oracle and that creation of perfectly aligned corpus can be achieved almost with the speed of human reading. Presented algorithms are language independent but in this paper we verify them with English–Lithuanian language pair on two types of text: law documents and fiction literature.

Keywords: Viterbi alignments, dynamic programming, string alignments, machine translation, natural language processing, rapid development, low-density languages.

1. Introduction

Let's assume that human translator is given a task to align each sentence in the bilingual corpus in word-by-word units as it described in Fig. 1 or in Table 1 or as an example in Zero Entropy Corpus (2010). The instructions for human translator will be: (1) align those words/phrases that are direct translations of each other or (2) mark them as erroneous translation or (3) align and mark as coreference or (4) leave some words unaligned and mark them as words with pure syntactic role in the sentence or (5) leave some words

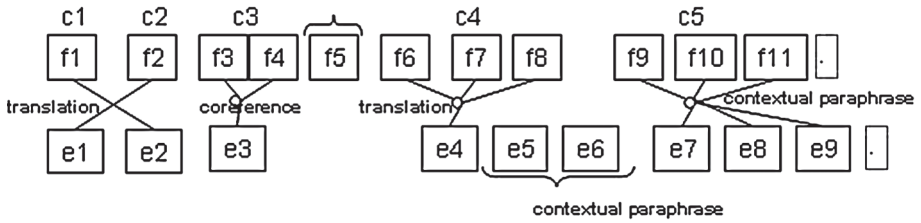

Fig. 1. Alignments within a sentence.

unaligned and mark them as supplementary in some semantic sense or (6) leave some words unaligned and mark them as an error in translation.

Such manually aligned corpus will be a useful resource for building translation memories and translation rules for automatic and semi-automatic translation systems. The main problem with such corpus is that it is very difficult to create and sometimes it is much easier to translate a given sentence than to align it. In fact, up to our knowledge, such kind of aligned corpus has not been created for research or translation purpose on a big scale. In most research projects hand made alignments has been created only on several hundred sentences just for the purpose to evaluate the quality of automatically made alignments. Automatically made alignments has been investigated in an area of statistical machine translation for the past twenty years and some progress has been made and many automatic procedures has been suggested by Brown *et al.* (1993), Och and Ney (2002), Laukaitis and Vasilecas (2008). Several extensions to Brown *et al.* (1993) has been tested in Toutanova *et al.* (2003). Those papers are single word based alignment models. Improvement can be made if we add phrase based alignment models as in Marcu and Wong (2002) or Watanabe *et al.* (2002).

But, the problem with automatic alignment methods is that they can generate errors and especially in cases of rare words and phrases. In those fully automatic alignment algorithms we receive a probabilistic model and a search in state space with such model for an optimal alignment is NP-Hard problem (Dzemyda and Sakalauskas, 2009, 2011; Dzemyda and Petkus, 2001). This is why algorithms and processes that give manual alignment quality but significantly reduce efforts required to make such alignments are important in the field of computational linguistics. Then, methodology and algorithms that helps to reduce time and labor needed to create "perfectly" aligned bilingual corpus is the main contribution reported in this paper.

It is important to emphasize what we mean by "perfectly" aligned bilingual corpus. Figure 1 displays some possible alignments between two sentences $f = f_1 f_2 f_3 f_4 f_5 f_6$ $f_7 f_8 f_9 f_1 0 f_1 1$ and $e = e_1 e_2 e_3 e_4 e_5 e_6 e_7 e_8 e_9$ where f_i and e_i stand for the words in position i in those sentences. Label c_i in Fig. 1 marks: (1) an irreducible phrase that can form a record in conventional bilingual dictionary, (2) a coreference (3) a contextual paraphrase. Some words can be untranslated and they are marked as such. Sometimes there can be an erroneous translation and human translator can spot such an error and mark it as such.

Table 1

An example of zero-entropy book alignment.

In$_{iš}$ the beginning$_{pradžių}$ he$_{jis}$ was$_{buvo}$ Christopher$_{Kristoferis}$ Bellew$_{Belju}$. By the time he was at college$_{Koledže}$ he$_{jis}$ had become$_{virto}$ Chris$_{Krisu}$ Bellew$_{Belju}$. Later$_{Vėliau}$, in the Bohemian crowd$_{bohema}$ of San$_{San}$ Francisco$_{Francisko}$, he$_{jį}$ was called$_{praminė}$ Kit$_{Kitu}$ Bellew$_{Belju}$. And in the end$_{Bet\ galiausiai}$ he$_{jis}$ was known by no other name than Smoke$_{Smoku}$ Bellew$_{Belju}$. And this history$_{istorija}$ of the evolution$_{keitimosi}$ of his$_{jo}$ name$_{vardo}$ is the history$_{istorija}$ of his$_{jo\ paties}$ evolution$_{keitimosi}$. Nor would$_{Jeigu\ nebūtų}$ it have happened$_{atsitikę}$ had he not had a fond$_{švelnios}$ mother$_{motinos}$ and$_{ir}$ an iron$_{geležinio}$ uncle$_{dėdės}$, and had he not received$_{negavęs}$ a letter$_{laiško}$ from$_{iš}$ Gillet$_{Džileto}$ Bellamy$_{Belamio}$.	Iš$_{in}$ pradžių$_{beginning}$ jis$_{he}$ buvo$_{was}$ Kristoferis$_{Christopher}$ Belju$_{Bellew}$. Koledže$_{college}$ jis$_{he}$ virto$_{become}$ Krisu$_{Chris}$ Belju$_{Bellew}$. Vėliau$_{Later}$ San$_{San}$ Francisko$_{Francisco}$ bohema$_{Bohemian\ crowd}$ jį$_{he}$ praminė$_{called}$ Kitu$_{Kit}$ Belju$_{Bellew}$. Bet$_{And\ in\ the\ end}$ galiausiai jis$_{he}$ tapo Smoku$_{Smoke}$ Belju$_{Bellew}$, ir kitaip jo niekas jau nebevadino. Tokia jo$_{his}$ vardo$_{name}$ keitimosi$_{evolution}$ istorija$_{history}$ buvo jo$_{his}$ paties keitimosi$_{evolution}$ istorija$_{history}$. Jeigu$_{Nor\ would}$ nebūtų jis turėjęs švelnios$_{fond}$ motinos$_{mother}$ ir$_{and}$ geležinio$_{iron}$ dėdės$_{uncle}$, negavęs$_{not\ received}$ laiško$_{letter}$ iš$_{from}$ Džileto$_{Gillet}$ Belamio$_{Bellamy}$, nieko panašaus nebūtų ir atsitikę$_{happened}$.

