José Aires Gomes Camacho

Licenciado em Engenharia Informática

Genuine Phrase-Based Statistical Machine Translation with Supervision

Dissertação para obtenção do Grau de Doutor em
Engenharia Informática

Orientador: José Gabriel Pereira Lopes, Doutor, Universidade Nova de Lisboa

Júri:

Presidente: Prof. Doutor Luís Manuel Marques da Costa Caires

Arguentes: Prof. Doutor Pavel Bernard Brazdil
Prof. Doutora Maria Teresa Rijo da Fonseca Lino

Vogais: Prof. Doutor António Manuel Horta Branco
Prof. Doutora Belinda Mary Harper Sousa Maia
Prof. Doutora Irene Pimenta Rodrigues
Prof. Doutor Nuno Miguel Cavalheiro Marques
Prof. Doutor João Miguel da Costa Magalhães

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Abril, 2015

Genuine Phrase-Based Statistical Machine Translation with Supervision

I dedicate this hard work to my dear family.

Acknowledgments

I want to thank my supervisor for his invaluable guidance, my university, Universidade Nova de Lisboa, for providing the conditions to carry out my work, the foundation, FCT/MCTES, for supporting me with a Ph.D. grant, and my coleagues for believing in me.

I also wanto to thank my dear family for supporting me, believing in me and keeping their faith in me, through all the ups and downs I have faced through this very long journey.

Abstract

This thesis addresses mainly two issues that have not been addressed in Statistical Machine Translation. One issue is that even though research has been evolving from word-based approaches to phrase-based ones, because words were consistently found to be inappropriate translation units, the fact is that words are still considered in the composition of phrases, either to determine translation equivalents or to check language fluency. Such consideration might result in the attempt of establishing relations between words within a phrase translation equivalent even when sometimes its phrases should be considered as a whole. Attempts to further partition such phrases would produce incorrect translation units that would introduce unwanted noise in the translation process. Besides, the internal fluency of an identified multi-word phrase should not require checking. As such, phrases should indeed be considered units, avoiding incorrect translation equivalents that might be identified from their partition, as well as only considering the fluency of a phrase with other phrases and not within the phrase itself. The other issue is that supervision, in the form of translation lexica, is generally overlooked, with SMT research focusing mainly on the identification of translation units without any human intervention and without considering already known translation units. As such, no importance has been attributed to the inclusion of verified lexica, with only some rarely used dictionaries to score translation candidates and not really as a source of translation units. Indeed, translation equivalents should be memorized, checked and used as a source of translation units, avoiding the need to keep identifying the same translation units, in particular if those are frequently used. This Thesis presents a truly Phrase-Based approach to SMT, using contiguous and non-contiguous phrases, along with Supervision, in which phrases are not divided and verified lexica is built, kept and used to propose translations of complete sentences.

Keywords: Statistical Machine Translation, Phrase-Based, Non-contiguous Equivalents, Suffix Arrays, Verified Lexicon.

Resumo

Esta Tese trata principalmente dois problemas que não foram devidamente tratados na Tradução Máquina Estatística. Um deles é que, apesar de a investigação ter passado a considerar multi-palavras como unidades e não apenas palavras, por se constatar que as palavras não constituem unidades de tradução adequadas, o facto é que as palavras são ainda consideradas na composição de multi-palavras, quer em equivalentes de tradução, quer na verificação de fluência de língua. Tal consideração pode resultar na tentativa de estabelecer relações entre palavras mesmo quando por vezes as multi-palavras devem ser consideradas como um todo. Particionar tais multi-palavras produzirá unidades de tradução incorrectas que introduzirão ruído no processo de tradução. Além disso, a fluência interna dessas multi-palavras não deve ser verificada. Como tal, multi-palavras deverão ser consideradas unidades, evitando equivalentes de tradução incorrectos resultantes da sua partição, e considerando apenas a fluência de multi-palavras com outras e não a sua fluência interna. O outro problema é que a supervisão, na forma de léxicos de tradução, é geralmente desvalorizada, com a investigação em SMT a ser focada na identificação de unidades de tradução sem qualquer intervenção humana e sem considerar unidades de tradução já conhecidas, com apenas algumas utilizações de dicionários para atribuir valores a candidatos de tradução e não propriamente como fonte de unidades de tradução. Efectivamente, equivalentes de tradução devem ser memorizados, verificados e utilizados como fonte de unidades de tradução, evitando a necessidade da sua repetida identificação, em particular os frequentemente usados. Esta Tese apresenta uma abordagem de SMT verdadeiramente baseada em multi-palavras, tanto contíguas como descontíguas, em conjugação com supervisão, na qual as multi-palavras não são divididas e o léxico verificado é construído, mantido e utilizado para propor traduções de frases completas.

Palavras-chave: Tradução Máquina Estatística, Baseada em Elementos Subfrásicos, Vectores de Sufixos, Equivalentes Descontíguos, Léxico Verificado.

Contents

List of Tables

List of Figures

List of Equations

1 INTRODUCTION

The work underlying this thesis began mainly within the framework of project PATRAS (contract POSC/PLP/61520/2004) where the main goal was to develop methodologies supported on Suffix Arrays with the purpose of extracting phrase[1] translations from aligned parallel texts[2]. The good results obtained provided the motivation to use those extracted phrase translations to produce translations of complete texts, this way leading research towards a Machine Translation (MT) system, a research mainly carried out under project ISTRION (contract PTDC/EIA-EIA/114521/2009), improving the developed translation work and extensively experimenting it on both translation directions of eight language pairs. The resulting MT system received the name of Transtor and implements a semi-supervised, phrase-based, statistical approach to machine translation that, as it will be shown in this thesis, is clearly distinguished from existing Statistical Machine Translation (SMT) approaches mainly developed after 1990 (Brown et al., 1990).

1.1 RESEARCH FRAMEWORK AND MOTIVATION

The research carried out on earlier projects (DIXIT, contract PRAXIS 2/2.1/TIT/1670/95; TRADAUT-PT, European contract MLIS-4005 TRADAUT-PT 26192; ASTROLABIUM, European contract MOBI-CT-2003-003344; PATRAS, contract POSC/PLP/61520/2004; and VIP-ACCESS, contract PTDC/PLP/72142/2006) led the research group I integrate to develop innovative, language independent, text mining procedures applied to raw text. Those developed procedures include:

- alignment of parallel texts, breaking those texts into segments that should continue to be translations of each other (Ribeiro, Lopes and Mexia, 2000a), (Ribeiro, 2002), (Gomes, Aires and Lopes, 2009);
- extraction of word and phrase translations (Ribeiro, Lopes and Mexia, 2000d), (Aires, Lopes and Gomes, 2009), (Gomes, Aires and Lopes, 2009);
- extraction of multi-word terms (Silva et al., 1999), (Aires, Lopes and Silva, 2008);
- clustering of documents (Silva et al., 2001), (Peleja, Silva and Lopes, 2011);
- identification of the language in which a document is written (Silva and Lopes, 2006);

[1] A phrase, as is usual in Machine Translation literature, is used as a string of words, independently of being or not a phrase in linguistically precise terms. From here onwards this designation, "phrase" will be used as a string of words.

[2] Two texts in different languages are parallel if they are a translation of each other.

- identification of phrases having similar meanings, because they occur in similar local lexical contexts (Gamallo, Agustini and Lopes, 2005), (Gamallo, Agustini and Lopes, 2008), (Casteleiro, Lopes and Silva, 2014); and
- identification of key terms in documents (Silva and Lopes, 2009), (Teixeira, Lopes and Ribeiro, 2013).

As a result of this wide range of activities, Machine Translation was set in the ISTRION project (contract PTDC/EIA-EIA/114521/2009) as one of our research group's leading goals. I, myself, took on the task of building a new approach to Machine Translation, following the preliminary results I had obtained in this area in the framework of PATRAS project, following a different and new approach to Phrase-Based Statistical Machine Translation (PBSMT) regarding the state-of-the-art in this research area (Och and Ney, 2004).

A first difference was introduced by assuming that word and phrase translations extracted from aligned parallel corpora should be validated, thus introducing a first level of supervision. We verified that alignment precision improved when correctly acquired term translations were iteratively reused in subsequent parallel corpora alignment. Such precision improvement, for the alignment of parallel texts belonging to the Portuguese (PT) English (EN) pair, ranged from a maximum of 75.5% precision, when absolutely no translation knowledge was used (Darriba Bilbao, Lopes and Ildefonso, 2005), to 84.5% precision (Gomes, Aires and Lopes, 2009) at a very early stage of the mentioned reuse iteration. The precision improvement of the alignment also contributes to the precision improvement of extracted unknown term translations, even for very low occurrence frequencies (Aires, Lopes and Gomes, 2009).

As a consequence of using this perspective, in which the automatic phrase translation extraction stage of the training process of a machine translation engine involves some remote supervision (prior to the realignment of the parallel corpora used to train our system), we managed to develop a competitive MT system capable of beating Moses[3], which is the state-of-the-art system for Statistical Machine Translation. Such conclusion was drawn after training Transtor and Moses on the same parallel corpora and realizing that Transtor had an average advantage of 12 BLEU points on both directions on the 8 analyzed language pairs between Portuguese, English, French, Spanish, and German (check Chapter 4 for a detailed analysis of results). The German-French and French-Spanish language pairs were not analyzed, but three very difficult language

[3] http://www.statmt.org/moses/

pairs which comprise the German language (German-English, German-Spanish and German-Portuguese) were included in the tests.

The idea of introducing supervision on phrase translations extraction resulted from the fact that the SMT approach does not include a memory of what has been learned earlier, and always learns everything from scratch. Indeed, it behaves as a ship commander who never knows where the borders/extremes of his ship are and needs some kind of device to measure/calculate from scratch where they are, whenever necessary. And this also applies to any frequently visited port. He never knows where wharf and piers are. He needs to recalculate everything prior to docking. Actually, it learns but everything is fuzzy. The importance of supervision is confirmed by the considerable improvement on translation quality produced by Transtor, our MT system.

A second difference is a direct consequence of our way of addressing alignment. In SMT, the alignment of parallel corpora requires parallel texts to be aligned at the sentence level. Then, for each pair of parallel sentences, word frequencies are determined, as well as the number of co-occurrences of words in each parallel sentence, for the whole parallel corpora. Each word in one sentence can be the translation of any word of the parallel sentence. This way the probability of each word being a translation of any word of the parallel corpora is determined. The alignment is then defined as the probability of a word being the translation of other words in the other language. This is the basis for word-based SMT. The evolution of word-based alignment into phrase-based alignment is, in most approaches, still dependent on word-based alignment that is further refined to produce alignments at the phrase level (Tillmann, 2003), (Zhang, Vogel and Waibel, 2003), (Zhao and Vogel, 2005), (Zhang and Vogel, 2005), (Setiawan, Li and Zhang, 2005).

Ever since we started working on this area, our perspective of alignment was a bit different. Given two parallel texts, we were interested in dividing them into parallel segments (composed of one or more tokens) for which there is a high degree of certainty that they are translations of each other, using those parallel segments as alignment anchors. A first approach to determine such certainty degree was accomplished by linear regression and applying statistical filters to remove outliers (Ribeiro, Lopes and Mexia, 2000a). The procedure was then recursively applied to each candidate of parallel segments until no more alignments could be discovered. Considering that some of the statistical filters were too strict and removed far more alignment anchors than necessary, an improvement was later introduced which consisted in replacing one of the statistical filters by an algorithmic filter (Ildefonso and Lopes, 2005). Further im-

provement led to using bilingual lexica of phrase translations that were automatically extracted and manually validated (Gomes, Aires and Lopes, 2009), an approach that keeps improving, particularly in the extraction of high quality phrase translations.

The major difference between the two methods of addressing alignment is related to the determination of word or phrase translation probabilities. SMT starts determining word translation probabilities for the whole parallel corpora. We start dividing the parallel corpora into candidates of parallel segments currently using a bilingual phrase lexicon. The determination of word and phrase translation probabilities is made later, at a stage where phrase translations are spatially located and trapped, thus becoming much more focused and computationally less heavy.

A third difference is a direct consequence of both differences mentioned above: the periodic supervision of term translation extraction enables us to take known phrase translation equivalents as possibly adequate anchors and filter out alignment outliers, this way reducing phrase realignment imprecision and enabling us to give more credit to obtained alignments. As a result, the extraction of new phrase translation equivalents with very low frequencies becomes more precise and its use in translation situations enables the achievement of better quality translations. It is not by chance that in SMT, either word-based or phrase-based, alignment is taken as a hidden variable because nothing is known for sure, since it is never assumed the existence of previous knowledge.

According to our understanding, another problem afflicting SMT is related to its intrinsic dependency on discovering word translations taken as words, even in their more recent evolutions towards Phrase-Based Statistical Machine Translation (PBSMT). As a matter of fact, we all know that many words need to be translated by multi-words, as is the simple case of "`penso`", in Portuguese, which requires two words to be translated into English ("`I think`") or into French ("`je pense`"). Other cases would be "`from`" <=> "`a partir de`", not to mention the case of "`counterclockwise`", which needs many words to be translated into its Portuguese equivalent "`no sentido contrário ao do movimento dos ponteiros do relógio`", or the many alternatives this phrase may assume. For this problem, word-based SMT requires considering empty word and word translation fertility. In our case, as a consequence of our aligning procedure, together with the vocabulary validation, and the extraction methods used during our training stage, we take for granted that a source phrase is translated by a target phrase (with either phrase having one or more words), depending on the degree of knowledge acquired by our system. Our

approach is clearly a phrase-based approach. Yet, within this topic, knowing that word order is not the same in the various languages, we distinguish two types of word order: local word order and long distance word order.

Local word order occurs inside linguistically motivated phrases. Some examples are:

- "social policy" <=> "política social", where the adjective occurs before the noun in English and after it in Portuguese;
- "acp-eu council of ministers" <=> "conselho de ministros acp-ue", where the nominal part "council of ministers" <=> "conselho de ministros" may be translated without no order change, but, as for the first example, the adjective part "acp-eu" <=> "acp-ue" occurs in different positions in both languages;
- but much more complex examples could be presented.

Some examples of long distance word order in sentences are:

- "the decision of the council of ministers was taken into account" <=> "considerou-se a decisão do conselho de ministros", where the subject of the sentence in the passive voice may appear after the passive verbal form in Portuguese.

Either SMT or PBSMT handle both problems by taking into consideration a word order model which determines the probabilities of word translations changing positions. Those movements are learned from parallel training corpora, usually defining as a parameter the maximum moving distance allowed for word translations. In contrast, we are capable of handling both order types with translation patterns which implement an approach comparable to the Hierarchical Phrase-Based Translation approach (Chiang, 2007). Local order patterns are mostly used for extracting phrase translations in the framework of the ongoing Ph.D. research of Luís Gomes and the result of this work is available in our validated bilingual phrase lexica. Longer distance patterns are mostly used for translation, as will be shown later in this thesis (Sub-Section 3.1.1.2).

Additionally, many of the new concepts introduced in this new approach depend on efficient retrieval operations from the base texts used to support the models involved in the translation process. Those retrieval operations are in turn carried out efficiently by indexing the base texts with Suffix Arrays and other structures built on top of the Suffix Arrays in order to make some of their properties stand out.

In sum, our approach to Machine Translation can be classified as Semi-Supervised Phrase-Based Statistical Machine Translation (SSPBSMT). As explained earlier, it diverges from main trends in PBSMT in several aspects, resulting from our own research background in the area. One of the main differences results from the fact that Transtor considers phrases as the nuclear units used in every model and feature involved (namely the translation model[4], the language model[5] and the penalty feature[6]), something that is in contrast with other approaches claiming to be phrase-based even though their phrase translation models are obtained from a word-based alignment and their language models are word-based. The mentioned differences led us towards a simpler translation process which enabled us reaching a BLEU translation scores (Papineni et al., 2002) higher on average than the ones obtained using MOSES, for several language pairs in both directions. In addition, the fact that we were using a much simpler translation machinery and did not consider any reordering model along with many other aspects taken by SMT or PBSMT, makes our advantage yet more impressive.

1.2 A VERY SHORT OVERVIEW OF COMPARABLE WORK

In the last few years, main trends in Statistical Machine Translation (SMT) have evolved from word-based (Brown et al., 1993) towards-phrase based (Och and Ney, 2004), (Lopez, 2008). This evolution aimed at rationalizing some of the central hypotheses in word-based SMT that mismatch reality, namely those hypotheses related to translation correspondences between words in two languages: not all words are translated by single words, there are words having no direct translation, some word translation correspondences are only valid in specific lexical contexts, etc. By moving towards phrases (uninterrupted sequences of words) some of these translation correspondences were simplified, as well as local (in phrase) word order. Despite this evolution, the alignment procedure remained basically the same: texts continued to be aligned at sentence level and only then sentences are aligned at a word level (Brown, Lai and Mercer, 1991b), (Gale and Church, 1993), (Och and Ney, 2003) and word and phrase translations are extracted together with their translation statistics. No supervision is made. As a consequence, a huge number of incorrect translations and corresponding statistics are extracted and used in the translation process, leaving the burden of selection to the search procedure of decoding (Germann et al., 2001),

[4] A translation model is the probability that a source string is a translation of a target string.
[5] A language model measures the fluency of the sequencing of translation phrases in the translation target language.
[6] The penalty feature tries to guarantee a balance between the number of words between a source language sentence and its translation.

making the translation process heavier. This probably explains, according to our perspective, why some supervision improved our own translation results. Some proposals have appeared in the last few years to improve this state of affairs, in particular for filtering out unproductive phrase table entries (Deng, Xu and Gao, 2008).

Yet, no one still assumes the need for manual validation of extracted phrase translations, not only to be used directly in the production of new translations, but also to be used for improving future validations of extracted phrase translations, something that could be accomplished by reusing the validated positive and negative examples to train good classifiers, implemented, for instance, with SVMs (Mahesh, Gomes and Lopes, 2011). Those classifiers would then be applied on not yet known newly extracted translations from newly realigned parallel corpora (Aires, Lopes and Gomes, 2009), having the results manually validated, having the classifiers retrained on the newly augmented set of validated phrase translations, and iterating this process until no improvement is achieved, if this can ever be achieved.

In order to better tackle long distance reordering phenomena, resulting from the fact that there are words and phrases that translate as gapped phrases (discontinuous patterns), (Chiang, 2007) proposed the use of hierarchical Phrase-Based SMT. In this thesis I will incorporate a similar proposal in our own translation engine. According to our perspective and experience, a huge number of gapped patterns result from coordination (apart from other sources) (Silva et al., 1999) and its extraction may be improved by taking into account phrase meaning similarity (Gamallo, Agustini and Lopes, 2005), (Gamallo, Agustini and Lopes, 2008), (Casteleiro, Lopes and Silva, 2014) whose extraction may be aided by handling phrase alignment results if we improve our robust phrase-based alignment approach (Gomes, Aires and Lopes, 2009). (Carbonell et al., 2006) use a technique quite similar to the one we used for word sense disambiguation (Gamallo, Agustini and Lopes, 2005), (Gamallo, Agustini and Lopes, 2008) for improving the extraction of translation equivalents from non-parallel corpora.

In (Barrachina et al., 2009) it is recognized that despite the advances made in SMT, translations still need to be edited to correct the errors. As a consequence, they propose an interactive phrase-based SMT that accepts corrections and adapts its own output to those corrections while translation is being made from left to right. This interaction just contributes to reduce the search space. It is not clear that it is intended for having a lasting effect on the knowledge confidence the machine will use henceforth. According to our perspective, translation er-

rors made by the system must be manually corrected and reused at the alignment stage for new phrase translation extraction and validation. They may also be used for manual extraction of problematic gapped phrases translations.

Text Mining applied to raw text requires the use of huge quantities of text and data structures powerful enough for efficient string frequency counting, full text indexing, and efficient string matching and retrieval. Suffix Arrays are known to support these requirements. (Yamamoto and Church, 2001) are among the users of this data structure for Text Mining purposes. Results reported by (Abouelhoda, Kurtz and Ohlebush, 2004), related to the construction and use of compressed and succinct suffix arrays, lead us to prefer suffix arrays to suffix trees in the framework of PATRAS project. In the last few years, many authors have resorted to use these data structures for indexing and representing huge parallel corpora and the translation tables extracted from that corpora. Recently it was discovered that suffix trees can be represented with a suffix array plus a tree layer (Sadakane, 2007), (Russo, Navarro and Oliveira, 2008), (Fischer, Mäkinen and Navarro, 2008), this layer requires only marginal space. In essence this result is similar to the one obtained by (Abouelhoda, Kurtz and Ohlebush, 2004), only that the space requirements are much smaller, around 9 times for natural text, and it supports more operations. Hence it is nowadays possible to provide a functional suffix tree layer for translation software, which simplifies the resulting algorithms while being time and space efficient. Due to this evolution we will also explore suffix trees in this project in order to compare both structures in the same application area. Just recently results on compression by (Costa et al., 2013), obtained in the research group I integrate, enabled very competitive data structures occupying 0.4% of the original text while being very efficient for carrying out online queries. However, more work will be necessary to integrate them in a framework as the one that is explained in this thesis, as query time response is adequate for human users (i.e. using a concordancer) but are still too slow for being directly used by an MT system to translate text. Probably they will be adequate for implementing a full interactive machine translation engine of the kind proposed by (Barrachina et al., 2009).

1.3 INNOVATIVE CONTRIBUTIONS

In the approach presented in this thesis, phrases are always considered as units and not just a composition of words which eventually might have to be individually analyzed. Most popular phrase-based approaches still have one or more components built on top of word-based concepts: phrase translation equivalents extracted from a word-based alignment, as well as text fluency determined by a word-based language model. On the contrary, the approach pre-

sented in this thesis is truly phrase-based in the sense that phrases are first degree citizens: translation units consist of phrases, the language model considers phrases and the penalty model is also accounted for phrases. In sum, phrases are not divided.

In this approach, the translation model is capable of producing balanced scores to equally correct translations, leaving to the language model the task of selecting the ones that compose the most fluent combinations of phrases. The language model does not produce an absolute model and does not use any techniques to model unseen data, only worrying about scoring the options available from observed data.

Both the translation model and the language model are supported by indexing structures based on Suffix Arrays which allow their efficient calculation. Those indexing structures also support the actual translation process as well, by identifying every phrase from the original text to then be used to retrieve their corresponding translations which will be then combined to produce a complete translation.

Producing either the translation model or the language model does not require the use of a whole set of available corpora in a single unit (taken as an indivisible whole). Several different corpora can be processed and used separately, this way allowing the production of translations using the corpora sets considered to be the most relevant for the translation in question. This greatly simplifies the management of different models produced with different corpora.

Previously acquired knowledge is included in the form of phrase-based bilingual lexica (for both contiguous and non-contiguous phrases), containing validated entries used as a source of phrase translations, with tests confirming their positive impact on translation results. Translation patterns are capable of modeling syntax supported in lexical evidence, constituting a serious alternative to using syntax supported by text tagging and avoiding the introduction of other errors that might result from the tagging procedure.

The translation probability score developed for the Transtor decoding is implemented by the sentence translation model, as an alternative to the log-linear approach. This is because I argue that the log-linear approach might not be the most appropriate framework to model the probability of a translation, particularly because of its definition as a product of scores which makes it very sensitive to low probabilities and does not allow null probabilities, a problem that requires smoothing techniques applied to any model that might produce such null probabilities. The sentence translation model from Transtor is capable of

better dealing with such situations because of its implementation as an average of scores.

1.4 READING PLAN

This section provides a general overview of how this thesis is organized from now on. It continues with the chapter "State of the Art", in which approaches to Statistical Machine Translation, most important and most relevant to the work proposed here, will be further analyzed under a critical point of view, along with the several concepts that support them. The chapter starts by introducing Statistical Machine Translation (Section 2.1) with its individual components and concepts. Translation evaluation (Section 2.2) is discussed for its importance in system comparison and improvement. The following sections will describe other concepts supporting Transtor, namely: a different approach on alignment, the FCT Alignment (Section 2.3), used to guide the translation extraction process; and the indexing structures (Section 2.4), used to support phrase identification and retrieval operations. This chapter will end with some conclusions (Section 2.5) in order to provide an overview on how Transtor overcomes some limitations or uses alternative methods to solve common problems more efficiently.

The chapter "Translation Process" will describe the concepts, structures and algorithms supporting Transtor in order to produce translations of complete texts. Section 3.1 is dedicated to the description of underlying concepts supporting this approach, like the phrase translation equivalents and the adapted indexing structures, which support many of the operations required by Transtor; Section 3.2 explains the pre-processing stage, responsible for ensuring the texts involved in the process follow a structure that improves their analysis; Section 3.3 describes the training stage supporting the translation process, like the identification of contiguous phrase translation equivalents from aligned parallel texts and the feature models involved in scoring translation candidates; Section 3.4 describes the translation stage, which integrates all the previous elements and concepts to produce the translations of full sentences; and Section 3.5 will highlight the main differences between Transtor and Moses.

The chapter "Results" will be dedicated to evaluate the results of the translations made by Transtor and compare them with those obtained by Moses (Koehn et al., 2007), the PBSMT system state-of-the-art. This will be made by presenting the BLEU evaluation scores of translations of several texts obtained by both Moses and Transtor, for several language pairs. A first section will describe the conditions set to produce the translations to be evaluated, another

will describe the data produced by each tool after training, another will show the results using common data, another will show additional evaluation scores for Transtor, and a final one will analyze and compare results from both systems.

The final chapter "Conclusions and Future Work" will be dedicated to the analysis of the developed work presented here, namely identifying the innovations introduced, and refer a few changes already planned in order to deal with identified limitations of Transtor, aiming at its improvement on both performance and quality of results.

2 STATE OF THE ART

This chapter is dedicated to the description of the state-of-the-art of the elements most relevant to the work presented in this thesis. This chapter will start by introducing the main concepts of SMT in Section 2.1, then mention automatic translation evaluation in Section 2.2, present the FCT Alignment in Section 2.3, introduce the indexing structures in Section 2.4 and provide conclusions in Section 2.5 in order to provide a glimpse on how Transtor overcomes some limitations or uses alternative methods to solve more efficiently the same problems faced by others.

2.1 STATISTICAL MACHINE TRANSLATION

In 1949, at a time when computers were first being considered for the problem of natural language translation, Warren Weaver suggested the application of statistical and cryptanalytic techniques (Weaver, 1955), after their successful use in breaking the Enigma Code. However, computational limitations at the time kept this approach from being further explored.

It was only when such limitation began to fade, in the late 1980's and early 1990's, that Statistical Machine Translation (SMT), an approach to Machine Translation (MT), appeared. Such an approach resulted from the back and forth movements in the research area of speech recognition, where the application of linguistic rules developed for phonetics led Frederick Jelinek to say "every time I fire a linguist, the performance of the speech recognizer goes up" because the use of Information Theory (Shannon, 1948) and statistics revealed to be more beneficial. Not only that, by the end of the eighties it was obvious that the logico-symbolic approach, heavily based on variations of Chomskian Linguistics, had failed to reveal Human Languages Code. SMT started to emerge as a new research paradigm based on Statistics and on unsupervised Machine Learning methods, producing results unthinkable until then (Brown et al., 1988), (Brown et al., 1990), (Brown et al., 1991a), (Brown, Lai and Mercer, 1991b), and (Brown et al., 1993).

SMT employs a learning algorithm to a large body of texts and their translations, forming the so called parallel text (or parallel corpus). From such application of the learning algorithm, the learner is then able to translate previously unseen sentences. With an SMT toolkit and enough parallel text an MT system can be built for any new language pair within a very short period of time. The accuracy of these systems depends crucially on the quantity, quality, and do-

main of the data, but there are many tasks for which even poor translation is useful (Lopez, 2008).

The groundbreaking approach, which resulted from the original work on SMT with the pioneer IBM Candide project, continues to influence SMT today, illustrating many common modeling concepts (Koehn, 2009). In less than two decades, SMT has come to dominate the academic MT research, and has gained an important share of the commercial MT market.

The following sections will describe the main principles and methods shared by the main SMT approaches that are known, in particular the probability of a translation in Sub-Section 2.1.1, the alignment in Sub-Section 2.1.2, the translation feature models in Sub-Section 2.1.3, and the decoding process in Sub-Section 2.1.4. Finally, an introduction is presented for the tree-based approaches in Sub-Section 2.1.5.

2.1.1 PROBABILITY OF A TRANSLATION

The main concept supporting SMT is the definition of $P(e|f)$ as the probability of a translation *e* given a foreign sentence *f*. Such probability is usually defined as the combination of the probability of smaller parts (words or phrases) that compose the given translation and its original sentence. This is because evidence to determine the probability of those smaller parts is more likely to be found than for the complete sentences. Following this, the purpose of SMT is then to find the combination of smaller parts that produce a translation of a complete original sentence for which the probability is the highest (or closest to the highest), a process that is called decoding (Sub-Section 2.1.4).

The following sub-sections will describe the main approaches developed to model a translation probability, first describing the pioneer noisy-channel approach (Sub-Section 2.1.1.1), followed by its generalization into the log-linear approach (Sub-Section 2.1.1.2).

2.1.1.1 Noisy-Channel Approach

The noisy-channel approach was the probability score applied in the first successful statistical approach to language translation, introduced by (Brown et al., 1988). That statistical approach was word-based and established the concept that a string of English words, *e*, can be translated into a string of French words, *f*, in many different ways, introducing the view that every French string, *f*, is a possible translation of *e*. As such, to every pair of strings (*e*, *f*), a number $P(f|e)$ is assigned, which can be interpreted as the probability of *f* being a translation of *e*. With these concepts, given a French string *f*, the purpose of an SMT system

is to find the English string $\hat{e}$ for which $P(e|f)$ is greatest, illustrated in Equation 1.

$$\hat{e} = \underset{e}{\operatorname{argmax}} P(e|f)$$

Equation 1. Obtaining the English string with the greatest probability

Using Bayes' theorem, $P(e|f)$ can be written as shown in Equation 2.

$$P(e|f) = \frac{P(e)P(f|e)}{P(f)}$$

Equation 2. Noisy-channel model

Considering that the denominator is independent of e, finding $\hat{e}$ corresponds to finding the translation e for which the product $P(e)P(f|e)$ is the largest. The resulting expression is the Fundamental Equation of Machine Translation, shown in Equation 3.

$$\hat{e} = \underset{e}{\operatorname{argmax}} P(e)P(f|e)$$

Equation 3. Fundamental equation of machine translation

The expression in Equation 3 benefits from the combination of the two factors. The translation model probability, $P(f|e)$, is higher for English strings that have the necessary words in them to translate the French, but those are not necessarily well-formed. The language model probability, $P(e)$, is higher for well-formed English strings, but those might not correctly translate the French. Together, both models produce a large probability for well-formed English strings that account well for the French. The advantage of this approach over modeling $P(e|f)$ directly is that two independent models can be applied to the disambiguation of e (Brown et al., 1990), something that is beneficial because the estimates of each model can contain errors. By applying them together it is hoped that the errors of each of the models are compensated by the other model.

This modeling approach follows the idea introduced by Weaver, who made the analogy to information theoretic work (Shannon, 1948), (Shannon, 1951) on signal transmission over a physical medium, called the noisy channel problem, hence the noisy-channel designation of this approach. Weaver suggested the following.

> One naturally wonders if the problem of translation could conceivably be treated as a problem in cryptography. When I look at an article in Russian, I say: "This is really written in English, but it has been coded in some strange symbols. I will now proceed to decode." (Weaver, 1955)

As a result of the idea introduced by Weaver, the process to recover e is called decoding, described in Sub-Section 2.1.4.

2.1.1.2 Log-Linear Approach

The log-linear approach (Berger, Pietra and Pietra, 1996), (Och and Ney, 2002) is a model structure that is well known in the machine learning community. It allows including several models (besides the translation and language models) for features that are suspected to contribute to the improvement of translation quality. The model is shown in Equation 4.

$$P(e|f) = \exp \sum_{i=1}^{n} \lambda_i h_i(e, f)$$

Equation 4. Log-linear model

More specifically, this approach determines the probability of a translation through the combination of n feature models h_i, with each feature model being attributed a feature weight λ_i to express how much the feature contributes to the total translation probability score.

The log-linear model can be seen as a generalization from the earlier noisy-channel approach (Sub-Section 2.1.1.1), as shown by the following conditions

- number of feature functions $n = 2$;
- $\lambda_1 = \lambda_2 = 1$;
- $h_1(e, f) = \log(P(f|e))$;
- $h_2(e, f) = \log(P(e))$.

with which the expression in Equation 5 is produced.

$$P(e|f) = \exp[1 \cdot \log(P(f|e)) + 1 \cdot \log(P(e))] = P(f|e)P(e)$$

Equation 5. Noisy-channel model expressed with the log-linear model

This model does not impose any limitation on the number of features nor on what those features are intended to model, which can range from the already mentioned translation and language models to morphologic, semantic or pragmatic models.

2.1.2 ALIGNMENT

The alignment is an operation carried out over a pair of parallel texts (texts in different languages, with each text being a translation of the other) with the purpose of mapping units in one language to units in the other language. Those mapped units will then support the creation of the translation tables which, in turn, support translation in SMT. Usually, alignment requires that the parallel

texts have the same number of lines, with the mapping being carried out within each parallel pair of lines.

The main challenge of the alignment comes from the fact that it is not known which units from the source language correspond to which units from the target language. For this reason, it is necessary to estimate the alignment model from incomplete data. As such, the alignment between translation units is "hidden from plain view" which is why the alignment is considered a hidden variable. SMT approaches depend on the resulting mapped units to build their translation lexica, which can then be used to produce translations. Those correspondences can be scored according to a defined translation model (Sub-Section 2.1.3.1).

The first alignment approaches were word-based, meaning that the units to be mapped consisted of single words. The word-based approach is described below in Sub-Section 2.1.2.1. The limitations faced by word-based approaches led further research towards phrase-based alignment proposals, described in Sub-Section 2.1.2.2. Additionally, the description of a phrase-based approach using word categories instead of literal words, the alignment template approach, is included in Sub-Section 2.1.2.3.

2.1.2.1 Word-Based Alignment

The purpose of word-based alignment is to map words from a sentence considered to be the target to words from a sentence considered to be the source (Brown et al., 1988), (Brown et al., 1990). Formally, given a sentence e_1^I with I words and a foreign sentence f_1^J with J words, the goal of word alignment is to establish word-to-word correspondences between those sentences. Those correspondences can be represented by a matrix A, such as $A \subset [1, I] \times [1, J]$, where $(i, j) \in A$ if word e_i is aligned with word f_j.

An example taken from (Koehn, 2009) is shown in Figure 1, where words in the English sentence (rows) are aligned to words in the German sentence (columns), as indicated by the filled points in the matrix.

The main challenge of this simple approach resides on the fact that a sentence expressed in different languages is not guaranteed to have the same number of words. This is not only because some words in one language can be translated by more than one word, but also because some words may have no direct correspondence in the other language. The concepts of "empty word" (a word could remain unmapped) and "word fertility" (a word may be mapped to one or more words) were introduced to deal with such situations.

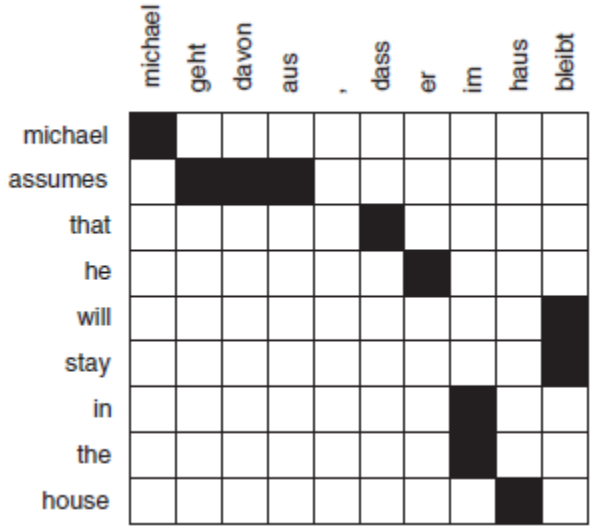

Figure 1. Example of word alignment

Further development of word-based alignment led to its symmetrization (Och, Tillman and Ney, 1999), (Koehn, Och and Marcu, 2003). It consists in taking a word-based alignment A_0 in one direction (*e* to *f*) and a word-based alignment A_1 in the other direction (*f* to *e*), producing a single symmetrized word-based alignment A. Some simple criteria of symmetrization consist of the intersection ($A = A_0 \cap A_1$), or the union ($A = A_0 \cup A_1$) of the alignments in each direction. The intersection produces a higher precision with lower recall, while the union produces a higher recall with lower precision.

An example, taken from (Koehn, 2009) and shown in Figure 2, illustrates the intersection (in black) and union (in either black or gray) of a pair of alignments taken in each language direction.

Alternatively, a refined symmetrization method starts from the intersection between alignments on each direction, iteratively extending the resulting alignment by including neighbor elements. In a first step, the intersection $A = A_0 \cap A_1$ is determined. Then, the alignment A is iteratively extended by adding alignments (i, j), occurring only in the alignment A_0 or in the alignment A_1, if neither e_i nor f_j have an alignment in A, or if the following conditions both hold:

- The alignment (i, j) has a horizontal neighbor $(i-1, j)$, $(i+1, j)$ or a vertical neighbor $(i, j-1)$, $(i, j+1)$ that is already in A.
- The set $A \cup \{(i, j)\}$ does not contain alignments with both horizontal and vertical neighbors.

The refined method is often able to improve precision and recall when compared with the initial directed word alignments.

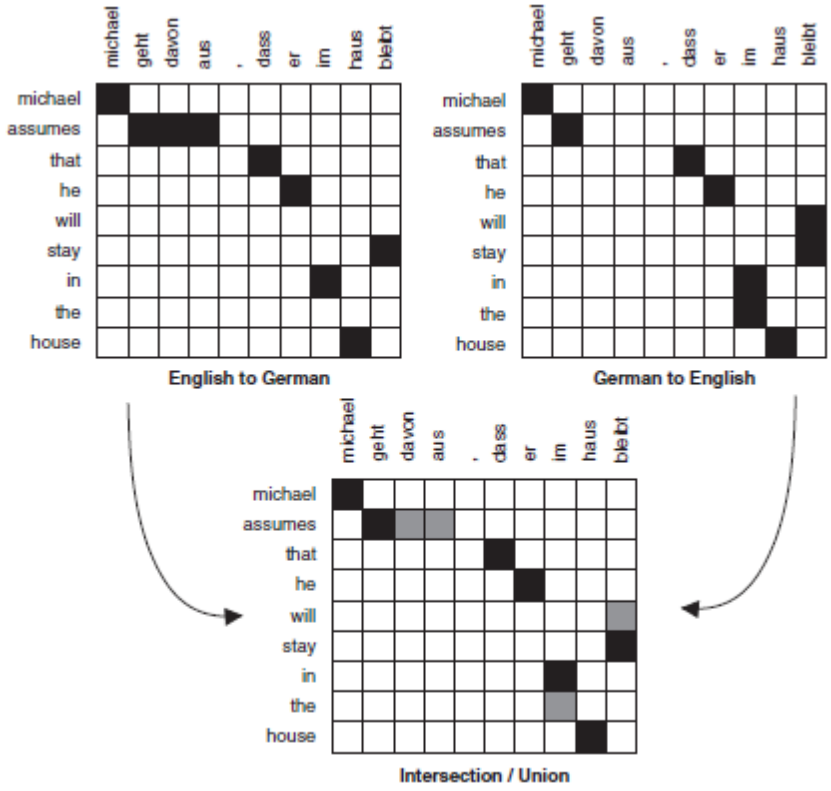

Figure 2. Example of symmetrization of IBM model alignments

However, even with symmetrization, single words are still considered single units, something that can represent a problem because single words are not the best candidates for the smallest units of translation equivalence. In fact, there are cases in which contiguous sequences of words are commonly translated as a unit, with idiomatic expressions representing a typical example of such cases, in which possible word correspondences only make sense in the context of the expression being considered.