Alignments as in Fig. 1 or in Table 1 can be produced by human translator with 100% confidence, i.e., alignments as the random variable will be interpreted as the probability $P = 1$. On the other hand statistical machine translation methods can produce the same alignment but it will be value of a random variable where the size of the values set is 2^{lm}, where l means number of words in the first sentence and m means number of words in the second sentence. Some statistical machine translation models such as Brown *et al.* (1993) Model 2 will be able produce such alignments as an optimal solution, i.e., Viterbi alignments other more complex models will produce alignments as some sub-optimal solution under given model. Additionally, by a traditional statistical training approach those alignments will be without any explanatory labels like c_i in Fig. 1. And again, if those alignments will be received from human translator then they will be treated as the alignments received from Oracle who knows exactly how to map each word and explain those mappings by classification labels that are outside traditional statistical induction. Nevertheless, our approach in this paper is that machine can suggest its solution of the alignments and present them to the human translator, who then can confirm it or correct it at some positions.

Then, in this paper we are mostly concerned with machine learning algorithms that help to achieve this goal, i.e., machine learning algorithms that consider human-translator as an "oracle" and use him in the learning process. Actually, at the present time there is an interest in optimization and human-machine communication processes and in particular when we are trying to integrate statistical inference with human translator as an oracle (Barrachina *et al.*, 2009). So, there we suggest model for creating 'perfectly" aligned bilingual corpus from scanned books collection. In most languages we can find thousands of translated books and models that allow to receive 'perfect" alignments will be

useful for dictionaries creation and automatic translation systems improvement. Additionally, this model can be useful in translation quality control as it marks all translation discrepancies.

English–Lithuanian language pair has been used in this research project where we scanned several translated books and aligned them with the method presented in the following sections. The Lithuanian language belongs to Baltic group of Indo-European languages and represents the East Baltic subgroup. It has a very rich morphology (Pajarskaite *et al.*, 2004; Lipeika, 2010) for all open word classes and it shows clear relation with Slavic morphology. The nouns in Lithuanian have 7 cases, 3 numbers and 2 genders. Verbs have a lot of forms: 4 tenses, 3 types of conjugation, 3 moods and a great plenty of verbal forms like participles, semi-participles, infinitives and supine (Vaiciunas *et al.*, 2004). We use dictionaries to solve the problems of morphology in the same way as it was done in Skripkauskas and Telksnys (2006).

Those introductional remarks define the rest of the paper which is organized as follows. In Section 2 we present the general architecture of the system for zero entropy corpus creation. Section 3 describes statistical inference model for finding words and phrases translation probabilities. Section 4 describes alignments procedure and algorithms that capture oracle behavior when he creates zero entropy corpus. Section 5 describes empirical tests of the suggested method. Finally, concluding remarks and future work are presented at the end of the paper.

2. General Framework

We described what we call perfect alignments or zero entropy alignments at the introduction. There is impossible to get automatically those alignments with current theoretical and practical techniques of artificial intelligence (AI) because if we do, then it will mean that we created an artificial intelligent agent that can pass any Turing AI test (i.e., the task is AI-complete). Then, for those alignments we need a solution of machine learning and software engineering because a human is used as an oracle and we would like to assist this oracle with machine learning software to optimize his activity.

From the point of view of software engineering it is important to note that one way to increase alignments productivity it is just to improve the user interface (UI). In the initial stage of our project we done exactly that but our findings in this "user-interface" question can be summarized as follows: make alignments "user-interface" as close as possible to resemble video-games user interface, i.e.,: (1) only computer pointing device (like PC mouse) must be used (no keyboard), (2) instant reaction to user actions (i.e., after user entered alignment, machine must recompute related information in several seconds), (3) computer must provide some scoring for the user action (in our case, amount by which corpus entropy has been reduced) so that user stays involved and continues aligning. Figure 2 shows UI that we used in this research project. There are 24 active elements which are activated by PC mouse click and we found that those 24 active elements are enough to create comfortable UI for human editor without PC keyboard.

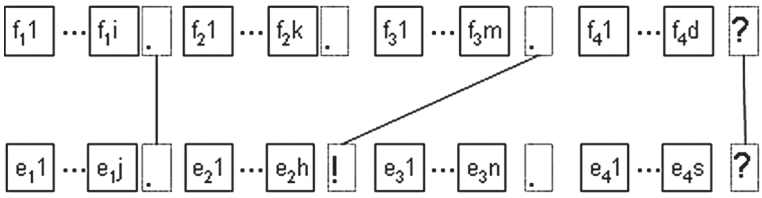

Fig. 2. Alignments between sentence punctuation marks.

Fig. 3. User interface for alignment review and corection.

Then, as we defined requirements for alignments "user-interface", the question is how to implement a machine learning algorithm that take those requirements into consideration? But, before to answer this question, we must look at the state of the art in the statistical inference area of machine translation because it shapes our systems architecture for books alignments. First of all, there are two types of alignment algorithms: the alignments between sentences and alignments between words within the sentence. Then, the system we developed has two distinct modules: one for alignments between sentences and one for alignments within sentences (see Figs. 1, 2, 3 and 4).

There is a number of papers dedicated for aligning documents on the sentences-to-sentence level (or, in other words, on sentence punctuation marks) with their translations in a bilingual corpus. The method developed by Brown*et al.* (1991) is based on sentence length and require some anchor points like the alignment of paragraphs and chapters to constrain the search and increase precision. Our research shows that if we have scanned the book by some optical character recognition (OCR) software then often those requirements on anchor points are not met due to inaccuracy of white space handling by OCR. Gale and Church (1993) presents impressive alignment accuracy with their two stage dynamic programming algorithm but again they require that paragraphs separation must be well defined. That often is difficult to achieve with OCR scanned book. Chen (1993) suggested to use bilingual lexicon for sentence alignments but in our case we found that we need a hybrid approach that use bilingual lexicon and dynamic programming algorithms on sequences.

The paper of Brown *et al.* (1993) is usually credited in the area word-to-word alignments within the sentence. The authors of the paper developed a simple Expectation Maximization (EM) algorithm by the model which they call Model 1. Then, the authors of the paper admits that the Model 1 leads to poor alignment and starts to increase model

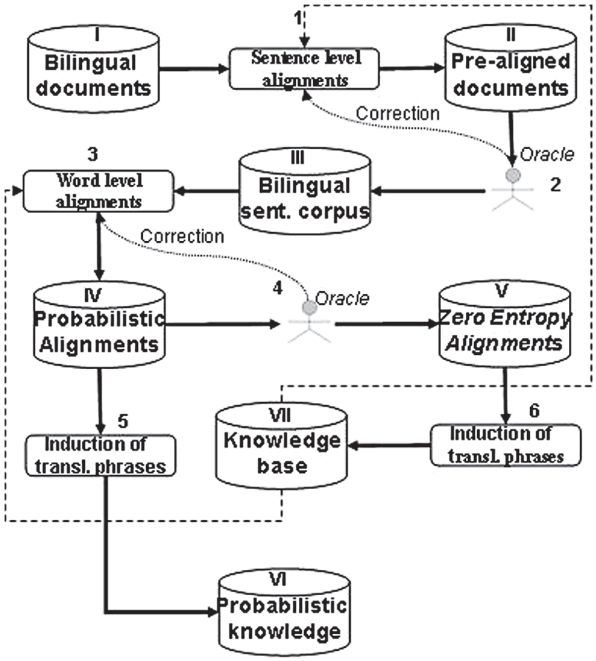

Fig. 4. The framework for zero entropy alignments generation.

complexity my introducing some common sense knowledge such as: translation often is monotonous, some words are not translated and some words tends to appear close to each other. Nevertheless, the last model from Brown *et al.* (1993), Model 5, does not bring satisfactory alignments especially when we deal with languages of complex morphology (Laukaitis and Vasilecas, 2007), (Laukaitis and Vasilecas, 2008). The recent trend in statistical machine translation was towards phrase based translations (Och and Ney, 2004), but the authors of that paper used the same Model 3 from Brown *et al.* (1993) to get alignments. The only difference is that they use it twice in both translation directions. In our algorithm we use some ideas of mentioned papers but there is one additional property of the suggested algorithm that gives it a better quality of the alignments. It consist of our ability to construct a decision model from all manual corrections that has been made by editing alignments suggested by stochastic optimization procedure.

But before we start describing algorithms in details, let us look at the whole framework that we developed during this research project and which is shown in Fig. 2. We can see that there are two processes that involve the oracle (number 2 and 4), two decision making processes that are based on machine learning models (processes number 1 and 3) and two knowledge induction processes (number 5 and 6). All computation results are stored in the database No. 7. Next, we describe these processes and resources in more details.

The whole alignment process starts with a bilingual document base. The most common resources for that kind of database are books that can be scanned with modern optical

character recognition software. For example, there are thousands of books translated from various languages into Lithuanian language but there is no publicly available Lithuanian language fictional books bilingual corpus on the scale which can be compared for example with the European Union document base (Official Journal of the European Union, 2010). During this research project we scanned 5 fictional books that have expired copyrights and then tested the quality of OCR software. We found that in general there was only about 0.3% of words that have been scanned with errors. This fact is in favor of using fiction books to generate translation memory and translation rules if only we were able to have algorithms for high precision alignments. Another universal resource for bilingual document database is legal documents and technical documentation. Sometimes those documents are already aligned on the sentence level or their alignments on the sentence level can be handled without any process of machine learning and oracle intervention (we used a small set of regular expressions in the case of well-structured technical documentation). On the other hand for the scanned books or documents without regular structure we need some optimization and revision process. In Fig. 2 the process named *sentence level alignments* is responsible for automatic documents alignments on the sentence-to-sentence level. Those pre-aligned documents then can be reviewed by the oracle.

One of the main ideas that we verified and implemented for alignments on the sentence-to-sentence level consists of generating two sequences of DNA symbols for each document. That allowed us to use free and off-the-shelf software packages used in bioinformatics research projects to align DNA sequences. One sequence is a document with all punctuation marks removed and another sequence is those punctuation marks. Then two different algorithms are used to align these sequences. The first sequence which consists only of words we treat as one big sentence where words are coded with DNA symbols (details are described in Section 3). The second sequence consists of punctuation marks coded to DNA letters. We receive two sequences that have close resemblance with genetic code sequences and as we mentioned there is a number of algorithms and software that tries to analyze and compare multi-genetic sequences. In this project we found that the product BioJava (2008) is especially well suited for our purpose.

Next, we describe processes and resources shown in Fig. 2 without algorithmic details which are described at length in Section 3. We start with the collection of OCR scanned books marked in Fig. 2 as *Bilingual documents*. Then, these books are processed by *sentence level alignments* (No. 1) process after witch we receive automatically aligned sentences. Next, these sentences are processed by the oracle (process No. 2 in Fig. 2). We found that the most productive way for the oracle at this process is to review those parts of sentences that have the biggest misalignments. If it happens that the oracle finds some bad alignment and corrects it, then, process of sentences alignment (No. 1) is involved again. The only difference compared with the previous processing is that corrections made by the oracle are used as constrains. We found that with such reactive approach from the system the whole book of 500 kilobytes can take several minutes of the oracle time to align it on the sentence-to-sentence level. More details on algorithms involved in those processing steps can be found in the section below.

The processes No. 1 and 2 are concerned with "perfect" alignment on the sentence-to-sentence level and the processes No. 3 and 4 tries to deal with "perfect" alignments

on word-to-word level in the similar way. The process No. 3 tries to achieve the best possible alignment by using Viterbi alignments and decision tree which is induced from zero entropy alignments. Oracle is responsible for the review of alignments in the process No. 4. As oracle confirms/rejects some of alignments the system tries to repeat process No. 3 by using decision marks from oracle as constrains. Again, more details on those processes are given in Section 3.

With this general framework we are able to increase the oracle alignments productivity, but zero entropy alignments on the word-to-word level always will be a small amount of all alignments made on the sentence-to-sentence level. Those concerns are reflected by the process No. 5. This process tries to filter out those alignments made automatically by process No. 3 that are compatible with the oracle knowledge embedded in zero entropy alignment corpus. Current implementation of process No. 5 is very simple. It looks at alignments from process No. 3 and checks if unaligned words have sufficient support from zero entropy corpus. It means that we simply count how many times some word is unaligned and how many times it appears in zero entropy corpus as aligned word. If the ratio exceeds threshold γ then we mark this words as compatible. If after those checks all words in the whole sentence are marked as aligned then the whole sentence is marked as compatible and it is put into the database No. VI.

Finally the process No. 6 is responsible for generating translation phrases from zero entropy corpus. Its algorithm is simple: it takes all possible combinations off translation phrases that satisfies one constrain. That constrain consist of requirement of the phrase ending and beginning words being aligned and requirement that all words in the middle of the phrase are unaligned or are aligned within the translation phrase.

3. Automatic Alignments Induction

This section presents algorithms that induce alignments automatically, i.e., without the use of human as an oracle. In the section above we mentioned that the whole process of alignments creation is split into two parts. In the first subsection we present the algorithm for alignments induction on sentence punctuation marks and in the next, alignments induction on a word-to-word basis is presented. Of course, the whole book or document can be treated as one single sentence and then, algorithm for alignments induction on a word-to-word basis can be used. Nevertheless, entire book will be very long sentence and we need additional steps to improve alignments when we deal with books or documents. Additionally, there are several important differences between alignments within the sentence and alignments on sentence punctuation marks. These differences are clearly seen in Figs. 1 and 2. First, within the sentence an alignment arc can intersect as in Fig. 1 for words $f1 - e2$ and $f2 - e1$. On the other hand, alignments on sentences punctuation marks are always monotonous and without intersection (rare exceptions can be ignored). Second important difference is that within the sentence we can have aligned different number of words as in Figs. 1 for words $(f3f4) - e3$. In the case of alignments on sentences punctuation marks, we always have one-to-one alignment.

3.1. *Alignments on Sentence-to-Sentence Level (i.e., on Punctuation Marks)*

The main idea presented in this paper behind alignments problem on sentence punctuation marks is to transform it to the longest common subsequence (LCS) problem and then, after solving LCS problem we decode solution to the original form of the alignments. There are algorithms that solves LCS problem in $O(mn)$ time, where m and n are the lengths of the two symbolic sequences. Additionally to this useful interpretation we can use software of the shelf that solves LCS effectively and that can be plugged in as a component in books alignment system.