A particular example is the translation equivalent "`de volta a a estaca zero`" <=> "`back to square one`", which cannot be properly translated word by word (a more literal translation into English would be "`back to pillar zero`"). In this case, the best option would be associating "`de volta a`" with "`back to`", which is correct, and "`a estaca zero`" with "`square one`", which is an association that should only happen within this context.

Another example is the translation equivalent "`clockwise`" <=> "`em o sentido de os ponteiros de o relógio`", in which 9 words in Portuguese correspond to one word in English, pose a challenge to the word-based ap-

proach. Situations like these are better handled with phrases, inspiring the development of phrase-based alignment, described below (Sub-Section 2.1.2.2).

2.1.2.2 Phrase-Based Alignment

In phrase-based alignment, the units mapped now consist of phrases which, in turn, consist of one or more words, not necessarily following grammatical concerns. As mentioned in word-based alignment (Sub-Section 2.1.2.1), translating contiguous sequences of words, like idiomatic expressions, can be more naturally accomplished with phrases. Both the "empty word" translation and "word fertility" from word-based alignment are abandoned. Instead, phrases from a phrase translation equivalent are nonempty and can have a different number of words.

A typical way to produce phrase-based alignment is from the symmetrization of the previous word-based alignments of parallel corpora (Tillmann, 2003), (Zhang, Vogel and Waibel, 2003), (Zhao and Vogel, 2005), (Zhang and Vogel, 2005), (Setiawan, Li and Zhang, 2005). Groups of words that may constitute phrase translation equivalents are then identified using consistency rules. A phrase pair (e_1^I, f_1^J) is said to be consistent with an alignment A if all words e_i from e, that have alignment points in A, have those alignment points with words f_j from f, and vice versa. So, basically, any aligned words from a phrase will only align to one or more words from the other.

As shown in Figure 3, also taken from (Koehn, 2009), the first example is consistent because all aligned words are included in the phrase pair, the second example is not consistent because one alignment point in the second column is not included in the phrase pair, and the third example is consistent because the unaligned word on the right is allowed by the consistency rules.

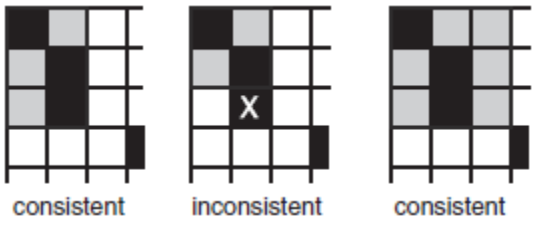

Figure 3. Examples of consistency and inconsistency

However, obtaining phrase-based alignment from previous word-based alignment can still face some challenges inherited from the limitations of word-based alignment. As noted before, idiomatic expressions, as well as other expressions, usually cannot be correctly aligned at a word level (as shown in the example "`de volta a a estaca zero`" <=> "`back to square one`"), which can

consequently limit the quality of phrase-based alignment obtained from word-based alignment.

As an alternative to using word-aligned parallel corpus, phrase alignment may be done directly from sentence-aligned corpora using a probabilistic model (Shin, Han and Choi, 1996), pattern mining methods (Yamamoto et al., 2003), or matrix factorization (Goutte, Yamada and Gaussier, 2004). Still, as referred in (Koehn, 2009) results are generally no better than learning phrases from word-alignments.

In the end, as mentioned in (Lopez, 2008), Phrase-based models have quickly become standard, as results have consistently proven to be better than the ones obtained by word-based approaches.

2.1.2.3 Alignment Template

The alignment template approach (Och and Ney, 2004) is a particular implementation of a phrase-based model. Instead of using explicit phrase-to-phrase translations, this approach associates phrases through an alignment template which consists of a reordering of the words composing the phrase. However, that reordering is based on word classes (or categories) rather than specific words. Those classes are automatically trained bilingual classes using the method described in (Och, 1999) and constitute a partition of the vocabulary of the source and target languages.

Using classes instead of the actual words improves generalization because, for instance, if there are classes in the source and target languages that contain town names, it is possible for an alignment template learned using a specific town name to be generalized to other town names. Once the words in the phrase have been assigned to classes, the words can then be translated using word-to-word translation, with the alignment templates being reordered as in phrase-based models.

In sum, the alignment templates are a generalization capability of a bilingual phrase lexicon in which words are replaced by word classes and contain the alignment information for each phrase pair. However, such alignment information is still word-based (or word-class-based, to be more precise), so it will still inherit limitations from word-based approaches, in particular when dealing with idiomatic expressions.

2.1.3 TRANSLATION FEATURE MODELS

In order to determine the probability of a translation (Sub-Section 2.1.1), translation feature models are considered with the purpose of trying to assess the

quality of different specific features that might contribute to the overall quality of a translation to be produced. Some feature examples include language fluency, translation relevance, or even the likeliness of occurrence of a specific word.

In the case of the noisy-channel approach (Sub-Section 2.1.1.1), only two translation feature models were considered (the translation model and the language model), unlike the log-linear approach (Sub-Section 2.1.1.2), which allows for any number of translation feature models. Particularly considering the widespread use of the log-linear approach, the identification of the features that provide the most significant information about translation quality (feature selection) is an open problem in SMT, as mentioned in (Lopez, 2008). In fact, as stated in (Koehn, 2009), millions of features can be introduced to score a translation candidate, such as features that can indicate the use of a specific phrase translation.

The following sub-sections will describe the most relevant and the most commonly used feature models, namely the translation model (Sub-Section 2.1.3.1), the language model (Sub-Section 2.1.3.2), the reordering or distortion model (Sub-Section 2.1.3.3), and the penalty models (Sub-Section 2.1.3.4).

2.1.3.1 Translation Model

The Translation Model (TM) is a statistical model developed with the purpose of scoring how likely a translation unit *e* in one language is to translate a given translation unit *f* in another language.

The most common approach, resulting from the initial work proposed in (Brown et al., 1988) and (Brown et al., 1990), considers the number of times each different translation *e* is associated with *f*, using the formula in Equation 6, in which the $count(e, f)$ function considers the number of associations established between *e* and *f*. The result is a non-null probability score, where the sum of the probabilities of every translation alternative is 1.0, this way constituting a probability distribution.

$$p_{tm}(e|f) = \frac{count(e, f)}{\sum_{i=1}^{n} count(e_i, f)}$$

Equation 6. Formula for the direct translation probability

Besides the direct translation probability $p_{tm}(e|f)$, the inverse translation probability $p_{tm}(f|e)$ can also be considered. However, such scores reflect how frequently each translation unit in one language has been found together with a given translation unit in another language, according to the aligned parallel texts from which the model has been produced. The main disadvantage of such an approach consists in the inability to provide a higher score to a perfectly cor-

rect translation that is not so frequently used, and assigning a high score to a very frequent translation that might not be the most appropriate choice within a given context. One such example is represented by the Portuguese source phrase (a single word) "`casa`", which might be translated into English by "`house`" or by "`gets married`", with the first one generally being much more frequent (depending on the texts being analyzed), even though the second one is also a perfectly valid translation.

Additionally, when phrases are the translation units being used, avoiding the overestimation of a rare phrase pair is accomplished by decomposing it into its word translations and checking how well they match up. The method is called lexical weighting, which is basically a smoothing method that uses the richer statistics provided by lexical translation to produce more reliable probability estimates. Either the direct lexical weighting $lex(e|f)$ or the inverse lexical weighting $lex(f|e)$, can be considered. However, the main problem with lexical weighting is that it cannot be correctly applied to expressions like "`clock-wise`" **<=>** "`em o sentido de os ponteiros de o relógio`".

From my point of view, the choice of a translation phrase, once its correction has been determined, should mainly depend on its context and not so much on how frequently it has been associated to an original phrase. The translation model proposed in this thesis minimizes such problem by admitting more than one source of phrase translations separately, as described in Section 3.3.2.1 ahead.

2.1.3.2 Language Model

The Language Model (LM) is a statistical model developed to score the fluency of a general sentence, initially proposed and inspired by (Brown et al., 1988) and (Brown et al., 1990), measuring how well a sequence of words might be followed by another word.

The generally accepted approach is the *n*-gram language model which inherits its form from speech recognition (Jelinek and Mercer, 1980) and uses the Markov assumption, which is an independence assumption that breaks a sentence probability into the product of the probability of each word given a limited history of preceding words, instead of a history considering every preceding word. More specifically, the fluency score of a sentence with m words $w_1, w_2, \dots, w_m$ is determined according to the preceding $n - 1$ words of each w_i, as shown in Equation 7.

$$p_{lm}(w_1, w_2, \dots, w_m) = \prod_{i=1}^{m} p(w_i | w_{i-n+1}, \dots, w_{i-1})$$

Equation 7. Formula for the general language model

The model is named the n-gram history model, with the most common value of n being 3 for the tri-gram language model (Equation 8).

$$p_{lm}(w_1, w_2, \dots, w_m) = \prod_{i=1}^{m} p(w_i | w_{i-2}, w_{i-1})$$

Equation 8. Formula for the tri-gram history language model

Such score reflects how frequently each word follows a set of previous n–1 words, according to the monolingual texts from which the model has been produced.

The main challenge in language models is dealing with sparse data because it will make unclear the distinction between something that never occurs and something that simply has not been observed. In order to deal with the sparseness, methods such as add-one smoothing, deleted estimation or Good-Turing smoothing, for which an efficient implementation is presented by (Gale and Sampson, 1995), take probability mass from evident events and assign it to unseen events, but this, in turn, prevents an impossible combination from getting its adequate probability of zero.

The main disadvantage of such model has to do with the fact that phrases have to be decomposed into their individual words to be checked for their language model, when very often those phrases should be considered as a single unit. The phrase "`in spite of`" is such an example, where the probability of "`of`" given the occurrence of the previous two words "`in spite`" (considering the tri-gram history model) is 1 in many texts, meaning that the word "`of`" is the only word that follows "`in spite`" in those texts. The phrase "`in accordance with`" can be a similar example. Besides, when both the previous phrase examples come after two other words, for instance "`the rules`", their contribution will be the same when their first word is analyzed, as both phrases begin by "`in`". More specifically, the fluency of the phrase "`the rules in`" will be the same in both "`the rules in accordance with`" and in "`the rules in spite of`".

From my point of view, phrases should always be considered as units, which do not need to have their internal fluency analyzed. The language model proposed in this thesis implements such an idea by determining the likeliness of

adjacent contiguous phrases according to other adjacent alternatives, as described in Section 3.3.2.2 ahead.

2.1.3.3 Reordering or Distortion Model

The purpose of the reordering or distortion feature, resulting from the original work by (Brown et al., 1988) and (Brown et al., 1990), is to model the likelihood of translation units changing places when translated. An example of such situation is the equivalent "`European Community`" <=> "`Comunidade Europeia`", in which a word-by-word translation still requires the words to be swapped in order to obtain a correct translation equivalent.

Modeling such reordering is typically accomplished by a distance-based reordering cost which allows translation units to be moved from their original position to another position on the translation. The model penalizes reordering in general, leaving to the language model (Sub-Section 2.1.3.2) the responsibility of justifying it. The movement of translation units is often limited to a maximum number of positions. However, translations between some language pairs may require large position movements that can easily exceed the generally small windows of words (usually three words) used by the language model. Given its limitations, disregarding the reordering model does not significantly affect translation quality, but reduces the complexity of the search problem from exponential to polynomial, turning decoding much faster. Yet, including limited reordering still yields better translation results.

In sum, admitting moves within a window of a few words can be handled by the language model, often representing the best that can be done with reordering. Larger reordering windows or completely unrestricted reordering, besides increasing complexity and execution time, often leads to worse results (Koehn, 2009).

The lexicalized reordering model (Tillmann, 2004) is an improvement over the one simply conditioned on movement distance described above. Such model intends to assign a translation unit with a score that will indicate how likely the translation unit is to be maintained, swapped with the previous, or placed discontinuously.

2.1.3.4 Penalty Models

The purpose of the penalty models is to keep a balance between the length of a translation and the corresponding source sentence. In the case of the word penalty model, it intends to maintain a balance between the number of words from the sentence being translated and its translation. This is mainly because of the

preference the language model (Sub-Section 2.1.3.2) has for shorter translations, simply because fewer trigrams have to be scored. As such, the penalty model helps preventing translations from being too short relatively to its original, and is generally responsible for a significant improvement in translation quality.

Yet, the word penalty model is unable to deal properly with situations in which an original sentence and its translation have a significant difference in their corresponding number of words. Sentences containing translation equivalents like "`clockwise`" **<=>** "`em o sentido de os ponteiros de o relógio`", in which 9 words in Portuguese correspond to 1 word in English, constitute an example.

To deal with such situations, the phrase penalty model was developed, in which phases are considered units, instead of single words. In this case, the choice is between using either fewer or more phrases. When fewer phrases are considered, each phrase will be generally longer than when more phrases are considered, in which case each phrase will generally be shorter. In practice, it is preferable to use longer phrases, even though these are less frequent. The lack of statistical support is generally compensated not only because longer phrases include more context but also because the translation model (Sub-Section 2.1.3.1) has a contribution in filtering bad phrase pairs.

2.1.4 DECODING

The objective of decoding is to determine the translation for which the highest score is achieved according to a defined translation probability score, usually one of the translation probability scores described in Sub-Section 2.1.1: the original noisy-channel approach (Sub-Section 2.1.1.1); and the log-linear approach generalization (Sub-Section 2.1.1.2), adopted from the machine learning field.

Decoding is a very hard problem because of the exponential number of possible choices for a specific input sentence. In fact, (Knight, 1999) has shown that the decoding problem is NP-complete. Consequently, examining all possible translations, scoring them, and determining the best is computationally too expensive.

Beam search decoding, described by (Wang and Waibel, 1997) and (Koehn, 2004), is a general framework followed by almost every approach. Such general framework is inspired by speech recognition algorithms which date back to (Jelinek, 1969).

Before translating a sentence, the applicable word or phrase translations are obtained from the translation table. Then, a translation is built word by word,

until all words have been contemplated in the decoding process. While composing the translation word by word, partial hypotheses are produced before obtaining a complete translation, represented by a complete hypothesis.

A hypothesis most notably contains information about what translation words have been produced, which source words have been covered, and the partial translation probability score. So, decoding starts with the empty hypothesis, expanding hypotheses whenever a new word is contemplated in the translation, until complete hypotheses are obtained.

As mentioned above, the computational complexity of decoding implies the need of restricting the search space. Such restriction is accomplished by:

- hypothesis recombination, in which partial hypotheses that cannot be part of the best translation are discarded;
- pruning out bad hypotheses early on, in which case the translation cost of the remaining untranslated words also has to be considered. A fair comparison of hypotheses covering different parts of the input sentence implies considering an estimate of the future cost of translating the rest of the input sentence, called rest cost or outside cost; and
- including limits on reordering, which significantly reduce the search space.

The methods mentioned above are heuristic because they do not guarantee to always find the best translation, but the best one could be found often enough, or at least a translation that is very close to being the best.

2.1.5 TREE-BASED APPROACHES

Given that the grammar structure of a sentence is represented in linguistic theories by a tree structure, it is only natural to try to extend those trees to express translation models. Such extension results in synchronous grammars expressed by pairs of trees, one for the original sentence and the other for the translation sentence.

A Context-Free Grammar (CFG), can be represented by a tuple (N, T, D), in which N represents a set of non-terminal symbols, T represents a set of terminal symbols, and D represents a set of productions $D = \{N \rightarrow \{N \cup T\} *\}$, where each production defines how a non-terminal symbol can be mapped to a sequence of terminal and non-terminal symbols. The popularity of CFG's in natural language parsing comes from the fact that terminal symbols can naturally represent words and that non-terminal symbols can represent syntactic categories.

Extending a CFG to express translations is accomplished with a Synchronous Context-Free Grammar (SCFG) from a tree-based approach. The grammar of an

SCFG is composed of a pair of productions and a definition of a correspondence between non-terminals from the productions. The elementary structures of an SCFG are rewrite rules of the form $X \rightarrow \langle E, F, M \rangle$, where X is a non-terminal, E and F are both strings of terminals and non-terminals, and M is a one-to-one correspondence between non-terminal occurrences in E and non-terminal occurrences in F.

The methods for building phrase models can be used to learn synchronous grammars:

- Extract all rules that are consistent with the carried out word alignment.
- Create hierarchical phrase pairs by allowing phrases to include other phrases.
- Use syntactic markup to create grammar rules with non-terminal nodes covering underlying phrase mappings.
- Estimate probability distributions for grammar rules based on relative counts.

It should be noted that the beam search decoding algorithm for phrase-based models (Sub-Section 2.1.4) does not work for tree-based models, since the translation cannot be built from left to right straightforwardly. Instead, decoding by parsing is carried out with a chart parsing algorithm, which may require efficient methods to access grammar rules, as well as recombination, pruning and grammar binarization to improve parsing efficiency. Chart parsing was first proposed by (Kay, 1985), for which a common approach is a variant of the Viterbi algorithm (Viterbi, 1967), but the Earley parser (Earley, 1970) is the one mainly used for parsing in computational linguistics, as is the case of SMT.

2.1.5.1 Hierarchical Approach

The hierarchical approach (Chiang, 2007) is a phrase-based system that uses SCFG rules that can be learned automatically from a parallel text without any syntactic annotation. The main productions use a single undifferentiated non-terminal X, allowing a maximum of two non-terminals in the right-hand side of any rule, as well as a number of terminal symbols in both languages. Each rule can represent a mapping between phrases, which may be reordered recursively. Consider the following grammar fragment.

- $H_1: X \rightarrow the\ X_1\ X_2 \mid X_2\ o\ X_1$
- $H_2: X \rightarrow has\ arrived \mid chegou$
- $H_3: X \rightarrow plane \mid avião$

Recursivity is expressed by rule H_1 which, besides containing the terminal equivalence "`the`" <=> "`o`" to ensure some lexical evidence, it has the non-terminals X_1 and X_2. Applying H_3 to instantiate X_1 with "`plane`" <=> "`avião`", and applying H_2 to instantiate X_2 with "`has arrived`" <=> "`chegou`", allows H_1 producing the translation "`the plane has arrived`" from "`chegou o avião`", and vice-versa.

The rule extraction is accomplished using a symmetrized word alignment obtained from a word-aligned parallel corpus calculated on both directions and exploring common sub-phrases between consistent aligned phrases, which will then have differing parts replaced with non-terminal symbols. The grammar is filtered using some constraints in order to avoid producing a very large number of rules, an undesirable consequence not only because it significantly burdens the process but also because it creates spurious ambiguity, producing many different derivations for the same translations. Some constraints examples are the following:

- limiting the phrases to a length of 10 words on either side;
- limiting the rules to five non-terminals plus terminals on the foreign side;
- rules can have at most two non-terminals, simplifying the decoder implementation;
- non-terminals cannot be adjacent on the foreign side, a major cause of spurious ambiguity; and
- a rule must have at least one pair of aligned words, so that translation decisions are always based on some lexical evidence.

Besides the extracted rules, the process includes a set of special rules, which are the glue rules and the entity rules.

Once rules have been extracted from the training data, X could be the start symbol of the grammar, translating new sentences only using the extracted rules. However, the grammar may divide a source sentence into segments, translating one segment at a time, something that is formalized using the glue rules, shown below. Glue rules analyze a start symbol S as a sequence of Xs which are then translated without reordering.

$$S \rightarrow \langle S_1 X_2, S_1 X_2 \rangle$$

$$S \rightarrow \langle X_1, X_1 \rangle$$

Finally, a specialized set of translation modules are applied in order to translate names, dates, numbers, and bylines that might occur in a sentence, inserting

those translations into the grammar as new rules. Such rules are called entity rules and a generalization example over numbers of years is shown below.

$$X \rightarrow \langle há\ X_1\ anos, X_1\ years\ ago \rangle$$

Penalties are applied on several components: on extracted rules in order to allow the model to learn a preference for longer or shorter derivations; on glue rules so that the model can learn a preference for hierarchical phrases over a serial combination of phrases; on the four types of entity rules (concerning numbers, dates, names and bylines) so that the model can learn how much to rely on each of them; and, finally, a word penalty, which only considers terminal symbols, is applied on all the rules to learn general preferences.

The hierarchical approach is a more powerful generalization than the alignment template approach (Sub-Section 2.1.2.3). Both use word-based alignment to build their generalization bilingual lexica, but the hierarchical approach can be recursively applied to any phrase, while the alignment template (Och and Ney, 2004) only admits words that fit into word classes (Och, 1999) within the phrase, with no recursion.

Because of its flexibility, this approach also faces some challenges. First of all, given that the derivations cannot be directly observed, training has to consider heuristic approximations. Then, implementing every translation decision as a rule application contributes to the very high complexity of the approach, implying the need of using heuristic methods and the need of introducing constraints, as already shown above in the derivation rules. Finally, the cubic complexity of the decoding algorithm is reduced to linear at the cost of introducing a length limit for spanned sub-strings and the language model is included using a cube pruning algorithm.

2.2 TRANSLATION EVALUATION

Evaluation of translations is a serious challenge, mainly because each original sentence admits many valid translations. In fact, (Hovy, King and Popescu-Belis, 2002) attribute to Yorick Wilks the remark that "more has been written about MT evaluation over the past 50 years than about MT itself".

Evaluation carried out by humans is very expensive and time consuming, which motivated the development of automatic metrics. Particularly for MT, it can be very useful having an automatic metric that quickly allows assessing the quality of MT systems, assessing if one system is better than another, or even assessing if a change in the system led to an improvement.

Automatic metrics involve the use of reference translations. These consist of a set of test sentences for which translations made by humans are already available. The reasoning behind the use of reference sentences is that a close resemblance to a human translation must be an indicator of the good quality achieved by an MT system (Papineni et al., 2002). These metrics are based on partial string matching between the output and the reference translations.

The most popular automatic evaluation metric, and the one used for the presented results (Chapter 4), is BLEU, described in Sub-Section 2.2.1 below.

2.2.1 BLEU

The currently most widely used evaluation score is the Bi-Lingual Evaluation Understudy (BLEU) (Papineni et al., 2002), in which the matches between the output and the reference sentence consider not only the single words but also the *n*-grams up to some maximum *n*, enabling the metric to reward sentences where local word order is closer to the local word order in the reference.

The generic *n*-gram precision of order *n* is expressed in Equation 9.

$$\text{precision}(n) = \frac{\text{matched\_ngrams}(n)}{\text{total\_ngrams}(n)}$$

Equation 9. The *n*-gram precision of order *n*

The generic BLEU metric, for a maximum order *n* of *n*-grams to be matched, is defined in Equation 10.

$$\text{BLEU}(n) = \text{brevity\_penalty} \cdot \exp \sum_{i=1}^{n} \lambda_i \cdot \log(\text{precision}(i))$$

Equation 10. The BLEU score of order *n*

BLEU considers the number of *n*-gram matches as a fraction of the number of total *n*-grams in the output sentence, making it a precision-oriented metric. The problem with such metrics is that dropping words is not penalized. Such problem is addressed by BLEU with a brevity penalty, which has the purpose of reducing the score of a produced sentence that is much shorter than its reference. The brevity penalty is defined in Equation 11.

$$\text{brevity\_penalty} = \min\left(1, \frac{\text{output\_length}}{\text{reference\_length}}\right)$$

Equation 11. Brevity penalty

Typically, the maximum order *n* of *n*-grams to be matched is set to 4, in which case the metric is then called BLEU-4. Besides, the weights λ_i for the different

precisions are all typically set to 1, resulting in the simplified BLEU-4 formula expressed in Equation 12.

$$\text{BLEU}(4) = \text{brevity\_penalty} \cdot \prod_{i=1}^{4} \text{precision}(i)$$

Equation 12. Simplified BLEU score of order 4

As an example, taken from (Koehn, 2009), consider the sentences below.

- **Reference**: `Israeli officials are responsible for airport security.`
- **System A**: `[Israeli officials] responsibility of [airport] safety.`
- **System B**: `[airport security] [Israeli officials are responsible].`

The matches from the output of System A consist of a 2-gram match for "`Israeli officials`" and a 1-gram match for "`airport`", while all output words of System B have a match, in particular "`airport security`" is a 2-gram match and "`Israeli officials are responsible`" is a 4-gram match. Given the *n*-gram matches, the *n*-gram precision can be computed, which is the ratio of correct *n*-grams of a certain order *n* in relation to the total number of generated *n*-grams of that order. Again, considering the example sentences above, the 1-gram, 2-gram, 3-gram and 4-gram precisions of each system are shown below:

- System A: the 1-gram precision is 3/6; the 2-gram precision is 1/5; the 3-gram precision is 0/4; the 4-gram precision is 0/3.
- System B: the 1-gram precision is 6/6; the 2-gram precision is 4/5; the 3-gram precision is 2/4; the 4-gram precision is 1/3.

The fact that if any of the *n*-gram precisions is 0 will result in the whole score also being 0, can become a problem, particularly because the larger the *n*-grams considered, the most likely it is to produce a 0 score. This is why BLEU scores are commonly calculated over an entire test set.

As a final note, BLEU scores are commonly calculated over an entire test set in order to reduce the probability of obtaining an *n*-gram precision of 0, something that is most likely to happen for the larger *n*-grams considered. Avoiding a precision of 0 is necessary because the whole score would also be 0.

2.3 FCT ALIGNMENT

As mentioned before, two texts in different languages are parallel if each one is the translation of the other, and their alignment divides them into text segments that should continue to be translations of each other. As such, aligning a pair of parallel text C and D implies the determination of their corresponding segmentations $\{C_1, C_2, \dots, C_k\}$ and $\{D_1, D_2, \dots, D_k\}$ such that C_i is a translation of D_i, $\forall i: 1 \leq i \leq k$.

The most explored alignment methodologies, like (Och, Tillman and Ney, 1999), (Och and Ney, 2004) or (Koehn, 2004), assume that the beginning and the end of both parallel texts should align. As such, with the definition of a rectangle from points (0,0), $(0,|D|)$, $(|C|,|D|)$ and $(|C|,0)$, where $|C|$ and $|D|$ represent the last offsets of parallel texts C and D, respectively, then points (0,0) and $(|C|,|D|)$ define a diagonal from which alignment anchors should not be far. So, the chosen alignment anchors will be the ones closer to this "golden" diagonal, where the criteria for determining such closeness differ with author.

Those alignment algorithms determine paragraph and sentence boundaries by using all the information available from marked-up texts. Alignment anchors for sentence boundaries are then determined using Dynamic Programming algorithms and some other hypothesis. Once sentence boundaries have been set, the alignment follows the methodologies described above in Sub-Section 2.1.2.

In contrast, (Ribeiro, Lopes and Mexia, 2000b), (Ribeiro, Lopes and Mexia, 2000c), (Ribeiro, Lopes and Mexia, 2000e), (Ribeiro, Lopes and Mexia, 2000f), (Ribeiro, Lopes and Mexia, 2000g), (Ribeiro et al., 2001) and (Ribeiro, 2002) consider the alignment problem as a global restriction on possible alignment anchors that should be near the golden diagonal. A first approach (Ribeiro, Lopes and Mexia, 2000e) used as possible anchors just homograph tokens (like numbers, proper names and punctuation signs) having the same number of occurrences in the two parallel texts, enabling (Ribeiro, 2002) to determine by linear regression the best fit to be closer the golden diagonal. Then, by applying statistical filtering algorithms, outliers were removed while unfiltered pairs were used as anchors to break the whole texts into smaller segments. The same procedures were applied recursively to these text segments until no more anchors were found.

In a subsequent approach (Ribeiro et al., 2001), the number of candidate alignment anchors was considerably enlarged by taking into account possible cognates with identical frequency. This decision was considered because European languages have a significant number of cognates (words having similar forms

and meaning the same, as is the case for "constitution" and "constituição"), for which homographs are a particular case. Following this idea, (Ribeiro et al., 2001) extracted gapped and contiguous homograph sequences of characters from the joining of the two texts to be aligned. The technique employed in extracting these sequences was the one which originated Gael's Ph.D. Thesis (Dias, 2002) and (Silva et al., 1999) now applied to character sequences instead of word sequences. The objective was to identify possible cognates.

By observing that the Confidence Band filter from the previous approach was processing demanding and discarded a significant number of perfectly good candidate alignment anchors, (Ildefonso and Lopes, 2005) substituted that filter by the Longest Sorted Sequence Algorithm and used the Levenshtein Distance (or edit distance) to determine possible candidate cognates, instead of using the method proposed by (Ribeiro et al., 2001) for cognate extraction.

In the latest approach, the FCT Aligner (Gomes, Aires and Lopes, 2009), statistical filters were dropped as well as the hypothesis of identical number of single-word and multi-word translations, that had been previously automatically extracted and validated and were used as candidate anchors. Also, it simplified many procedures, abandoned the filters used as mentioned above, and established the phrase as the unit considered for alignment anchors. This alignment approach was named after the institution in which it was developed: FCT/UNL.

The FCT Aligner, used by Transtor, produces a symmetrical phrase-based alignment, relying on validated bilingual phrase lexica to identify relations between phrases of parallel texts to produce a monotonic alignment. The relations between text fragments are expressed through the alignment segments, which are classified as "recognized" if the relations result from any evidence indicating that the text fragments involved constitute a translation equivalent.

EN	PT
Eurojust ' s mission shall be to support and strengthen coordination and cooperation between national investigating and prosecuting authorities in relation to serious crime affecting more than one European country .	A Eurojust tem por missão apoiar e reforçar a coordenação e a cooperação entre as autoridades nacionais competentes para a investigação e o exercício de a acção penal em matéria de criminalidade grave que afecte mais de o que um país Europeu .

Table 1. Fragment of parallel texts.

Table 1 shows a sample of a pair of parallel texts, in English and Portuguese. The equivalence between those texts can be further refined, establishing correspondences between smaller parallel text fragments as the ones produced by

the FCT Alignment presented in Table 2, where the "recognized" column contains '*' characters marking phrase pairs considered to be translations because some evidence has been found confirming such relation, like being found in a validated bilingual phrase lexicon used to assist in the alignment process. Such recognized phrase pairs are called "alignment anchors".

Any segments occurring between recognized segments (or between a recognized segment and either the beginning or the end of the text) are considered to be implicit, because they are not directly supported by acquired and validated translation knowledge about word and multi-word translations. Those implicit segments can occur either because they are not indeed proper translations, because the aligner did not have enough evidence to identify them as translations, or because of an error resulting from some misalignment, like the one in the example shown in Table 4, described ahead.

#	EN	recognized	PT
1	Eurojust ' s		A Eurojust tem por
2	mission	*	missão
3	shall be to support		apoiar
4	and	*	e
5	strengthen		reforçar
6	coordination and cooperation	*	a coordenação e a cooperação
7	between	*	entre
8	national investigating and prosecuting		
9	authorities	*	as autoridades
10			nacionais competentes para a investigação e o exercício de a acção penal
11	in relation to	*	em matéria de
12			criminalidade
13	serious	*	grave
14	crime		
15	affecting	*	que afecte
16	more than	*	mais de o que
17	one	*	um
18	European country	*	país Europeu
19	.	*	.

Table 2. An alignment example

Even though the FCT Aligner was unable to find evidence to recognize the implicit segments, many of those segments represent correct phrase translations. Examples of both situations can be found on Table 2 above, where implicit entry 5 corresponds to a correct equivalent ("strengthen" is a translation for

"reforçar") while implicit entry 3 does not ("shall be to support" is not a translation for "apoiar"). Additionally, a recurring pattern in the English/Portuguese language pair, already explored in (Aires, Lopes and Gomes, 2009), can be identified from the structure of the alignment segments, consisting of a word or phrase aligning with nothing before and after an alignment anchor. Entries from Table 2 "" <=> "criminalidade" (entry 12), "serious" <=> "grave" (entry 13), and "crime" <=> "" (entry 14) verify such pattern and, guided by it, it is possible to extract the correct translation equivalent "serious crime" <=> "criminalidade grave".

Exploring the above allows the identification of additional phrase translations which contribute to a higher phrase translation coverage, or higher recall. The extraction of phrase translations is described ahead in more detail (Section 3.3.1), and those phrase translations are then used to compose translations of complete texts (Section 3.4).

However, as noted above, these implicit segments do not always correspond to correct equivalents and should be considered with caution, simultaneously trying to avoid incorrectly associated entries while trying to avoid discarding correct associations (Section 3.3.2.1).

Another important feature of the FCT Aligner is its ability to improve the alignment quality with the improvement of the verified bilingual phrase lexicon. Table 3 shows a small sample of a pair of parallel texts that will be used to illustrate how the alignment evolved between iterations in which new information is included in the verified bilingual phrase lexicon.

EN	PT
… use the following bridging tables in their regular monitoring of the consistency between …	… utilizam as seguintes tabelas de correspondência a o controlarem regularmente a coerência entre …

Table 3. Sample of a pair of parallel texts

Table 4 shows an alignment, obtained for the pair of parallel texts illustrated in Table 3, which suffers from some misalignment. Such situation results from the incorrect association between the English article "the" and the Portuguese preposition "a" in segment 9, when the correct association would have been between the English article "the" in segment 9 and the Portuguese article "a" in segment 10. Without enough information, the choice is ambiguous and, in this case, the wrong choice was made.

#	EN	recognized	PT
1	use	*	utilizam
2	the	*	as
3	following	*	seguintes
4	bridging		
5	tables	*	tabelas
6	**in their regular monitoring**		
7	**of**	*	de
8			correspondência
9	the	*	**a**
10			**o controlarem regularmente** a
11	consistency	*	coerência
12	between	*	entre

Table 4. First iteration alignment

However, the alignment is significantly improved when the equivalent "in their regular monitoring of" <=> "a o controlarem regularmente" (a correct nonliteral translation, highlighted in bold in Table 4) is inserted, in which case a subsequent processing of the aligner produces the alignment depicted in Table 5, where the mentioned phrase translation equivalent is captured in segment 7, improving the alignment quality and identifying the translation equivalent "the" <=> "a", in segment 8, between the correct instances of "the" and "a".

#	EN	recognized	PT
1	use	*	utilizam
2	the	*	as
3	following	*	seguintes
4			tabelas de
5	bridging	*	correspondência
6	tables		
7	**in their regular monitoring of**	*	**a o controlarem regularmente**
8	the	*	a
9	consistency	*	coerência
10	between	*	entre

Table 5. Improved alignment after iteration

While in situation depicted in Table 4, the alignment was made by the known translation pair "tables" <=> "tabelas" due to local misalignment, in Table 5, due to better alignment, the aligner prefers to use the translation pair "bridging" <=> "correspondência" because it is longer, in number of characters than "tables" <=> "tabelas". Such improvement allows the identification of the equivalent "bridging tables" <=> "tabelas de correspondência" in a sub-sequent use of the re-aligned parallel text to extract con-

tiguous phrase translation equivalents (Section 3.3.1). With the additional identified equivalent, the alignment would be further improved, as shown in Table 6.

#	EN	recognized	PT
1	use	*	utilizam
2	the	*	as
3	following	*	seguintes
4	bridging tables	*	tabelas de correspondência
5	**in their regular monitoring of**	*	**a o controlarem regularmente**
6	the	*	a
7	consistency	*	coerência
8	between	*	entre

Table 6. Further alignment improvement

As a conclusion, it should be pointed out that other alignment procedures do not keep any memory of previously identified translation equivalents, so retraining those translation engines might produce slightly different results simply because of approximation techniques like hill-climbing, but any improvement will never be significant. With the FCT Alignment, using translation pairs that were previously memorized allows alignments to evolve and improve at each subsequent retraining process and, with it, the quality of newly extracted translation pairs and the quality of the translations made also improves.

2.4 INDEXING STRUCTURES

Several calculation stages in Transtor require the identification of unique phrases, fast counting of occurrences and fast access to all their occurrences. A unique phrase is considered a unique word or multi-word term despite the number of occurrences the phrase has in a text. As an example, consider the text "`rose is a rose is a rose is a rose`". The text has 3 unique phrases of one word ("`rose`", "`is`", "`a`") and 1 unique phrase of 3 words ("`is a rose`") which occurs 3 times.

Many of those requirements have already been met in previous work done in phrase translation extraction (Aires, Lopes and Gomes, 2009) and, before that, in multi-word expressions extraction (Aires, Lopes and Silva, 2008), which involved the use of Suffix Arrays and other support structures. Such previous work was also reused and adapted to support the following Transtor tasks:

- extraction of contiguous phrase translations from aligned parallel corpora (Section 3.3.1);
- calculation of the phrase translation model (Section 3.3.2.1); and
- calculation of the phrase language model (Section 3.3.2.2).

This section is dedicated to the description of the structures, in their original character-based form, for a generic text *T* having a generic size of *N* characters. Suffix Arrays (Sub-Section 2.4.1) are the basic indexing structures, having additional structures built on top of them to support and improve some text analysis features, like term frequency, occurrence identification and unique phrase identification. Those additional structures consist of the LCP Array (Sub-Section 2.4.2), the Suffix Class Array (Sub-Section 2.4.3) and the Term Array (Sub-Section 2.4.4). Each individual structure is described in the following subsections, using text *T* = `to_be_or_not_to_be` when necessary to exemplify some concepts, where the '_' character represents a space.

2.4.1 SUFFIX ARRAY

Suffix Arrays (Manber and Myers, 1990) are an indexing structure, calculated using a very efficient suffix sort algorithm (Larsson and Sadakane, 1999), allowing efficient access to term occurrences and efficient determination of term frequency. To better understand this structure, consider the text *T* along with the offsets of each character, as shown in Table 7.

T	t	o	_	b	e	_	o	r	_	n	o	t	_	t	o	_	b	e
i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17

Table 7. Text *T* with its offsets

First of all, each offset *i* represents a suffix from *T*, consisting on a string composed by the characters from offset *i* through offset *N*–1, represented by *T*[*i*, *N*–1], or simply *T*[*i*]. Table 8 shows the suffixes of *T*.

i	*T*[*i*]
0	to_be_or_not_to_be
1	o_be_or_not_to_be
2	_be_or_not_to_be
3	be_or_not_to_be
4	e_or_not_to_be
5	_or_not_to_be
6	or_not_to_be
7	r_not_to_be
8	_not_to_be
9	not_to_be
10	ot_to_be
11	t_to_be
12	_to_be
13	to_be
14	o_be
15	_be
16	be
17	e

Table 8. Suffixes from *T*

Each index *i* of a Suffix Array contains an index *SA*[*i*] of text *T*, representing a suffix *T*[*SA*[*i*]]. The Suffix Array will store the suffixes in lexicographical order such that *T*[*SA*[*i*]] < *T*[*SA*[*i*+1]], as shown in Table 9. Such order allows efficient binary phrase search in $P \cdot log(N)$ time complexity, with *P* corresponding to the character length of the phrase being searched.

i	*SA*[*i*]	*T*[*SA*[*i*]]
0	15	_be
1	2	_be_or_not_to_be
2	8	_not_to_be
3	5	_or_not_to_be
4	12	_to_be
5	16	be
6	3	be_or_not_to_be
7	17	e
8	4	e_or_not_to_be
9	9	not_to_be
10	14	o_be
11	1	o_be_or_not_to_be
12	6	or_not_to_be
13	10	ot_to_be
14	7	r_not_to_be
15	11	t_to_be
16	13	to_be
17	0	to_be_or_not_to_be

Table 9. Suffix Array of *T*

As a consequence of the lexicographical order, references to common prefixes of suffixes are located next to each other. Some examples are the range of entries 10 through 13 of Table 9, which share the common prefix "o", the range of entries 15 through 17, which share the common prefix "t", and entries 16 and 17, which share the prefixes enumerated in Table 10. The Suffix Classes (Sub-Section 2.4.3) take advantage of this locality property to efficiently identify unique phrases sharing the same frequency.

prefix
to_be
to_b
to_
to
t

Table 10. Character prefixes of the phrase to_be

The Suffix Array will have some of its features enhanced by the structures described in the following sub-sections: the LCP Array, the Suffix Class Array and the Term Array. Those structures will improve the identification of unique phrases, the determination of the term frequency of those unique phrases, and the identification of every occurrence of a given unique phrase.

2.4.2 LCP ARRAY

The Longest Common Prefix (LCP) between two character strings consists of the number of contiguous common characters, determined from the start of both strings. In other words, the counting process starts from the first characters of both strings and increments the count for every common character between those strings until a different one is found. Table 11 shows a few examples, where the common prefix between "phrase 0" and "phrase 1" is highlighted in bold and explicitly shown in "common prefix".

phrase 0	phrase 1	common prefix	LCP
abc	**ab**d	ab	2
abc	bbc	<empty>	0
abc	**a**cb	a	1

Table 11. lcp values for some phrases

The LCP Array (Yamamoto and Church, 2001) keeps the LCP values for every adjacent Suffix Array entry, in which $LCP[i]$ consists of the *lcp* value between suffixes $T[SA[i]]$ and $T[SA[i-1]]$. Table 12 shows the LCP Array of the Suffix Array depicted in Table 9. An LCP Array will have $N+1$entries, with its limit values set to 0. The limit values are:

- $LCP[0]$ – referring to suffixes $T[SA[-1]]$ and $T[SA[0]]$, in which $SA[-1]$ is out of the Suffix Array bounds; and
- $LCP[N]$ – referring to suffixes $T[SA[N-1]]$ and $T[SA[N]]$, in which $T[SA[N]]$ is out of the Suffix Array bounds.

As an example, entry 17 from Table 12 corresponds to suffix array entries 16 and 17 represented in Table 9. Those entries correspond to "`to_be`" and "`to_be_or_not_to_be`", respectively, having "`to_be`" as their longest common prefix. This common prefix has a length of 5 characters and, therefore, the LCP value between those entries is 5.

i	*T*[*SA*[*i* – 1]]	*T*[*SA*[*i*]]	common prefix	*LCP*[*i*]
0	<out of bounds>	_be	<empty>	0
1	_be	_be_or_not_to_be	_be	3
2	_be_or_not_to_be	_not_to_be	_	1
3	_not_to_be	_or_not_to_be	_	1
4	_or_not_to_be	_to_be	_	1
5	_to_be	be	<empty>	0
6	be	be_or_not_to_be	be	2
7	be_or_not_to_be	e	<empty>	0
8	e	e_or_not_to_be	e	1
9	e_or_not_to_be	not_to_be	<empty>	0
10	not_to_be	o_be	<empty>	0
11	o_be	o_be_or_not_to_be	o_be	4
12	o_be_or_not_to_be	or_not_to_be	o	1
13	or_not_to_be	ot_to_be	o	1
14	ot_to_be	r_not_to_be	<empty>	0
15	r_not_to_be	t_to_be	<empty>	0
16	t_to_be	to_be	t	1
17	to_be	to_be_or_not_to_be	to_be	5
18	to_be_or_not_to_be	<out of bounds>	<empty>	0

Table 12. LCP Array of Suffix Array from text T

The main purpose of this structure is to assist in the construction of the Suffix Class Array, described in Sub-Section 2.4.3 below.

2.4.3 SUFFIX CLASS ARRAY

A Suffix Class represents a set of Suffix Array prefixes sharing the same frequency. The Suffix Class Array is calculated by an algorithm (Yamamoto and Church, 2001) based on suffix arrays for computing the term frequency (and other statistics, like document frequency) for all substrings in a corpus in $O(N \log N)$ time, even though there are $N(N + 1)/2$ such substrings in a corpus of size N, grouping the $N(N + 1)/2$ substrings into at most $2N - 1$ equivalence classes. By grouping substrings in this way, many of the statistics of interest can be computed over the relatively small number of classes, which is manageable, rather than over the quadratic number of substrings. Table 13 shows the Suffix Class Array of *T*.

i	LBL	SIL	*lb*	*rb*	*tf*	phrase
0	0	1	0	4	5	_
1	1	3	0	1	2	_be
2	3	16	1	1	1	_be_or_not_to_be
3	1	10	2	2	1	_not_to_be
4	1	13	3	3	1	_or_not_to_be
5	1	6	4	4	1	_to_be
6	0	2	5	6	2	be
7	2	15	6	6	1	be_or_not_to_be
8	0	1	7	8	2	e
9	1	14	8	8	1	e_or_not_to_be
10	0	9	9	9	1	not_to_be
11	0	1	10	13	4	o
12	1	4	10	11	2	o_be
13	4	17	11	11	1	o_be_or_not_to_be
14	1	12	12	12	1	or_not_to_be
15	1	8	13	13	1	ot_to_be
16	0	11	14	14	1	r_not_to_be
17	0	1	15	17	3	t
18	1	7	15	15	1	t_to_be
19	1	5	16	17	2	to_be
20	5	18	17	17	1	to_be_or_not_to_be

Table 13. Suffix Class Array of Suffix Array from text T

The column headers represent the following:

- LBL (Longest Bounding LCP): only prefixes with length greater than this number belong to the Suffix Class. For instance, Suffix Class 7, with an LBL of 2, represents phrases "be_", "be_o", …, "be_or_not_to_be", excluding prefixes "b" and "be" because their lengths are not greater than 2, belonging to Suffix Class 6.
- SIL (Shortest Interior LCP): corresponds to the length limit of the prefixes represented by the Suffix Class.
- *lb*: the index of the left most Suffix Array entry having a prefix belonging to the Suffix Class. As an example, the *lb* of Suffix Class 11 is 10.
- *rb*: the index of the right most Suffix Array entry having a prefix belonging to the Suffix Class. As an example, the *rb* of Suffix Class 11 is 13.
- *tf*: the term frequency of all the phrases represented by the Suffix Class, corresponding to *rb*–*lb*+1.

prefix
t_to_be
t_to_b
t_to_
t_to
t_t
t_

Table 14. Prefixes represented by Suffix Class 18

As another example, Table 14 shows the character prefixes represented by the Suffix Class Array entry 18 on Table 13, to which prefix "t" does not belong because it does not share the same frequency, being represented by the Suffix Class Array entry 17. Each Suffix Class identifies the occurrences of the prefixes it represents through the range of Suffix Array indices represented by *lb* and *rb*.

A Suffix Class Array is built with the aid of the LCP Array and there can be at most 2*N*–1 Suffix Classes, as shown in (Yamamoto and Church, 2001). Now, the fact that common prefixes with a common frequency are represented by the same Suffix Class poses a problem when those common prefixes need to be distinguished. The Term Array was designed to overcome such limitation, described in Sub-Section 2.4.4 below.

2.4.4 TERM ARRAY

As noted above, two different phrases in which one is a prefix of the other and having both the same frequency are represented by the same Suffix Class, but it might be necessary to deal with both phrases separately. For instance, if "presidente de a república" occurs the same number of times as "presidente", they will be represented by the same Suffix Class, but they need to be separated to allow the identification of their separate translations. The Term Array (Aires, Lopes and Silva, 2008) is built by unfolding every Suffix Class to show all their prefixes, and each such prefix originates a Term Array entry. As an example, Suffix Class entry 18 is unfolded into the prefixes it represents, shown in Table 14. Table 15 shows the Term Array of text *T*.

i	SC index	phrase
0	0	_
1	1	_b
2	1	_be
3	2	_be_
4	2	_be_o
5	2	_be_or
6	2	_be_or_
7	2	_be_or_n
8	2	_be_or_no
9	2	_be_or_not
10	2	_be_or_not_
11	2	_be_or_not_t
12	2	_be_or_not_to
13	2	_be_or_not_to_
14	2	_be_or_not_to_b
15	2	_be_or_not_to_be
16	3	_n
…	…	…
133	19	to
134	19	to_
135	19	to_b
136	19	to_be
137	20	to_be_
138	20	to_be_o
139	20	to_be_or
140	20	to_be_or_
141	20	to_be_or_n
142	20	to_be_or_no
143	20	to_be_or_not
144	20	to_be_or_not_
145	20	to_be_or_not_t
146	20	to_be_or_not_to
147	20	to_be_or_not_to_
148	20	to_be_or_not_to_b
149	20	to_be_or_not_to_be

Table 15. Term Array (partial)

The SC index is the index of the corresponding entry in the Suffix Class Array. A Term Array entry represents a unique phrase, allowing access to all the corresponding occurrences of the given phrase through the corresponding Suffix Class (through the SC index).

Introducing a limit of L characters to avoid the analysis of extremely large terms, and knowing that there will be at most $2N$–1 Suffix Classes (Yamamoto and Church, 2001), each Suffix Class will generate L Terms in the worst case, so the number of elements of this structure would be $(2N–1)L$. Knowing that L is a constant much smaller than N, the resulting size complexity is $O(N)$.

2.5 CONCLUSIONS

This section discusses the main SMT concepts along with their implementations, introduced in the previous sub-sections, focusing on flaws and proposing improvements, this way preparing for the introduction of the advantages achieved by the alternatives supporting Transtor.

Starting by the probability score of a translation (Sub-Section 2.1.1), being defined as a product of probabilities (the log-linear model, described in Sub-Section 2.1.1.2, is an alternative way to represent a product of scores) of smaller parts or components, the main flaw that could be identified is its sensitivity to low values, reflecting those low values throughout a complete translation candidate being scored. In particular, null values have to be avoided, possibly introducing other errors. The alternative presented here (Sub-Section 3.4.7) consists in the definition of a simple score based on an average of feature models which prevents any low values from ruining the score of a generically acceptable translation candidate, even allowing them to be null. From my perspective, such an approach allows a better assessment of the quality of a translation candidate, as will be shown ahead.

In case of the alignment and extraction of translation equivalents, the original word-based approaches (Sub-Section 2.1.2.1) are being gradually replaced by phrase-based ones (Sub-Section 2.1.2.2), but these are still mostly based on a primary word-based stage that is processed to produce a phrase-based result. Even the generalization proposal of the alignment template (Sub-Section 2.1.2.3) considers categories of words. None of these cases considers previous knowledge, neither in an earlier nor in a later stage. This is in contrast with the alignment approach (Section 2.3) that supports the system presented in this document, which considers phrases, not as a simple composition of words, but as basic translation units, and also considers previous knowledge already validated.

The widely used log-linear approach (Sub-Section 2.1.1.2) admits the inclusion of any number of translation feature models that will contribute to assess the quality of a translation candidate. Yet, the most common translation feature models are the translation model (Sub-Section 2.1.3.1), the language model

(Sub-Section 2.1.3.2), the reordering model (Sub-Section 2.1.3.3) and the penalty model (Sub-Section 2.1.3.4). In fact, the ability to include a great number of feature models might not necessarily be an advantage since the task of identifying relevant features can also be overwhelming (Sub-Section 2.1.3). Transtor, however, only focuses on a very simple set of feature models (translation model, language model and penalty model), also common amongst SMT approaches, but the ones supporting Transtor can be set apart from the main approaches, as will be seen ahead.

The translation model (Sub-Section 2.1.3.1) applied to phrases, besides considering the conditional probability of a phrase translation equivalent, on both translation directions, based on the number of times they have been associated in a given parallel corpus, it can also consider the lexical weighting by analyzing the individual words composing the phrases involved in the equivalent so, once again, words are considered. The translation model approach presented here (Sub-Section 3.3.2.1) will consider phrases as units not to be decomposed and will simply use a simple conditional probability based on the number of times they have been associated in a given parallel corpus aligned with the FCT Alignment (Section 2.3). The language model (Sub-Section 2.1.3.2) is another example in which the word-based approach prevails, instead of a truly phrase-based approach. Such model will consider the n-gram history of words composing a sentence translation candidate independently of how those words were produced, either from a single phrase of n words or from a composition of n individual words. The language model approach presented here (Sub-Section 3.3.2.2) distinguishes such different situations of sentence composition while considering combinations of phrases with one or more words and not just single words. The penalty model (Sub-Section 2.1.3.4) is yet another feature model that is frequently considered at the word level by most approaches whereas Transtor takes into account the number of phrases. In sum, most approaches claim to be phrase-based simply because they identify translation equivalents of phrases, but those phrase translation equivalents are mostly produced from word-based alignments, as well as the remaining models will also consider those phrases at their word level.

In this thesis, I argue that a word level approach is not the most adequate for translation since there are many phrases that should not be analyzed at the word level, as shown by the English/Portuguese translation equivalents "`in accordance with`" **<=>** "`de acordo com`", "`however`" **<=>** "`em o entanto`", "`considering`" **<=>** "`tendo em conta`", or "`back to square one`" **<=>** "`de volta a a estaca zero`", to name only a few examples. As

such, phrases should be considered as units, not only while extracted but also on the remaining components and stages of the translation process. Such an approach is the one followed by Transtor, which also includes validated phrase translation lexica, mainly because phrase translations should not have to be learned over and over again. The inclusion of supervision by validating translation pairs extracted and classified and by including post-edited versions of new texts translated by the system provides an additional level of quality.

The most popular decoding algorithm (Sub-Section 2.1.4) produces partial hypotheses on-demand and requires a cover vector to keep track of the words already covered in the translation. In contrast, Transtor will first produce a complete graph of the possible translations which will then be traversed with a few optimizations, while the phrases covered are implicitly tracked by the graph (Sub-Section 3.4.8).

The representation power of a tree-based approach implementation like the hierarchical approach (Sub-Section 2.1.5.1) is ensured at the cost of representing every phrase translation as a rule and at the cost of including some special set of rules (the glue rules). Such decision implies carrying out its decoding as chart parsing because there is no obvious way to build the translation from left to right as with beam search decoding. In contrast, the translation patterns (Sub-Section 3.1.1.2) provide Transtor with additional representation power by simply enabling the rearrangement of the translation graph. This allows the initial contiguous phrase translation equivalents (Sub-Section 3.1.1.1), which cover a significant number of cases, to be used as always while leaving unchanged the initial decoding algorithm developed for Transtor (Sub-Section 3.4.8).

Additionally, the use of suffix arrays and their related structures (Sub-Section 2.4) provides an efficient and compact support for the retrieval operations required by Transtor.

3 TRANSLATION PROCESS

The translation process presented in this thesis, Transtor, has several essential procedures and components that differ from the ones presented in the state-of-the-art, allowing Transtor to obtain the competitive results presented in Chapter 4. It is possible to state that the translation process is divided into three important steps: pre-processing, training and translation.

In this section, I introduce the implementation details of Transtor, distinguishing the three mentioned stages of the complete process. After the details, Section 3.5 presents the main differences between Transtor and Moses, the most used state-of-the-art translation engine, as well as the justifications for the choices made in Transtor implementation.

3.1 UNDERLYING CONCEPTS

Before moving to the details, I introduce some key concepts which are crucial for the several stages of the translation process, namely the notion of phrase translation equivalents, contiguous and non-contiguous, the importance of a verified bilingual phrase lexicon and the data structures presented in Section 2.4, with some adaptations made specifically for Transtor.

3.1.1 PHRASE TRANSLATION EQUIVALENTS

A phrase translation equivalent consists of a pair of phrases, in different languages, where each phrase is a translation of the other (some examples are "`house`" <=> "`casa`" and "`considering`" <=> "`tendo em conta`"). Phrase translation equivalents are the building blocks of the system presented here since translations of complete texts are produced by dividing the text into phrases (taken as sequences of words, not necessarily with any linguistic or grammatical sense), obtaining the translations of those phrases and combining those phrase translations to produce the translation of the whole text. These equivalents can either be contiguous (Sub-Section 3.1.1.1) or non-contiguous (Sub-Section 3.1.1.2), where both types complement each other and are explained in detail in the sub-sections below.

3.1.1.1 Contiguous Phrase Translation Equivalents

Both phrases composing a contiguous phrase translation equivalent are contiguous, as implied by the type name. Table 16 shows some examples for the English-Portuguese language pair, which has some highlighted parts that will become evident in the next sub-section.

#	EN	PT
1	house	casa
2	park	parque
3	considering	tendo em conta
4	in case of	em o caso de
5	in accordance with	de acordo com
6	with respect to	em o que toca a
7	in spite of	apesar de
8	European Community	Comunidade Europeia
9	European Council	Concelho Europeu
10	serious crime	criminalidade grave
11	international law	legislação internacional
12	social policy	política social
13	**turn** *the light* **off**	**desligar** a luz
14	**turn** *all lights* **off**	**desligar** *todas as luzes*
15	**turn** *each light* **off**	**desligar** *cada luz*
16	**turn** *every light* **off**	**desligar** *todas as luzes*
17	**turn** *many lights* **off**	**desligar** *muitas luzes*
18	**turn** *some lights* **off**	**desligar** *algumas luzes*
19	**bring** *a case* **against**	**instaurar** *um caso* **contra**
20	**bring** *the process* **against**	**instaurar** *o processo* **contra**
21	**bring** *all cases* **against**	**instaurar** *todos os casos* **contra**

Table 16. Contiguous phrase translation equivalents

Contiguous phrase translation equivalents cover a significant number of translation equivalents, which can be composed by one of the following: a pair of single-words; a single-word and a multi-word; or a pair of multi-words.

Equivalents composed by a pair of single-words are mandatory (like entries 1 and 2 in Table 16), but there are also many situations in which a single-word requires a multi-word translation and vice-versa, like "considering" **<=>** "tendo em conta" (entry 3 in Table 16), strengthening the idea that some multi-word phrases should be considered units. Now, translations involving a pair of multi-words are also very common. In some cases, those equivalents can result from the composition of smaller translation equivalent units, like "in case of" **<=>** "em o caso de" (entry 4 in Table 16), sometimes involving reordering, like "European Community" **<=>** "Comunidade Europeia" (entry 8 in Table 16). Still, such equivalents are included in the verified bilingual phrase lexicon because they can be used very frequently, this way avoiding additional processing with a simple retrieval operation over such lexicon.

Nonetheless, for translation equivalents involving a pair of multi-words, establishing internal relations is not always clear, for which idiomatic expressions are very good examples. Some of those equivalents only allow a partial internal

equivalence relation, like "in accordance with" <=> "de acordo com" (entry 5 in Table 16), in which internal equivalence relations "accordance" <=> "acordo" and "with" <=> "com" can be correctly established, leaving "in" to be related to "de", but this last one is generally not correct unless it occurs before "accordance with" <=> "acordo com". There are some other examples in which "in" and "de" might be considered equivalents, like "in many ways" <=> "de muitas maneiras", but there are many more examples in which the relation is incorrect, like "in the house" <=> "em a casa". Other translation equivalents involving a pair of multi-words do not allow any clearly correct internal equivalence, like "how old" <=> "que idade", reinforcing the idea of such equivalents being considered units.

In such cases where internal equivalence relations are not very clear, there is still the option of establishing them in order to be used to compose the larger ones, a decision that is complemented with the expectation that any models involved in the translation process will contribute to their correct selection, particularly because some of those internal relations can be very close translations (like "how" with "que", and "old" with "idade"), but this decision comes with the overhead of allowing many generally incorrect entries, so discarding such unclear internal relations would be more efficient than allowing them and depending on the models to use them properly.

In cases like the ones described so far, and because the phrases are contiguous, there is no problem in considering them as units (in the example, the unit would be "in accordance with" and "de acordo com"), avoiding any uncertain internal equivalence relations that would create an unwanted overhead. However, such a simple solution is not always very efficient. As an example, consider a similar case as the equivalent "in accordance with" <=> "de acordo com" described above, presented by the equivalent "bring a case against" <=> "instaurar um caso contra" (entry 19 in Table 16). In this last example, internal equivalence relations "a" <=> "um", "case" <=> "caso" and "against" <=> "contra" can be established, leaving open the option of relating "bring" with "instaurar". If this last relation was correct, the whole equivalent could be produced as a monotonic combination of the equivalents composing it, but since the equivalent "bring" <=> "instaurar" is not generally correct, allowing it could produce the same negative effects as allowing the equivalent "in" <=> "de" in the previous example, for which the solution was to consider the larger phrase as a unit.

The problem in this case is that, even though applying such a solution of considering the larger phrase would produce a correct unit, it would not be very efficient because it would not allow translating "`bring all charges against`", despite the fact that the English phrases only differ by two contiguous words ("`the case`" occurring in the resulting equivalent, and "`all charges`" occurring in the phrase to be translated). An alternative solution requires looking at the other similar examples (entries 20 and 21 in Table 16), in which their generalization shows that "`bring`" <=> "`instaurar`" should not be associated to the whole equivalent, but only to "`against`" <=> "`contra`", while allowing some other phrases in the middle: "`bring * against`" <=> "`instaurar * contra`". This introduces the concept of non-contiguous phrases, where some fixed phrase literals are separated by some variable phrases.

With the previous example, one might still argue that the generally incorrect equivalent "`bring`" <=> "`instaurar`" could be allowed while depending on the models to make the right choice, particularly because the monotonic composition of the larger equivalent can be obtained with such generally incorrect equivalent, this way avoiding the complexity of introducing non-contiguous phrases. However, avoiding generally incorrect equivalents is not the only advantage of non-contiguous phrases. In fact, examples like "`turn * off`" <=> "`desligar *`" (a generalization from entries 13 through 18 in Table 16), in which a single word is translated by a non-contiguous phrase, show that non-contiguous phrases are absolutely necessary, allowing phrase translation generalization. Another example illustrating such need is the French negation that requires the two non-contiguous words "`ne * pas`" to translate the Portuguese negation "`não *`".

Cases benefitting from non-contiguous phrases can be very common but are also the most challenging ones, requiring being handled with a different methodology. Phrase lexicon equivalents do not support their generalized application because they simply replace a fixed phrase by another, which motivated the development of translation patterns, described in Sub-Section 3.1.1.2 below.

3.1.1.2 Non-Contiguous Phrase Translation Equivalents

As mentioned in the previous sub-section, the non-contiguous phrase translations (or translation patterns) were introduced to support phrase translation generalizations. Those generalizations consist of a pair of non-contiguous phrases (pattern phrases), each containing the same number of variables, and a mapping establishing an association between the variables in one pattern

phrase to variables in the other pattern phrase, which provides support for the order change of variables. Both pattern phrases can be used either as a source or as a target, where the translation of a phrase, associated to a given source variable, will be associated to its corresponding target variable, on the target phrase.

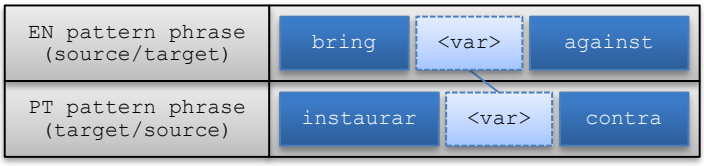

Figure 4. Example of a translation pattern

As an example, entries 19 through 21 in Table 16 have some common phrases (highlighted on the table), which allow those entries to be generalized to produce the translation pattern depicted in Figure 4, where the variables on each pattern phrase, connected by an edge, represent the contiguous phrase translation equivalents not shared between those entries ("`a case`" **<=>** "`um caso`", "`all cases`" **<=>** "`todos os casos`", and so on). In this case, as mentioned earlier, the relation between "`bring`" and "`instaurar`" is intrinsically dependent on this phrase translation pattern, which includes "`against`" and "`contra`". With this feature, it is not established any kind of equivalence relation between "`bring`" and "`instaurar`" outside the translation pattern. From my point of view, allowing such equivalence relation while depending on other methodologies like, for instance, word sense disambiguation (Casteleiro, Lopes and Silva, 2014), will only increase the complexity of the solution, so it is preferable to avoid those generally incorrect equivalence relations, something that is addressed in more detail in Sub-Section 3.1.1.2.4 below.

As another example, entries 13 through 18 in the same table share the (single-word) phrases "`turn`" and "`off`" on the English part, and "`desligar`" on the Portuguese part, enabling those entries to be generalized to the non-contiguous phrase translation equivalent "`turn <var> off`" **<=>** "`desligar <var>`", depicted in Figure 5.

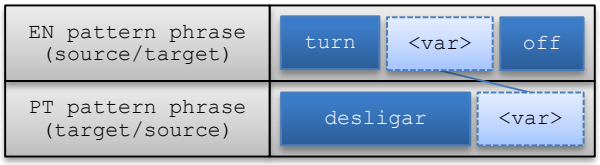

Figure 5. Another example of a translation pattern

This translation pattern allows the translation of the phrase "`turn all lights off`" even if its literal translation is not available in the verified bilingual phrase lexicon, as long as it is possible to translate "`all lights`": the

contiguous phrase "all lights" would be associated to the variable in the English phrase, translated as "todas as luzes" (if available), and placed after "desligar" in the Portuguese phrase. However, it should be noted that the generalization process described here is only used to explain the motivation behind the feature, since Transtor is only responsible for using translation patterns that have been previously and separately identified.

The following sub-chapters will describe the translation pattern feature in more detail, namely its structure and its constraints (Sub-Section 3.1.1.2.1), the aspects of its application (Sub-Section 3.1.1.2.2), the numerical translation patterns (Sub-Section 3.1.1.2.3), and discuss some final considerations about the translation pattern feature (Sub-Section 3.1.1.2.4).

3.1.1.2.1 Translation Pattern Composition

As mentioned above, translation patterns are composed by a pair of non-contiguous phrases and a mapping. Each non-contiguous phrase is composed by a number of variable elements separated by literal elements. The variable elements (represented by both <var> in either Figure 5 or Figure 4) represent phrase sections that can hold a yet unknown phrase, while the literal elements represent phrase elements that have to be matched exactly ("instaurar" or "against" in Figure 4), providing lexical evidence about the pattern application to a candidate. Both non-contiguous phrases composing a translation pattern will have the same number of variable elements, with the mapping establishing a relation of translation equivalence between a variable element in one non-contiguous phrase and a variable element in the other non-contiguous phrase, with every variable element in one non-contiguous phrase being connected to only one variable element in the other non-contiguous phrase. Adjacent variable elements are not allowed for translation purposes, meaning that they must always be separated by a literal element, this way avoiding ambiguity and also improving efficiency. Both non-contiguous phrases can either work as a source or as a target. Table 17 shows some translation pattern examples for the English-Portuguese language pair.

#	EN (source / target)	PT (target / source)
1	bring <var> against	instaurar <var> contra
2	shut <var> off	desligar <var>
3	allow <var> to be attained	atingir <var>
4	<var> should be awaited	é oportuno esperar por <var>
5	bring <var> to the attention of	comunicar <var> a
6	bring <var> closer	aproximar <var>
7	turn <var> off	desligar <var>

Table 17. Translation pattern examples

Empty variables are not allowed on translation pattern applications, meaning that a variable instantiation requires a non-empty contiguous phrase. This decision was taken mainly because allowing empty variables will turn non-contiguous phrases into contiguous ones, and these are highly likely covered by the verified bilingual phrase lexicon (Sub-Section 3.1.2). This way redundancy is avoided in favor of lexicon entries because they are applied more efficiently.

Another reason to avoid empty variables has to do with the fact that it is not very clear that it will always produce correct translations. As examples, entry 1 in Table 17 can produce a perfectly fine contiguous phrase translation equivalent, but entry 3 in the same table and, most definitely entry 4, need an element to which the verb action is applied, in which case the empty variable should not be allowed.

Additionally, in the case of entry 2 in the table above, the empty variable will result in the contiguous phrase translation equivalent "`shut off`" <=> "`desligar`", but contiguous "`shut off`" should also have as contiguous phrase translation equivalents "`desligado`", "`desligada`", "`desligados`" and "`desligadas`", which should all be present in the verified bilingual phrase lexicon.

For the reasons above, translation patterns are required to have all their variables instantiated, leaving contiguous phrase translation equivalents to the verified bilingual phrase lexicon.

3.1.1.2.2 Translation Pattern Application

The application of a translation pattern to a candidate phrase to be translated has to consider the matching of the source phrase to the candidate according to the fixed parts and identifying the variable parts.

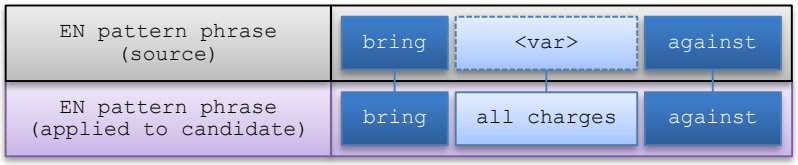

Figure 6. Matching of a candidate phrase

Taking the English phrase from the translation pattern depicted in Figure 4 and applying it to the candidate phrase "`bring all charges against`", as shown in Figure 6, both fixed parts ("`bring`" and "`against`") find a match on the candidate phrase. The remaining part (the phrase "`all charges`") is then associated to the variable part, resulting in the translation of the phrase "`all`

charges" being associated to the variable part on the Portuguese phrase from the same translation pattern.

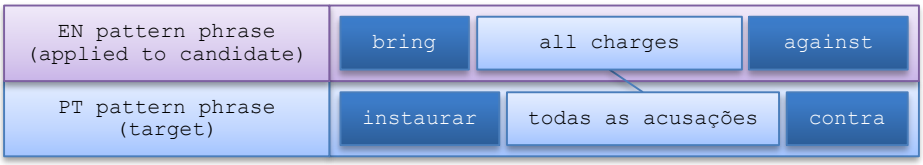

Figure 7. Application of a general translation pattern

The final application of the translation pattern, while having the contiguous phrase translation equivalent "all charges" <=> "todas as acusações", will produce the final phrase translation "instaurar todas as acusações contra", as shown in Figure 7, which is a correct translation.

The example above shows a situation in which the phrase associated to the variable ("all charges") is translated by a single phrase ("todas as acusações"), which simplifies the explanation of this translation pattern feature. However, this is not always the case, and the situations where there is more than one phrase translation available require making a choice that depends on other concepts not yet introduced. Such situations will be discussed in Section 3.4.5.

3.1.1.2.3 Numerical Translation Patterns

Considering the significant number of phrase translations that deal with numbers (law articles, measurements, and so on), and also considering how efficient the application of a translation pattern is when its variable elements consist of numbers, numerical translation patterns were also developed.

The numerical translation patterns represent non-contiguous phrases where all the variable elements refer to numbers and their efficiency comes from the fact that searching for a number (the variable element) simply requires a string match to the number tag, represented by the literal <number>, because the text is normalized (Sub-Section 3.2.2).

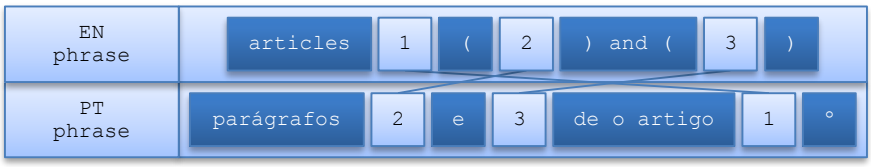

Figure 8. Numerical translation correspondences

The example in Figure 8 shows the phrase translation equivalent "articles 1 (2) and (3)" <=> "parágrafos 2 e 3 de o artigo 1 °", which shows how the numbers can change order and how some elements do not have

a correspondence with each other: the parentheses on the English phrase are not used on the Portuguese phrase; "parágrafos" on the Portuguese phrase does not have a correspondence on the English phrase; and even "articles" (plural) is not equivalent to "artigo" (singular). Such lack of correspondences could be tackled with the flexibility option described ahead (Sub-Section 3.1.1.2.4), but its negative consequences would far surpass its benefits.

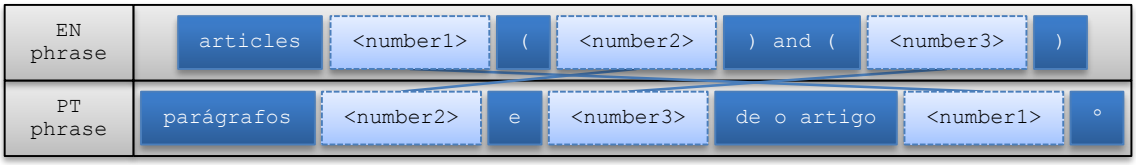

Figure 9. Numerical translation pattern

The phrase translation equivalence in Figure 8 is supported by the numerical translation pattern "articles <number1> (<number2>) and (<number3>)" **<=>** "parágrafos <number2> e <number3> de o artigo <number1> °", depicted in Figure 9, where the number contained on source <number1> will be placed on target <number1>, and so on.

EN	PT
article <number>	o artigo <number> °
article <number1> (<number2>)	parágrafo <number2> de o artigo <number1> °
articles <number1> (<number2>) and (<number3>)	parágrafos <number2> e <number3> de o artigo <number1> °
<number> ml flask	balão de <number> ml
<number> kg weight	peso de <number> kg
<number> km away	a uma distância de <number> km

Table 18. Textual representation of numerical translation patterns

Table 18 shows the textual representation of a few examples of numerical translation patterns.

3.1.1.2.4 Advantages of Translation Patterns

Without the translation pattern feature, the best way to propose a translation for "bring a case against" would be to have the translation of the whole phrase, but this could not be used to deal with new cases like "bring all charges against". A more flexible alternative might try to take advantage of the fact that the literal elements "against" **<=>** "contra" are a phrase translation equivalent in their own right, which opens the possibility of considering the literal elements "bring" **<=>** "instaurar" also as a phrase translation equivalent and count on the language model or word sense disambiguation to be able to select it according to the context, even though this last equivalence can only be considered as such in the context of the shown translation pattern.

However, the flexibility of allowing such phrase translations with the purpose of them being used in very particular conditions (again, "bring" <=> "instaurar" only makes sense when occurring near "against" <=> "contra") has two immediate problems:

- It increases the possibility of producing wrong translations, along with an increase in the processing time for considering such additional phrase translations. Such overhead can be avoided by the pattern feature which will only allow the translation whenever the context conditions (established by the literal elements) are met.
- Admitting such phrase translations is not always possible, with Figure 5 showing such a situation. Unlike the previous case (Figure 4), in which a relation could be allowed between "bring" and "instaurar" (in spite of it being generally wrong), in this case one literal ("desligar") is translated by two literals ("turn" and "off") separated by a phrase, which means that one of those words (typically "off") will not have a corresponding literal. The numerical pattern example depicted in Figure 9 also shows that the flexibility option would not really help because allowing equivalents like "articles" <=> "parágrafos", "(" <=> "e", ") and (" <=> "de o artigo", or ")" <=> "°", would be a tremendous stretch, resulting in more problems than benefits. Such situations do not represent a problem to the translation pattern feature, because explicit relations between literals are not required.

With the introduction of the translation pattern feature, the problems described above can be avoided because such feature allows the generalization of translation equivalents by establishing relations between phrases that should be considered as non-contiguous units, which allows discarding generally incorrect translation equivalents like the ones mentioned above.

3.1.2 VERIFIED BILINGUAL PHRASE LEXICON

Supervision in Transtor is present in several levels: in the verified bilingual phrase lexicon; in the verification of parallel texts used as a base; and in the produced post-edited translations.

The verified bilingual phrase lexicon is a set of translation equivalents, verified by human translators and linguists, which classify translation candidates according to their correction. Many of the first verified entries came from previous contiguous phrase translations extractions carried out by the procedures developed in previous work (Aires, Lopes and Gomes, 2009). Many more are

periodically being extracted with new procedures being developed in the framework of the ongoing Ph.D. research of Luis Gomes.

The supervision introduced with the verification of phrase translation entries, either contiguous or non-contiguous, is justified not only because it allows a more accurate alignment (as shown in Section 2.3), but also because it allows the system to produce higher quality translations. As the base knowledge increases with the correctly evaluated and accepted translation candidates (the verified bilingual phrase lexicon gains new translation equivalents), the quality of subsequent extractions also improves.

The verified bilingual phrase lexicon used by Transtor is the result of the human validation of automatically extracted contiguous phrase translation equivalents, which can be classified as follows:

- **Accepted**: the phrase translation has been accepted as correct by some human validator.
- **Rejected**: the phrase translation has been considered as incorrect; this is used by the extractor of translation equivalents for validation, to avoid reconsidering translations that were already rejected, but might arise systematically as a consequence of the statistical methods used in the process.
- **Postponed**: equivalents that might raise some doubts to the evaluators and that require some more thought and attention.
- **Correct, but longer than necessary**: correct entries that are obtained by a monotonic composition of smaller equivalents. For example, "`a combined market share of`" **<=>** "`uma quota de mercado combinada de`" is longer than necessary because "`a`" translates as "`uma`", "`combined market share`" translates as "`quota de mercado combinada`" and "`of`" translates as "`de`". As such, the correct translation "`a combined market share of`" **<=>** "`uma quota de mercado combinada de`" is longer than necessary for alignment purposes.
- **Correct, but shorter than necessary**: actually incorrect entries that require one or more additional words to be entirely correct. This is mainly applied in the alignment of declined languages like German, Czech and others.
- **Unverified**: for those extracted entries that have not yet been validated.

All of the entries above, except the unverified ones, are very helpful in the FCT Alignment process as well, as they provide it with additional filters (either positive or negative) to improve the final result of the alignment and to avoid making the same mistakes over and over again.

Prior to the human validation there is a step of automatic classification which helps speeding up the validation process and increasing the precision of subsequent extractions and validations. Such classification is implemented with an SVM classifier, trained with the translation equivalents validated as correct and incorrect (Mahesh, Gomes and Lopes, 2011). The result of this automatic classification step is also a set of possibly correct and incorrect translation equivalents, which are used to filter the entries to be validated, since it is more productive for a human validator to look for correct occurrences in a set of candidates classified as possibly correct.

3.1.3 ADAPTED INDEXING STRUCTURES

The original indexing structures described in Section 2.4 are character-based (allowing full-text search and analysis) and were generically developed to allow the analysis of every string occurring in a text, regardless of the size of the strings. Such situation allows the inclusion of a great number of strings which can be considered too large, and therefore having from little to no statistical interest, significantly increasing the number of analyzed elements, without a corresponding significant gain. Another circumstance to consider was the fact that Western languages were the ones analyzed while developing Transtor, inspiring a word-based solution to better focus on complete words. These reasons led to the development of new versions of the indexing structures, adapted in order to: discard string entries corresponding to incomplete words; and introduce a limit on the number of words to avoid the analysis of indefinitely large strings.