Then, the success of alignments depends on how we transform books content to string sequences on which we can run LCS algorithms. Our goal in this research project was to achieve at least 0.05 precision error rate of alignments and 0.3 of recall error rate. At the beginning of our research we tested very naive book content transformation, i.e., we removed all words from the book and then used LCS solver on remaining punctuation marks. In that case we received only 0.43 alignments precision error rate. The algorithm that we present next is a final solution to this problem of book alignments on sentence punctuation marks and it achieves about 0.02 precision error rate of alignments (Zero Entropy Corpus, 2010).

Algorithm (Alignments on Sentence-to-Sentence Level)

1. Compute words count ratio ϕ. We count how many words are in one language book (let mark it as $WordCount_e$) and how many in another (let mark it as $WordCount_f$). Then, $\phi = \frac{WordCount_e}{WordCount_f}$.
2. Transform each book in language pair to symbolic sequences (β_1 will be sequence for the first book and β_2 will be sequence for the second book) by following transformation.

```
1: β1 ⇐'' {initially we set these symbolic sequences to empty strings}
2: β2 ⇐''
3: for each book i such that 1 ⩽ i ⩽ 2 do
4:    for each sentence s in book i do
5:       for each token t in sentence s 1 ⩽ t ⩽ NumberOfToken_s do
6:          if token is word then
7:             if i = 1 OR (i = 2 AND t ⩽ NumberOfToken_s * φ ) then
8:                βi ⇐ βi +' Y'
9:             end if
10:         else if token has these separation marks .?! then
11:            βi ⇐ βi +' A'
12:         else
13:            βi ⇐ βi+''
14:         end if
15:      end for
16:   end for
17: end for
```

3. We compute alignments A_s on sequences β_1 and β_2 by finding the longest common subsequence τ_β and removing all symbols $'Y'$ from it. We run Needleman–Wunsch algorithm (1970) on β_1 and β_2 to find LCS. Needleman–Wunsch algorithm is most know in the field bio-informatics for Deoxyribonucleic acid (DNA) sequence analysis but we found it can be used successfully for punctuation mark sequences, as well. It is possible to make its usage even more straightforward if we replace punctuation marks with DNA codes (it is why we used $'Y'$ and 'A' in constructing β_1 and β_2). Then, off-the-shelf software like BioJava can be used to find LCS.
4. Next, we make another alignment A_l which we call a lexical alignment. For that we create new symbolic sequences γ_1 and γ_2 for each book in language pair (γ_1 will be sequence for the first book and γ_2 will be sequence for the second book). The following algorithm is used to create these sequences.

```
1: γ1 ⇐'' {initially we set these symbolic sequences to empty strings}
2: γ2 ⇐''
3: PhrasesSortedStack ⇐ CreatePhrasesSortedStack()
4: SortedStack1 ⇐ NewStack()
5: SortedStack2 ⇐ NewStack()
6: while PhrasesSortedStack IS NOT EMPTY do
7:    Phrase ⇐ PhrasesSortedStack.GetNext()
8:    if Phrase has no intersection conflict in PhrasesSortedStack2 then
9:       SortedStack2 ⇐ SortedStack1.Put('Y')
10:      SortedStack2 ⇐ SortedStack2.Put('Y')
11:   end if
12: end while
13: for each book i such that 1 ⩽ i ⩽ 2 do
14:   for each sentence s in book i do
15:      for each token t in sentence s 1 ⩽ t ⩽ NumberOfToken_s do
16:         if token has these punctuation marks .?! then
17:            SortedStack_i ⇐ SortedStack_i.Put('G')
18:         end if
19:      end for
20:   end for
21: end for
22: for each book i such that 1 ⩽ i ⩽ 2 do
23:   while SortedStack_i IS NOT EMPTY do
24:      γ_i ⇐ γ_i + SortedStack_i.GetNext()
25:   end while
26: end for
```

where $CreatePhrasesSortedStack()$ is a function that returns stack container of translation phrases and words found in the dictionary. Sorting is done by translation phrase length where most lengthy phrase is on the top of the stack. $SortedStack_1$ and $SortedStack_2$ are sorted by sentence sequence number and words sequence number in the sentence.

5. We proceed with alignments A_l the same as with A_s only we use sequences γ_1 and γ_2 instead of sequences β_1 and β_2.
6. We intersect A_s and A_l alignments to produce $A_F = A_s \bigcap A_l$, the final book alignments on sentence punctuation marks.

3.2. *Alignments Within the Sentence*

The algorithm above gives 0.02 precision error rate of alignments on sentence punctuation marks. The algorithm is an example of dynamic programming and it has no stochastic component because our alignments are relatively simple compared with alignments within the sentence. Then, in this section we present an alignment algorithm that gives us up to 0.23 error rate of alignments within the sentence on fictional books and up to 0.08 error rate of alignments on technical documentation (details in Section 5). Algorithm consists of two parts. At the first part of the algorithm we model alignments as hidden variable in stochastic translation process. We calculate Viterbi alignments under this stochastic model. We get only 0.25 (in the case of fictional books) error rate of alignments under this stochastic model and that is better than alignments we get under Model 4 from Brown *et al.* (1993). The second part of the algorithm consist of correcting Viterbi alignments from the first stage by applying set of induced rules from zero entropy corpus. That only slightly improves alignments. We can interpret those rules as the program that mimics human translator behavior when he corrects zero entropy alignments.

3.2.1. *Viterbi Alignments Induction by the Use of Dictionaries*

The statistical approach for the machine translation is an attractive method because it suggests fully automated process of learning and inference. Nevertheless, *"pure"* (i.e., using only data and model) statistical methods are insufficient for practical translation. Then, hybridization by incorporating linguistic knowledge into statistical framework can be the answer. In the following model the idea of factoring probability distribution of one language given another is extended by incorporating language pair dictionary, morphology analysis and semantics dictionaries.

Let e stands for English and f for foreign language sentence representation. Then the starting point for the probability representation can be expressed with alignment a as the hidden parameter:

$$P(f|e) = \sum_a P(f, a|e). \tag{1}$$

Probability $P(f, a|e)$ can be modeled in many ways. In our framework we suggest to modify Model 1 and Model 2 from Brown *et al.* (1993) in such a way that knowledge from various dictionaries will be included.

Let $e = e_1, e_2, \ldots, e_l$ be a sentence with l words and $f = f_1, f_2, \ldots, f_m$ be a sentence with m words. Let D be a bilingual dictionary consisting of the pairs of words $(\dot{e}, \dot{f})$, where $\dot{e}$ is a word in English and $\dot{f}$ is a word in a foreign language. Let

$dictranslate(\cdot)$ be a function that returns the set of the word translations from dictionary D, $lemmas(\cdot)$ be a function that returns set of lemmas of the word and $semantics(\cdot)$ be a function that returns set of hypernyms, hyponyms and synonyms of the word. Additionally, functions $semantics(\cdot)$ and $lemmas(\cdot)$ returns empty word (null) value if a given word is from close class of words (i.e., if the word is determiner, conjunction, particle etc.). We define the function $match(wordset1, wordset2)$ which returns true if there is at least one word in the set $wordset1$ and in the set $wordset2$.