Both changes contributed not only to keep the focus on the smaller strings (which are statistically more significant), but also to reduce the number of strings to analyze, with a corresponding reduction on the size of the structures. An additional change consisted in making the character string comparisons case insensitive, ensuring a match when comparing, for instance, `community` with `Community` or `COMMUNITY`.

As with the original versions of the indexing structures (Section 2.4), the following sub-sections will describe each structure individually for a generic text T of size N. When necessary, T = `_to_be_or_not_to_be_or_not_to_be` will be used to help in the description and a limit of 5 words will be used as an example. Note that the text T used in this section is different than the one used in Section 2.4 ("`or_not_to_be`" is repeated), changed with the purpose of highlighting some properties.

3.1.3.1 Adapted Suffix Array

As a consequence of dealing with complete words, Suffix Array entries referring to partial words became unnecessary. However, since determining the order between two suffixes requires access to their individual characters, even in a word-based solution, the suffix sort algorithm (Larsson and Sadakane, 1999), optimized for full-text indexing, was maintained. This way, the word-based Suffix Array of a text is obtained from the original character-based Suffix Array of the same text by simply discarding the entries corresponding to partial words. As an example, consider the character-based Suffix Array of *T*, shown in Table 19.

i	*SA*[*i*]	*T*[*SA*[*i*]]
0	29	_be
1	16	_be_or_not_to_be
2	3	_be_or_not_to_be_or_not_to_be
3	22	_not_to_be
4	9	_not_to_be_or_not_to_be
5	19	_or_not_to_be
6	6	_or_not_to_be_or_not_to_be
7	26	_to_be
8	13	_to_be_or_not_to_be
9	0	_to_be_or_not_to_be_or_not_to_be
10	30	be
11	17	be_or_not_to_be
12	4	be_or_not_to_be_or_not_to_be
13	31	e
14	18	e_or_not_to_be
15	5	e_or_not_to_be_or_not_to_be
16	23	not_to_be
17	10	not_to_be_or_not_to_be
18	28	o_be
19	15	o_be_or_not_to_be
20	2	o_be_or_not_to_be_or_not_to_be
21	20	or_not_to_be
22	7	or_not_to_be_or_not_to_be
23	24	ot_to_be
24	11	ot_to_be_or_not_to_be
25	21	r_not_to_be
26	8	r_not_to_be_or_not_to_be
27	25	t_to_be
28	12	t_to_be_or_not_to_be
29	27	to_be
30	14	to_be_or_not_to_be
31	1	to_be_or_not_to_be_or_not_to_be

Table 19. Character-based Suffix Array from *T*

Creating the word-based Suffix Array of *T* is as simple as keeping the suffix entries starting by a space character (in this case, entries 0 through 9), and discarding the remaining entries. The kept entries are then incremented by one so that the suffixes refer to the first character of the corresponding words and not the space characters preceding those words. The resulting word-based Suffix Array of *T* is represented in Table 20.

i	*SA*[*i*]	*T*[*SA*[*i*]]
0	30	be
1	17	be_or_not_to_be
2	4	be_or_not_to_be_or_not_to_be
3	23	not_to_be
4	10	not_to_be_or_not_to_be
5	20	or_not_to_be
6	7	or_not_to_be_or_not_to_be
7	27	to_be
8	14	to_be_or_not_to_be
9	1	to_be_or_not_to_be_or_not_to_be

Table 20. Word-based Suffix Array of *T*

This structure only refers to the beginning of each string, so changes to ensure a limitation on the number of words are only required in the remaining indexing structures, described in the following sub-sections.

3.1.3.2 Adapted LCP Array

Following the word-based Suffix Arrays, the entries of word-based LCP Arrays must also correspond to complete words. This means, as an example, that the word-based LCP between "Europe" and "European" will be 0 because the common prefix shared ("Europe") does not correspond to a complete word in "European". The word-based LCP value will consist of the number of characters of the complete words shared by a pair of character strings. The reason why the number of characters is used instead of a word count has to do with the fact that many required operations involving offset manipulation can be executed more efficiently. Some examples of LCP values are shown in Table 21.

#	phrase 0	phrase 1	common prefix	LCP
1	European_Community	European_Community	European_Community	18
2	European_Communities	European_Community	European	8

Table 21. Other LCP example

The first entry of the table has both words ("European" and "Community") shared by both phrases, so the LCP corresponds to the character length of the complete phrases which include the separating space (18), but the second entry only has the first word ("European") shared by both phrases, so the LCP value corresponds to the character length of that word (8).

i	*LCP*[*i*]	common phrase
0	0	<empty>
1	2	be
2	15	be_or_not_to_be
3	0	<empty>
4	9	not_to_be
5	0	<empty>
6	12	or_not_to_be
7	0	<empty>
8	5	to_be
9	18	to_be_or_not_to_be
10	0	<empty>

Table 22. Word-based LCP Array

Table 22 shows the word-based LCP Array of text *T*, calculated from the corresponding word-based Suffix Array shown in Table 20.

i	*LCP*[*i*]	common phrase
0	0	<empty>
1	2	be
2	15	be_or_not_to_be
3	0	<empty>
4	9	not_to_be
5	0	<empty>
6	12	or_not_to_be
7	0	<empty>
8	5	to_be
9	15	to_be_or_not_to
10	0	<empty>

Table 23. Limited word-based LCP Array

Table 23 shows the resulting LCP Array when a limit of 5 words is considered, where entry number 9 has now a length of 15 characters (corresponding to the 5 words limit), as opposed to entry number 9 in Table 22, where such word count limitation was not imposed.

With the introduction of the word count limitation, an adapted Suffix Array with *N* entries will have an adapted LCP Array with *N* + 1 entries.

3.1.3.3 Adapted Suffix Class Array

As with the previous adapted structures, the word-based Suffix Class Array required making sure the entries correspond to full words. This means that "europe" and "european" should belong do distinct entries even when they share the same *tf* because the only allowed word prefix of "european" is "european" itself.

i	LBL	SIL	*lb*	*rb*	*tf*	phrase
0	0	2	0	2	3	be
1	2	15	1	2	2	be_or_not_to_be
2	15	28	2	2	1	be_or_not_to_be_or_not_to_be
3	0	9	3	4	2	not_to_be
4	9	22	4	4	1	not_to_be_or_not_to_be
5	0	12	5	6	2	or_not_to_be
6	12	25	6	6	1	or_not_to_be_or_not_to_be
7	0	5	7	9	3	to_be
8	5	18	8	9	2	to_be_or_not_to_be
9	18	31	9	9	1	to_be_or_not_to_be_or_not_to_be

Table 24. Word-based Suffix Class Array

Table 24 shows the resulting word-based Suffix Class Array of text *T*, calculated from the corresponding word-based Suffix Array and word-based LCP Array.

i	LBL	SIL	*lb*	*rb*	*tf*	phrase
0	0	2	0	2	3	be
1	2	15	1	2	2	be_or_not_to_be
2	0	9	3	4	2	not_to_be
3	9	16	4	4	1	not_to_be_or_not
4	0	12	5	6	2	or_not_to_be
5	12	15	6	6	1	or_not_to_be_or
6	0	5	7	9	3	to_be
7	5	15	8	9	2	to_be_or_not_to

Table 25. Limited word-based Suffix Class Array

Table 25 shows the resulting Suffix Class Array when a limit of 5 words is considered, built with the assistance of the limited word-based LCP Array shown in Table 23. This Suffix Class Array can be seen as the result of applying the necessary changes to the Suffix Class Array shown in Table 24 in order to respect the limit of 5 words. In the example, such changes imply the removal of entries 2 and 9, since their terms within the given word count limit are already covered by their previous entries (1 and 8, respectively) and would also imply the adjustment of entries 4, 6, and 8 in order to only consider the terms within the same word count limit, becoming entries 3, 5 and 7, respectively, in Table 25.

With the introduction of the word count limitation, the worst case space complexity of the Suffix Class Array is not affected, so an adapted Suffix Array with N entries still produces a Suffix Class Array with at most $2N - 1$ entries.

3.1.3.4 Adapted Term Array

Following the same concerns as with the previous structures, the word-based Term Array required the prefixes to correspond to complete words. Table 26 shows the word-based Term Array calculated from the previous structures.

i	SC index	phrase
0	0	be
1	1	be_or
2	1	be_or_not
3	1	be_or_not_to
4	1	be_or_not_to_be
5	2	be_or_not_to_be_or
6	2	be_or_not_to_be_or_not
7	2	be_or_not_to_be_or_not_to
8	2	be_or_not_to_be_or_not_to_be
9	3	not
10	3	not_to
11	3	not_to_be
12	4	not_to_be_or
13	4	not_to_be_or_not
14	4	not_to_be_or_not_to
15	4	not_to_be_or_not_to_be
16	5	or
17	5	or_not
18	5	or_not_to
19	5	or_not_to_be
20	6	or_not_to_be_or
21	6	or_not_to_be_or_not
22	6	or_not_to_be_or_not_to
23	6	or_not_to_be_or_not_to_be
24	7	to
25	7	to_be
26	8	to_be_or
27	8	to_be_or_not
28	8	to_be_or_not_to
29	8	to_be_or_not_to_be
30	9	to_be_or_not_to_be_or
31	9	to_be_or_not_to_be_or_not
32	9	to_be_or_not_to_be_or_not_to
33	9	to_be_or_not_to_be_or_not_to_be

Table 26. Word-based Term Array

As it can be confirmed, this version is significantly smaller than the original character-based one (Sub-Section 2.4.4). In this particular case, the gain reaches an order of 9 times smaller.

i	SC index	phrase
0	0	be
1	1	be_or
2	1	be_or_not
3	1	be_or_not_to
4	1	be_or_not_to_be
5	2	not
6	2	not_to
7	2	not_to_be
8	3	not_to_be_or
9	3	not_to_be_or_not
10	4	or
11	4	or_not
12	4	or_not_to
13	4	or_not_to_be
14	5	or_not_to_be_or
15	6	to
16	6	to_be
17	7	to_be_or
18	7	to_be_or_not
19	7	to_be_or_not_to

Table 27. Limited word-based Term Array

Table 27 shows the resulting Term Array when a limit of 5 words is considered, built from the Suffix Class Array shown in Table 25.

As noted in (Aires, Lopes and Gomes, 2009), when considering a word count limitation of L words, each adapted Suffix Class will produce at most L terms, so an adapted Suffix Array with N words will produce an adapted Term Array with at most $(2N - 1)L$ entries.

3.1.3.5 Merging Process for the Indexing Structures

The indexing structures described in the previous sub-sections grow much larger than their corresponding text, resulting in the impossibility to fit all structures in main memory for relatively large corpora. To cope with such limitation, all the calculated structures are stored on disk because they take significantly more space and can be accessed through efficient binary searches, while the text, much smaller in comparison and needing a more immediate access (because its access is generally random and widespread), is kept on main memory.

Still, during the normal calculation process of the Suffix Array of a text, both the text and its Suffix Array need to fit in main memory at the same time because, in general, very distant memory addresses need to be accessed, which can re-

sult in severe swap memory thrashing, therefore resulting in a severe performance degradation. However, as long as the whole text fits in main memory (without its Suffix Array), such degradation can be greatly minimized by fragmenting the text in chunks. Each chunk should allow the corresponding text fragment to fit in main memory along with its Suffix Array, processing each chunk individually and merging them in the end. This approach is more efficient because it takes advantage of the sequential access of the Suffix Array of each chunk from disk (which can benefit from reading disk blocks) while keeping only the text in main memory.

The set of chunks will consist of a partial order, in which the order of every suffix within each chunk is established, but the order of suffixes between chunks, necessary to merge them, is not determined. In other words, in generic terms, the merging problem consists in determining a complete order between character strings $\{a_0, \dots, a_n\}$ from chunk A, and character strings $\{b_0, \dots, b_m\}$ from chunk B, knowing that $a_i < a_{i+1}, \forall i: 0 \leq i < n$, and $b_j < b_{j+1}, \forall j: 0 \leq j < m$. Determining a complete order implies determining a relation between all strings a_i and b_j. The final order of the merged chunks will only need examining the first suffix of each chunk to determine which one comes first, taking the following suffix from the chunk that provided the previously selected suffix, and repeating the process until every suffix of every chunk has been processed. For instance, suffix a_0 from chunk A and suffix b_0 from chunk B are compared: if $a_0 < b_0$, then a_0 becomes the first suffix of the merged chunk, and a_1 and b_0 are compared next; otherwise, b_0 becomes the first suffix of the merged chunk, and a_0 and b_1 are compared next; and so on.

Nonetheless, however, the suffixes of the sorted chunks can share many long prefixes, resulting in the need to carry out a large number of very long comparisons to be able to decide their lexicographical order, seriously affecting the merging efficiency. In particular, it can be very common to find a generic number of l suffixes of a chunk sharing an LCP with a suffix of the other chunk, which can lead to the need of carrying out l·LCP comparisons to determine their relative order, with longer LCP's causing greater degradation in performance. Fortunately, such long comparisons can be avoided by exploring some mathematical properties of character strings and their LCP values.

The following sub-sections will describe the improved merging process in detail, starting by presenting the properties of the LCP values (Sub-Section 3.1.3.5.1), which allow improving string comparison, describing the actual merging of two chunks of Suffix Arrays and LCP Arrays (Sub-Section 3.1.3.5.2)

and explaining an exceptional case that can arise during the merging process (Sub-Section 3.1.3.5.3).

3.1.3.5.1 Using LCP Values to Improve String Comparison

As a simple example, consider a partial order $a < b$ and $a < c$, established between character strings a, b and c. In this case, in order to produce a complete order, it will be necessary to determine if $b < c$ (leading to the complete order $a < b < c$) or $c < b$ (leading to the complete order $a < c < b$). Establishing that $x[i]$ represents the character located at offset i on a generic character string x, without any additional knowledge, determining if $b < c$ or if $c < b$ requires comparisons between $b[i]$ and $c[i]$, starting at $i = 0$, until a different character is found. So, the more initial characters are shared by both strings, the more comparisons are required. However, if LCP(a, b) and LCP(a, c) are determined, something that could be accomplished very easily at the same time the relations $a < b$ and $a < c$ were established, the merging process can be carried out more efficiently, as will be shown below.

Once established that LCP(x, y) is the length of the longest common prefix between generic character strings x and y, the following conditions hold:

(1) $x[i] = y[i], \forall i: 0 \leq i < \text{LCP}(x, y)$;

(2) $x < y <=> x[\text{LCP}(x, y)] < y[\text{LCP}(x, y)]$.

So, using the conditions above with the previously calculated LCP(a, b) and LCP(a, c), it is possible to establish a complete order between a, b and c just by comparing LCP(a, b) with LCP(a, c).

First of all, knowing the mentioned LCP's and from condition (1) above, it follows that:

(1.1) $a[i] = b[i], \forall i: 0 \leq i < \text{LCP}(a, b)$; and

(1.2) $a[i] = c[i], \forall i: 0 \leq i < \text{LCP}(a, c)$.

So, if the LCP comparison determines that LCP(a, c) < LCP(a, b), from (1.1) and (1.2), it follows that:

(1.3) $a[i] = b[i] = c[i], \forall i: 0 \leq i < \text{LCP}(a, c)$.

But, for the particular case in which $i = \text{LCP}(a, c)$, it follows that:

(1.4) $a[i] = b[i] < c[i]$.

This way, from (1.3), (1.4) and (2), it can be concluded that LCP(a, c) = LCP(b, c) and $b < c$, therefore concluding that $a < b < c$. The same reasoning can be applied

to conclude that $a < c < b$, if the lcp comparison above determines that LCP(a, b) < LCP(a, c).

So far, it has been shown that, as long as the LCP values are different, a comparison between them is enough to make a decision about the complete order of the strings a, b and c above. However, in case LCP(a, b) = LCP(a, c) = lcp, it is necessary to analyze the actual b and c strings, but only from offset lcp onward, because the characters are equal until offset lcp–1. In such case, the lcp value is incremented every time the characters are equal until different characters are found, updating the corresponding lcp value at the same time the order between b and c strings is determined.

With this analysis, the merging process of Suffix Arrays can be significantly improved. Going back to the example in which a generic number of l suffixes of one chunk share the same LCP with a suffix of the other chunk, this approach allows the number of comparisons to be reduced from $l \cdot$LCP to just l, which can be a very significant gain, particularly because such shared LCP's are very common.

Using these results to improve the merging process implies the calculation of the LCP Array of each chunk, with a final LCP Array being produced corresponding to the final merged Suffix Array. The process is described in Sub-Section 3.1.3.5.2 below.

3.1.3.5.2 Merging the Structures

Confronted with the need of having to partition a text in smaller chunks, it is necessary to merge those chunks in order to obtain a final single text, a final single Suffix Array and a final single LCP Array. The process will be described, using the results shown in Sub-Section 3.1.3.5.1 above, for an example text divided into two texts (t_0 and t_1), along with the corresponding Suffix Arrays (sa_0 and sa_1) and corresponding LCP Arrays (la_0 and la_1).

First of all, t_0 and t_1 are merged back into a single text (mt), something that is accomplished by concatenating t_1 to the end of t_0. As a consequence, the Suffix Arrays and the LCP Arrays can be used unchanged, but the offsets of sa_1 require a base offset equal to the size of t_0 (len_0) to be able to refer to its corresponding t_1 suffixes, now placed after t_0 in the merged text mt. So, considering that $mt[i]$ represents the mt suffix starting at offset i, element i_0 from sa_0 represents suffix $mt[i_0]$ while element i_1 from sa_1 represents suffix $mt[i_1+len_0]$. Besides mt, the merging process will produce the corresponding msa (from sa_0 and sa_1), and mla (from la_0 and la_1).

Producing a merged array will consist in taking the elements at the head of the arrays to be merged, to then insert them in order at the tail of the merged array. Using the properties above requires the analysis of LCP(*mt*[tail(*msa*)], *mt*[head(sa_0)]) = head(la_0) and LCP(*mt*[tail(*msa*)], *mt*[head(sa_1)]) = head(la_1), which greatly simplify the selection between both head elements. So, generically:

if head(la_0) < head(la_1) then

update tail(*msa*) to head(sa_1)+len_0; advance *msa* tail; advance sa_1 head

update tail(*mla*) to head(la_1); advance *mla* tail; advance la_1 head

if head(la_0) > head(la_1) then

update tail(*msa*) to head(sa_0); advance *msa* tail; advance sa_0 head

update tail(*mla*) to head(la_0); advance *mla* tail; advance la_0 head

if head(la_0) = head(la_1) = *lcp* then

determine *nlcp* = LCP(*mt*[head(sa_0)+*lcp*], *mt*[head(sa_1)+len_0+*lcp*])+*lcp*

if *mt*[head(sa_0)+*nlcp*] < *mt*[head(sa_1)+len_0+*nlcp*] then

update head(la_0) to *nlcp*

update tail(*msa*) to head(sa_0); advance *msa* tail; advance sa_0 head

update tail(*mla*) to head(la_0); advance *mla* tail; advance la_0 head

if *mt*[head(sa_0)+*nlcp*] > *mt*[head(sa_1)+len_0+*nlcp*] then

update head(la_1) to *nlcp*

update tail(*msa*) to head(sa_1)+len_0; advance *msa* tail; advance sa_1 head

update tail(*mla*) to head(la_1); advance *mla* tail; advance la_1 head

As mentioned in the description of the structure (Sub-Section 2.4.2), an LCP Array is padded with a 0 *lcp* value at the beginning of the array and another at the end, so the same applies to the LCP Array of each chunk. This means that the first comparison needs to be a complete one, which is consistent with the fact that the order between the very first suffixes of each chunk is not known and a complete comparison is required to establish such order.

This way, by following the procedure above, selecting which suffix comes first will imply: putting the corresponding 0 *lcp* value to the tail of *mla*; updating the other 0 *lcp* value to the resulting *lcp* from the comparison of the first suffixes of each chunk; and adding a final 0 *lcp* value to the end of *mla* when the process is finished. This ensures that *mla* is also padded with 0 *lcp* values at its beginning and at its end.

Once all the elements from both chunks have been processed with the comparisons above, the merging process for the Suffix Arrays and the LCP Arrays of the individual chunks is finished.

Both the Suffix Class Array and the Term Array do not represent such a serious problem since the locality of their data avoids the previous swap memory thrashing, allowing them to be processed in sequential contiguous chunks that fit into the available memory along with the text, from both the final Suffix Array and the final LCP Array. Nonetheless, the final Suffix Class Array is completely determined before moving on to determining the final Term Array.

3.1.3.5.3 Exceptional Cases while Merging the Suffix and LCP Arrays

The concatenation of the texts of two separately processed chunks being merged might raise some exceptional cases. To illustrate those cases, consider the Suffix Array *SA* of a text *T* = "3211" shown in Table 28.

i	*SA*[*i*]	*T*[*SA*[*i*]]
0	3	1
1	2	11
2	1	211
3	0	3211

Table 28. Suffix Array from text *T*

Depending on the text being merged to *T*, the final order of the Suffix Array might require additional attention. As a first example, consider concatenating "0" to *T*, producing the text *MT* = "32110", with its Suffix Array *MSA* shown in Table 29.

j	*MSA*[*j*]	*MT*[*MSA*[*j*]]
0	4	0
1	3	10
2	2	110
3	1	2110
4	0	32110

Table 29. Suffix Array from text *T*, after a concatenation with no consequences

As it can be seen, such concatenation represents no problem. The final *MSA* only requires an additional entry corresponding to the concatenated "0" but the relative order of the initial *SA* is kept (consider *i* = 0…3 in Table 28 and *j* = 1…4 in Table 29).

However, the initial order of *SA* is not always kept. Consider the concatenation of "2" to *T*, this time producing the text *MT* = "32112", with its corresponding Suffix Array *MSA* shown in Table 30.

j	*MSA*[*j*]	*MT*[*MSA*[*j*]]
0	3	**12**
1	2	**112**
2	4	2
3	1	2112
4	0	32112

Table 30. Suffix Array from text *T*, after a concatenation with consequences

In this case, the concatenation of "2" to the end of *T* results in the lexicographically out of order entries *MT*[*MSA*[0]] and *MT*[*MSA*[1]], highlighted in the table. Switching the order of *MSA*[0] and *MSA*[1] results in the correct Suffix Array for *MT* = "32112", shown in Table 31.

j	*MSA*[*j*]	*MT*[*MSA*[*j*]]
0	2	**112**
1	3	**12**
2	4	2
3	1	2112
4	0	32112

Table 31. Corrected Suffix Array from text *T* after the previous concatenation with consequences

In sum, when merging the Suffix Arrays and the LCP Arrays of a pair of texts already concatenated, suffixes $sa_0[i]$ and $sa_0[i+1]$ may require: their order to be changed; their *lcp* value to be updated; or both. Such requirements will only need to be checked if $sa_0[i]+la_0[i+1] = len_0$, for any $0 <= i < len_0$, in which case a stack is used in order to ensure the correct final order of the entries. So, after determining that $sa_0[i]+la_0[i+1] = len_0$, it will be necessary to confirm the order between suffixes $sa_0[i]$ and $sa_0[i+1]$. In case it is determined that they change their order, $sa_0[i]$ will be placed in the stack and index i is moved forward. Whenever the stack is not empty, the above comparisons will not only involve the head entries from the blocks being merged, but also the head entries from the stack.

3.2 PRE-PROCESSING

Texts involved in the translation process (either to support the language model, to support the translation model or to be translated) are first pre-processed, for text normalization (Sub-Section 3.2.1) and number replacement (Sub-Section 3.2.2), as described and justified in the sub-sections below, in order to simplify all processes required for translation.

All the adapted indexing structures (Section 3.1.3) are calculated after the pre-processing stage, which has been developed precisely to improve the identifica-

tion of individual words and to ensure that the analysis corresponds to a generic number entity instead of a specific number literal.

3.2.1 TEXT NORMALIZATION

The purpose of text normalization is to format the texts in order to relieve the system from carrying out extra checks that would burden the process. The text normalization consists of contractions expansion (Sub-Section 3.2.1.1), and tokenization (Sub-Section 3.2.1.2), described below.

3.2.1.1 Contractions Expansion

This procedure is carried out particularly for the Portuguese, Spanish, French and German languages and consists in separating the elements (usually prepositions and articles) hidden in a contraction, like the examples shown in Table 32.

Portuguese contraction	Corresponding expansion	English translation
da	de a	of the
do	de o	of the
na	em a	on the
neste	em este	in this
daquele	de aquele	of that
daquela	de aquela	of that

Table 32. Examples of contractions in Portuguese

Such separation allows a clear identification of the words involved, which in turn allows the factorization of their translations. For instance, with contractions, the phrase translation equivalent "daquele" <=> "of that" would have to be kept as a whole, unlike when the contraction is expanded, which would convert the equivalent into "de aquele" <=> "of that", which is longer than necessary because it can be produced with the individual phrase translation equivalents "de" <=> "of" and "aquele" <=> "that".

PT	EN
de	of
em	in
o	the
a	the
este	this
aquele	that
aquela	that

Table 33. Some article and preposition Portuguese entries with their English translations

With expansion of contractions, the previous 6 entries in Table 32 can be converted in the 7 entries in Table 33, which are more productive as they adequate-

ly express the information needed for alignment and translation purposes. Keeping the contractions would require entries from both Table 32 and Table 33 because entries from either of the tables could not be used to produce entries of the other table. Moreover, it would make the translation problem more difficult as the preposition depends on some word that generally appears to the left of the contraction (and preposition), while the article depends on a word that appears to the right of the contraction (and the article it includes). This will be better understood if we look at examples in Table 34.

PT	EN
a partir da	from the
a partir do	from the
a partir das	from the
a partir dos	from the
a partir daquele	from that
a partir daquela	from that
gosto da	I like the
gosto do	I like the
gosto das	I like the
gosto da	I like the
gosto daquele	I like that
gosto daquela	I like that

Table 34. Unexpanded Portuguese entries and their English translations

Portuguese phrase "a partir de" works as a unit (a compound preposition) and should not be divided into its constituents. As a matter of fact, it is translated by single word preposition "from" and can appear in many contexts translated by "after", "as from", "as of", "beginning in", "with effect from", and by many other expressions. If any of these translations would be appended with the English article or demonstrative adjective, one can figure out the huge number of entries that would be necessary. Moreover, the use of an article in Portuguese (or in English) does not necessarily require the use of its translation in the other language. As a consequence, the number of bilingual phrase lexicon entries should still increase to tackle the presence or absence of an article in any of the languages of the pair considered. If this is a problem for a pair of languages where one of the languages (English) is morphologically poor, when one considers a pair of languages not morphologically poor, where articles may be marked with gender, number and/or case, the explosion of entries necessary would be huge.

Another example is the Portuguese phrase "gosto de", translated as the English phrase "I like", where Portuguese preposition "de" is not expressly translated as well as the English pronoun "I" does not need to be explicitly

translated. Allowing contractions would require the translation of several Portuguese expressions with contractions to be merged into the translation of a single English expression, as shown in Table 34, where its entries, along with entries from Table 35 and Table 36 would all be required in the translation process, again, because entries from either one would be unable to produce entries from any other.

PT	EN
a	the
o	the
as	the
os	the
aquele	that
aquela	that

Table 35. Portuguese article and demonstrative adjectives and their English translations

Expanding the contractions makes it possible to classify the entries from Table 34 as longer than necessary because the translations contained in such table can be obtained through the combination of entries from Table 35 and Table 36. Without this expansion of contractions, the bilingual phrase lexicon would have to be much larger.

PT	EN
a partir de	from
gosto de	I like

Table 36. Phrase entries that can be considered units

The expansion of contractions will also benefit the language model (Section 3.3.2.2) because of the clear identification of previously contracted words.

3.2.1.2 Tokenization

Since the several structures used in the translation process find exact matches and rely on single space characters to identify individual tokens, the tokenization process aims at facilitating such identification. Texts are not guaranteed to have such a clear separation between tokens, so a single space is used to replace contiguous blank characters (spaces or tabs) and to separate words from punctuation. Table 37 shows some tokenization examples, where the character "_" represents a space. Without the elimination of extra blanks, the first three terms would be considered different from each other, as the extra blank spaces will result in different lengths. In the last three terms from the same table, the absence of the blank space prevents the identification of the three instances of word "world".

#	Original	Tokenized
1	European_Community	European_Community
2	European____Community	European_Community
3	European__Community	European_Community
4	world.	world_.
5	world,	world_,
6	world	world

Table 37. Tokenization examples

Also, to avoid limit checks, a single space character is added at the beginning of a text, as will become clear ahead (Section 3.1.3).

3.2.2 NUMBER REPLACEMENT

The occurrence of a numerical literal in a text should not be treated as any other type of text. For instance, having to translate the phrase "Article 101" should not depend on actually having a translation for the literal "Article 101" but rather on having a translation for its generalized representation "Article <number>". Besides, for both the translation and the language models, it should not be considered the co-occurrence of "Article" with the literal "101" but rather the co-occurrence of "Article" with a general number, not to mention it would be impossible to store translations for every number literal (the number of law articles is limited, but number literals occur in many situations in which such limitation does not hold).

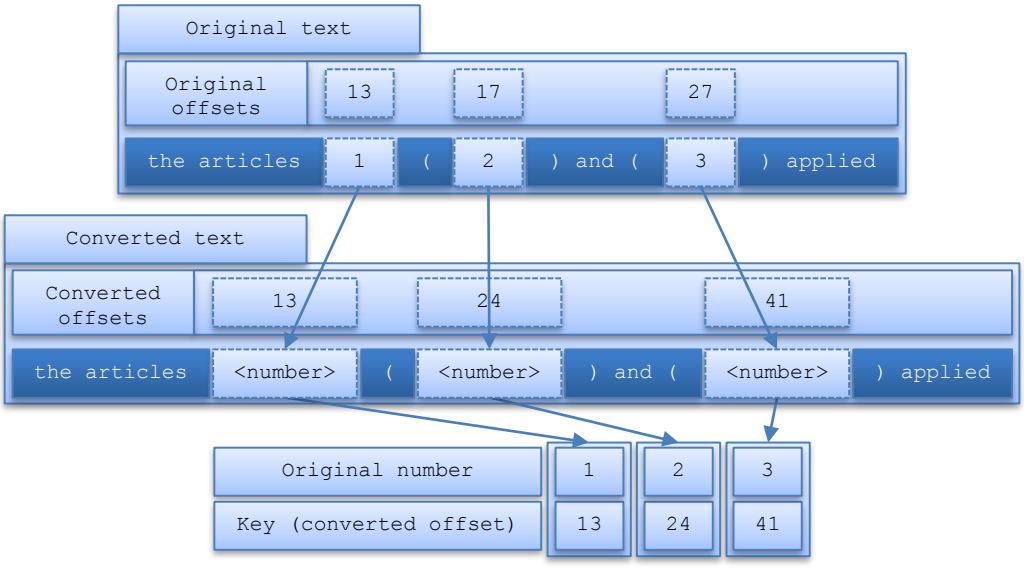

Figure 10. Example of number replacement on a text to be translated

This replacement greatly simplifies the process, considering most structures and procedures are kept unchanged because the number tags are also represented in plain text (in the example, the numbers '1', '2' and '3' are each re-

placed by the plain text tag "<number>"). For these reasons, number literals are replaced by a general tag <number>, but those replacements follow different concerns depending on the purpose of the texts, as described below.

- Monolingual texts: These texts can have their numbers replaced without having to obey any constraints. However, this should only be done after normalization to avoid converting the tag <number> into the set of tokens <_number_>, a consequence of the tokenization described above.
- Aligned parallel texts: The number replacement on these texts must take place after the alignment (Section 2.3) since the actual number literals are useful alignment anchors. The alignment is represented by the limit offsets of the aligned phrases in each text, and replacing the number literals by the <number> tags (which do not necessarily share the same character length) will have an impact on those offsets, which have to be updated to preserve the consistency of the aligned phrases.
- Texts to be translated: In this case the original numbers have to be kept to be appropriately restored on the final translation and, again, this must be done after the normalization stage described above. A hash table is used to keep the original numbers, which are accessed by the offsets of their corresponding tag in the converted text. An example is shown in Figure 10.

The only situation requiring more attention is when the texts are to be translated because it is necessary to store the original numbers so that they can be restored for the final translation.

3.3 TRAINING STAGE

The training stage involves the preparation of the models to be used in the translation process. As such, it involves: the extraction of phrase translation equivalents, which are the building blocks of the solution presented in this thesis; and the calculation of the models, which will provide some guidance as to how those building blocks are to be displaced to produce a sentence translation.

3.3.1 CONTIGUOUS PHRASE TRANSLATION EXTRACTION FROM ALIGNED PARALLEL CORPORA

By using a genuine phrase-based approach, Transtor needs phrase translation equivalents to translate complete texts. Apart from the translations existing in the human verified bilingual phrase lexicon, the extraction of contiguous phrase translations (probably longer than those existing in the verified bilingual phrase lexicon, as well as other possible translations that do not yet exist in that lexicon) is a very important task and component. This component supports the

translation process by providing contiguous phrase translation equivalents even when a verified bilingual phrase lexicon is not available (Aires, Lopes and Gomes, 2009). When a verified bilingual phrase lexicon is indeed available, extracted contiguous phrase translation equivalents allow a higher precision because its entries are known to be correct, and a higher recall by increasing the number of phrases for which a translation is available.

The implementation of the contiguous phrase translations extraction relies on: the adapted indexing structures (Section 3.1.3) to identify phrases in each language; and on parallel corpora that has been aligned with the FCT Aligner (Section 2.3) to help establishing an association between the identified contiguous phrases in one language to the identified contiguous phrases in the other language.

The following sub-sections describe the extraction process in detail, starting by a general description about how the alignment is used to guide the identification of contiguous phrase translation equivalents (Sub-Section 3.3.1.1), moving on to the identification of unique phrases (the vocabulary) in each language (Sub-Section 3.3.1.2), and finishing with the association of the identified unique phrases, in both languages, to express their translation equivalence (Sub-Section 3.3.1.3), to be included in the translation stage (Section 3.4).

3.3.1.1 Alignment Guidance

As mentioned before, besides the many phrase translations used as anchors by the alignment, additional phrase translations can be extracted to increase the number of phrases for which a translation is available (a higher recall), which particularly benefits the process of translating new texts. Table 38 shows a very small alignment improvement over the alignment represented in Table 2 (located in Sub-Section 2.3), where the equivalent "`support`" <=> "`apoiar`", on entry 4, is now recognized and used as an anchor.

#	EN	recognized	PT
1	Eurojust ' s		A Eurojust tem por
2	mission	*	missão
3	shall be to		
4	support	*	apoiar
5	and	*	e
6	strengthen	*	reforçar
7	coordination and coopera-tion	*	a coordenação e a coopera-ção
8	between	*	entre
9	national investigating and prosecuting		
10	authorities	*	as autoridades
11			nacionais competentes para a investigação e o exercí-cio de a acção penal
12	in relation to	*	em matéria de
13			criminalidade
14	serious	*	grave
15	crime		
16	affecting	*	que afecte
17	more than	*	mais de o que
18	one	*	um
19	European country	*	país Europeu
20	.	*	.

Table 38. Improved alignment example

The exemplified alignment already recognizes a few phrase translations (marked with '*'), like "mission" <=> "missão" on entry 2, and "strength-en" <=> "reforçar" on entry 6, but additional phrase translation equivalents can be extracted with the guidance of such alignment.

#	correct	EN	PT
1		Eurojust ' s	A Eurojust tem por
2		Eurojust ' s mission	A Eurojust tem por missão
3	*	Eurojust ' s mission shall be to	A Eurojust tem por missão
4	*	Eurojust ' s mission shall be to support	A Eurojust tem por missão apoiar
5	*	Eurojust ' s mission shall be to support and	A Eurojust tem por missão apoiar e
6	*	Eurojust ' s mission shall be to support and strengthen	A Eurojust tem por missão apoiar e reforçar
7	*	Eurojust ' s mission shall be to support and strengthen co-ordination and cooperation	A Eurojust tem por missão apoiar e reforçar a coordenação e a cooperação
8	*	Eurojust ' s mission shall be to support and strengthen co-ordination and cooperation between	A Eurojust tem por missão apoiar e reforçar a coordenação e a cooperação entre
9		Eurojust ' s mission shall be to support and strengthen co-ordination and cooperation between national investigating and prosecuting	A Eurojust tem por missão apoiar e reforçar a coordenação e a cooperação entre
10		national investigating and prosecuting authorities	as autoridades
11	*	national investigating and prosecuting authorities	as autoridades nacionais competentes para a investigação e o exercício de a acção penal
12	*	national investigating and prosecuting authorities in relation to	as autoridades nacionais competentes para a investigação e o exercício de a acção penal em matéria de
13		national investigating and prosecuting authorities in relation to	as autoridades nacionais competentes para a investigação e o exercício de a acção penal em matéria de criminalidade
14		national investigating and prosecuting authorities in relation to serious	as autoridades nacionais competentes para a investigação e o exercício de a acção penal em matéria de criminalidade grave
15	*	national investigating and prosecuting authorities in relation to serious crime	as autoridades nacionais competentes para a investigação e o exercício de a acção penal em matéria de criminalidade grave
16		serious	criminalidade grave
17	*	serious crime	criminalidade grave
18		serious crime	grave

Table 39. Some extracted phrase translation equivalents

Table 39 shows some examples of the translation equivalents that can be extracted from the previous alignment while using a count limit of 20 words. En-

tries 1 through 9 show extracted equivalents beginning with "`Eurojust ' s`" (entry 1 on Table 38), entries 10 through 15 show extracted equivalents beginning with "`national investigating and prosecuting`" (entry 9 on Table 38), and the last 3 entries show the possible combinations of translation equivalents that can be extracted from the alignment entries 13 through 15 from Table 38. The column "correct" indicates correct entries marked with a "*" character, showing that not every extracted equivalent is correct.

The identification of such translation equivalents can be carried out by considering a given phrase, checking the corresponding segments on which the phrase occurs, and identifying the corresponding phrase translation. Such correspondence will only consider combinations of segments without partitioning any of them. As an example, considering again the alignment in Table 38, it would not be possible to identify a translation for "`mission shall be`" (segment entry 2 and partial segment entry 3), "`prosecuting`" (partial segment entry 9) or "`relation`" (partial segment entry 12) because all of them involve at least one partial segment.

The number of tokens allowed for a phrase is limited mainly for practical reasons and is applied to both phrases of a translation equivalent candidate. For example, if a limit of 6 tokens were to be considered, it would not be possible to extract a translation for "`Eurojust ' s mission shall be to`" because it has 7 tokens, but if a limit of 7 tokens was to be considered it would be possible to extract the translation equivalent "`Eurojust ' s mission shall be to`" **<=>** "`A Eurojust tem por missão`" because it can be obtained by a composition of one or more complete segments (in this case, segment entries 1, 2 and 3), with one phrase having 7 tokens and the other having 5. In Chapter 4, results will show how the number of tokens affects translation quality.

Additionally, it has been noted that equivalents "delimited" by empty segments (an empty segment in the beginning of one language and an empty segment in the end of the other language) are a common pattern (Aires, Lopes and Gomes, 2009) found when parallel texts have been aligned using the procedures and concepts described in Section 2.3. A couple of examples are identified in entries 9 through 11 and in entries 13 through 15, taken from Table 38 and shown again in Table 40 with the empty segments highlighted. Those entry sets allow the identification of translation equivalents "`national investigating and prosecuting authorities`" **<=>** "`as autoridades nacionais competentes para a investigação e o exercício de a acção penal`" and "`serious crime`" **<=>** "`criminalidade grave`", respectively. An observation that could justify this is the fact that any other combination

would leave an alternative aligning with an empty segment while "consuming" another. For instance, admitting "serious crime" <=> "grave <empty>" would, in a way, imply "<empty>" <=> "criminalidade", which does not make sense because a phrase can never be translated by an empty phrase, and so the segments should be merged.

#	EN	recognized	PT
…	…		…
9	national investigating and prosecuting		
10	authorities	*	as autoridades
11			nacionais competentes para a investigação e o exercício de a acção penal
…	…		…
13			criminalidade
14	serious	*	grave
15	crime		
…	…		…

Table 40. Examples surrounded by empty segments

The reasoning supporting the use of surrounding empty segments is similar to the reasoning supporting the application of consistency rules in other phrase translation extraction approaches (Section 2.1.2.2).

The quality of each extracted translation equivalent is left up to the translation model (Section 3.3.2.1), and the context relevance of their use is left up to the target language model (Section 3.3.2.2). The following sub-sections will describe the contiguous phrase translation extraction process in more detail, particularly the phrase identification stage (Sub-Section 3.3.1.2), and the phrase association stage (Sub-Section 3.3.1.3).

3.3.1.2 Phrase Identification Stage

Unlike the alignment, which establishes relations between phrase instances or occurrences, the phrase translation extraction will relate unique phrases, but it will do so through their aligned phrase occurrences.

In order to identify the unique phrases along with their individual occurrences in both languages, their corresponding monolingual texts are first processed separately, consisting in the calculation of the adapted indexing structures (Section 3.1.3) for each language text, namely the Suffix Array, the LCP Array, the Suffix Class Array and, finally, the Term Array, with this last one being the goal structure because it contains the information about the unique phrases.

Figure 11 intends to show a simple representation of the English and Portuguese Term Arrays, calculated from a given pair of aligned parallel texts in those languages. Those Term Arrays will contain every unique phrase present in their corresponding languages, but only some entries are highlighted in the figure in order to simplify the description, including the English phrase "`serious crime`" and the Portuguese phrase "`criminalidade grave`", present in the alignment example in Table 38 above. In an actual implementation, tests have been carried out with phrases having up to 20 tokens, allowing the identification of phrases like "`Eurojust ' s mission shall be to support and strengthen`" in English and "`A Eurojust tem por missão apoiar e reforçar`" in Portuguese.

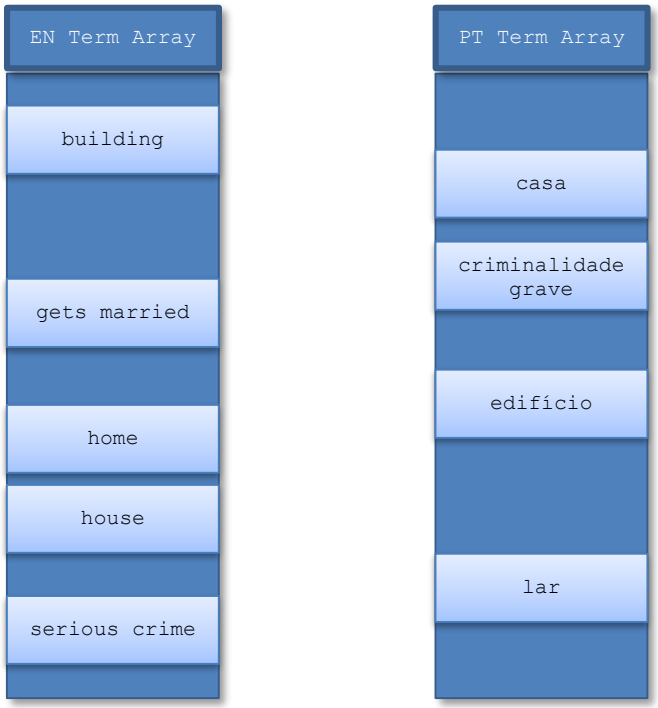

Figure 11. Term Arrays from both languages

Once the several unique phrases present in each language have been determined, it is possible to establish the associations between them, which will represent the identified contiguous phrase translation equivalents to be extracted, as explained in the phrase association stage below (Sub-Section 3.3.1.3).

3.3.1.3 Phrase Association Stage

With both Term Arrays calculated in the identification stage described above (Sub-Section 3.3.1.2), entries from each array can be associated to entries from the other array, which will be equivalent to having unique phrases in each lan-

guage associated to unique phrases in the other language, which in turn represent the wanted phrase translation equivalents.

Figure 12 shows the associations established between the highlighted entries of the pair of Term Arrays depicted in Figure 11, with the associations indicating, for instance, that the English phrases "building", "gets married", "home" and "house" are phrase translation equivalents of the Portuguese phrase "casa". Again, in an actual implementation, carried out tests with phrases having up to 20 tokens allowed the association of the example contiguous phrases presented in the previous sub-section to produce the translation equivalent "Eurojust ' s mission shall be to support and strengthen coordination and cooperation" <=> "a Eurojust tem por missão apoiar e reforçar a coordenação e a cooperação".

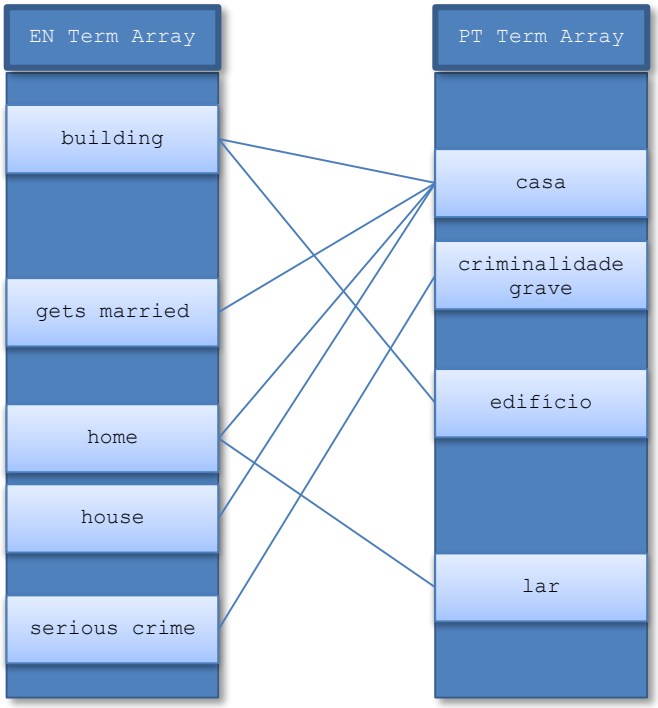

Figure 12. Some associated Term Arrays entries

Each target Term Array entry associated to a source Term Array entry is represented on the source entry by an association element containing the identifier of the target entry. Figure 13 illustrates how the associations of Figure 12 are actually implemented where, for instance, PT Term Array entry 20 (representing the phrase "casa") has its 4 phrase associations represented by EN Term Array entries 19 (corresponding to "building"), 36 (corresponding to "gets married"), 81 (corresponding to "home") and 87 (corresponding to "house"). For practical reasons, this example only shows a few phrase associations but, for instance, a lot more could be considered for "serious crime", like "crimi-

nalidade grave", "crime grave", "crimes graves" and "infracções graves".

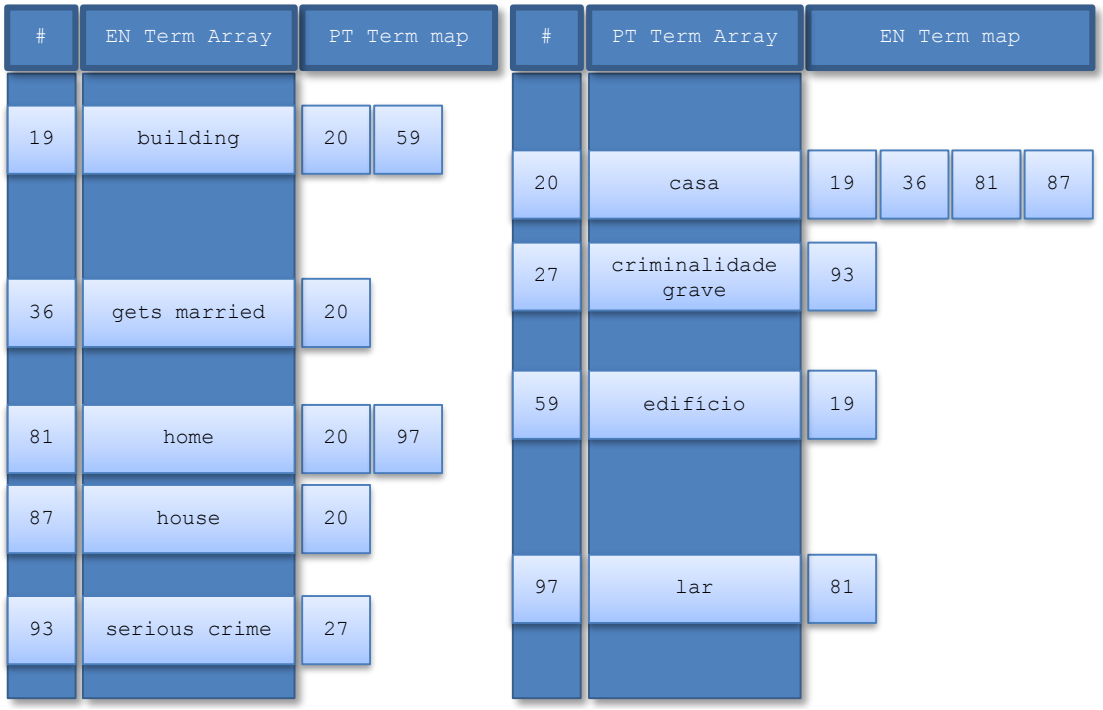

Figure 13. Implementation for the association of Term Array entries

Besides the identifier of the target entry, the association element contains the corresponding number of matches and the translation model score (Section 3.3.2.1). Table 41 shows how the target entries associated to the source entry representing the phrase "casa" are actually implemented, where: "ID" is the Term Array entry of the associated phrase; "matches" is the number of times each phrase has been matched to "casa"; and "score" is the translation model score of each phrase relatively to "casa". The "represented phrase" is only added to the table for illustrative purposes to show which phrase is represented by the ID.

represented phrase	ID	matches	score
house	87	296	0.5451
home	81	129	0.2376
gets married	19	92	0.1694
building	36	26	0.0479

Table 41. Associations established for Portuguese phrase "casa"

The association elements are processed for each Term Array entry using the alignment of its individual occurrences (because those are the ones that have been aligned, either explicitly or implicitly). Each of those occurrences is used to determine the corresponding source segments containing the occurrence, using

those source segments to obtain the corresponding target segments, and identifying the phrases contained in those target segments. Again, both explicit and implicit entries will be considered, but those different alignment types will be analyzed in the results (Chapter 4) to check how they can affect the translation model (Sub-Section 3.3.2.1).

Each found target phrase is used to retrieve the corresponding target Term Array entry, using that entry to confirm if it has already been associated to the source entry. In case an association entry is found it will be incremented, otherwise a new association entry is created and included. The result of processing every occurrence of a Term Array entry will be the set of association elements for that entry, which could be empty if no associations are found.

As an example, consider the individual occurrence of the source Term Array entry representing the phrase "`serious crime`", which is found on the alignment represented on Table 38. The occurrence is contained in complete segments 14 and 15, but since this is a situation in which an empty segment (segment 13) is adjacent to the segments containing the wanted phrase, segments 13 through 15 are considered, obtaining the corresponding target phrase "`criminalidade grave`", which is used to find its corresponding Term Array entry (check the use of surrounding empty segments introduced in Sub-Section 3.3.1.1). The found target entry is then used to check if it has been associated to the source entry, adding an association element if no previous association has been established or incrementing the counter of the corresponding association element by one.

Once all the occurrences of a given source entry have been processed, the association elements of the resulting set will be sorted from highest to lowest number of matches, keeping the first n most matched association elements and discarding the remaining ones. Only then the translation model score is calculated for every association element. As an example, with translations for "`casa`" shown in Table 41, if n=3 then the entry corresponding to "`building`" would be discarded, and if n>=4 none of the entries would be discarded. This association process is carried out for every Term Array entry, in both directions.

Carried out tests (Chapter 4) have considered a limit of 7 translation equivalents for each phrase. Such limit has proven to be adequate not only because the vast majority of most significant equivalents are included with such limit, but also because performance is improved by avoiding a great number of translation equivalents that have very low translation model scores.

The result of the association stage described above is stored with the purpose of being used in the translation process, particularly because they can be required very frequently during the translation of many texts. Once processed and stored, translations for a phrase having a length of P characters can be retrieved with a complexity of $O(P \log N)$ (Manber and Myers, 1990).

In sum, this solution first sequentially accesses all source Term Array entries, sequentially takes the occurrences of each entry, identifies the alignment segments of each occurrence through binary searches, and finally accesses the corresponding target Term Array entries through additional binary searches. Thanks to the alignment guidance, the structures that allow efficient identification, storage and retrieval of phrases, and the word limitation, the complexity of this solution, when processing a parallel corpus of size $2N$ words (N words for each corpus in each language), is $O(N \times (\log_2 N)^2)$ (Aires, Lopes and Gomes, 2009), far better than the $O(N^4)$ that would be required if all possible N^2 phrases from one text were to be combined with the N^2 phrases from the other text.

3.3.2 MODELS INVOLVED IN TRANSLATION SCORING

In order to score the translation candidates produced during the decoding process (Sub-Section 3.4.8), a model has been developed for Transtor as a combination of other two models, which will in turn evaluate meaning and fluency separately. Models for meaning and fluency are typically present in every SMT approach, but the implementations of those models in this approach have a few differences, as will be discussed ahead.

The phrase translation model (Sub-Section 3.3.2.1) is calculated to be applied for meaning and the phrase language model (Sub-Section 3.3.2.2) is calculated to be applied for fluency, where each model can be trained with separate dedicated corpora (Sub-Section 3.4.6). Both models are finally combined in the sentence translation model (Sub-Section 3.4.7), which is the model actually responsible for scoring translations of complete sentences.

3.3.2.1 Phrase Translation Model

The phrase translation model (*ptm*) developed for Transtor is responsible for assigning a score to each of the phrase translation equivalents of an original phrase. In general terms, given an original phrase f, with n extracted phrase translations e_i, the *ptm* score of *e*, as a translation of *f*, is expressed in Equation 13, resulting in a score between 0.0 and 1.0, inclusively.

$$ptm(e|f) = \frac{matches(e|f)}{\sum_{i=1}^{n} matches(e_i|f)}$$

Equation 13. Phrase translation model within a source

Sometimes $ptm(e|f)$ is simplified to $ptm(e)$, in which case the original phrase *f* is implicit. The function $matches(e|f)$ returns the number of times a given target phrase *e* has been associated to a source phrase *f* within the source from where the phrase translation equivalent *e* <=> *f* has been extracted. Table 42 presents the *ptm* scores for the examples introduced in Table 41 for a given aligned parallel corpus, already presented in Sub-Section 3.3.1.3.

translations of "casa"	matches	*ptm*
house	296	0.5451
home	129	0.2376
gets married	92	0.1694
building	26	0.0479
total	543	1.0000

Table 42. The *ptm* scores from an example aligned parallel corpus

So far, the model will assign high scores to frequent translations and low scores to infrequent ones. In an attempt to produce more balanced scores, several separate and independently analyzed phrase translation sources can be combined using their individual scores. The concern about trying to assign more balanced scores is the main distinction between this model and the one presented in Sub-Section 2.1.3.1, which only expresses association frequency and lexical weighting in both translation directions. The combination of several sources is discussed ahead in Sub-Section 3.4.6.1.

3.3.2.2 Phrase Language Model

The phrase language model (*plm*) developed for Transtor is responsible for assigning a score to each of the translation phrases that follow a given translation phrase. In general terms, given a translation phrase *e*, with *n* following translation phrases fe_i, the *plm* score of translation phrase *fe*, as a following of (occurring exactly after) *e*, is expressed in Equation 14, resulting in a score between 0.0 and 1.0, inclusively.

$$plm(fe|e) = \frac{context\_value(e, fe)}{\sum_{i=1}^{n} context\_value(e, fe_i)}$$

Equation 14. Phrase language model

In order to understand the concept of "following phrase translation" (and before explaining the "context_value"), consider the example shown in Figure 14,

illustrating the analysis of a fragment of an original English sentence "the procedures described in". The figure shows the original phrase "the" being followed by original phrases "procedures", "procedures described" or "procedures described in", as shown by the corresponding edges connecting those phrases. As a consequence, available phrase translations of "the" might be followed by available phrase translations of "procedures", "procedures described" and "procedures described in", also represented by edges on the figure. More specifically, translations "o", "a", "os" or "as" of the English word "the" might be followed by phrase translations "acções", "procedimentos", "acções descritas", "procedimentos descritos", "acções descritas em" or "procedimentos descritos em".

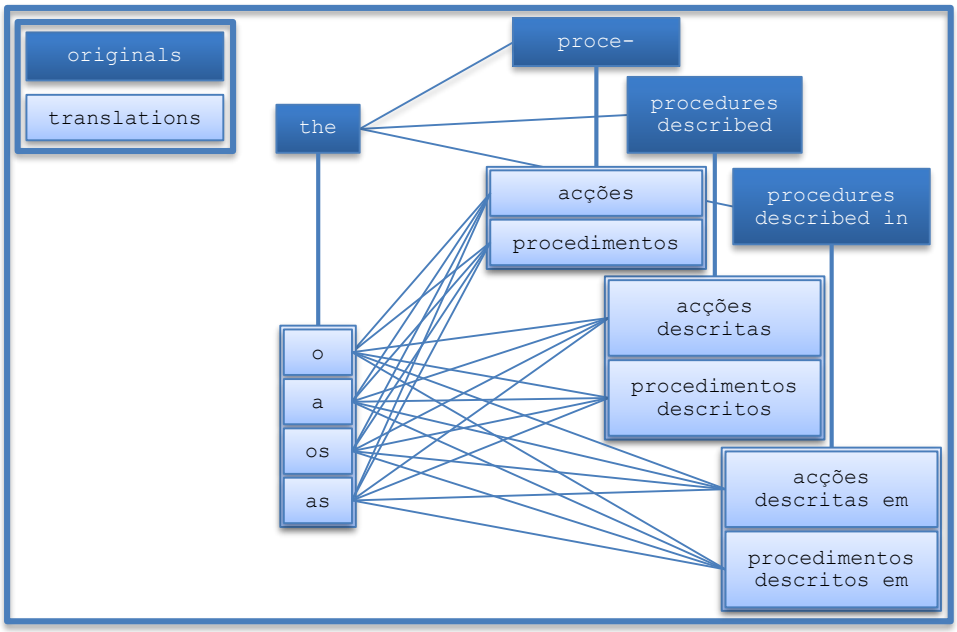

Figure 14. Original adjacent phrases with their translations

The purpose of the *plm* score is to evaluate how likely a translation phrase (named as *fe*, above) may occur after a given *e* phrase. Determining such score depends on the "context value" introduced in Equation 15, in which *tf* corresponds to the term frequency of a phrase, the '+' sign represents a concatenation operation and the "_" character represents the space character.

$$context\_value(e, fe) = tf(e + "\_" + fe)$$

Equation 15. Context value of an adjacent pair of phrase translations

So, the context value is determined by the term frequency of the phrase resulting from the concatenation of *e* and *fe*, separated by a space character. As an example, considering the current phrase "os" and the following phrase

"acções", the context value consists of the term frequency of the phrase "os acções". The term frequency is obtained from one or more indexed monolingual corpora, used as a base for *plm*. Table 43 shows the concatenation of each "current" phrase with each of their "following" phrases from Figure 14. The columns represent the "current" phrase and the rows represent the "following" phrase, with each cell representing the described concatenations used to obtain the corresponding "context values".

the					
o	a	os	as		
o acções	a acções	os acções	as acções	acções	**procedures**
o procedimentos	a procedimentos	os procedimentos	as procedimentos	procedimentos	
o acções descritas	a acções descritas	os acções descritas	as acções descritas	acções descritas	**procedures described**
o procedimentos descritos	a procedimentos descritos	os procedimentos descritos	as procedimentos descritos	procedimentos descritos	
o acções descritas em	a acções descritas em	os acções descritas em	as acções descritas em	acções descritas em	**procedures described in**
o procedimentos descritos em	a procedimentos descritos em	os procedimentos descritos em	as procedimentos descritos em	procedimentos descritos em	

Table 43. Concatenation of adjacent phrases

Table 44 shows the context values obtained for Figure 14, corresponding to the term frequencies of the entries from Table 43. The totals are also included in Table 44, representing the sum of all the values in each column, and are used to normalize the values to produce the actual *plm* scores, shown in Table 45.

A person familiar with English and Portuguese languages would know that: "o", as a translation of "the", would be almost for sure not supported by any of those other proposed translations following it; "a" as a possible translation of "the" may be supported because in Portuguese "a" can also be a preposition; "os" may support to be followed by "procedimentos", "procedimentos descritos" or "procedimentos descritos em"; and "os" will not support "acções", "acções descritas" or "acções descritas em". The kind of knowledge just described is related to the concept of language modeling, and the model presented here tries to infer the same conclusions through the frequencies of the concatenations of a phrase translation with each of its following phrase translations, as shown in Table 44.

the					
o	a	os	as		
0	371	0	797	acções	**procedures**
0	418	523	0	procedimentos	
0	284	0	765	acções descritas	**procedures described**
0	396	492	0	procedimentos descritos	
0	281	0	726	acções descritas em	**procedures described in**
0	349	480	0	procedimentos descritos em	
0	2099	1495	2288	**total**	

Table 44. Context values

Table 45 shows the *plm* scores, obtained after normalizing the values from Table 44 with the totals of the corresponding columns. As an example, the *plm* score between "a" and "acções" is 371/2099=0.1768. If the total is 0, as is the case with the "o" column, the score is also defined to be 0. The "o" column also shows that this is not a probability distribution because the sum of every score is not guaranteed to be 1.0.

the					
o	a	os	as		
0.0000	0.1768	0.0000	0.3483	acções	**procedures**
0.0000	0.1991	0.3498	0.0000	procedimentos	
0.0000	0.1353	0.0000	0.3344	acções descritas	**procedures described**
0.0000	0.1887	0.3291	0.0000	procedimentos descritos	
0.0000	0.1339	0.0000	0.3173	acções descritas em	**procedures described in**
0.0000	0.1663	0.3211	0.0000	procedimentos descritos em	

Table 45. The *plm* scores from the example

Intuitively, the *plm* score expresses the likelihood of a given phrase occurring after another, considering the alternatives, unlike the history model which is absolute and, therefore, independent of such alternatives. Using translation phrases as units (and not checking for fluency within a phrase) can be justified by the fact that those phrases already have some degree of quality ensured by the *ptm* score (Section 3.3.2.1). As an example, the *plm* score evaluates how likely the phrase "the rules" is to be followed by, for instance, the phrase "in spite of", without the need to check for the internal fluency of either "the rules" or "in spite of". The model only intends to determine how well a unit phrase is to be followed by another unit phrase.

This model considers phrases as units, unlike the *n*-gram history model described in Sub-Section 2.1.3.2, which evaluates the fluency of a sentence through the combined analysis of every word that composes the sentence with its previous *n*–1 words. As such, this model can be viewed as a local history model of

adjacent pairs of phrases, independently of the number of words making up those phrases, even though a limit is imposed on each phrase, but such limit can reach 20 words (Chapter 4). Also, another difference of this model is its ability to combine individual scores from several separate and independently analyzed monolingual sources. Such ability is discussed ahead in Sub-Section 3.4.6.2.

3.4 TRANSLATION STAGE

In this final step of the whole translation process, all the concepts and results from the previous steps are combined to support several techniques, which together with some additional concepts, leads to the final translation result.

The translation stage begins with the partitioning of the source text to be translated into all its possible unique phrases (Sub-Section 3.4.1), limited by a parameterized maximum phrase length. Then, the process moves on to obtaining the translation model scores and determining the language model scores. These scores will support the determination of the final sentence model score.

The partitioned phrases are then used to retrieve their corresponding phrase translation equivalents: contiguous phrase translation equivalents will be described first (Sub-Section 3.4.2), while the inclusion of translation patterns, which depend on additional concepts, are described later (Sub-Section 3.4.5).

Once the phrase translations have been retrieved for the unique phrases identified in the text partitioning stage, the several occurrences of the unique phrases for which translations are available are used to create the cover graph (Sub-Section 3.4.3) with the main purpose of assisting in the creation of the translation graph (Sub-Section 3.4.4). With the translation graph, the *plm* score can be effectively included in the process (Sub-Section 3.4.6.3), finally allowing the traversal of the translation graph (Sub-Section 3.4.8) in order to determine a combination of translation phrases that maximizes the *stm* score. Untranslatable phrases can result in problems that have to be considered, namely the existence of untranslatable sections (Sub-Section 3.4.9.1) and the existence of dangling nodes (Sub-Section 3.4.9.2). Finally, once the decoding traversal is completed, it is necessary to traverse the translation graph once more (Sub-Section 3.4.10) in order to present the identified combination of phrase translations that maximizes the *stm* score, which will be the final translation.

3.4.1 TEXT PARTITIONING

The text partitioning consists in taking the text to be translated and identifying the several phrases contained in it, which will then be used to retrieve their

phrase translations and produce a final translation for the whole original text. In order to simplify the explanation, text *T* = "`the rules in accordance with the national law`" will be used as an example. Table 46 shows the offsets corresponding to the beginning of each word in *T* which, along with the lengths of the phrases, are used to represent the several phrases present in *T*.

text	`the`	`rules`	`in`	`accordance`	`with`	`the`	`national`	`law`
offset	1	5	11	14	25	30	34	43

Table 46. Example of a text to be partitioned

The text partitioning task is efficiently accomplished by calculating the adapted Term Array (Sub-Section 3.1.3.4) of the text, which will represent every occurrence of every unique phrase present in a text, within the word count limitation introduced. The Term Array of *T* is represented in Table 47, showing that only entry 22 (corresponding to the term "`the`"), with a length of 3 characters, has two occurrences, corresponding to offsets 1 and 30. Every other entry of the Term Array represented by the table occurs only once.

Table 47 also shows, for instance, that entries 6 and 7 both start at offset 11, but entry 6 (corresponding to the phrase "`in`") has a length of 2 characters while entry 7 (corresponding to the entry "`in accordance`") has a length of 13 characters. This table represents every possible phrase occurring in *T*, with the length corresponding to the phrase length in characters and the offset corresponding to the starting point, also in characters, where the phrase occurs in *T*. In this situation, the character-based lengths are used instead, to better support operations requiring offset calculations.

#	phrase/term	length	offsets
1	accordance	10	14
2	accordance with	15	14
3	accordance with the	19	14
4	accordance with the national	28	14
5	accordance with the national law	32	14
6	in	2	11
7	in accordance	13	11
8	in accordance with	18	11
9	in accordance with the	22	11
10	in accordance with the national	31	11
11	in accordance with the national law	35	11
12	law	3	43
13	national	8	34
14	national law	12	34
15	rules	5	5
16	rules in	8	5
17	rules in accordance	19	5
18	rules in accordance with	24	5
19	rules in accordance with the	28	5
20	rules in accordance with the national	37	5
21	rules in accordance with the national law	41	5
22	the	3	1 | 30
23	the national	12	30
24	the national law	16	30
25	the rules	9	1
26	the rules in	12	1
27	the rules in accordance	23	1
28	the rules in accordance with	28	1
29	the rules in accordance with the	32	1
30	the rules in accordance with the national	41	1
31	the rules in accordance with the national law	45	1
32	with	4	25
33	with the	8	25
34	with the national	17	25
35	with the national law	21	25

Table 47. Term Array representing the partitioned text example

With the entries from the Term Array representing every phrase present in *T*, they can then be used to obtain their corresponding phrase translations. Since obtaining translations for the several individual occurrences of a phrase would necessarily return the same results for each occurrence (for instance, retrieving translations of the word "the" would always return the same results for any of its occurrences), using unique phrases represents an advantage because a trans-

lation is obtained and made available for every occurrence with a single operation. For contiguous phrase translations equivalents (Sub-Section 3.1.1.1), the operation involved is a simple retrieval operation, described in Sub-Section 3.4.2 below, while for translation patterns (Sub-Section 3.1.1.2), the operation involved is a more elaborate matching operation, described later in Sub-Section 3.4.5.

3.4.2 RETRIEVAL OF CONTIGUOUS PHRASE TRANSLATION EQUIVALENTS

Once the adapted Term Array (Sub-Section 3.1.3.4) of a text has been produced in the partitioning stage described in Sub-Section 3.4.1 above, the entries from the Term Array are then used to obtain their corresponding contiguous phrase translation equivalents (Sub-Section 3.1.1.1) that have been previously extracted (Section 3.3.1), to be later combined according to the several models and produce the final translation of the text. Contiguous phrase translation equivalents, that have been previously extracted, are retrieved using a simple string match, along with their corresponding *ptm* scores (Sub-Section 3.3.2.1). Table 48 shows some examples produced from an aligned parallel corpus.

original	translation	score
the	o	0.3264
	a	0.2613
	os	0.2117
	as	0.2007
national	nacional	0.6170
	nacionais	0.3830
law	lei	0.6462
	legislação	0.3538
national law	legislação nacional	0.4865
	leis nacionais	0.5135
rules	regras	0.7073
	normas	0.2927
the rules	as regras	1.0000
in	em	0.5862
	dentro	0.4138
with	com	0.5966
	acompanhado	0.2017
	acompanhada	0.2017
in accordance with	de acordo com	1.0000

Table 48. Entries with scores

In the end, Term Array entries for which translations are found are then used to create the cover graph, described in Sub-Section 3.4.3 below. If more than one source is used, the scores from each source are combined to produce a single score, as described ahead in Sub-Section 3.4.6.1. However, Term Array entries for which phrase translation equivalents are not found represent untranslatable phrases and are disregarded from the remainder of the process. The untranslatable phrases might lead to untranslatable sections (Sub-Section 3.4.9.1) and to dangling nodes (Sub-Section 3.4.9.2), which require additional processing but, for simplicity, those will be described later.

3.4.3 COVER GRAPH

A cover of an original text consists of a combination of its phrases that completely rebuild the original text. One such example is depicted in Figure 15 for the text *T*, also represented in Table 46. Other covers for the same text exist.

Figure 15. An example of a valid cover

A valid cover does not contain any gaps or overlaps. An example of a cover with a gap is shown in Figure 16, in which the single-word phrase "`accordance`" is missing and does not enable to obtain a complete cover of *T*.

Figure 16. A cover with a gap

An example of a cover with an overlap is shown in Figure 17, where the single-word phrase "`accordance`" is covered by both "`in accordance`" and "`accordance with`".

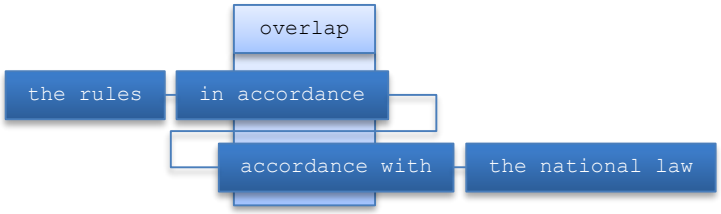

Figure 17. A cover with an overlap

Considering the many ways in which a text can be partitioned, there can be many possible covers. All those covers can be represented by a cover graph, with any individual path on such graph representing a single cover. A cover graph example for text *T* is shown in Figure 18, where the cover from Figure 15 is

highlighted. Two artificial nodes are included: one, labeled as "<start>", to keep a reference to the starting cover nodes; and another, labeled as "<end>", to which the ending cover nodes will refer.

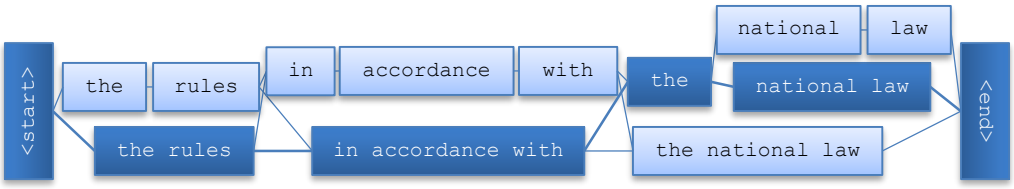

Figure 18. A cover graph example

The cover graph representation has the following properties: it only allows its traversal in one direction (from the node <start> to the node <end>); it does not contain cycles; and it allows taking advantage of common sub-paths shared by alternative covers. Identifying common sub-paths allows avoiding duplicate processing and consequently improving performance: the complexity goes from the number of paths to the number of edges. Considering a text of N words, and considering a limit of L words for any given phrase from the text, there will be $O(NL)$ phrases. Each phrase will connect to at most L following phrases, so there will be $O(NL^2)$ possible edges, while there will be $O(L^{NL})$ possible paths.

The main purpose of the cover graph, in the Transtor translation process, is to simplify the translation graph creation, which is why the original phrases considered for this stage are the ones that are indeed translatable (having at least one phrase translation equivalent). Using the cover graph simplifies the process of ensuring the phrase translations do not have gaps or overlaps according to their original phrases: using the cover from Figure 16 would result in a translation that would not contain a translation for "accordance"; using the cover from Figure 17 would result in a translation that would contain a duplicate translation for "accordance".

Producing the cover graph requires using every instance (or occurrence) of the identified translatable entries (Sub-Section 3.4.2) from the Term Array calculated previously (Sub-Section 3.4.1). Each instance will produce a cover node containing the information about the offset and the length of the phrase instance represented by the node, as well as a set containing the references to the nodes that follow it on the cover graph. The set of references on every cover node support the cover edges.

The set of references on a cover node is determined using the corresponding ending offset (obtained with the offset and length of the node) to identify the cover nodes having an offset consistent with the given ending offset and separated by a blank space character. The node <start> will reference the starting

cover nodes (the ones with an offset of 1), and the node <end> will be referenced by the ending cover nodes (the ones having the sum of their offset and their length equal to the length of the original text). Once the cover graph is complete, the translation graph can be created from it, as explained in Sub-Section 3.4.4 below.

3.4.4 TRANSLATION GRAPH

The cover graph (Sub-Section 3.4.3) allows establishing a meaningful traversal of the translatable original phrases, but in order to produce the translations, the goal is to traverse the corresponding translation phrases, which is supported by the translation graph. Again, the cover graph will assist in creating the translation graph, simplifying the edge creation between the phrase translations of adjacent original phrases and ensuring every original phrase is covered only once.

First, each node of the cover graph will produce a set of translation nodes. Such set will contain one translation node for each phrase translation of the original phrase represented by the cover node. Then, each translation node produced by a cover node n_0 will be connected to each translation node produced by a cover node n_1, for every pair of cover nodes n_0 and n_1 connected by an edge. Figure 19 illustrates the several stages on how cover edges produce translation edges: stage (1) presents two cover nodes connected by a cover edge; stage (2) presents the translation nodes produced by the cover nodes presented in stage (1); and stage (3) presents the translation edges connecting the translation nodes presented in stage (2).

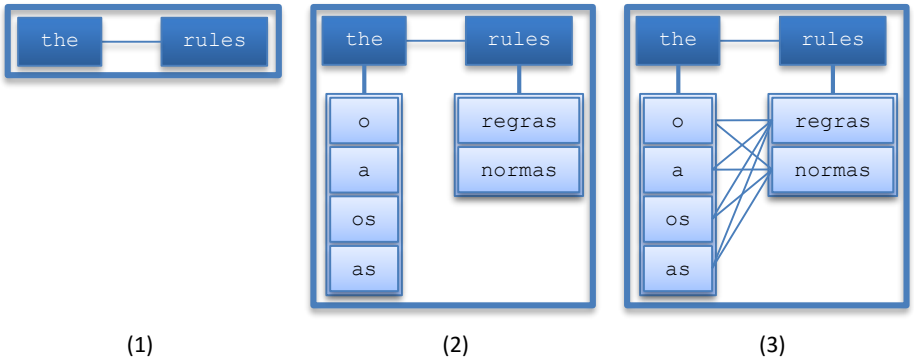

Figure 19. The several stages for producing translation edges from a cover edge

An example of a translation graph produced from a cover graph is shown in Figure 20. As with the cover graph, artificial starting and ending nodes are included in the translation graph.

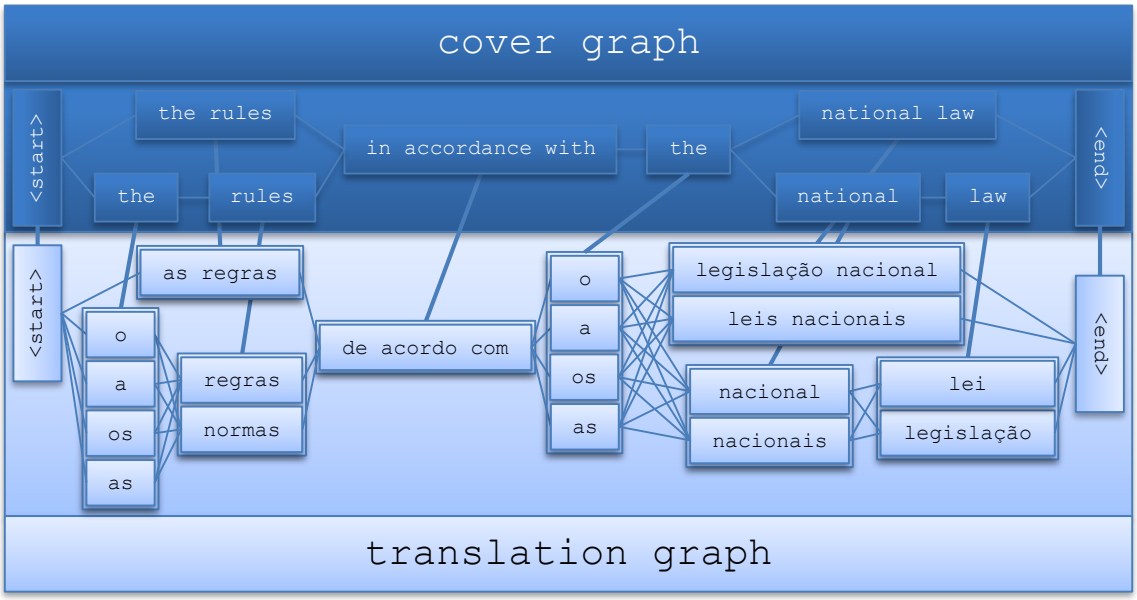

Figure 20. Translation graph resulting from a cover graph

The translation graph enjoys the same properties as the cover graph: unidirectional; no cycles; and exploring common sub-paths. Consequently, the complexity improvement from the number of paths to the number of edges is also verified in this case.

Once the translation graph has been created, the nodes already include the *ptm* scores (Section 3.3.2.1), but the *plm* scores (Section 3.3.2.2) are still missing for the edges, so an additional stage has to be carried out in order to include them, with such stage being described in Sub-Section 3.4.6.3 below.