Then, with sentences e and f from parallel corpora on sentence-to-sentence level we consider the set of alignments $A = \{i, j\},\ i \in [1, .., l],\ j \in [1, .., m]$ where one of the followings holds:

1. $match(dictranslate(e_i), f_j) = true$.
2. $match(dictranslate(lemmas(e_i)), f_j) = true$.
3. $match(dictranslate(e_i), lemmas(f_j)) = true$.
4. $match(dictranslate(lemmas(e_i)), lemmas(f_j)) = true$.
5. $match(dictranslate(semantics(e_i)), lemmas(f_j)) = true$.

Let d_i be the distance between English word e_i and it's aligned translation f_{a_j}. Then, probability $P(f, a|e)$ can be decomposed similarly as IBM Model 1:

$$P(f, a|e) = \epsilon \prod_{j}^{m} \frac{1}{d_{a_j}} t(f_j|e_{a_j}). \tag{2}$$

By defining indicator function $I(a_j)$ which is equals 1 when function $match(\cdot, \cdot)$ is equal to $true$ and 0 otherwise we write

$$P(f|e) = \epsilon \sum_{a_1}^{l} \cdots \sum_{a_m}^{l} \prod_{j}^{m} \frac{1}{d_{a_j}} t(f_j|e_{a_j}) I(a_j),$$

subject to

$$\sum_{f} t(f|e) = 1.$$

Unconstrained auxiliary function

$$h(t|\lambda) = \epsilon \sum_{a_1}^{l} \cdots \sum_{a_m}^{l} \prod_{j}^{m} \frac{1}{d_{a_j}} t(f_j|e_{a_j}) I(a_j) - \sum_{e} \lambda_e \Big(\sum_{f} t(f|e) - 1 \Big).$$

The partial derivative of h with respect to $t(f|e)$

$$\frac{\partial h}{\partial t(f|e)} = \epsilon \sum_{a_1}^{l} \cdots \sum_{a_m}^{l} \sum_{j}^{m} \delta(f, f_j) \delta(e, e_{a_j}) t(f|e)^{-1} \prod_{k}^{m} \frac{1}{d_{a_k}} t(f_k|e_{a_k}) I(a_k) - \lambda_e.$$

If this partial derivative is zero then

$$t(f|e) = \lambda_e^{-1}\epsilon \sum_{a_1}^{l} \cdots \sum_{a_m}^{l} \sum_{j}^{m} \delta(f, f_j)\delta(e, e_{a_j}) \prod_{k}^{m} \frac{1}{d_{a_k}} t(f_k|e_{a_k})I(a_k). \quad (3)$$

Alignments A that maximizes $P(f, a|e)$ are called Viterbi alignments. An iterative estimation of Viterbi alignments by using (3) we mark as $V_1(f|e)$. Next, we treat $V_1(f|e)$ as a constrain and we search to find the remaining words translation probabilities $t(f|e)$ using Model 2 (Brown *et al.*, 1993) and then the new evaluation of $V_2(f|e) = \max P(A)$ can be received by preserving $V_1(f|e)$ as the constrain.

The $V_2(f|e)$ is an optimal solution under the Model 3. It incorporates our knowledge in the form of dictionaries but it is a solution without any oracle involvement. On the other hand, oracle has a knowledge that is not incorporated into parallel corpus and then he can correct $V_2(f|e)$ alignments and produce $V_3(f|e)$ alignments. The question is how to change the Model 3 so that alignments $V_2(f|e)$ and $V_3(f|e)$ will be as close as possible. In the next section we suggest an algorithm on how to improve alignments by using knowledge from oracle in the form of zero entropy corpus.

4. Alignments Induction Based on Oracle Knowledge

4.1. *Alignments on Sentence-to-Sentence Level*

The algorithm that we suggested for alignments induction on sentence punctuation marks gives 0.02 error rate. We found that in practice, there is no need for oracle involvement if we use suggested algorithm. Nevertheless the following rule of thumb improves alignments if we want to improve recall error rate:

1. Get sentence-to-sentence level alignments $A_F = A_s \bigcap A_l$ by using the algorithm that we described above.
2. Select aligned sentences pair which has longest normalized difference in length between the sentence and its translation.
3. Ask the oracle to get at least one correct alignment in select aligned sentences pair.
4. Start from Step 1 by using alignments edited by oracle as constrains.

We can confirm that by using this simple rule we can improve recall error rate by using only several iterations of it. In general, we found that an ordinary book of about 500 kilobytes can be aligned in one hour if we use the approach that we just described and we can achieve error up to 0.01.

4.2. *Alignments Within the Sentence*

In the previous section we presented an algorithm for the case when alignments are treated as hidden parameters. Sometimes Viterbi alignments $V_2(f|e)$ can match exactly the choice made by oracle but sometimes oracle can choose to correct Viterbi alignments

Table 2
Set of actions that oracle can take and their labels used as classification labels for decision tree

	Action label	Description
1	CLEAR	Clear assigned alignment, i.e., oracle can decide that assigned alignment position is wrong and clear it.
2	JOINT	Joint nearby standing words into single phrase.
3	COREFERENCE	Set coreference mark, i.e., oracle can decide that word/phrase expression in a sentence and its translation refer to the same thing but aren't translations of each other.
4	CONFIRMATION	Set a confirmation mark, i.e., oracle agrees with alignment made by the stochastic optimization procedure presented above.

at some sentence positions. In that case we would like to code some of the oracle knowledge into a separate model. We can use any model suited for fully observable data because zero entropy corpus represents data that can be interpreted as data without any hidden parameters. In this project we decided to use decision tree as an additional model to correct Viterbi alignments. The main reason for that is that decision trees are well suited for human comprehension. Next, we present basic steps for decision tree construction that we used in this research project.

There are many algorithms on how to build a decision tree but we decided to use C4.5 (Quinlan, 1993), one of the most popular model for decision tree inference. Additional reason for use of C4.5 is off-the-shelf software that implements this algorithm. We used WEKA (Hall *et al.*, 2009) data mining software due to its integration flexibility. Then, what we need is just a data table in which all decisions from the oracle are coded. The structure of that table consist of one classification attribute which represents action labels that human translator can do when he corrects Viterbi alignments $V_2(f|e)$ (see Table 2) and the remaining attributes are feature functions that reflects the knowledge about language pair in our corpus (see Table 3).

The feature functions in the Table 3 we divided into two sets: one set reflects our knowledge about language morphology and the second reflects such knowledge as transaction probability and positions in the sentence. Such division was made because we wanted to test the impact of morphology knowledge on alignments quality.