3.4.5 Including the Translation Pattern Feature

The translation pattern feature (Sub-Section 3.1.1.2) is included in the translation process of Transtor through three different stages: matching; validation; and graph integration. The matching stage, described in Sub-Section 3.4.5.1, will confirm if a translation pattern can be applied to a candidate phrase and will instantiate the pattern variables with the corresponding sub-phrases of the candidate phrase. The validation stage, described in Sub-Section 3.4.5.2, will ensure the instantiated pattern variables from a previously matched translation pattern are translatable. Finally, the graph integration stage, described in Sub-Section 3.4.5.3, will integrate a previously validated translation pattern into the translation graph. As a final note, the translation pattern feature requires the inclusion of at least one source of contiguous phrase translation equivalents.

3.4.5.1 Matching a Translation Pattern to a Candidate Phrase

Matching the source part of a translation pattern to a candidate phrase requires that the literal parts find their exact matches on the candidate, while the remainder sub-phrases (not matched with the literal parts) of the candidate will instantiate their corresponding pattern variable elements.

A matching example is shown in Figure 21, in which it is intended to obtain the Portuguese (target language) translation of the English (source language) candidate phrase *T* = "`turn the lights off`" using the translation pattern on entry number 7 of Table 17. Matching the corresponding source and target languages on the translation pattern, "`turn <var> off`" is considered the source pattern and "`desligar <var>`" is considered the target pattern.

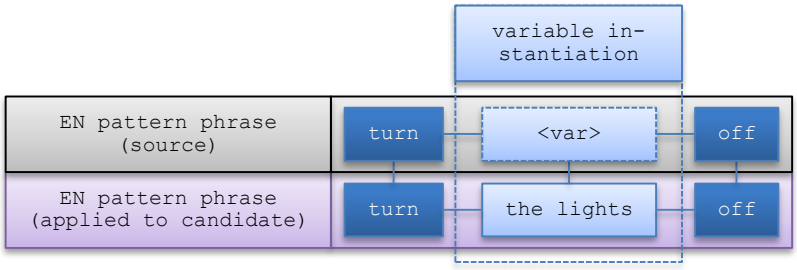

Figure 21. Matching a translation pattern to a candidate

The figure illustrates the source pattern literals "`turn`" and "`off`" being matched by the candidate sub-phrases "`turn`" and "`off`" respectively, leaving the variable from the translation pattern to be instantiated with the candidate sub-phrase "`the lights`".

Once the variables have been effectively instantiated with their corresponding sub-phrases, a validation process needs to be carried out on those sub-phrases in order to make sure each one is translatable. Going back to the example from Figure 21, it will be necessary to confirm that the sub-phrase "`the lights`" is translatable.

Such verification is required to ensure the matched translation pattern is translatable, and is carried out in the validation process, described in Sub-Section 3.4.5.2 below.

3.4.5.2 Validating a Matched Translation Pattern

A matched translation pattern (Sub-Section 3.4.5.1) is only confirmed to match the candidate phrase, which means the literals have found their matches while the variables have been instantiated with their corresponding sub-phrases on the candidate phrase.

However, the matching of a translation pattern does not ensure it is translatable, which is a necessary condition for the translation pattern to be considered in the translation process. A matched translation pattern is translatable only if all of its instantiated variables are also translatable. An instantiated variable is translatable if the sub-phrase instantiating it can be translated. The translatability of such sub-phrases follows the same criteria used for the translation of complete sentences, requiring each sub-phrase to be translated either by a single phrase or by a combination of phrases covering the complete sub-phrase, with no gaps or overlaps.

The several possible ways to translate sub-phrases associated to pattern variables are represented by variable cover graphs, which will, in turn, produce the corresponding variable translation graphs. Again, taking the example from Figure 21, the sub-phrase "`the lights`" (associated to the variable of the source pattern) needs to be translated either by the single phrase "`the lights`" or by both "`the`" and "`lights`" individually, so that a traversal from the node `<start>` until the node `<end>` of the variable cover graph is possible. In a case where a translation is available for all the mentioned phrases, the sub-phrase associated to the variable can be translated in two possible ways, as shown by the variable cover graph represented in Figure 22.

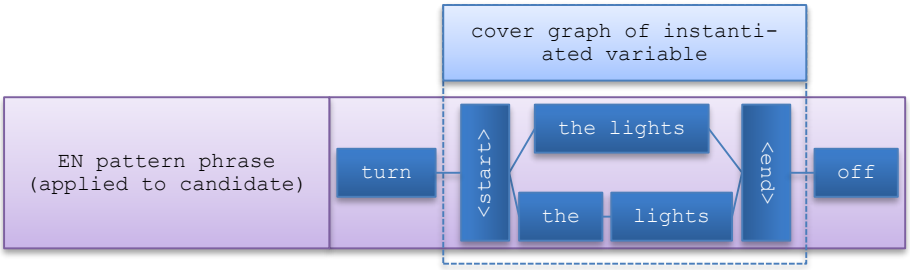

Figure 22. Valid matched translation pattern

Once it has been confirmed that every variable from a matched translation pattern is translatable, the translation pattern can then be safely integrated into the translation graph, as described in Sub-Section 3.4.5.3 below.

3.4.5.3 Integrating a Validated Translation Pattern

Obtaining a translation through the application of a validated matched translation pattern (Sub-Section 3.4.5.2) requires its integration into the main translation graph. Such integration implies the placement of the cover graph of the translation pattern in its corresponding position in the main cover graph in order to produce the corresponding final translation graph.

As an example, consider the partial sentence S = "`must turn the lights off when`", which has T = "`turn the lights off`" as a sub-phrase that can be translated by the translation pattern shown in the previous sub-sections. The corresponding cover graph is shown in Figure 23, where the variable cover graph is still generically represented by a single node.

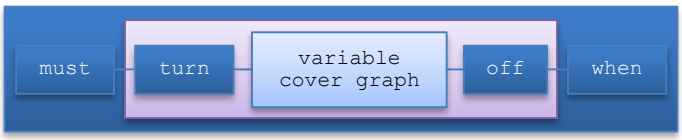

Figure 23. Cover graph integration

Again, the purpose of the cover graph is to simplify the creation of the translation graph. Figure 24 shows the resulting translation graph from Figure 23, where the translation graph from the variable is also generically represented by a single node.

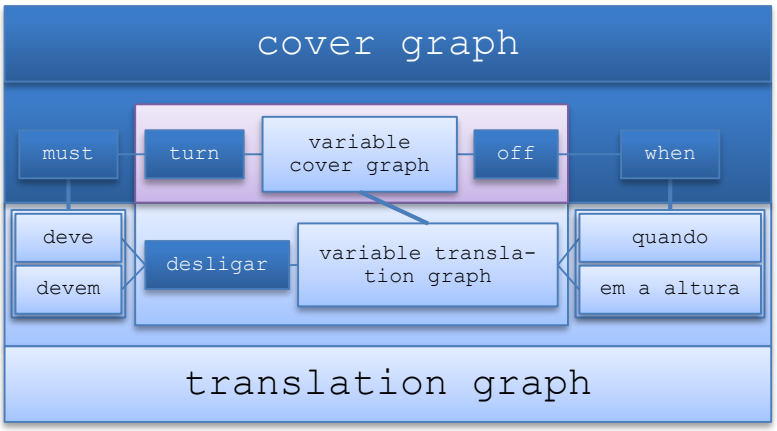

Figure 24. Translation and cover graphs

Before moving on the final graphs, the variable cover graph presented in Figure 22 will produce the variable translation graph represented in Figure 25. Those variable cover and translation graphs correspond to the ones generically represented in the previous figures.

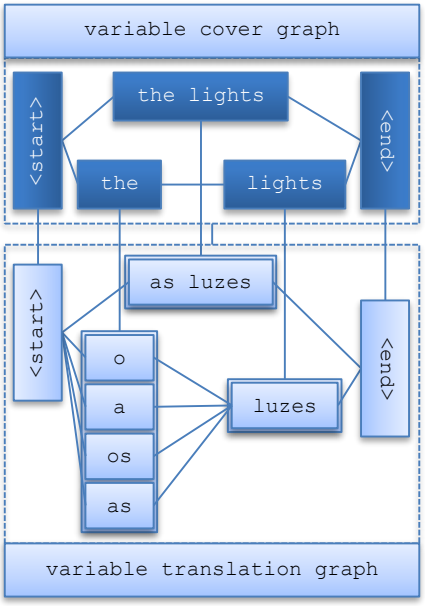

Figure 25. Translation graph from the variable

Having introduced the generic cover and translation graphs in Figure 24, and the variable cover and translation graphs in Figure 25, it becomes easier to explain how those are integrated to obtain the final cover and translation graphs in Figure 26.

The figure shows that the phrase translations from "must" ("deve" and "devem") are connected to the first element from the target part of the translation pattern (the literal "desligar"), the literal "desligar" is connected to the following element from the translation pattern (the variable), and the phrase translations from "when" ("quando" and "em a altura") are connected to the last element from the target part of the translation pattern (the variable).

So far, the variable has been connected to the corresponding elements but, in order to obtain the final graph, the elements from the variable are the ones that have to be connected. In the example, this means, the literal "desligar" is connected to the elements which are in turn connected to the artificial node <start> from the variable translation graph ("as luzes", "o", "a", "os" and "as") and, following the same logic, the elements connected to the artificial node <end> from the variable translation graph ("as luzes" and "luzes") will now be connected to "quando" and "em a altura".

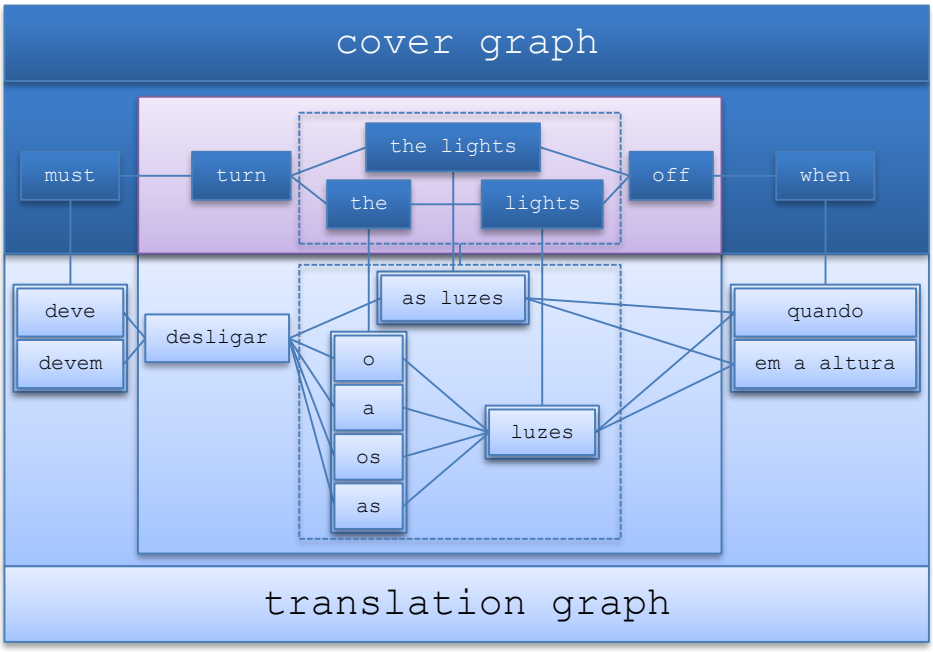

Figure 26. Translation graph integration

With the integrated translation pattern, in which the nodes already include their translation model, the remaining graph traversals can be carried out, namely for including the language model (Sub-Section 3.4.6.3), decoding (Sub-Section 3.4.8), and showing the final translation (Sub-Section 3.4.10).

Since the translation patterns used in Transtor have all been verified (another manifestation of supervision), and since the literal elements from one part of the translation pattern do not necessarily correspond to another literal on the other part of the translation pattern, the *ptm* scores of the literal elements is decided to be 1.0, expressing the certainty of their correction.

3.4.6 Including the Translation and Language Models

Once the graph is composed, the translation and language models have to be used in the translation stage. The previous sub-sections described the *ptm* and *plm* models trained from a single source. However, as mentioned above, each model can use several sources to produce a combined final score. Using several distinct sources allows them to be kept separate, for instance by topics, domains or subjects, to then use the ones considered most relevant to any translation in question.

3.4.6.1 Combining Several Sources for the Translation Model

In this approach, several phrase translation sources (aligned parallel corpora) can be trained separately, as described above (Sub-Section 3.3.2.1), so that the ones considered most relevant for a given translation can be combined to pro-

duce a final *ptm* score. So, for a selection of n different sources, each with a model ptm_i, a combined *ptm* is obtained using the formula expressed in Equation 16.

$$ptm(e|f) = \max_i(ptm_i(e|f))$$

Equation 16. Combining several phrase translation model scores

The combined *ptm* allows a more balanced score between the different alternatives. With this solution, it would be possible to assign a score of 1.0 to several (or even all) alternative translations for a phrase, which is consistent with the goal of assigning balanced scores to equally correct entries. Because it is possible to assign a score of 1.0 to more than one alternative translation for a given phrase, it is not ensured that the sum of the probabilities of those alternative translations is 1.0, this way showing that the *ptm* score does not correspond to a probability distribution.

3.4.6.2 Combining Several Sources for the Language Model

As with the *ptm* score, in this approach, several separate monolingual sources can be trained separately, as described above (Sub-Section 3.3.2.2). So, for a selection of n different sources, where each source provides a $context\_value_i$, the combined *plm* score of translation phrase *fe*, as a following of (occurring exactly after) a translation phrase *e* with m following translation phrases fe_j, is obtained using the formula expressed in Equation 17. As with the individual *plm*, if every context value is 0, the final combined value is also defined to be 0.

$$plm(fe|e) = \frac{\sum_{i=1}^{n} context\_value_i\,(e, fe)}{\sum_{i=1}^{n}\sum_{j=1}^{m} context\_value_i\,(e, fe_j)}$$

Equation 17. Combining several phrase language model scores

In this case, such definition allows the results to be the same as merging all sources and training them as a single source but, again, this solution has the advantage of using combinations of sources considered most relevant for a given translation.

3.4.6.3 Adding the Language Model

The *plm* score (Section 3.3.2.2) considers a set of phrases following a current phrase and evaluates how likely each phrase is to follow the current one. Such relations between current and following phrases are established by the edges from the translation graph (Sub-Section 3.4.4), which is why the *plm* score depends on the translation graph to be determined first.

Calculating the *plm* score for every edge is accomplished through a traversal of the translation graph, using a depth-first traversal to process each individual node along with their corresponding following nodes. Processing an individual node requires determining the corresponding context value for each of its following nodes (if any), with their sum being used to normalize each of the context values individually, producing the intended *plm* score.

Once the *plm* score has been determined for all the edges from the translation graph, the decoding process can begin, as described in Sub-Section 3.4.8 below.

3.4.7 SENTENCE TRANSLATION MODEL

The sentence translation model (*stm*) developed for Transtor combines the phrase translation model (Section 3.3.2.1) and the phrase language model (Section 3.3.2.2) to determine the score of a translation produced for a sentence, this way allowing the selection of the best translation amongst several possible ones in the decoding stage described ahead (Sub-Section 3.4.8).

When determining the *stm* score of a translation candidate of a sentence, every pair of adjacent translation phrases is analyzed by combining the *ptm* scores of the corresponding phrase translations and the *plm* score between the phrases of an adjacent pair. More specifically, for a particular pair of adjacent translation phrases *e* and *fe*, the *ptm* and *plm* scores involved are combined as shown by the edge score function $es(e, fe)$ defined in Equation 18.

$$es(e, fe) = tw \cdot ptm(e) + lw \cdot plm(e, fe) + tw \cdot ptm(fe)$$

Equation 18. Edge score between adjacent translation phrases

The edge score considers $ptm(e)$ and $ptm(fe)$, evaluating how likely *e* and *fe* translate their corresponding original phrases, and $plm(e, fe)$, evaluating how likely phrase translation *fe* is to follow phrase translation *e*, where *tw* is the weight of the *ptm* score and *lw* is the weight of the *plm* score.

Since both score weights can be represented as a function of the other, there is only one degree of freedom. In this case, *tw* is represented as a function of *lw*, as $tw = \frac{1-lw}{2}$, with *lw* admitting a value between 0.0 (the language model is not considered) and 1.0 (the translation model is not considered). This way, the edge translation model represents an interpolation of the contribution of the translation model and the language model.

Now, considering that, in general terms, an original sentence can be partitioned into *n* non-empty contiguous phrases $f_1, f_2, \ldots, f_n$ to produce a translation sentence $e_1, e_2, \ldots, e_n$, with each e_i being a non-empty phrase translation of the original phrase f_i, the *stm* score for the translation sentence will evaluate every ad-

jacent pair of phrase translations, using the edge score presented in Equation 18, as shown in Equation 19.

$$stm(e_1, e_2, \dots, e_n) = \sum_{i=1}^{n-1} \frac{es(e_i, e_{i+1})}{(n-1)^{\alpha}}$$

Equation 19. Score of a combination of translation phrases

The division in Equation 19 allows the fair comparison between translation sentences with different number of phrases, otherwise translation sentences with a higher number of phrases would have an unfair (and unwanted) advantage. The alpha parameter in *stm* allows increasing the penalization of translation sentences with more phrases: the higher the alpha, the lower the scores for sentences with more phrases, consequently benefitting sentences with fewer phrases. In fact, tests have shown that translation sentences with fewer phrases have generally better quality, which is consistent with the fact that the longest possible phrase combinations represent word by word translations, which are known to be generally bad translations. This alpha parameter intends to accomplish the same purpose as the penalty models described in Sub-Section 2.1.3.4, which in this case is a phrase-based penalty model. Finally, in cases where a sentence can be translated by a single phrase f_1, the translation score is simply determined by $ptm(e_1)$.

The formula described here shows how a single translation possibility is evaluated, but the problem consists in finding the best translation amongst a great number of translation candidates. Finding the best translation is accomplished in the Transtor decoding stage, described ahead in Sub-Section 3.4.8, where it is shown how the potentially immense number of possibilities can be processed efficiently.

3.4.8 DECODING THROUGH GRAPH TRAVERSAL

The *stm* score, presented in Section 3.4.7, evaluates a single translation candidate. However, the translation process can produce a great number of translation candidates and the purpose is to find one that maximizes the *stm* score. Such maximization process is called decoding which, in Transtor, is accomplished by traversing the translation graph (Sub-Section 3.4.4). However, the translation graph can have millions of different paths, with each individual path representing a single translation candidate, and calculating their individual scores, in order to be compared against each other, to then select the one having the highest score, would render the process impractical. Fortunately, decoding can be efficiently processed following the observations described below.

First of all, e_n will represent the first phrase of a sentence and e_1 will represent the last, so a sentence divided into n phrases will be represented as $e_n, e_{n-1}, \dots, e_1$. Then, to enable a more natural application of the *stm* score expressed in Equation 19 on the graph, the recursive form of the *stm* score (the *rstm* score) is presented in Equation 20 (the equivalence between both formulas is shown in the Annex). These settings allow a more direct relation between the recursive formula and the actual graph traversal, where the cumulative score of a phrase depends on the cumulative score of its following phrase.

$$rstm(e_n) = \begin{cases} 0, & n = 1 \\ \dfrac{es(e_n, e_{n-1}) + rstm(e_{n-1}) \cdot (n-1)^{\alpha}}{n^{\alpha}}, & n > 1 \end{cases}$$

Equation 20. Recursive form of the *stm* score

The *stm* formula, either in its original or in its recursive form, scores a single translation candidate but the goal of decoding is to find the translation candidate with the score that maximizes the formula, so decoding is a maximization problem. Taking advantage of the graph structure, which allows reusing calculations shared by common sub-paths, in a maximization problem consists in keeping only the maximum cumulative values (because lower values will never contribute to a higher total value), so a depth-first traversal is carried out, where every translation node *e*, with *n* following translation nodes fe_i, is traversed only once, having its maximum cumulative *stm* (*mstm*) score calculated with the formula presented in Equation 21, where $d(e)$ represents the depth of node *e*.

$$mstm(e) = \begin{cases} 0, & d(e) = 1 \\ \max\limits_{i} \dfrac{es(e, fe_i) + mstm(fe_i) \cdot d(fe_i)^{\alpha}}{(d(fe_i) + 1)^{\alpha}}, & d(e) > 1 \end{cases}$$

Equation 21. Maximum cumulative *stm* score

Unlike with *rstm*, in which the *n* is used in the formula because the number of phrases is known, in *mstm* the depth is used instead because it will depend on the depth of the following node *fe* with which the highest cumulative score is achieved. Each translation node includes information about the following translation node that maximizes the total up to that point, namely the reference to the node, the corresponding *mstm* and the corresponding depth. Once every translation node has been traversed, the path that provides the highest *stm* score can be reconstructed using the mentioned references. This procedure can be interpreted as the concern of determining which following phrase *fe* should be considered once the current phrase *e* has been reached.

In case the cover graph is composed by a single node, representing a single original translatable phrase, the translation graph will have one or more paths, each consisting of a single translation node, none of them having any adjacent translation phrases for which to calculate the *plm* score. Such situation results in the translation node with the highest *plm* score to be presented as the final translation.

In the end, this traversal determines the information needed to traverse the graph once more in order to present the final translation, as described in Sub-Section 3.4.10 ahead.

3.4.9 NON-TRIVIAL CASES

The description made so far assumed perfect conditions. However, this is not always the case and some non-trivial cases may surface during the translation stage. Those situations are explained in the sub-sections below.

3.4.9.1 Untranslatable Sections

As already mentioned above, the existence of untranslatable phrases might originate untranslatable text sections (portions of text that are not covered by any translation). For this reason, translation sections (either translatable or untranslatable) are identified in order to apply the translation process to the sections that are indeed translatable.

#	phrase/term
6	`in`
12	`law`
13	`national`
14	`national law`
15	`rules`
22	`the`
25	`the rules`
32	`with`

Table 49. Availability of phrase translations

Consider again the text T = "`the rules in accordance with the national law`", with Table 49 showing the phrases from Table 47 for which translations are available. The presented translatability of the phrases results in the text sectioning depicted in Figure 27, which has an untranslatable section composed by the phrase "`accordance`". The figure shows that there is a connection between the node representing the first occurrence of the phrase "`the`" and node "`in`" through node "`rules`", a connection between node "`with`" and node "`law`" through the node representing the second occurrence of the phrase "`the`" and node "`national`", but there is no way of reaching node "`with`"

from node "in", resulting in the untranslatable section formed by the phrase "accordance".

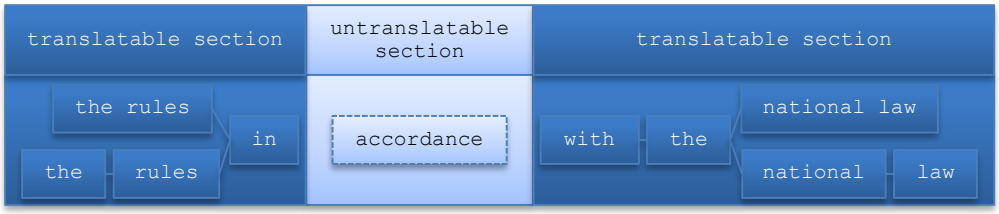

Figure 27. Translation sections

The untranslatable section present in the graph shown in Figure 27 would not occur if Table 49, on which the graph is based, signaled a translation availability of either "in accordance", "accordance with" or "in accordance with". For instance, Figure 28 shows the text sectioning that would result from also having a translation for "in accordance with", which consists of a single section covering *T* completely.

Figure 28. Translatable section covering the whole text

The identification of translation sections uses the offset ranges of the translatable phrases to first determine the translatable sections, also represented by offset ranges. The identified offset ranges representing the translatable sections are then used to identify the ranges from the untranslatable sections through a complement set.

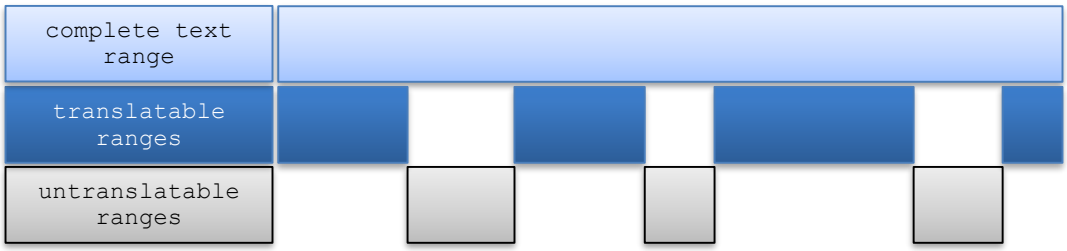

Figure 29. Translatable and untranslatable ranges

It should be noted that before checking for translation sections, it is necessary to validate any matched translation patterns (Sub-Section 3.4.5.2) and that each translation section is considered separately to generate the cover graph of its corresponding text (Sub-Section 3.4.3). Untranslatable sections are presented unchanged.

3.4.9.2 Dangling Nodes

As mentioned above, besides the untranslatable sections, untranslatable original phrases can also produce dangling nodes, which have no contribution in the production of complete translations, unnecessarily burdening the process. Such nodes are dangling on the graph, either because they cannot reach the end of the graph (end dangling nodes), or because they cannot be reached from the beginning of the graph (begin dangling nodes). Removing those nodes requires a consistency check since any nodes linked to them might become dangling too.

In the particular example depicted in Figure 28, there are two dangling nodes, corresponding to phrases "in" (end dangling node) and "with" (begin dangling node), shown in lighter colors in Figure 30.

Figure 30. Dangling nodes in a cover graph

The removal of such nodes is carried out through a recursive depth-first traversal of the graph, starting from the node <start> and analyzing each node with the purpose of reaching the node <end>.

The analysis of a current node consists in checking all its following nodes before analyzing the current one, implementing the depth-first traversal. Once all the following nodes have been analyzed, the analysis of the current node consists in removing the references to its following nodes that do not reach the node <end>. In a case in which every following node is removed, the current node is marked as having been analyzed and as unable to reach the end. In the end of the traversal process, references to the end dangling nodes have been removed, making them inaccessible, and the begin dangling nodes were never accessed because the traversal started from the node <start>. Once the dangling nodes have been isolated, the cover graph is ready to support the translation graph creation (Sub-Section 3.4.4) while ignoring those identified dangling nodes.

3.4.10 Presenting the Final Translation

The previous graph traversal carried out for decoding (Sub-Section 3.4.8) will leave each node with information about which is the best translation node to follow from that point onward. Such information is used to follow a left-to-right traversal of the path corresponding to the combination of phrases that maximize the *stm* score, presenting the phrase translation associated to each traversed

node, beginning at the artificial node <start> and finishing at the artificial node <end>. In other words, the artificial node <start> points to the first node of the best path, which in turn points to the following node of the best path, and so on, until the artificial node <end> is reached. The nodes presented along the way will constitute the final translation.

3.5 DIFFERENCES BETWEEN TRANSTOR AND MOSES

Both Moses and Transtor depend on parallel corpora to calculate their translation models and depend on monolingual corpora to calculate their language models. However, before introducing the results obtained with both systems, it is important to stress out their differences.

First of all, as mentioned earlier, the main distinctive character is the fact that Transtor is a truly phrase-based approach, since phrases are considered to be units by themselves and not just a composition of words. This is true in every element that composes the system, namely the alignment used as a base, the phrase translation extraction, the translation model, the language model and the penalty parameter.

In the case of Moses, extracted translation equivalents and, consequently, the translation model, also apply to phrases. However, extracted phrase translation equivalents are obtained from a word-based alignment and the remaining language model, reordering model and penalty model all consider words as their units.

Another distinctive feature already mentioned is that Transtor considers supervision, mostly manifested through the use of validated bilingual phrase lexicon. These and other differences will be discussed individually and in detail in the sub-sections below.

3.5.1 ALIGNMENT AND PHRASE TRANSLATION EQUIVALENTS

In Moses, a parallel corpus is aligned using GIZA++[7], which implements the IBM models (Brown et al., 1993). It requires sentence aligned texts and produces a word-based alignment without any supervision. Such alignment, run in both language directions, to produce a symmetrized word-based alignment, supports the extraction of phrase translation equivalents by considering phrases consistent with the symmetrized alignment. Yet, as mentioned earlier, a word-based alignment will hardly deal with situations in which there is no clear relation between words, as shown by the phrase translation equivalent example “clockwise” <=> “em o sentido de os ponteiros de o relógio”

[7] The GIZA++ toolkit (http://www.isi.edu/~och/GIZA++.html).

(Sub-Section 2.1.2.1). Such example can be used to show not only that phrases should be considered units, but also that lexica validation (Sub-Section 3.1.2) is necessary to prevent those problems, since there is no obvious way to associate such equivalents.

When the alignment process only uses statistics to align words or phrases, it can be very difficult to avoid many recurring misalignment situations. These can be prevented more effectively with the use of supervision, where known entries, previously verified by a user, can help a system improve its process of detecting correct translation equivalents.

In the case of Transtor, a parallel corpus is aligned with the FCT Aligner (Sub-Section 2.3), which is phrase-based and uses previous knowledge in the form of verified bilingual phrase lexica to improve its quality (previously detected translation equivalents are used to provide support in detecting additional translation equivalents). Transtor then uses the resulting alignment not only to consider the phrases that were recognized in the alignment process, but also to identify additional phrase translations not yet recognized by the system, using some methodology which explores common alignment patterns (Sub-Section 3.3.1.1).

Also, as main SMT research, Moses is only focused on creating a bilingual phrase lexicon from scratch. In a way, it assumes that previous knowledge in the form of a validated bilingual phrase lexicon is never available. However, there can be information available about how to translate a great number of phrases, as well as there are language constructs that follow very specific rules. In the approach supporting Transtor, such knowledge can be expressed both in the literal form of bilingual phrase lexica (contiguous phrases) and in the more generic form of translation patterns (non-contiguous phrases).

3.5.2 TRANSLATION MODEL

Once Moses has identified the phrase translation equivalents, it computes five different phrase translation scores that compose its translation model (Sub-Section 2.1.3.1):

- the direct phrase translation probability $p(e|f)$;
- the inverse phrase translation probability $p(f|e)$;
- the inverse lexical weighting $lex(f|e)$;
- the direct lexical weighting $lex(e|f)$; and
- the phrase penalty, which is always $exp(1) = 2.718$.

The purpose of combining all those probabilities is to improve the probability of assigning high scores to phrases that are highly likely to translate other phrases. However, phrases like "`clockwise`" <=> "`em o sentido de os ponteiros de o relógio`" will also present problems to lexical translation probabilities, and inverse translation probabilities are not intuitive, since they follow the inverse direction of translation.

Transtor simplifies all of the above by using a simple translation model, consisting of the direct phrase translation probability (Sub-Section 3.3.2.1). The combination of translation models from several sources (Sub-Section 3.4.6.1) reduces the problem of producing a score which would mainly reflect how frequently a phrase translation has been associated on another phrase.

3.5.3 LANGUAGE MODEL

One of the language models used by Moses (and the one used in the tests presented in Chapter 4) is IRSTLM[8], which is an implementation of an n-gram history model (Sub-Section 2.1.3.2). Such model uses an absolute n-gram analysis to determine how likely a given word is to appear after a set of n – 1 words for any given sentence, so the analysis is done at the word level. Also, because of the nature of the translation probability score (Equation 22 in Sub-Section 3.5.5 below), this model needs to avoid producing null values because they would nullify the whole translation probability, something that is avoided through smoothing techniques.

The language model used by Transtor is very different. It begins by considering phrases as units, so the internal fluency of phrases like "`in spite of`" or "`in accordance with`" will not be considered (unlike with the n-gram history model). Instead, the model takes pairs of adjacent phrases, with each phrase having one or more words, and determines how likely they are to be found together according to their adjacent alternatives, as described in Section 3.3.2.2. Also, like the translation model, the language model can be the combination of language models from several sources (Sub-Section 3.4.6.2). As a final difference, null probabilities are not avoided in this model because, due to of the nature of the translation probability score employed by Transtor (Equation 24 in Sub-Section 3.5.5 below), they do not represent a problem, even though such probability would not enable to discriminate between "never occurs" and "has not been observed". However, with the alternative of preventing zero probabilities with smoothing techniques, which assigns some small value to unobserved events, it would not be possible to discriminate between "never occurs", "has

[8] The IRSTLM language modeling toolkit (http://sourceforge.net/projects/irstlm).

not been observed" and "barely occurs". This way, a model should be capable of dealing with unobserved events which could be compensated by other models or elements. In other words, a model should be prepared for the likely situation in which a particular event has not been observed, particularly by taking advantage of other events that have indeed been observed.

3.5.4 PENALTY MODEL

The penalty model used in Moses considers the number of words in the sentence, expressed as $W(e) = \exp(length(e))$. As explained earlier (Sub-Section 2.1.3.4), the purpose of this model is to maintain a balance between the number of words from the sentence being translated and its translation, since the language model (Sub-Section 2.1.3.2) employed by Moses prefers shorter translations. However, the translation equivalent "`counterclockwise`" <=> "`em o sentido contrário a o de os ponteiros de o relógio`" also represents a good example of how the word-based penalty model faces challenges, since there is no clear balance when a sentence and its translation involve such phrase translation equivalent.

This is in contrast with Transtor, which can deal with such situations, particularly if such equivalent is available in the verified bilingual phrase lexicon, considering it as a single element and not as a set of several words. Transtor uses a phrase-based penalty parameter (Sub-Section 3.4.7) that considers phrases as units, penalizing translations with the most phrases and, this way, favoring translations with fewer phrases. This penalty parameter also provides additional balance between the phrase translation model and the phrase language model. In fact, smaller adjacent phrases might have higher language model scores because they are more likely to be found together but their overall quality might not be the best, while larger adjacent phrases usually have lower language model scores but their overall quality might be very acceptable because their lower language model scores can be counterbalanced by the (general) greater quality of each phrase. As an example, translating "`in spite of`" with three phrases (each phrase corresponds to a single word) will have a higher penalty than translating it as a single phrase. Still, a translation having fewer phrases does not mean it has fewer words. Some translations with fewer phrases may be longer because each phrase may also contain more words.

3.5.5 TRANSLATION PROBABILITY SCORE

Besides the fact that some models are not shared between Moses and Transtor, and the fact that the ones that are shared have some differences, the translation probability score that combines the models on each system is also different.

In Moses, the probability $p(e|f)$ of the translation sentence *e* given the original sentence *f* is broken up into the four models, as shown in Equation 22:

- the translation model $p_{tm}(f|e)$;
- the language model $p_{lm}(e)$;
- the distortion model $D(e,f)$; and
- the word penalty $W(e) = \exp(length(e))$.

Each of the four models is weighted by a weight w_i.

$$p(e|f) = p_{tm}(f|e)^{w_{tm}} \times p_{lm}(e)^{w_{lm}} \times D(e,f)^{w_d} \times W(e)^{w_w}$$

Equation 22. Translation probability score applied in Moses

In fact, given that the Moses decoder internally uses logarithms (hence, the "log-linear" designation), what is indeed computed is expressed in Equation 23, but both formulas are equivalent.

$$p(e|f) = \exp(w_{tm} \log p_{tm}(e|f) + w_{lm} \log p_{lm}(e) + w_d \log D(e,f) + w_w \log W(e))$$

Equation 23. Log-linear model form of the translation probability score

The log-linear approach is a very popular model, adopted from the machine learning field, used to determine the probability of a translation as a combination of several feature models (like the translation model, the language model, the reordering model and the penalty model). Taking advantage of already developed concepts, as well as allowing any number of feature models, are certainly advantages in favor of the log-linear model. However, from my point of view, its definition as a product of features can be considered a disadvantage since it nullifies total results when confronted with any null values. Avoiding such situation requires smoothing techniques for any models involved that could produce such values (for which the language model is an example) but at the risk of those techniques introducing other errors. Another problem resulting from being defined as a product of features has to do with low individual values having a big impact on the final result.

The approach followed by Transtor is much simpler, using an average of the translation and language models, penalized by the number of phrases (Sub-Section 3.4.7), as shown in Equation 24.

$$stm(e_1, e_2, \ldots, e_n) = \sum_{i=1}^{n-1} \frac{tw \cdot ptm(e_i) + lw \cdot plm(e_i, e_{i+1}) + tw \cdot ptm(e_{i+1})}{(n-1)^{\alpha}}$$

Equation 24. The *stm* score applied in Transtor

Knowing that a translation candidate is produced by decomposing a source sentence into smaller units (words or phrases) which are translated individually

and then recombined to produce a complete translation for the source sentence, the purpose of either the log-linear model or the *stm* is to take the scores of those smaller units in order to produce a score for the whole translation candidate.

To illustrate the advantage of *stm* over the log-linear model, consider a toy example in which there are two alternative combinations, X and Y, which have to be evaluated in order to decide which combination is the best. Consider that each alternative combination has three elements (words or phrases), with the scores of alternative X as $\{X\} = \{0.2,0.2,0.2\}$ and the scores of alternative Y as $\{Y\} = \{0.02,0.5,0.5\}$. The scores of the individual elements of each alternative are then used to produce a final score for each alternative and those scores are then used to decide between X and Y.

At a first glance, the scores from X are very balanced in comparison with the ones from Y. However, Y has two elements with a relatively high score and a third one with a very low score. Intuitively, even though the scores from Y are not as balanced scores as the ones from X, it could also mean that the two elements from Y having the highest scores are very adequate options, requiring Y only one edition on its element with the lowest score to turn the set into an acceptable choice, while in the case of X the balanced low scores could correspond to an average mediocre alternative which might require editing all of its elements. This intuitive analysis is better expressed with the average, as will be shown below.

Going back to the need of deciding between X and Y, the product of scores will first be used, in which case $P(X) = 0.2 \times 0.2 \times 0.2 = 0.008$ and $P(Y) = 0.02 \times 0.5 \times 0.5 = 0.005$, so $P(X) > P(Y)$, resulting in the selection of combination X. This example shows that a very low score easily penalizes a total result when the probability is defined as a product of scores. Alternatively, if the simple average was used instead, $P(X) = (0.2 + 0.2 + 0.2)/3 = 0.2$ and $P(Y) = (0.02 + 0.5 + 0.5)/3 = 0.34$, so $P(X) < P(Y)$, in which case combination Y would be the one selected, in agreement with the intuitive observation in the paragraph above. This shows that while using the average, low values (and even null values) are only reflected locally without a significant negative impact on the complete set, as opposed to using a product for which the negative impact of low scores is easily propagated, ruining the whole score.

Another example supporting the intuitive notion above can be a situation in which a translation path is being selected from partial information. Consider another toy example in which a set of adjacent elements (again, words or

phrases) {A, B, C, D} has to be scored. For that purpose, it will be necessary to analyze the context between A and B, B and C, and C and D. In such analysis, there might not be available evidence supporting the consideration of B and C together. However, the information available between A and B and between C and D might compensate for the lack of evidence between B and C. In other words, B can be selected because of A, and C can be selected because of D. This is a flexible solution that does not require evidence for every adjacent pair to make a decision. Besides, if the translation model provides a high likelihood for a translation that is not supported by the language model (not because it does not occur but because it has not been observed), it could mean that a translation is being proposed to a perfectly admissible new context. As such, the approach presented here is flexible enough to deal with unobserved alternatives that might actually be correct, possibly compensated by adjacent alternatives, which should not be immediately discarded. With this approach, it is possible to admit "`first`", "`last`", "`best`", or any other phrase to fill the blank (as long as it is admissible by the translation model) of, for instance, "`the <blank> rule`", even if there is no actual evidence supporting the choice. Additionally, if the translation model suggests "`best`", finding some partial evidence (like finding evidence for "`the best`" and not for "`best rule`") could still justify the choice of "`the best rule`" as the final translation.