The integration of the decision tree into the whole alignment framework is straightforward. At first, we use only automatic alignments build upon the stochastic model presented above. Then, when a significant portion of data are added (next chapter formalize this) we rebuild the decision tree based on all data in zero entropy corpus. Then, when we have the decision tree the alignments algorithm run as follows:

1. Get $V_2(f|e)$ alignments.
2. Run decision tree to imply constrains.
3. If decision tree has made some suggestions then create $V_3(f|e)$ alignments by recalculating $V_2(f|e)$ with constrains implied by decision tree.

Table 3
Set of feature functions that reflects the knowledge about language pair in the corpus.

	Description
Set one	
1	Label that identifies Noun or Verb phrase.
2	English Part-of-speech (POS) tag.
3	Lithuanian Part-of-speech (POS) tag.
Set two	
4	Is it named entity.
5	Difference between positions in the sentence.
6	Length of sequential match.
7	Length of the phrase.
8	Translation probability.

Experimental part shows that decision tree slightly improves alignments quality.

5. Evaluation

We have evaluated suggested method on documents from the Official Journal of the European Union (2010) and on translated fictional books (Zero Entropy Corpus, 2010). In both cases we produced hand alignments for comparison with alignments generated automatically. The automatically generated and hand-made alignments were in close agreement on documents from the Official Journal of the European Union and more discrepancies were found in the case of OCR scanned fictional books. In both cases we found that the biggest difference between hand-made alignments and the automatically generated alignments are affected by errors made in translation (like missed paragraph in translation) or too free interpretation of the text by the translator. Then, in addition to the translation memory, suggested algorithms can be used to detect errors that occur during the translation process.

We aligned 7854 sentences by hand from the Official Journal of the European Union document base on sentence-to-sentence level and 8317 sentences from fictional OCR scanned books. The main goal of the experiment on that alignment level was to measure the dependency of precision and recall error rates from the size of the text that is supplied to the system in one portion. As a benchmark to compare our method with the state-of-the-art that is currently available in research papers, we used the method described in Moore (2002; we named it in our experiment as "Hybrid Length + Words").

In order to test the alignments on the word-to-word level we built English-Lithuanian corpus aligned by hand of 248 389 words. We put it on the Internet in order to show results and for reuse in others research projects (for Internet URL use Zero Entropy Corpus, 2010). It is important to note that for documents from the Official Journal of the European Union we aligned by hand only 19 587 words and the rest are alignments from

scanned fiction books. The reason for this was that the creation of technical document alignments is a tedious job and we aligned only what was needed for our experiment. On the other hand, we found that alignments of fiction books can create an involvement that is comparable to involvement created by computer games. It is important aspect in our research because we think that books alignments can be created on a voluntary base like other Internet based collaboration on voluntary base (i.e., Wikipedia).

There is a question how to evaluate the efficiency of suggested method. Some subjectivity is unavoidable because the method involves human decisions on what is correct. Nevertheless, we assume that human translator is the oracle who knows the exact answer for the alignments and the following simple metrics were used:

$$Err_s = \frac{1}{S}\sum_{i=1}^{S} MismatchCount_i, \tag{4}$$

$$Err_a = \frac{1}{S}\sum_{k=1}^{S} \frac{MismatchAlignments_k}{N_k}, \tag{5}$$

where S is the number of sentences, $MismatchCount_i$ is an indicator which is equal to 1 if sentence i has been misaligned on sentence-to-sentence level, $MismatchAlignments_k$ is number of alignments that have been misaligned in the sentence k and N_k is the number of correct alignments (verified and corrected by human editor) in the sentence k. Those metrics are precision error of the method. For the recall error we suggest following:

$$RecErr_s = \frac{1}{S}\sum_{k=1}^{S} MissedCount_i, \tag{6}$$

$$RecErr_a = \frac{1}{S}\sum_{k=1}^{S} \frac{MissedAlignments_k}{N_k}, \tag{7}$$

where S is the number of sentences, $MissedCount_i$ is an indicator which is equal to 1 if sentence i has been missed (not spotted) on sentence-to-sentence level and $MissedAlignments_k$ is number of alignments that have been missed (not spotted) in the sentence k and N_k is the number of correct alignments in the sentence k. Precision metric for sentences alignment is similar to metric used in Brown *et al.* (1991) and Moore (2002) and recall error metric for sentences alignment is similar to metric used in Moore (2002).

5.1. *Alignments on Sentence-to-Sentence Level*

In order to measure alignments on the sentence-to-sentence level we decided to test two groups of documents: technical documentation where translation has been done by some computer aided software and the second group is fictional books that represents the "hard" case. In the recent paper on sentence-to-sentence level alignments (Moore, 2002) author reported low error rate for technical documentation (we call their method "Hybrid Length + Words" in Table 4). We thought that it is important to compare our method with

Table 4
Euro documents sentence-to-sentence alignment error rate for different methods

Alignment method	Size				
	50	100	500	2000	7854
Err_s					
Hybrid length + words	0.02	0.04	0.025	0.024	0.027
Punctuation marks LCS	0.04	0.06	0.034	0.042	0.068
Matched words LCS	0.02	0.04	0.028	0.026	0.028
Intersection LCS	0.02	0.02	0.020	0.018	0.020
$RecErr_s$					
Hybrid length + words	0.02	0.02	0.014	0.029	0.03
Punctuation marks LCS	0.04	0.07	0.044	0.042	0.052
Matched words LCS	0.02	0.024	0.028	0.026	0.026
Intersection LCS	0.04	0.07	0.044	0.042	0.052

the method suggested in that paper on the similar kind of documents. The results that we received are shown in the Table 4.

In the case of the fictional books we done a separate test and in Table 5 we can see the results. Because the translation of the fictional books differs from the translation of technical documents we can expect an increase in error rates. But as we see from Table 5 the increase in error rates is not very big. Additionally, we can see that in our suggested method increase in error rates is smaller than in the case of "Hybrid Length + Words", the method reported in Moore (2002). We investigated the phenomena of it by manually inspecting alignment errors and we come to conclusion that the suggested method performs slightly better because of very big dictionary we compiled for the alignment process. Additionally, we can note that recall error rate relatively increased more than precision error when we compare technical and fictional text. We can explain this by recalling that suggested method consists of the intersection of two different alignments. This is hardly a surprise, but we can note that recall error rate is not as important as precision error rate in the case of sentence-to-sentence alignments. In our experiments we found that human editor can tolerate lower recall if the is precision is increased.

5.2. *Alignments Within the Sentence*

The extensive study of various methods for computing word alignments using statistical or heuristic models has been presented in Och and Ney (2003). The authors of this paper have concluded that the best model for the alignments is their Model 6 which consist of log-linear combination of HMM and Model 4 from Brown *et al.* (1993). The second important conclusion that was made by the authors of that paper was that conventional bilingual dictionary and use of word classes have only a minor effect on the quality of alignments. Method that is suggested in this paper heavily depend on dictionaries and morphological analysis and that's why we decided to test in this research project the

Table 5
Fictional literature sentence-to-sentence alignment error rate for different methods

Alignment method	Size				
	50	100	500	2000	8317
Err_s					
Hybrid length + words	0.06	0.07	0.05	0.042	0.047
Punctuation marks LCS	0.08	0.08	0.052	0.054	0.049
Matched words LCS	0.04	0.04	0.04	0.038	0.036
Intersection LCS	0.02	0.02	0.024	0.025	0.026
$RecErr_s$					
Hybrid length + words	0.04	0.04	0.024	0.027	0.039
Punctuation marks LCS	0.08	0.07	0.064	0.042	0.066
Matched words LCS	0.02	0.03	0.024	0.029	0.022
Intersection LCS	0.02	0.04	0.02	0.023	0.027

Table 6
Euro documents word-to-word alignments error rate dependency on the size of dictionary and conventional bilingual corpus

Alignment method	Err_a	$RecErr_a$
IBM M4 corpus size (1M words)	0.15	0.12
IBM M4 (10M words)	0.11	0.09
Lexical EM (1M words) (dic size 10 000)	0.275	0.219
Lexical EM (10M words) (dic size 10 000)	0.273	0.213
Lexical EM (1M words) (dic size 200 000)	0.144	0.157
Lexical EM (10M words) (dic size 200 000)	0.13	0.10
Lexical EM (1M words) (dic size 800 000)	0.078	0.075
Lexical EM (10M words) (dic size 800 000)	0.074	0.073
Lexical EM + Dtree (1M words) (dic size 800 000)	0.074	0.072
Lexical EM + Dtree (10M words) (dic size 800 000)	0.071	0.072

dependency of alignments quality from the size of dictionary and the size of conventional bilingual corpus. Those dependencies are shown in the Tables 6 and 7. In Tables 6 and 7 model "IBM M4" is the Model 4 from Brown *et al.* (1993), "lexical EM" stand for the model described in Section 3.2 of this paper and "Lexical EM + Dtree" stand for the model described in Section 4.

In the Table 6 we can see that dictionary size must be at least 200 000 entries in order to achieve error rates that are close to error rates from "IBM M4" model. We can see that dictionary of 800 000 entries (about 360 000 word-to-word translations and the rest short phrases) reduces precision and recall error rates and which now are lower than error rates achieved with IBM M4 model. We can see that if we add decision tree (model "Lexical EM + Dtree") then only a slight improvement in error rates is achieved.

Table 7

Fiction books word-to-word alignments error rate dependency on the size of dictionary and conventional bilingual corpus

Alignment method	Err_a	$RecErr_a$
IBM M4 corpus size (1M words)	0.34	0.28
IBM M4 (10M words)	0.28	0.25
Lexical EM (1M words) (dic size 10 000)	0.489	0.425
Lexical EM (10M words) (dic size 10 000)	0.474	0.413
Lexical EM (1M words) (dic size 200 000)	0.331	0.294
Lexical EM (10M words) (dic size 200 000)	0.321	0.288
Lexical EM (1M words) (dic size 800 000)	0.245	0.211
Lexical EM (10M words) (dic size 800 000)	0.239	0.205
Lexical EM + Dtree (1M words) (dic size 800 000)	0.226	0.202
Lexical EM + Dtree (10M words) (dic size 800 000)	0.224	0.201

Interesting results are presented in Table 7. There are presented error rates for aligned fiction books. We can see that error rates are much bigger than in the former case of technical documents. Additionally, we can see that the impact of big dictionary on error rates improvement is bigger as well. Again, the impact of decision tree is minor.

5.3. *Process of Alignments with Humans Involvement*

Next experiment that we conducted was a subjective one but it reveals some important aspects of the model presented in this paper. Our goal was to establish an error rate of the oracle. Error rate of the oracle sound controversially as we assume that oracle knows the exact answer. Nevertheless human translator can make an error and our goal was to determine the error rate dependency from the experience of the oracle. The importance of this experiment is in the fact that if we want to produce perfectly aligned corpus on the big scale then we need to implement its production by using social collaboration framework (i.e., like Wikipedia). This experiment allows us empirically to establish the necessary amount of experience human translator needs to produce alignments with low error rate. Additionally, we conducted experiment on two types of corpus: technical documentation and fictional books. After this experiment the error rates have been evaluated by a professional translator.

For the experiment we have chosen 8 volunteers with good knowledge of English and Lithuanian but without any experience with the system. We divided them into two equal groups one of them used interface of alignments system but without suggestions from the system. Another group reviewed the alignments that have been produced by the system automatically and if necessary, corrected them. Tables 8 and 9 presents the results. We can see from the tables that productivity can increase about four times when alignments are made automatically and user only reads them and corrects them if necessary.

We can see from Table 7 that there is required only about 1000 aligned words to achieve low precision error rate and after which improvement becomes negligible. From

Table 8
Alignment error rate of the oracle and it dependency from experience (hundreds of sentences) (Err_a metric)

Euro doc. (no support from the system)					
Size (numb. of words)	0	432	1247	2811	3291
Err_a	0.0143	0.008	0.0074	0.0079	0.0072
$RecErr_a$	0.0274	0.0253	0.0259	0.0242	0.0221
Euro doc. (by using the system)					
Size (numb. of words)	0	625	1358	2942	4537
Err_a	0.0033	0.0028	0.0024	0.002	0.0018
$RecErr_a$	0.0048	0.0032	0.0039	0.0035	0.0036
Fiction books (no support from the system)					
Size (numb. of words)	0	617	1287	2876	4522
Err_a	0.024	0.029	0.021	0.027	0.017
$RecErr_a$	0.049	0.043	0.037	0.036	0.03
Fiction base (with system)					
Size	0	10 243	20 189	30 334	40 072
Err_a	0.0042	0.0024	0.0027	0.0028	0.0032
$RecErr_a$	0.0035	0.0033	0.0037	0.0034	0.0031

Table 9
Time words/per secong

Fiction books (no support from the system)					
Size (numb. of words)	0	617	1287	2876	4522
Time (word per hour)	954	927	892	948	914
Fiction base (with system)					
Size (numb. of words)	0	10 243	20 189	30 334	40 072
Time (word per hour)	2172	3247	3426	3591	3877

the same table we can see that recall error rate achieves its limit only after more alignments are made, i.e., we need about 10 000 aligned words. From Table 9 we can see that alignments time improves until about 1000 words and then it remains almost the same.

6. Conclusions

In this paper we suggested the method to ease human translator work when he creates translated book alignments. Additionally, we presented empirical evidence that suggested method can increase productivity of alignments creation as well as an accuracy of it. Presented algorithms for alignments creation heavily rely on knowledge base and as has been demonstrated, they can create high quality alignments if dictionaries are big enough.

The main motivation for the research project which has been presented in that paper, was in fact that there are thousands of translated books that can be used to improve automatic translation systems. In the future, we hope to collect corpus of several million words and develop algorithms for such improvement.

References

Barrachina S., Bender, O., Casacuberta, F., Civera, J., Cubel, E., Khadivi, S., Lagarda, A., Net, H., Tomas, J., Vidal, E., Vilar, J.M. (2009). Statistical approaches to computer-assisted translation. *Computational Linguistics*, 35(1), 3–28.