3.5.6 DECODING

Both approaches implement decoding as a best path search in a directed acyclic graph, but there are still differences to be pointed out.

In the case of Moses, the search is implemented as the expansion of partial hypothesis, from left to right, starting with an empty hypothesis until a complete hypothesis covering all words is found. A cover vector is required during the search to determine which words have already been covered in the partial translation built so far (Sub-Section 2.1.4).

Unlike Moses, Transtor builds a complete directed acyclic graph with all the available phrase translations, implicitly determining which phrases (not simply words) are already covered. Each path represents a possible sentence translation (Sub-Section 3.4.8). Decoding is then accomplished through the depth-first traversal of the directed acyclic graph in order to determine the path with the highest score. Such path represents the combination of phrases to be presented as the translation of a full sentence.

4 RESULTS

Having introduced Transtor as a new approach to Machine Translation, it is important to assess its quality. This chapter will present a translation evaluation carried out through a set of tests that were meant to be run by Transtor and Moses (Koehn et al., 2007). Moses was chosen to provide reference results because it "is considered the de facto benchmark for SMT"[9]. The results from both systems are then evaluated using BLEU (Papineni et al., 2002), which is the most commonly used evaluation score. Scores from both systems are then analyzed and compared for two sets of parallel corpora. Transtor is further analyzed with three more sets of parallel corpora, not only because it trains much faster, but also because Moses was not able to process the largest parallel corpora, which was the most interesting one to analyze.

Yet, a note should be added about parameter tuning. The purpose of the tuning process is to determine the best values for the several weight parameters being used in translation and is the most expensive stage of SMT in terms of processing time. Such process requires an additional parallel corpus, independent from the corpus used in the models training, but it should be of a similar domain as the texts being translated (Pecina, Toral and Genabith, 2012), since different tuning corpora will produce different sets of weight values which will reflect how the SMT system behaves with each given tuning corpus. In particular, in a case when an evaluation is to be carried out on several language pairs, ideally the same parallel corpora should be available between them in order to ensure the comparability of their results. On top of all that, tuning might be faced with local maxima that, given the non-deterministic nature of the process, might prevent the global maximum from being found.

As such, given the difficulty to fulfill all the requirements above for every language pair, at the cost of a tremendously significant increase in processing time per language pair (as mentioned, for instance, in the Moses manual), and at the risk of only minor improvements when compared to the default weight values made available by Moses (Haddow, Arun and Koehn, 2011), (Pecina, Toral and Genabith, 2012), a decision was made to not carry out the tuning process. This way, the purpose of this analysis is to produce translations with Moses using its default weight values for decoding and compare them with the ones produced by Transtor using a set of parameters chosen according to the intuitive notion of the relative relevance between the models and the importance of the number of

[9] Quote from the Moses manual.

phrases composing a translation candidate. The parameter choice made for Transtor might not be the best or might even be some fortunate guess, also depending on the corpora used for training to translate a given text, but the same applies to the default weight values from Moses, so the conditions are comparable between both systems.

A more detailed analysis of Transtor parameter values will be carried out for the language pair providing the best results and the one providing the worst results. This will be done not only to see if there is any margin for improvement, but also with the purpose of determining how parameter values affect translation quality, hopefully determining a range of values for which it is most likely to obtain the best results. Such further analysis for Transtor is affordable because of the limited number of parameters involved (one to determine how relevant will be the language model against the translation model, and another to determine how much the number of phrases will penalize a total score) and their limited range of values.

4.1 DESCRIPTION OF TEST SETTINGS

As mentioned before, comparability of results is ensured by translating a set of reference documents after training both MT systems with the same parallel corpora. The quality achieved by each MT system is assessed with the BLEU scores of the translations obtained from the reference documents. The languages tested are "en" (English), "pt" (Portuguese), "es" (Spanish), "fr" (French) and "de" (German), while the language pairs, analyzed in both their directions, are de-en, de-es, de-pt, en-es, en-fr, en-pt, es-pt and fr-pt.

Some characteristics of the corpora can influence the translation quality: being larger usually provides more information but processing it can also present a challenge; being from a domain relevant to the documents being translated can also contribute to improve quality; and having a controlled vocabulary, in which case the text is written more carefully, avoiding alternative meanings and using limited vocabulary, can also reduce problems. With this in mind, further testing was carried out for Transtor in order to see how it behaved with different types of corpora, like being significantly larger in size, being from a not so relevant domain, and having a non-controlled vocabulary.

The additional corpora were not processed with Moses because it takes a considerably longer time to carry out the training process. In fact, we were unable to process the corpus APERTIUM EURLEX without Moses crashing. Besides, judging by the difference in processing time between Transtor and Moses for

the common corpora analyzed (Section 4.3), it would take around 100 hours of processing time for Moses to complete its training over APERTIUM EURLEX.

Finally, tests were run on a machine consisting of a 64-bit, 4-core, 8-thread processor running at 3.4 MHz, having 16 GB of RAM and 4 TB of disk space.

4.1.1 REFERENCE DOCUMENTS

Information about the reference documents, used to evaluate both systems, namely the number of words and their size in bytes, is found in Table 50 below, with the values representing the averages between all the languages involved. The reference documents are composed of five documents belonging to the Official Journal of the European Union[10] (with "eurlex" prefix) and four documents belonging to the European Constitution[11] (with "euconst" prefix).

File	Words count	Size in bytes
eurlex1	2399	9870
eurlex2	2794	15020
eurlex3	1587	7435
eurlex4	6740	38474
eurlex5	669	3097
euconst1	4245	28072
euconst2	9553	65292
euconst3	5057	32993
euconst4	4483	31110

Table 50. Data about the reference documents

It should be noted that the reference documents are not included in any of the parallel corpora used to train the models, but there might be some document portions contemplated in some of those parallel corpora. Still, this is a realistic scenario and is presented to both systems.

4.1.2 PARALLEL DATA

Tests involving the use of the same parallel corpus with both systems were carried out with OPUS EUCONST[12] and DGT-TM[13]. OPUS EUCONST is a very small parallel corpus, but with a very specific and controlled vocabulary, so using it provides an idea of how the systems behave in such "perfect" conditions. DGT-TM is a significantly larger parallel corpus, but also more generic, so using it provides an idea of how the systems can handle a significant amount of

[10] http://eur-lex.europa.eu/{de,en,es,fr,pt}/consleg/latest/index.htm
[11] http://eur-lex.europa.eu/{de,en,es,fr,pt}/treaties/index-old.htm
[12] The European Constitution (http://opus.lingfil.uu.se/EUconst.php).
[13] The Directorate-General for Translation (DGT) of the European Commission has made its multilingual Translation Memory (TM) publicly accessible (http://ipsc.jrc.ec.europa.eu/?id=197).

data. The number of millions of words (MW) and the size in Megabytes (MB) for each corpus, presented as the average between the languages analyzed, is shown in Table 51 below.

Corpus	MW	MB
OPUS EUCONST	0.138	0.810
DGT-TM	54	305

Table 51. Data about the common parallel data

With the same set of reference documents, additional tests were carried out for Transtor using APERTIUM EURLEX[14], OPUS EMEA[15], and OPUS EUROPARL[16]. APERTIUM EURLEX is the largest parallel corpus and is used with the purpose of analyzing the performance when presented with such a large amount of data. OPUS EMEA is a large corpus about medicines and is used with the purpose of analyzing the translation quality when using a very specific domain not related to the domain of the reference documents. OPUS EUROPARL is a parallel corpus consisting of transcripts of the European Parliament and is used with the purpose of analyzing the behavior of the tool when presented with data in which the language is not so controlled. The data about these additional parallel corpora is shown Table 52 below, also as an average between the languages. As a side note, the corpora prefixed by OPUS where obtained from the OPUS[17] project.

Corpus	MW	MB
APERTIUM EURLEX	79	456
OPUS EMEA	13	77
OPUS EUROPARL	32	183

Table 52. Data about the additional parallel corpora

The information presented in both Table 51 and Table 52 is an average between all the languages involved.

In the case of Transtor, besides using the parallel corpus aligned with the FCT Aligner (Sub-Section 2.3), it also includes the verified bilingual phrase lexica (Section 3.1.2) and the translation patterns (Sub-Section 3.1.1.2) in the test sets. The verified bilingual phrase lexica do not have the same degree of development for every language pair, as shown in Table 53 below.

Language pair	de-en	de-es	de-pt	en-es	en-fr	en-pt	es-pt	fr-pt
Count (thousands)	130	70	217	217	298	749	218	371

Table 53. Number of entries of verified bilingual phrase lexica

[14] Official Journal of the European Union (http://apertium.eu/data).
[15] Documents of the European Medicines Agency (http://opus.lingfil.uu.se/EMEA.php).
[16] European Parliament Proceedings (http://opus.lingfil.uu.se/Europarl3.php).
[17] The Open Parallel Corpus (http://opus.lingfil.uu.se/index.php).

Just like the verified bilingual phrase lexica, the translation patterns do not share the same level of development between every language pair, as shown in Table 54 below.

Language pair	de-en	de-es	de-pt	en-es	en-fr	en-pt	es-pt	fr-pt
Count	8	51	109	705	35	1589	2	9

Table 54. Translation patterns count

Producing translation patterns require an abstract thinking that has proven to be a challenge for the linguists that collaborated with us to produce them. Still, since the English-Portuguese language pair has been processed for a longer period of time, such effort is reflected in its noticeable higher number of entries.

4.1.3 TOOLS SETTINGS

Transtor and Moses each include a set of several parameters that have to be set. Some parameters affect the training stage while other parameters influence the decoding stage. Both stages of Moses were carried out using its default settings, with Release 1.0 (the most recent to date).

The fixed parameters set for Transtor are described in Table 55. The limit of 20 words for the language model was set to be able to account for a pair of phrase translation equivalents, which have a limit of 7 words applied to the translation model. Depending on the relevance of the results, the language model weight, the translation model weight and the penalty parameter will be analyzed further with other values in Sub-Section 4.5.

Parameter	Value
translation model word limit per phrase	7
language model word limit per phrase	20
limit of translation equivalents per phrase	7
language model weight	0.5
translation model weight	0.5
penalty parameter (α)	3

Table 55. Parameters values from Transtor

In the case of Moses, many different parameters are available, as shown in Table 56 below. The parameters comparable with the ones from Transtor are highlighted and on top of the table. It should also be noted that the translation model is composed of 5 different scores, as mentioned in Sub-Section 3.5.2 above. The bottom three parameters from Moses have no direct correspondent parameter from Transtor.

Parameter	Value
translation model word limit per phrase	7
language model word limit per phrase	3
limit of translation equivalents per phrase	20
language model weight	0.5
translation model weight (5 in total)	all set to 0.2
word penalty	–1
distance reordering weight	0.3
lexicalized reordering weight (6 in total)	all set to 0.3
distortion limit	6

Table 56. Parameter set from Moses

The limit of 20 words for the language model used by Transtor compares against the language model used by Moses which only considers 3 words. Moreover, Moses considers 20 translation equivalents per phrase while Transtor considers only 7, which contributes to improve performance and can be justified by the good quality of the considered phrase translation equivalents enabled by supervision.

4.2 RESULTS OF TRAINING THE DATA

The training stage required by the tools needs some time to be carried out and produces a certain amount of data, which also constitute important elements to be compared between the tools. The following sub-sections will present the information relative to the corpora used for both tools (Sub-Section 4.2.1) and the additional corpora (Sub-Section 4.2.2) used to further analyze Transtor. All the information is presented as an average between the languages studied.

4.2.1 DATA ABOUT COMMON CORPORA

Starting with OPUS EUCONST, Table 57 shows such information, as well as the relative gains achieved by Transtor relative to Moses. The table shows that Transtor only takes 5% of the time taken by Moses to train the models and only consumes 53% of the size of the structures produced by Moses.

	Moses	Transtor	Transtor gain
Time (mm:ss)	05:28	00:17	5%
Size (GB)	0.249	0.132	53%

Table 57. Training data for OPUS EUCONST

Once trained, the tools take some time to translate the reference documents, as shown in Table 58, which also includes the relative gains achieved by Transtor. In this case, the table shows that Transtor takes, on average, 15% of the time Moses takes to produce a translation.

File	Moses (mm:ss)	Transtor (mm:ss)	Transtor gain
eurlex1	00:20	00:02	10%
eurlex2	00:38	00:04	11%
eurlex3	00:18	00:03	17%
eurlex4	01:52	00:13	12%
eurlex5	00:07	00:01	14%
euconst1	01:16	00:13	17%
euconst2	03:03	00:34	19%
euconst3	01:36	00:18	19%
euconst4	01:11	00:15	21%
Average transtor gain			15%

Table 58. Translation times using OPUS EUCONST

Table 59 shows the corresponding training information for DGT-TM. Again, the average gains achieved by Transtor in relation to Moses are shown. With this larger corpus, Transtor takes 6% of the time and 64% of the space required by the structures in comparison with Moses.

	Moses	Transtor	Transtor gain
Time (hh:mm:ss)	45:15:19	02:46:49	6%
Size (GB)	51	33	64%

Table 59. Training data for DGT-TM

Now, the corresponding translation times are presented in Table 60, again showing the gains achieved by Transtor in relation to Moses. On average, using the corpus DGT-TM, Transtor takes 26% of the time Moses takes to produce a translation.

File	Moses	Transtor	Transtor gain
eurlex1	01:32	00:23	25%
eurlex2	03:15	00:33	17%
eurlex3	01:34	00:35	37%
eurlex4	07:07	01:36	22%
eurlex5	00:39	00:20	51%
euconst1	07:57	01:36	20%
euconst2	15:53	03:15	20%
euconst3	06:18	01:34	25%
euconst4	05:60	00:53	15%
Average transtor gain			26%

Table 60. Translation times using DGT-TM

The tables above show that Transtor is capable of producing less training information in significantly less time, allowing the system to start producing

translations sooner while consuming less computational resources. This is due to the simplicity of the models (*plm* and *ptm*) and the efficiency of the structures supporting those models. This space and time efficiency is even more important when dealing with significant amounts of data (like APERTIUM EURLEX below). Such time efficiency is also accompanied by the Transtor translation process, which is also significantly faster than Moses. On top of that, and despite the reduction of space and processing time, the results for the translation quality (Section 4.3) confirm that Transtor is very competitive.

4.2.2 DATA ABOUT ADDITIONAL CORPORA

As mentioned before, additional corpora were used to further analyze Transtor. As such, APERTIUM EURLEX was used to see how the system deals with a significant amount of data, OPUS EMEA was used to see how the quality is affected when a corpus from a not so relevant domain is used, and OPUS EUROPARL was used to see how the system behaves when presented with a corpus having a vocabulary that is not so controlled. Table 61 shows the time taken and the resulting data size for the training stage for each additional corpus.

	APERTIUM EURLEX	OPUS EMEA	OPUS EUROPARL
Time (hh:mm:ss)	05:37:42	01:56:03	01:17:46
Size (GB)	50	7	30

Table 61. Training data for the additional corpora

After training the additional corpora, the reference documents were translated in the average times shown in Table 62.

File	APERTIUM EURLEX (mm:ss)	OPUS EMEA (mm:ss)	OPUS EUROPARL (mm:ss)
eurlex1	01:45	00:15	00:36
eurlex2	00:47	00:08	00:19
eurlex3	00:36	00:06	00:12
eurlex4	05:51	00:17	00:55
eurlex5	00:15	00:03	00:05
euconst1	02:35	00:21	00:57
euconst2	07:26	00:30	01:36
euconst3	01:13	00:12	00:33
euconst4	00:51	00:09	00:32

Table 62. Translation times using the additional corpora

Particularly when considering the times taken with DGT-TM, the translation times continue to be reasonable. The translation quality results are presented in Section 4.4.2 below.

4.3 RESULTS USING THE COMMON DATA

This section presents the results obtained with both Moses and Transtor after being trained with the same corpora sets, in order to translate the reference documents using the test settings described in Section 4.1 above. Besides the scores obtained by each system for each corpora set, a table is included to show the differences between those scores, in which positive values reveal an advantage in favor of Transtor while the negative values reveal an advantage in favor of Moses.

A small but specific corpus, OPUS EUCONST, is analyzed first in Sub-Section 4.3.1, and then, a larger and more generic corpus, DGT-TM, is analyzed in Sub-Section 4.3.2.

4.3.1 OPUS EUCONST

This sub-section presents the results obtained using OPUS EUCONST to train the models for each system. As mentioned before, this corpus set is relatively small but because it is very well-behaved, it still allows the production of interesting results. Table 63 shows the results obtained with Moses.

	eurlex1	eurlex2	eurlex3	eurlex4	eurlex5	euconst1	euconst2	euconst3	euconst4	avg
de-en	0.6489	0.2148	0.4549	0.2615	0.5024	0.4296	0.4418	0.4713	0.6001	0.4473
en-de	0.6146	0.1499	0.3547	0.1883	0.3682	0.4058	0.3477	0.4509	0.5575	0.3820
de-es	0.5041	0.1328	0.2618	0.2446	0.3394	0.3767	0.3450	0.4472	0.5963	0.3609
es-de	0.5470	0.1185	0.2862	0.1825	0.3165	0.3320	0.2789	0.3759	0.5875	0.3361
de-pt	0.4676	0.0768	0.2417	0.1191	0.2487	0.1917	0.1748	0.2296	0.4006	0.2390
pt-de	0.5082	0.0976	0.3025	0.1450	0.2752	0.2055	0.1932	0.2726	0.4657	0.2739
en-es	0.6027	0.1783	0.3071	0.3105	0.3116	0.4722	0.4455	0.5322	0.7383	0.4332
es-en	0.6501	0.2393	0.4351	0.3031	0.4503	0.5351	0.5037	0.5823	0.7522	0.4946
en-fr	0.3368	0.1709	0.3229	0.2071	0.3566	0.3870	0.3712	0.4178	0.5499	0.3467
fr-en	0.4183	0.2198	0.4116	0.2390	0.4744	0.4283	0.4542	0.4747	0.6052	0.4139
en-pt	0.4977	0.1237	0.2788	0.1708	0.2535	0.2910	0.2510	0.3217	0.4756	0.2960
pt-en	0.5409	0.1984	0.4061	0.2641	0.3891	0.4193	0.4000	0.4890	0.6155	0.4136
es-pt	0.5579	0.2295	0.4270	0.2543	0.4021	0.3663	0.3238	0.4178	0.5363	0.3906
pt-es	0.6404	0.3186	0.5751	0.4150	0.5882	0.4560	0.4703	0.5282	0.7305	0.5247
fr-pt	0.5505	0.2209	0.3478	0.2450	0.3981	0.3025	0.3186	0.3381	0.4885	0.3567
pt-fr	0.5764	0.2387	0.3898	0.2796	0.4982	0.3883	0.4147	0.4277	0.5391	0.4169

Table 63. Moses results using OPUS EUCONST

Table 64 shows the results obtained with Transtor. Again, in spite of being small, this corpus still allows Transtor to obtain interesting results, mostly for the reference documents from the European Constitution (with "euconst" prefix), a set of documents for which this corpus is more relevant.

	eurlex1	eurlex2	eurlex3	eurlex4	eurlex5	euconst1	euconst2	euconst3	euconst4	avg
de-en	0.6837	0.2626	0.4724	0.2919	0.4905	0.5724	0.4474	0.6324	0.5899	0.4937
en-de	0.6313	0.1921	0.4227	0.1927	0.4420	0.5458	0.3793	0.6099	0.5509	0.4407
de-es	0.6396	0.1888	0.4375	0.2877	0.4936	0.5210	0.3892	0.6163	0.6282	0.4669
es-de	0.6146	0.1487	0.3788	0.2090	0.4215	0.4450	0.2855	0.5832	0.5838	0.4078
de-pt	0.6558	0.2201	0.4691	0.3159	0.4947	0.4609	0.3802	0.5227	0.6824	0.4669
pt-de	0.5997	0.1604	0.4107	0.2144	0.4455	0.3851	0.2956	0.4766	0.6071	0.3995
en-es	0.6196	0.2518	0.4452	0.3353	0.5055	0.6757	0.5082	0.7542	0.8785	0.5527
es-en	0.6543	0.2505	0.4674	0.3362	0.4468	0.6922	0.5384	0.7707	0.8856	0.5602
en-fr	0.6760	0.2376	0.4353	0.2837	0.5380	0.6920	0.5411	0.7450	0.8770	0.5584
fr-en	0.6053	0.2536	0.4481	0.2605	0.4981	0.6724	0.5724	0.7350	0.8721	0.5464
en-pt	0.7108	0.2812	0.6066	0.3893	0.5172	0.6133	0.5328	0.7171	0.8608	0.5810
pt-en	0.6683	0.2773	0.5746	0.3410	0.5348	0.6404	0.5508	0.7412	0.8338	0.5736
es-pt	0.7216	0.3640	0.6617	0.5012	0.6459	0.6320	0.5726	0.7337	0.8849	0.6353
pt-es	0.7227	0.3751	0.6596	0.5027	0.6868	0.6478	0.5795	0.7389	0.9056	0.6465
fr-pt	0.7673	0.3323	0.5338	0.3860	0.5955	0.5856	0.5720	0.6708	0.8627	0.5896
pt-fr	0.6901	0.3110	0.4421	0.3773	0.5546	0.6173	0.5699	0.6916	0.8828	0.5707

Table 64. Transtor results using OPUS EUCONST

Table 65 shows the differences between the results obtained by Transtor and Moses. These results show that the gains with Transtor are considerable. The small size of OPUS EUCONST is compensated by its topic relevance, showing that the approach supporting Transtor is more capable of taking advantage of a corpus relevant to the translation in question, despite its small size.

	eurlex1	eurlex2	eurlex3	eurlex4	eurlex5	euconst1	euconst2	euconst3	euconst4	avg
de-en	0.0348	0.0478	0.0175	0.0304	-0.0119	0.1428	0.0056	0.1611	-0.0102	0.0464
en-de	0.0167	0.0422	0.0680	0.0044	0.0738	0.1400	0.0316	0.1590	-0.0066	0.0588
de-es	0.1355	0.0560	0.1757	0.0431	0.1542	0.1443	0.0442	0.1691	0.0319	0.1060
es-de	0.0676	0.0302	0.0926	0.0265	0.1050	0.1130	0.0066	0.2073	-0.0037	0.0717
de-pt	0.1882	0.1433	0.2274	0.1968	0.2460	0.2692	0.2054	0.2931	0.2818	0.2279
pt-de	0.0915	0.0628	0.1082	0.0694	0.1703	0.1796	0.1024	0.2040	0.1414	0.1255
en-es	0.0169	0.0735	0.1381	0.0248	0.1939	0.2035	0.0627	0.2220	0.1402	0.1195
es-en	0.0042	0.0112	0.0323	0.0331	-0.0035	0.1571	0.0347	0.1884	0.1334	0.0657
en-fr	0.3392	0.0667	0.1124	0.0766	0.1814	0.3050	0.1699	0.3272	0.3271	0.2117
fr-en	0.1870	0.0338	0.0365	0.0215	0.0237	0.2441	0.1182	0.2603	0.2669	0.1324
en-pt	0.2131	0.1575	0.3278	0.2185	0.2637	0.3223	0.2818	0.3954	0.3852	0.2850
pt-en	0.1274	0.0789	0.1685	0.0769	0.1457	0.2211	0.1508	0.2522	0.2183	0.1600
es-pt	0.1637	0.1345	0.2347	0.2469	0.2438	0.2657	0.2488	0.3159	0.3486	0.2447
pt-es	0.0823	0.0565	0.0845	0.0877	0.0986	0.1918	0.1092	0.2107	0.1751	0.1218
fr-pt	0.2168	0.1114	0.1860	0.1410	0.1974	0.2831	0.2534	0.3327	0.3742	0.2329
pt-fr	0.1137	0.0723	0.0523	0.0977	0.0564	0.2290	0.1552	0.2639	0.3437	0.1538

Table 65. Differences for OPUS EUCONST

From the table above it can be seen that Transtor is capable of producing consistently better results, only losing in five different cases, but only for a small margin, and yet it always wins on average. When using a larger corpus, Transtor is still capable of producing interesting results, as shown in the following sub-section.

4.3.2 DGT-TM

This sub-section presents the results obtained using DGT-TM, which is significantly larger than OPUS EUCONST, and also about a broader set of topics. Table 66 shows the results obtained with Moses.

	eurlex1	eurlex2	eurlex3	eurlex4	eurlex5	euconst1	euconst2	euconst3	euconst4	avg
de-en	0.6546	0.3767	0.7113	0.4230	0.8009	0.3183	0.4522	0.3518	0.3964	0.4984
en-de	0.6489	0.2946	0.6318	0.3259	0.6507	0.2157	0.3392	0.2429	0.3259	0.4084
de-es	0.6608	0.3014	0.6159	0.3723	0.6957	0.2532	0.3694	0.2746	0.3875	0.4368
es-de	0.5398	0.2488	0.6231	0.3090	0.6587	0.1986	0.3044	0.2311	0.3121	0.3806
de-pt	0.5541	0.2219	0.5016	0.2397	0.4837	0.1575	0.2090	0.1584	0.2702	0.3107
pt-de	0.5605	0.2328	0.6173	0.2759	0.5769	0.1581	0.2711	0.1707	0.2911	0.3505
en-es	0.6336	0.3678	0.6176	0.4788	0.6834	0.3334	0.4600	0.3466	0.4672	0.4876
es-en	0.5593	0.3986	0.6988	0.4780	0.7396	0.3961	0.5263	0.4187	0.4850	0.5223
en-fr	0.6658	0.3239	0.5493	0.3697	0.5542	0.3140	0.4052	0.3151	0.3864	0.4315
fr-en	0.5682	0.3708	0.7357	0.4353	0.7794	0.3941	0.5035	0.4257	0.4439	0.5174
en-pt	0.4837	0.2790	0.5268	0.3103	0.4826	0.2292	0.2821	0.2340	0.3096	0.3486
pt-en	0.5718	0.3566	0.7290	0.4354	0.7108	0.3723	0.4597	0.3916	0.4405	0.4964
es-pt	0.6349	0.2981	0.5723	0.3188	0.5664	0.2832	0.3109	0.2768	0.3586	0.4022
pt-es	0.6964	0.4072	0.6790	0.4851	0.7132	0.3877	0.5073	0.4107	0.5258	0.5347
fr-pt	0.6097	0.3227	0.5422	0.3353	0.5513	0.2797	0.3228	0.2725	0.3651	0.4001
pt-fr	0.7038	0.3593	0.6198	0.4212	0.6273	0.3505	0.4335	0.3437	0.4236	0.4759

Table 66. Moses results using DGT-TM

Table 67 shows the results obtained with Transtor. This corpus allows a significant improvement on the translation of the reference documents from the Official Journal of the European Union (with "eurlex" prefix), but its broader set of topics results in a significant deterioration on the translation of the reference documents from the European Constitution (with "euconst prefix"), emphasizing the importance of topic relevance in achieving higher scores.

	eurlex1	eurlex2	eurlex3	eurlex4	eurlex5	euconst1	euconst2	euconst3	euconst4	avg
de-en	0.7485	0.3136	0.5932	0.3789	0.5712	0.3203	0.4178	0.3515	0.3321	0.4475
en-de	0.7740	0.2322	0.5793	0.2713	0.5780	0.2206	0.3212	0.2366	0.2154	0.3810
de-es	0.6883	0.2514	0.5821	0.3482	0.5857	0.3005	0.3522	0.3469	0.2895	0.4161
es-de	0.7547	0.2079	0.5757	0.2621	0.5372	0.2111	0.2502	0.2370	0.2147	0.3612
de-pt	0.7477	0.2634	0.6142	0.3766	0.5904	0.3198	0.3736	0.3333	0.3363	0.4395
pt-de	0.6874	0.2027	0.5837	0.2655	0.5656	0.2149	0.2717	0.2171	0.2395	0.3609
en-es	0.7526	0.3661	0.6525	0.4420	0.6333	0.3903	0.4492	0.4230	0.4271	0.5040
es-en	0.7869	0.3548	0.6502	0.4520	0.6390	0.4146	0.4893	0.4563	0.4278	0.5190
en-fr	0.8152	0.3172	0.5755	0.3406	0.5751	0.4662	0.4804	0.4616	0.4237	0.4951
fr-en	0.7947	0.3125	0.6824	0.3964	0.6087	0.4694	0.4910	0.4903	0.4312	0.5196
en-pt	0.7629	0.3610	0.6653	0.4752	0.5969	0.5365	0.5229	0.5998	0.7213	0.5824
pt-en	0.7341	0.3770	0.6918	0.4644	0.6604	0.5343	0.5609	0.6133	0.6754	0.5902
es-pt	0.7319	0.4292	0.7205	0.5237	0.6402	0.4955	0.5567	0.5167	0.5479	0.5736
pt-es	0.7291	0.4393	0.7670	0.5361	0.7231	0.4875	0.5642	0.5101	0.5635	0.5911
fr-pt	0.7806	0.3850	0.6394	0.4523	0.6238	0.4398	0.5020	0.4521	0.4852	0.5289
pt-fr	0.7299	0.3694	0.5588	0.4262	0.5719	0.4489	0.5268	0.4642	0.4889	0.5094

Table 67. Transtor results using DGT-TM

Both tools achieve better results with this significantly larger corpus, showing that the amount of data also plays an important role to improve translation quality. Table 68 shows the differences between Transtor and Moses for the DGT-TM corpus. Unlike with OPUS EUCONST, results between both tools are more balanced. Still, Transtor keeps a general advantage over Moses, only losing on average on both directions of the de-en and de-es language pairs. Additionally, even though Transtor loses in one direction of the en-es language pair by a small difference, the average on both directions of such pair is still favorable to Transtor.

	eurlex1	eurlex2	eurlex3	eurlex4	eurlex5	euconst1	euconst2	euconst3	euconst4	avg
de-en	0.0939	-0.0631	-0.1181	-0.0441	-0.2297	0.0020	-0.0344	-0.0003	-0.0643	-0.0509
en-de	0.1251	-0.0624	-0.0525	-0.0546	-0.0727	0.0049	-0.0180	-0.0063	-0.1105	-0.0274
de-es	0.0275	-0.0500	-0.0338	-0.0241	-0.1100	0.0473	-0.0172	0.0723	-0.0980	-0.0207
es-de	0.2149	-0.0409	-0.0474	-0.0469	-0.1215	0.0125	-0.0542	0.0059	-0.0974	-0.0194
de-pt	0.1936	0.0415	0.1126	0.1369	0.1067	0.1623	0.1646	0.1749	0.0661	0.1288
pt-de	0.1269	-0.0301	-0.0336	-0.0104	-0.0113	0.0568	0.0006	0.0464	-0.0516	0.0104
en-es	0.1190	-0.0017	0.0349	-0.0368	-0.0501	0.0569	-0.0108	0.0764	-0.0401	0.0164
es-en	0.2276	-0.0438	-0.0486	-0.0260	-0.1006	0.0185	-0.0370	0.0376	-0.0572	-0.0033
en-fr	0.1494	-0.0067	0.0262	-0.0291	0.0209	0.1522	0.0752	0.1465	0.0373	0.0635
fr-en	0.2265	-0.0583	-0.0533	-0.0389	-0.1707	0.0753	-0.0125	0.0646	-0.0127	0.0022
en-pt	0.2792	0.0820	0.1385	0.1649	0.1143	0.3073	0.2408	0.3658	0.4117	0.2338
pt-en	0.1623	0.0204	-0.0372	0.0290	-0.0504	0.1620	0.1012	0.2217	0.2349	0.0938
es-pt	0.0970	0.1311	0.1482	0.2049	0.0738	0.2123	0.2458	0.2399	0.1893	0.1714
pt-es	0.0327	0.0321	0.0880	0.0510	0.0099	0.0998	0.0569	0.0994	0.0377	0.0564
fr-pt	0.1709	0.0623	0.0972	0.1170	0.0725	0.1601	0.1792	0.1796	0.1201	0.1288
pt-fr	0.0261	0.0101	-0.0610	0.0050	-0.0554	0.0984	0.0933	0.1205	0.0653	0.0336

Table 68. Differences for DGT-TM

These results, together with the results from the previous sub-section, confirm that the size, the controlled vocabulary, and the relevance of the corpus in relation to the documents being translated are very important factors in translation quality.

4.4 TRANSTOR ADDITIONAL RESULTS AND ANALYSIS

This section is dedicated to the further analysis of Transtor. Sub-Section 4.4.1 is dedicated to the check the impact of removing the translation patterns and then also removing the verified lexica from the translation process, an analysis carried out for the en-pt language pair precisely because it is the most developed one. Then, Sub-Section 4.4.2 will check how Transtor deals with other corpora having different features. Lastly, Sub-Section 4.4.3 will consider the average behavior of the corpora in order to analyze which corpora features are the most important for translation.

4.4.1 IMPACT OF USING LEXICA AND TRANSLATION PATTERNS

The results shown so far have included all the sources available, namely the verified lexica and the translation patterns. In order to check for the effects of using those sources, as opposed to just using an aligned parallel corpus, the following results are shown in the tables below. To avoid the overwhelming amount of data, results are presented as averages. Additionally, the language pair that will be analyzed with more detail is en-pt, given that this is the one that has the largest number of translation patterns and the largest lexicon. The

losses verified in relation to using all sources are also included, where negative values represent an effective loss while positive values will actually represent a gain over the use of all sources.

	Average scores		Loss	
	en-pt	**pt-en**	**en-pt**	**pt-en**
eurlex1	0.7119	0.6673	0.0011	-0.0010
eurlex2	0.2840	0.2786	0.0028	0.0013
eurlex3	0.6095	0.5694	0.0029	-0.0052
eurlex4	0.3883	0.3413	-0.0010	0.0003
eurlex5	0.5172	0.5392	0.0000	0.0044
euconst1	0.6385	0.6505	0.0252	0.0101
euconst2	0.5397	0.5508	0.0069	0.0000
euconst3	0.7513	0.7535	0.0342	0.0123
euconst4	0.8673	0.8323	0.0065	-0.0015
Average loss			**0.0087**	**0.0023**

Table 69. OPUS EUCONST without patterns

First, Table 69 shows the BLEU scores obtained using OPUS EUCONST, for each reference document and for each language direction, without using the translation patterns. From the table, it can be seen that there are only four cases in which the exclusion of the translation patterns provides worse results (eurlex4 for en-pt; and eurlex1, eurlex3 and euconst4 for pt-en) and there are two cases where their exclusion will have no impact (eurlex5 for en-pt; and euconst2 for pt-en). However, on average, excluding the translation patterns will actually have a positive impact on results, but the gain is lower than 0.01 in both language directions. This could be because the translation patterns do not properly cover the reference documents, but also because using the translation patterns might require a different set of parameters. Nevertheless, the importance of the inclusion of the translation patterns has been shown in Sub-Section 3.1.1.2.4.

	Average scores		Loss	
	en-pt	**pt-en**	**en-pt**	**pt-en**
eurlex1	0.6462	0.5496	-0.0646	-0.1187
eurlex2	0.2308	0.2157	-0.0504	-0.0616
eurlex3	0.5140	0.4650	-0.0926	-0.1096
eurlex4	0.3315	0.2857	-0.0578	-0.0553
eurlex5	0.4971	0.4604	-0.0201	-0.0744
euconst1	0.6071	0.5819	-0.0062	-0.0585
euconst2	0.5029	0.5100	-0.0299	-0.0408
euconst3	0.7220	0.7223	0.0049	-0.0189
euconst4	0.8547	0.8001	-0.0061	-0.0337
Average loss			**-0.0359**	**-0.0635**

Table 70. OPUS EUCONST without both patterns and lexicon

Still for OPUS EUCONST, this time Table 70 presents the results obtained when neither the translation patters nor the lexicon are included in the translation process. This time every reference document, in both language directions, present an effective loss of score, confirming the importance of using a lexicon.

	Average scores		Loss	
	en-pt	**pt-en**	**en-pt**	**pt-en**
eurlex1	0.7638	0.7332	0.0009	-0.0009
eurlex2	0.3604	0.3744	-0.0006	-0.0026
eurlex3	0.6666	0.6912	0.0013	-0.0006
eurlex4	0.4768	0.4616	0.0016	-0.0028
eurlex5	0.6067	0.6486	0.0098	-0.0118
euconst1	0.5377	0.5386	0.0012	0.0043
euconst2	0.5274	0.5564	0.0045	-0.0045
euconst3	0.6002	0.6168	0.0004	0.0035
euconst4	0.7291	0.6991	0.0078	0.0237
Average loss			**0.0030**	**0.0009**

Table 71. DGT-TM without patterns

When analyzing the corpus DGT-TM, Table 71 shows that not including the translation patterns results in a score loss in seven different cases (eurlex2 for en-pt; and all eurlex documents and euconst2 in pt-en). In this case, as with OPUS EUCONST, not considering the translation patterns improves results but by an even smaller margin than with OPUS EUCONST.

	Average scores		Loss	
	en-pt	pt-en	en-pt	pt-en
eurlex1	0.7630	0.7198	0.0001	-0.0143
eurlex2	0.3578	0.3583	-0.0032	-0.0187
eurlex3	0.6662	0.6908	0.0009	-0.0010
eurlex4	0.4696	0.4504	-0.0056	-0.0140
eurlex5	0.6062	0.6486	0.0093	-0.0118
euconst1	0.4380	0.4392	-0.0985	-0.0951
euconst2	0.4915	0.5214	-0.0314	-0.0395
euconst3	0.4697	0.4729	-0.1301	-0.1404
euconst4	0.4867	0.5022	-0.2346	-0.1732
Average loss			**-0.0548**	**-0.0564**

Table 72. DGT-TM without both patterns and lexicon

Finally, Table 72 shows the results obtained when neither the translation patterns nor the lexicon are included with DGT-TM. There are three exceptions (eurlex1, eurlex3, and eurlex5 for en-pt), again for a very low margin, but on average the scores are lower by more than 0.05 for each language direction, confirming once again the importance of using a verified lexicon.

In the end, even though the importance of translation patterns has not been indisputably confirmed by the results above, this might be due to their low coverage for the reference documents, and even by the need of using a different set of translation parameters for Transtor. In other words, in spite of the results, translation patterns (Sub-Section 3.1.1.2) are necessary to deal with translation situations that could not be tackled by simple contiguous phrase translations (Sub-Section 3.1.1.1).

4.4.2 ADDITIONAL CORPORA

With the purpose of further analyzing Transtor, this section presents results for additional corpora sets, analyzed individually in the following sub-sections. Again, these corpora were not processed with Moses because it takes a considerably greater amount of time, with the particular case of not being able to process the corpus APERTIUM EURLEX, which was the most interesting one because of its size and broad range of topics.