Brown, P.F., Lai, J.C., Mercer, R.L. (1991). Aligning sentences in parallel corpora. In: *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics*, pp. 169–176.

Brown, P.F., Della Pietra, V.J., Della Pietra, S.A., Mercer, R.L. (1993). The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19(2), 263–311.

Chen S.F. (1993). Aligning sentences in bilingual corpora using lexical information. In: *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, pp. 9–16.

Dzemyda, G., Petkus, T. (2001). Application of computer network to solve the complex applied multiple criteria optimization problems. *Informatica*, 12(1), 45–60.

Dzemyda, G., Sakalauskas, L. (2009). Optimization and knowledge-based technologies. *Informatica*, 20(2), 165–172.

Dzemyda, G., Sakalauskas, L. (2011). Large-scale data analysis using heuristic methods. *Informatica*, 22(1), 1–10.

Gale, W.A., Church, K.W. (1993). A program for aligning sentences in bilingual corpora. *Computational Linguistics*, 19(1), 75–102.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H. (2009). The WEKA data mining software: an update. *SIGKDD Explorations*, 11(1).

Holland, R.C.G., Down, T.A., Pocock, M., Prlic, A., Huen, D., James, K., Foisy, S., Dräger, A., Yates, A., Heuer, M., Schreiber, M.J. (2008). BioJava: an open-source framework for bioinformatics. *Bioinformatics*, 24(18), 2096–2097.

Laukaitis, A., Vasilecas, O. (2007). Asymmetric hybrid machine translation for languages with scarce resources. In: *Proceedings of 8th International Conference on Intelligent Text Processing and Computational Linguistics CICLing, LNCS*, Vol. 4394, pp. 397–408.

Laukaitis, A., Vasilecas, O. (2008). Multi-alignment templates induction. *Informatica*, 19(4), 535–554.

Lipeika, A. (2010). Optimization of formant feature based speech recognition. *Informatica*, 21(3), 361–374.

Marcu, D., Wong, W. (2002). A phrase-based, joint probability model for statistical machine translation. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Philadelphia. pp. 87–99.

Moore, R.C. (2002). Fast and accurate sentence alignment of bilingual corpora. In: *Proceedings of the 5th Conference of the Association for Machine Translation in the Americas, LNAI*, Vol. 2499, pp. 135–144.

Needleman, S.B., Wunsch, C.D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3), 443–453.

Och, F.J., Ney, H. (2002). Discriminative training and maximum entropy models for statistical machine translation. In: *The 40th Annual Meeting of the Association for Computational Linguistics*. pp. 295–302.

Och, F.J., Ney, H. (2003). A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1), 19–51.

Och, F.J., Ney, H. (2004). The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4), 417–449.

Official Journal of the European Union. Accessed July 20010. `http://eur-lex.europa.eu`.

Quinlan, J.R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann, Los Altos.

Pajarskaite, G., Griciute, V., Raskinis, G., Kuper, J. (2004). Designing HMM-based part-of-speech tagger for Lithuanian language. *Informatica*, 15(2), 231–242.

Skripkauskas, M., Telksnys, L. (2006). Automatic transcription of lithuanian text using dictionary. *Informatica*, 17(4), 587–600.

Toutanova, K., Ilhan, H.T., Manning, C.D. (2003). Extensions to HMM-based statistical word alignment models. In: *Proceedings of Empirical Methods in Natural Langauge Processing*, Philadelphia. pp. 87–94.

Vaiciunas, A., Kaminskas, V., Raskinis, G. (2004). Statistical language models of lithuanian based on word clustering and morphological decomposition. *Informatica*, 15(4), 565–580.

Watanabe, T., Imamura, K., Sumita, E. (2002). Statistical machine translation based on hierarchical phrase alignment. In: *Proceedings of the 9th International Conference on Theoretical and Methodological Issues in Machine Translation*, Japan, pp 188–198.

Zero Entropy Corpus. Research project. Accessed July 2010.
`http://e-stud.vgtu.lt/?p=0.72323&preview&lang=en`

A. Laukaitis has graduated from Vilnius University Faculty of Physics in 1992. He received the PhD degree from the Institute of Mathematics and Informatics, Vilnius in 2002. He is an associated professor of the Information Systems Department of Vilnius Gediminas Technical University. His research interests include text mining, natural language interfaces, machine translation systems and knowledge management.

O. Vasilecas has graduated from Kaunas University of Technology in 1967. He received the PhD degree from Vilnius University, Vilnius in 1979. He is a professor of Information System Department, head of the Information System Scientific Laboratory of the Vilnius Gediminas Technical University (VGTU) and a professor of the Computer Science Department of Klaipė da University. His research interests include intellectual information system engineering, business, information and program system engineering, knowledge reflection and concept modeling, mathematical modeling of engineering processes.

D. Plikynas PhD from Vilnius University, Vilnius in 2003. He is a professor of Vilnius Management Academy, head of the Artificial Intelligence Laboratory. Research interests include, but are not limited to various computational Intelligence and complexity approaches (like neural networks, fuzzy systems, genetic algorithms, chaos&complexity theory, evolutionary dynamics, artificial life, social simulations etc) used for forecasting, recognition, modeling-simulating in social domain.

R. Laukaitis PhD from Vilnius Gediminas Technical University, Vilnius in 2001. Research interests include applied artificial intelligence and its use for social science.

Pusiau automatinis lygiagretaus tekstyno sudarymas su entropija, lygia nuliui

Algirdas LAUKAITIS, Olegas VASILECAS, Ričardas LAUKAITIS, Darius PLIKYNAS

Šiame straipsnyje pristatomas metodas leidžiantis sukurti tikslius lygiagrečius dvikalbius tekstynus su mažomis žmogaus rankinio darbo sąnaudomis. Straipsnyje žmogus-vertėjas yra traktuojamas kaip orakulas, kuris žino tiksliai, kaip reikia anotuoti tekstyną, o sistemos tikslas – minimizuoti šio orakulo panaudojimą. Tai mes išmatuojame matuodami greitį su kuriuo žmogus sukuria tikslius lygiagrečius tekstynus. Žodis „tikslus“ tekstyno kūrimo kontekste naudojamas norint pabrėžti, kad orakulas anotuoja tekstyną su tikimybe lygia 1, t.y. be klaidų. Šiame straipsnyje pateikiami anotavimo algoritmai tiek sakinio lygmenyje, tiek žodžio lygmenyje, be to pasiūlytas metodas leidžia integruoti šiuos du algoritmų tipus į vieningą sistemą. Pasiūlytas metodas nepriklauso nuo kalbų pasirinkimo, tačiau šiame straipsnyje mes pateikiame eksperimentus, kurie buvo atlikti su anglų–lietuvių kalbų tekstynais. Straipsnyje parodome, kad pasiūlytas metodas ypač naudingas, kai mėginama anotuoti verstas grožinės literatūros knygas.