4.4.2.1 APERTIUM EURLEX

APERTIUM EURLEX is the largest corpus and, since it covers a wide range of topics while using a controlled vocabulary, both its size and its topic coverage contribute to a generally significant improvement on the quality of every reference document over the quality achieved either with OPUS EUCONST or with DGT-TM. Table 73 show the results obtained using APERTIUM EURLEX.

	eurlex1	eurlex2	eurlex3	eurlex4	eurlex5	euconst1	euconst2	euconst3	euconst4	avg
de-en	0.7513	0.3401	0.5296	0.3722	0.5143	0.7226	0.7591	0.7951	0.7935	0.6198
en-de	0.7370	0.2016	0.4941	0.2479	0.4643	0.6033	0.6952	0.6787	0.7294	0.5391
de-es	0.6516	0.2312	0.5332	0.3538	0.5102	0.6537	0.7168	0.7928	0.8116	0.5839
es-de	0.6490	0.1640	0.4815	0.2480	0.4336	0.5287	0.6539	0.6865	0.7140	0.5066
de-pt	0.6781	0.2637	0.5747	0.3690	0.5330	0.6263	0.7088	0.7657	0.8357	0.5950
pt-de	0.6351	0.1829	0.5042	0.2681	0.4647	0.4798	0.5944	0.6285	0.7270	0.4983
en-es	0.7528	0.3533	0.6330	0.4499	0.6658	0.6881	0.7647	0.7943	0.8353	0.6597
es-en	0.7541	0.3509	0.6312	0.4593	0.5852	0.7166	0.7996	0.7949	0.8237	0.6573
en-fr	0.7738	0.2847	0.4719	0.3449	0.5020	0.7328	0.7216	0.7912	0.8365	0.6066
fr-en	0.8002	0.3182	0.6505	0.3962	0.5594	0.7248	0.7219	0.7731	0.7672	0.6346
en-pt	0.7657	0.3833	0.7091	0.4957	0.6365	0.6788	0.7570	0.8062	0.8449	0.6752
pt-en	0.7464	0.3782	0.7560	0.4963	0.7042	0.6724	0.7778	0.8026	0.8202	0.6838
es-pt	0.7104	0.4154	0.7255	0.5337	0.6797	0.6719	0.7583	0.8017	0.8332	0.6811
pt-es	0.6742	0.4362	0.6733	0.5343	0.6876	0.6575	0.7606	0.8014	0.8420	0.6741
fr-pt	0.7387	0.3707	0.6037	0.4460	0.6534	0.6461	0.7008	0.7418	0.7649	0.6296
pt-fr	0.7268	0.3541	0.5312	0.4274	0.5957	0.6586	0.7143	0.7402	0.8334	0.6202

Table 73. Transtor results using APERTIUM EURLEX

In general, results improve when compared to either OPUS EUCONST or DGT-TM, as will be seen in Sub-Section 4.4.3.

4.4.2.2 OPUS EMEA

OPUS EMEA is also a corpus with a significant size, but being about a very specific and different topic keeps the system from getting better results, in particular for the reference documents from the European Constitution (with "euconst" prefix). Table 74 shows the results obtained using OPUS EMEA, confirming that its very distinct topic from the reference documents prevent the system from obtaining competitive results even with the smallest corpus used (OPUS EUCONST).

	eurlex1	eurlex2	eurlex3	eurlex4	eurlex5	euconst1	euconst2	euconst3	euconst4	avg
de-en	0.6032	0.2328	0.4543	0.2494	0.4351	0.1624	0.2448	0.1584	0.1774	0.3020
en-de	0.4455	0.1424	0.3312	0.1325	0.2605	0.0856	0.1486	0.0977	0.1127	0.1952
de-es	0.6052	0.1459	0.3343	0.1890	0.3243	0.1420	0.1976	0.1750	0.1426	0.2507
es-de	0.6225	0.1084	0.3111	0.1250	0.3050	0.0822	0.1274	0.0896	0.0901	0.2068
de-pt	0.6242	0.1963	0.4320	0.2555	0.3970	0.1721	0.2555	0.1901	0.2377	0.3067
pt-de	0.3272	0.1452	0.3574	0.1715	0.3597	0.0896	0.1502	0.0888	0.1374	0.2030
en-es	0.4218	0.2228	0.3799	0.2535	0.4077	0.2354	0.2764	0.2407	0.2374	0.2973
es-en	0.5435	0.2352	0.4551	0.2572	0.5167	0.2253	0.2622	0.2410	0.2203	0.3285
en-fr	0.4813	0.2178	0.4038	0.2030	0.3893	0.1742	0.2626	0.1862	0.1868	0.2783
fr-en	0.5549	0.2284	0.4401	0.2226	0.4824	0.1926	0.2801	0.2070	0.2122	0.3134
en-pt	0.4554	0.2588	0.4948	0.3619	0.4261	0.5111	0.4723	0.5850	0.7606	0.4807
pt-en	0.3566	0.2629	0.5025	0.3347	0.4609	0.4625	0.4564	0.5523	0.6763	0.4517
es-pt	0.7001	0.3405	0.6328	0.4425	0.6323	0.3956	0.4553	0.4088	0.4504	0.4954
pt-es	0.5068	0.3519	0.6194	0.4526	0.6280	0.3900	0.4482	0.4020	0.4528	0.4724
fr-pt	0.7111	0.3184	0.5683	0.4082	0.6166	0.2901	0.4020	0.2939	0.3727	0.4424
pt-fr	0.3660	0.2780	0.4457	0.3286	0.5257	0.2771	0.3718	0.2755	0.3311	0.3555

Table 74. Transtor results using OPUS EMEA

Unlike APERTIUM EURLEX, results do not generally improve, as will be confirmed in Sub-Section 4.4.3 ahead.

4.4.2.3 OPUS EUROPARL

OPUS EUROPARL is another significantly large corpus, but since these are transcripts from the European Parliament, its language and vocabulary are not very controlled, keeping it from having a greater contribution on translation quality. Table 75 shows the results obtained using OPUS EUROPARL.

	eurlex1	eurlex2	eurlex3	eurlex4	eurlex5	euconst1	euconst2	euconst3	euconst4	avg
de-en	0.6609	0.2730	0.4084	0.2929	0.4120	0.1938	0.3494	0.2023	0.3572	0.3500
en-de	0.5959	0.1897	0.3748	0.1827	0.3447	0.1190	0.2715	0.1290	0.2650	0.2747
de-es	0.6024	0.1949	0.3837	0.2736	0.4372	0.1734	0.3125	0.2044	0.3336	0.3240
es-de	0.6342	0.1525	0.3226	0.1818	0.4050	0.0962	0.2174	0.1005	0.2241	0.2594
de-pt	0.5783	0.2164	0.3899	0.3087	0.4745	0.2037	0.3268	0.2176	0.3662	0.3425
pt-de	0.5866	0.1623	0.3489	0.2018	0.3641	0.1221	0.2402	0.1100	0.2471	0.2648
en-es	0.6317	0.2853	0.4481	0.3468	0.4091	0.2845	0.4186	0.2897	0.3957	0.3899
es-en	0.6501	0.2780	0.4651	0.3427	0.4793	0.2924	0.4346	0.3147	0.4324	0.4099
en-fr	0.6561	0.2248	0.3569	0.2876	0.4134	0.2476	0.4145	0.2531	0.4335	0.3653
fr-en	0.6828	0.2564	0.4298	0.3070	0.4301	0.2320	0.4149	0.2671	0.4474	0.3853
en-pt	0.6743	0.3190	0.5803	0.4281	0.5263	0.5021	0.4952	0.5682	0.7289	0.5358
pt-en	0.6352	0.2936	0.5512	0.3797	0.5231	0.4760	0.5062	0.5462	0.6606	0.5080
es-pt	0.6777	0.3659	0.6420	0.5048	0.6200	0.4088	0.5208	0.4223	0.5735	0.5262
pt-es	0.6840	0.3810	0.6032	0.4900	0.5711	0.3968	0.5182	0.4094	0.5611	0.5128
fr-pt	0.7380	0.3433	0.5236	0.4138	0.5374	0.3394	0.4810	0.3530	0.4998	0.4699
pt-fr	0.6816	0.3240	0.4512	0.3761	0.5617	0.3420	0.4785	0.3451	0.4867	0.4497

Table 75. Transtor results using OPUS EUROPARL

Again, unlike APERTIUM EURLEX, results do not generally improve, as will be seen in Sub-Section 4.4.3, below.

4.4.3 CORPORA INFLUENCE ON TRANSTOR RESULTS

Having produced different sets of results according to different corpora, it would be interesting to see how each affects translation. Considering the different features of each corpus, such analysis will allow inferring which features benefit translation and which ones will degrade it. Table 76 presents the gains, for the average scores of the reference documents by language pair, of corpus1 over corpus2, with positive values representing a gain for corpus1 while a gain for corpus2 is represented by negative values.

	OPUS EUCONST over DGT-TM	APERTIUM EURLEX over OPUS EUCONST	APERTIUM EURLEX over DGT-TM	OPUS EMEA over DGT-TM	OPUS EUROPARL over DGT-TM
de-en	0.0462	0.1261	0.1723	-0.1455	-0.0975
en-de	0.0598	0.0983	0.1581	-0.1858	-0.1063
de-es	0.0508	0.1170	0.1678	-0.1654	-0.0921
es-de	0.0466	0.0988	0.1454	-0.1544	-0.1018
de-pt	0.0274	0.1281	0.1555	-0.1328	-0.0970
pt-de	0.0386	0.0988	0.1374	-0.1579	-0.0961
en-es	0.0487	0.1070	0.1557	-0.2067	-0.1141
es-en	0.0412	0.0970	0.1383	-0.1905	-0.1091
en-fr	0.0634	0.0482	0.1115	-0.2167	-0.1298
fr-en	0.0268	0.0882	0.1150	-0.2063	-0.1343
en-pt	-0.0014	0.0942	0.0928	-0.1018	-0.0466
pt-en	-0.0166	0.1102	0.0936	-0.1385	-0.0822
es-pt	0.0617	0.0458	0.1075	-0.0782	-0.0474
pt-es	0.0554	0.0276	0.0830	-0.1187	-0.0783
fr-pt	0.0606	0.0400	0.1007	-0.0865	-0.0590
pt-fr	0.0613	0.0494	0.1107	-0.1539	-0.0598
average	0.0419	0.0859	0.1278	-0.1525	-0.0907

Table 76. Results comparison by corpora

The table above shows that, on average, OPUS EUCONST wins over DGT-TM. Such relation means that a corpus with a controlled language and from a topic relevant to the documents being translated are two properties that are more important than size. The exception is for both directions of the en-pt language pair, something that could be explained by the significantly larger lexicon for that language pair. The systematic gain of APERTIUM EURLEX over OPUS EUCONST and over DGT-TM shows that when size is added to controlled language and topic relevance, the results always improve. On the contrary, when analyzing OPUS EMEA and OPUS EUROPARL, both lose to DGT-TM, with OPUS EMEA losing more severely because of the lack of topic relevance, while OPUS EUROPARL, consisting of transcripts of the European Parliament, loses because of its non-controlled language.

4.5 TRANSTOR PARAMETER ANALYSIS

With the purpose of determining if there is any room for improvement, the parameters *lw* (and, indirectly *tw*) and α were changed to see how they affect translation quality. In order to avoid an overwhelming amount of data, only the language pairs for which, compared to Moses, the results are worst (de-en) and best (en-pt).

The tables below will present the current values obtained with the parameters *lw*=0.5 and α=3, comparing them with the maximum values. The potential gain and the corresponding parameters (*lw*;α) are also included in the tables.

	Current value		Maximum value		Potential gain		Parameters	
	de-en	en-de	de-en	en-de	de-en	en-de	de-en	en-de
eurlex1	0.6837	0.6313	0.6885	0.6313	0.0048	0.0000	0.3;7	0.5;3
eurlex2	0.2626	0.1921	0.2644	0.1941	0.0018	0.0020	0.5;2	0.5;4
eurlex3	0.4724	0.4227	0.4759	0.4240	0.0035	0.0013	0.3;7	0.5;2
eurlex4	0.2919	0.1927	0.2919	0.1950	0.0000	0.0023	0.5;3	0.6;3
eurlex5	0.4905	0.4420	0.5035	0.4552	0.0130	0.0132	0.5;2	0.3;6
euconst1	0.5724	0.5458	0.5947	0.5629	0.0223	0.0171	0.9;6	0.9;5
euconst2	0.4474	0.3793	0.4505	0.3801	0.0031	0.0008	0.7;3	0.6;4
euconst3	0.6324	0.6099	0.6562	0.6374	0.0238	0.0275	0.9;7	0.8;3
euconst4	0.5899	0.5509	0.6032	0.5862	0.0133	0.0353	0.6;7	0.9;5

Table 77. Analysis with OPUS EUCONST for de-en

Table 77 shows that the best score using OPUS EUCONST was only achieved for a single reference document, which is the en-de language direction of eurlex1. The highest improvements, on average, are achieved for the euconst1, euconst3 and euconst4 reference documents.

	Current value		Maximum value		Potential gain		Parameters	
	de-en	en-de	de-en	en-de	de-en	en-de	de-en	en-de
eurlex1	0.7485	0.7740	0.8005	0.7937	0.0520	0.0197	0.3;3	0.3;3
eurlex2	0.3136	0.2322	0.3386	0.2441	0.0250	0.0119	0.4;7	0.5;5
eurlex3	0.5932	0.5793	0.7040	0.6148	0.1108	0.0355	0.4;7	0.5;5
eurlex4	0.3789	0.2713	0.4071	0.2928	0.0282	0.0215	0.8;6	1.0;6
eurlex5	0.5712	0.5780	0.7069	0.6559	0.1357	0.0779	0.3;7	1.0;3
euconst1	0.3203	0.2206	0.3284	0.2500	0.0081	0.0294	0.6;7	0.8;6
euconst2	0.4178	0.3212	0.4216	0.3251	0.0038	0.0039	0.5;7	0.8;7
euconst3	0.3515	0.2366	0.3668	0.2722	0.0153	0.0356	0.6;7	0.8;6
euconst4	0.3321	0.2154	0.3377	0.2256	0.0056	0.0102	0.6;3	1.0;7

Table 78. Analysis with DGT-TM for de-en

Table 78 shows that both language directions of every reference document can improve when using corpus DGT-TM. The most significant improvement is achieved for the eurlex3 and eurlex5 reference documents, in particular for the de-en language direction. Generally, the best values are achieved with a higher phrase penalty.

	Current value		Maximum value		Potential gain		Parameters	
	en-pt	pt-en	en-pt	pt-en	en-pt	pt-en	en-pt	pt-en
eurlex1	0.7108	0.6683	0.7132	0.6979	0.0024	0.0296	0.4;4	0.2;2
eurlex2	0.2812	0.2773	0.2957	0.2868	0.0145	0.0095	0.2;3	0.4;2
eurlex3	0.6066	0.5746	0.6173	0.5746	0.0107	0.0000	0.3;3	0.5;3
eurlex4	0.3893	0.341	0.3926	0.3423	0.0033	0.0013	0.4;4	0.3;2
eurlex5	0.5172	0.5348	0.5214	0.5433	0.0042	0.0085	0.4;4	0.1;2
euconst1	0.6133	0.6404	0.6184	0.6666	0.0051	0.0262	0.8;6	0.8;7
euconst2	0.5328	0.5508	0.5343	0.5627	0.0015	0.0119	0.5;2	0.8;6
euconst3	0.7171	0.7412	0.7243	0.7684	0.0072	0.0272	0.8;6	0.9;5
euconst4	0.8608	0.8338	0.8704	0.8509	0.0096	0.0171	0.9;6	0.9;7

Table 79. Analysis with OPUS EUCONST for en-pt

As for the en-pt language pair, Table 79 shows that the best value using OPUS EUCONST was only obtained in a single situation (eurlex3, pt-en), but on average it is still possible to improve results when using other parameters, in particular for the pt-en language direction of the euconst3 reference document, which can still improve a value of 0.7412, a value that can already be considered good, to 0.7684, gaining 0.0272 BLEU points. The results also show that the eurlex reference documents benefit from a lower *lw* (higher *tw*) and a lower phrase penalty, unlike the euconst reference documents which, on average, benefit from a higher *lw* (lower *tw*) and a higher phrase penalty, for which the en-pt language direction of euconst2 is an exception.

	Current value		Maximum value		Potential gain		Parameters	
	en-pt	pt-en	en-pt	pt-en	en-pt	pt-en	en-pt	pt-en
eurlex1	0.7629	0.7341	0.8017	0.7751	0.0388	0.0410	0.2;3	0.4;5
eurlex2	0.3610	0.3770	0.3909	0.3989	0.0299	0.0219	0.1;4	0.4;5
eurlex3	0.6653	0.6918	0.7331	0.7778	0.0678	0.0860	0.3;6	0.4;5
eurlex4	0.4752	0.4644	0.4935	0.4819	0.0183	0.0175	0.3;5	0.4;7
eurlex5	0.5969	0.6604	0.7045	0.7155	0.1076	0.0551	0.9;7	0.3;6
euconst1	0.5365	0.5343	0.5398	0.5418	0.0033	0.0075	0.3;7	0.4;4
euconst2	0.5229	0.5609	0.5304	0.5609	0.0075	0.0000	0.3;7	0.5;3
euconst3	0.5998	0.6133	0.6103	0.6136	0.0105	0.0003	0.3;4	0.3;3
euconst4	0.7213	0.6754	0.7386	0.6780	0.0173	0.0026	0.3;2	0.3;3

Table 80. Analysis with DGT-TM for en-pt

Similarly to the previous example, Table 80 shows that the best value using DGT-TM is only achieved in a single situation, this time for the pt-en language direction of euconst2. In this case, the most significant gains are achieved for the eurlex5 reference document, with most documents generally benefiting from a lower *lw* (higher *tw*) and a higher phrase penalty.

To provide a more detailed picture of how the parameter change affects the translation quality, below are presented some tables which show the BLEU scores obtained on average when using APERTIUM EURLEX when translating the eurlex and the euconst reference documents, on both language directions of the en-pt language pair. The eurlex and euconst documents are shown separately since their quality behavior also differs. The tables show higher values highlighted with darker green, lower values highlighted with darker red, and intermediate values highlighted with yellow.

The first two tables below show the score change for the eurlex reference documents, one table for each language pair direction.

		phrase penalty						
		1	2	3	4	5	6	7
tw	0.0	0.5324	0.5882	0.5966	0.6054	0.6058	0.6058	0.6062
	0.1	0.5335	0.5916	0.6018	0.6098	0.6090	0.6099	0.6099
	0.2	0.5316	0.5976	0.6010	0.6090	0.6083	0.6085	0.6084
	0.3	0.5310	0.5958	0.6036	0.6099	0.6106	0.6102	0.6097
	0.4	0.5175	0.5968	0.6005	0.6103	0.6103	0.6101	0.6097
	0.5	0.4953	0.5932	0.5981	0.6116	0.6128	0.6140	0.6133
	0.6	0.4209	0.5233	0.5557	0.5627	0.5639	0.5650	0.5651
	0.7	0.3207	0.4753	0.5236	0.5464	0.5545	0.5549	0.5544
	0.8	0.2341	0.4548	0.4905	0.5120	0.5260	0.5467	0.5535
	0.9	0.1846	0.4259	0.4698	0.4926	0.5002	0.5088	0.5146
	1.0	0.1739	0.3622	0.4090	0.4569	0.4670	0.4793	0.4887

Table 81. Eurlex reference documents for en-pt language direction

Table 81 shows that the higher scores for the en-pt language direction are obtained for lower *lw* values (higher *tw* values) and higher phrase penalty values.

		phrase penalty						
		1	2	3	4	5	6	7
tw	0.0	0.4552	0.5913	0.6044	0.6096	0.6118	0.6116	0.6104
	0.1	0.4595	0.5964	0.6143	0.6208	0.6225	0.6222	0.6210
	0.2	0.4554	0.5987	0.6161	0.6205	0.6230	0.6230	0.6242
	0.3	0.4612	0.6075	0.6207	0.6243	0.6272	0.6270	0.6283
	0.4	0.4613	0.6189	0.6229	0.6281	0.6294	0.6297	0.6305
	0.5	0.4576	0.6094	0.6162	0.6224	0.6233	0.6238	0.6245
	0.6	0.3805	0.5642	0.5679	0.5755	0.5792	0.5805	0.5801
	0.7	0.3306	0.5459	0.5579	0.5631	0.5722	0.5739	0.5747
	0.8	0.2816	0.5216	0.5514	0.5602	0.5641	0.5658	0.5683
	0.9	0.2600	0.4795	0.5152	0.5236	0.5307	0.5206	0.5412
	1.0	0.2389	0.4379	0.4628	0.4537	0.4548	0.4832	0.4944

Table 82. Eurlex reference documents for pt-en language direction

Table 82 shows that the higher scores for the pt-en language direction are achieved in very similar circumstances as with the en-pt language direction.

The following two tables show the score behavior for the euconst reference documents, which is a bit different from the eurlex reference documents.

		phrase penalty						
		1	2	3	4	5	6	7
tw	0.0	0.5161	0.6842	0.7165	0.7361	0.7438	0.7477	0.7506
	0.1	0.5281	0.6989	0.7355	0.7538	0.7613	0.7654	0.7666
	0.2	0.5314	0.7093	0.7429	0.7586	0.7659	0.7687	0.7712
	0.3	0.5305	0.7238	0.7539	0.7665	0.7719	0.7756	0.7774
	0.4	0.5124	0.7299	0.7582	0.7698	0.7755	0.7782	0.7795
	0.5	0.4692	0.7533	0.7717	0.7797	0.7828	0.7847	0.7851
	0.6	0.3883	0.7684	0.7848	0.7878	0.7909	0.7909	0.7911
	0.7	0.2914	0.7722	0.7910	0.7953	0.7983	0.8004	0.8007
	0.8	0.2086	0.7621	0.7955	0.8045	0.8075	0.8094	0.8104
	0.9	0.1669	0.7229	0.7913	0.8063	0.8106	0.8158	0.8168
	1.0	0.1463	0.6774	0.7713	0.7945	0.8007	0.8043	0.8075

Table 83. Euconst reference documents for en-pt language direction

Table 83 shows that the higher scores for the en-pt language direction are achieved for higher *lw* values (lower *tw* values) and for higher phrase penalty values.

		phrase penalty						
		1	2	3	4	5	6	7
tw	0.0	0.4100	0.6566	0.7015	0.7211	0.7248	0.7262	0.7272
	0.1	0.4344	0.6723	0.7216	0.7384	0.7426	0.7438	0.7453
	0.2	0.4442	0.6866	0.7285	0.7426	0.7470	0.7476	0.7504
	0.3	0.4396	0.7006	0.7401	0.7484	0.7530	0.7539	0.7550
	0.4	0.4283	0.7207	0.7542	0.7590	0.7626	0.7631	0.7641
	0.5	0.4027	0.7466	0.7683	0.7736	0.7744	0.7732	0.7726
	0.6	0.3627	0.7626	0.7834	0.7860	0.7866	0.7859	0.7852
	0.7	0.3040	0.7636	0.7955	0.8025	0.7996	0.8004	0.7989
	0.8	0.2524	0.7425	0.7934	0.8047	0.8097	0.8068	0.8063
	0.9	0.2151	0.7129	0.7813	0.7970	0.8059	0.8089	0.8088
	1.0	0.1881	0.6546	0.7510	0.7735	0.7823	0.7858	0.7883

Table 84. Euconst reference documents for pt-en language direction

Table 84 shows that the higher scores for the pt-en language direction are also achieved in very similar circumstances as with the en-pt language direction.

Most likely the main reason behind the difference between the eurlex and euconst reference documents has to do with the fact that the eurlex documents have smaller sentences than the euconst documents and, intuitively, given the

size of the n-grams used (up to 7 words), it is not surprising that the language model does not require such a high relevance for the eurlex documents as for the euconst documents.

The fact that the worse values are always achieved for α=1 is a common factor shared between every language pair, in every language direction, and in every corpus analyzed. Another common factor is that the best values are usually clustered together, with the values often changing very slightly between them.

These results contribute to provide some guarantee that choosing a value from a range of possible values will produce good results. This is because it will be reasonable to assume that even if the best value is not the one chosen, there is a good chance that another value will be very close to it.

4.6 RESULTS ANALYSIS

The results presented in Section 4.3 above show that the gains with Transtor are most considerable for the smaller corpus OPUS EUCONST, which topic is also the most relevant to the reference documents. This means that Transtor is more capable of taking advantage of a corpus with a topic relevant to the translation in question even if it is very small. The larger corpus DGT-TM is not as relevant to the reference documents as the smaller one and, therefore, its results are not as good as with OPUS EUCONST. Yet, on average, Transtor is still capable of producing better results than Moses when using DGT-TM.

When also considering the additional results presented in Section 4.4 above, it can be seen that a bilingual phrase lexicon can have a significant positive impact on translation quality. Additionally, it can be established that size can be a very important factor to achieve good results, but topic relevance as well as a controlled language and vocabulary are even more important in translation quality.

Additionally, the differences in space and time efficiency between both tools in favor of Transtor, presented in Section 4.2, have shown that the structures used were very appropriate, even though they were not explored to their full potential as they still allow some optimizations. Even with a rough estimate, it will be possible to achieve an improvement in the order of at least 50% for the space occupied by the training data, in future work. This is space saving estimate results from the realization that some data members from the structures are redundant while others can be represented with fewer bytes.

From the data presented on Section 4.5 above, it can be confirmed that there is room for translation quality improvement when using other parameters, show-

ing that the higher values obtained when compared to Moses on Section 4.3 and the good values obtained for additional corpora on Section 4.4.2 did not result from a simple stroke of luck.

As a final remark, it should be noted that some scores involving the Portuguese language have been penalized because changes in written language with the orthographical agreement were not completely reflected into the validated bilingual phrase lexica, as well as because the corpora were still using the previous Portuguese writing while some of the reference documents already had the new Portuguese writing agreement.

5 CONCLUSIONS AND FUTURE WORK

Transtor, the system presented in this thesis, proposes a truly phrase-based approach to SMT (Section 2.1), since no sub-phrasal analysis is carried out in any of its components. The system includes a feature implementation that allows the expression power of tree-based approaches, as well as the inclusion of validated knowledge. The approach is supported by efficient structures and methodologies (Chapter 3) even though they still allow room for further improvement. Some of the main differences have already been highlighted and discussed when comparing Transtor against Moses above (Section 3.5), but it is still important to stress out some of its main features.

The main challenges faced by a word-level analysis of translations can be represented by English/Portuguese translation equivalents like "in order to" <=> "a fim de", "however" <=> "em o entanto", "considering" <=> "tendo em conta", or "back to square one" <=> "de volta a a estaca zero", where there is no clear word-to-word correspondence between the equivalents. In such cases, phrases should not be partitioned into their composing words. The inclusion of validated phrase translation lexica also prevents the system from having to identify the same phrase translation equivalents over and over again, this way improving subsequent translations. This is in contrast with main approaches (Sub-Section 2.1.2) because each time they are presented with the same parallel corpora they learn exactly the same they had learned in previous training sessions, with the only factor that might contribute to get some different results is the heuristic nature of some procedures involved in their training process.

Again, this approach is truly phrase-based since phrases are considered to be units by themselves and not just a composition of words, unlike many of the approaches that claim to be phrase-based while considering individual words at any given point in their process. The phrase decomposition into words can be a disadvantage, particularly when considering the problems faced in some cases, for which idiomatic expressions constitute very good examples, like "back to square one" <=> "de volta a a estaca zero". On the contrary, Transtor keeps phrases undivided in every component from the system, namely the alignment used, the phrase translation extraction, the translation model, the language model, the penalty model and its final translation probability score (Sub-Section 3.4.7).

The contiguous phrase translation equivalents used by Transtor (Sub-Section 3.1.1.1) are identified (Sub-Section 3.3.1) from parallel corpora that have been

aligned (Section 2.3) at the phrase level and considering previous validated knowledge, unlike most approaches claiming to be phrase-based at the cost of an initial bidirectional word-based alignment stage (Sub-Section 2.1.2), with the disadvantages noted above, and never considering previously validated knowledge. The use of validated knowledge allows focusing on the identification of new knowledge and reducing systematic mistakes in alignment or in phrase translation identification.

The sentence translation probability score (*stm*) used by Transtor (Sub-Section 3.4.7) is also different from the widely used log-linear model approach (Sub-Section 2.1.1.2). The latter is defined as a product of probabilities which is overly sensitive to low local scores that will penalize a whole translation candidate, and require the models involved to have smoothing techniques applied to them in order to avoid any probabilities of zero but at the risk of introducing other errors. On the contrary, *stm* is defined as a weighted average of the translation model and the language model, normalized by the number of phrases composing the translation candidate, which is not so sensitive to lower local scores, as well as zero local scores do not invalidate a whole candidate, particularly if a high adjacent score ends up compensating for such low values. Besides, its penalty parameter helps in the choice of translation candidates which have fewer and, therefore, longer phrases, which is an advantage because, as mentioned in (Koehn, 2009), shorter phrases occur more frequently so they will more often be applicable to previously unseen sentences but longer phrases capture more local context and help translating larger chunks of text at one time.

Additionally, the log-linear approach does not limit the number of translation feature models that can be used to define the quality score of a translation candidate. However, such limitation absence is not necessarily an advantage, not only because the most prevalent models are only four (translation model, language model, reordering model and penalty model) but also because identifying relevant features can be a very challenging task (Sub-Section 2.1.3).

On the contrary, Transtor only depends on three simple feature models, which are the translation model, the language model and the penalty model. These models, while common to many approaches, still have some differences in relation to their main implementations.

The translation model *ptm* Sub-Section (3.3.2.1) is simply defined by the direct translation probability of the translation phrase in relation to the original phrase, a probability obtained based on the alignment produced from a given parallel corpus. No additional inverse translation probabilities or lexical transla-

tion probabilities are included in *ptm*. Lexical probabilities analyze a phrase translation equivalent at their word-level, again facing the problems mentioned above (Sub-Section 2.1.3.1).

The language model *plm* Sub-Section (3.3.2.2) is the one having the most significant differences, not only because it considers phrases without decomposing them into their words, but also because it is determined according to adjacent phrase translation alternatives and not as an absolute model. The prevalent n-gram history model (Sub-Section 2.1.3.2) will consider "`in spite of`", "`in spite`" "`of`", "`in`" "`spite of`" or "`in`" "`spite`" "`of`" to be same. Since phrases are not decomposed, the internal fluency of a phrase like "`in spite of`" is not determined by *plm*, only considering the fluency of such phrase against other adjacent phrases.

In a sense, the current version of the language model establishes a degree of agreement between adjacent phrases through their literal representation. However, sometimes such degree of agreement has to be established between phrases that are not adjacent, as well as it is necessary to generalize beyond their literal form or determine the meaning of phrases. For this purpose, it is planned to integrate word sense disambiguation work (Casteleiro, Lopes and Silva, 2014) to help select phrase translations according to context.

The decoding algorithm developed for Transtor (Sub-Section 3.4.8) is implemented as best path finding by the traversal of the complete graph created after the sentence to be translated, implicitly determining the original phrases that have been covered in the translation. Every possible partition of a sentence is currently considered in decoding, but future work might include the analysis of other elements like capitalized words or the identification of functional words, like "`of`" and "`in`", which might provide additional clues as to how to partition a sentence.

Contiguous phrase translations (Sub-Section 3.1.1.1) cover a significant number of situations required to translate a text and are very efficient to apply, but these are static and some translation cases require some flexibility. For this reason, translation patterns (Sub-Section 3.1.1.2) were introduced in order to support translation generalizations that can additionally implement reordering with some lexical evidence. The translation patterns constitute a feature that can be compared to the hierarchical approach (Sub-Section 2.1.5.1), but the representation power of the latter is implemented in a way that requires every phrase translation to be represented as a rule, including some special set of rules (the glue rules), while its decoding has to be carried out by chart parsing. In con-

trast, the translation patterns introduced in this approach simply allow Transtor to rearrange the translation graph in order to accomplish the potential for the same expression power, this way keeping the application efficiency allowed by contiguous phrase translation equivalents, which are the most common equivalents, while keeping the same decoding algorithm developed before the translation patterns were contemplated by Transtor. This is because the translation patterns simply provide information for alternative placements of phrase translations on the translation graph, applying the models as if such placement resulted from the normal combination of the simple phrase translations, so the models do not require any additional change.

In a sense, translation patterns are used as hint elements that provide additional information about how to rearrange the graph in order to produce non-monotonic translations. Yet, it should be noted that the immediate purpose of the inclusion of the translation pattern feature is to assess their impact on results (Sub-Section 4.4.1). The next challenge is to ensure the importance of translation patterns is reflected in an increase in translation quality. Such reflection should be achieved by increasing the translation pattern coverage, including recursivity and implementing them in a more efficient way while preventing its complexity to escalate to the point where results can no longer be obtained in useful time, besides extending the translation pattern feature to include other generic symbols like letters, admitting the use of mixed tags in a single pattern, like <var> and <number>, and the ability to deal with patterns at a sub-word level, or morphology level.

Additionally, all the introduced SMT approaches (Section 2.1) contribute to the idea that producing a translation is a process for which there is never a clue on how to translate a text from one language to another, being largely focused on learning translation models and producing translations from a stage where no previously acquired or validated knowledge is available, but storing such knowledge to be used in future translations is a realistic approach that should deserve more attention. In fact, as mentioned before, most research is focused on creating a bilingual phrase lexicon from scratch without ever considering the advantages of using bilingual phrase lexicon entries that are already known and have been validated. Actually, even when known entries are considered, these are superficially approached, simply using a dictionary (the closest thing to a bilingual phrase lexicon) for scoring translations, as is the case of the lexical probability applied to phrase translation equivalents. However, there can be information available about how to translate a great number of phrases, as well as there are language constructs that follow very specific rules. In the approach

supporting Transtor, such knowledge can be expressed both in the literal form of bilingual phrase lexica (contiguous phrases) and in the more generic form of translation patterns (non-contiguous phrases).

All the retrieval operations required by Transtor are supported by indexing structures based on suffix arrays (3.1.3), which provide an efficient and compact support for such operations. In fact, those indexing structures allow using a simple integer value as an identifier of an individual unique phrase, not only avoiding the replication of the text, but also improving space efficiency, since this decision enables the monolingual phrases (with sizes ranging from 1 byte to 100 bytes or more) to be represented by a single integer (having a fixed size of 4 or 8 bytes). The mentioned identifiers can be obtained through a binary search of the phrase on the corresponding indexing structure, so space efficiency is accompanied by search efficiency. Nevertheless, the indexing structures still have room for significant improvement, namely in the occupied space, for which the integration with compressed indexing structures (Costa et al., 2013) is planned.

In sum, this thesis has introduced Transtor, an innovative and truly phrase-based SMT approach which, despite of being very simple, the sum of all the features mentioned above, along with the way they are combined, certainly play an important role on the higher quality of the results. Its simplicity is manifested not only in the definition of the procedures involved, but also by the smaller amount of data produced and the shorter period of processing time. In fact, the amount of data and the processing time both have significant room for improvement since the current version of the tool has been mostly developed as a "proof of concept". Nevertheless, even at its current stage of development, the results obtained so far are an encouragement to proceed exploring and developing this new approach for Machine Translation.

BIBLIOGRAPHY

Abouelhoda, M.I., Kurtz, S. and Ohlebush, E. (2004) 'Replacing Suffix Trees with Enhanced Suffix Arrays', *Journal of Discrete Algorithms*, no. 2, pp. 53-86.

Aires, J., Lopes, J.G.P. and Gomes, L. (2009) 'Phrase translation extraction from aligned parallel corpora using Suffix Arrays and related structures', EPIA'09, Aveiro.

Aires, J., Lopes, J.G.P. and Silva, J.F.d. (2008) 'Efficient Multi-Word Expressions Extractor Using Suffix Arrays and Related Structures', ACM iNEWS'08, Napa Valley, California, USA.

Barrachina, S., Bender, O., Casacuberta, F., Civera, J., Cubel, E., Khadivi, S., Lagarda, A., Ney, H., Tomás, J., Vidal, E. and Vilar, J.M. (2009) 'Statistical Approaches to Computer-Assisted Translation', *Computational Linguistics*, vol. 35, no. 1, pp. 3-28.

Berger, A.L., Pietra, S.A.D. and Pietra, V.J.D. (1996) 'A maximum entropy approach to natural language processing', vol. 22, no. 1, pp. 39-71.

Brown, P.F., Cocke, J., Pietra, S.D., Pietra, V.J.D., Jelinek, F., Lafferty, J.D., Mercer, R.L. and Roossin, P.S. (1990) 'A statistical approach to machine translation', *Computational Linguistics*, vol. 16, no. 2, pp. 76-85.

Brown, P., Cocke, J., Pietra, S.D., Pietra, V.D., Jelinek, F., Mercer, R. and Roossin, P. (1988) 'A statistical approach to language translation', *Proceedings of the 12th International Conference on Computational Linguistics (COLING-88)*, Aug, pp. 71-76.

Brown, P.F., Della-Pietra, S.A., Della-Pietra, V.J. and Mercer, R.L. (1991a) 'Word-sense disambiguation using statistical methods', *In Proceedings of the 29th Annual Meeting of the Association of Computational Linguistics (ACL)*.

Brown, P.F., Della-Pietra, S.A., Della-Pietra, V.J. and Mercer, R.L. (1993) 'The mathematics of statistical machine translation', *Computational Linguistics*, vol. 19, no. 2, pp. 263-313.

Brown, P.F., Lai, J.C. and Mercer, R.L. (1991b) 'Aligning sentences in parallel corpora', *In Proceedings of the 29th Annual Meeting of the Association of Computational Linguistics (ACL)*.

Carbonell, J., Klein, S., Miller, D., Steinbaum, M., Grassiany, T. and Frey, J. (2006) 'Context-based machine translation', *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas*, pp. 8-12.

Casteleiro, J., Lopes, G.P. and Silva, J. (2014) 'Bilingually Learning Word Senses for Translation', *Computational Linguistics and Intelligent Text Processing, 15th International Conference, CICLing 2014*.

Chiang, D. (2007) 'Hierarchical phrase-based translation', *Computational Linguistics*, vol. 33, no. 2.

Costa, J., Gomes, L., Lopes, G.P., Russo, L.M.S. and Brisaboa, N.R. (2013) 'Compact and Fast Indexes for Translation Related Tasks', *Progress in Artificial Intelligence - 16th Portuguese Conference on Artificial Intelligence*, pp. 504-515.

Darriba Bilbao, V.M., Lopes, J.G.P. and Ildefonso, T. (2005) 'Measuring the impact of cognates in parallel text alignment', *Proceedings of the Portuguese Conference on Artificial Intelligence*, pp. 338-343.

Deng, Y., Xu, J. and Gao, Y. (2008) 'Phrase Table Training for Precision and Recall: What Makes a Good Phrase and a Good Phrase pair?', *Proceedings of ACL (HLT)*, pp. 81-88.

Dias, G. (2002) 'Extraction Automatique d'Associations Lexicales à partir de Corpora', *Ph.D. Thesis*.

Earley, J. (1970) 'An efficient context-free parsing algorithm', *Communications of the ACM*, vol. 13, no. 2, pp. 94–102.

Fischer, J., Mäkinen, V. and Navarro, G. (2008) 'An(other) Entropy-Bounded Compressed Suffix Tree', *Proceedings of CPM*, pp. 152-165.

Gale, W.A. and Church, K.W. (1993) 'A program for aligning sentences in bilingual corpora', *Computational Linguistics*, vol. 19, no. 1, pp. 75–102.

Gale, W.A. and Sampson, G. (1995) 'Good–Turing frequency estimation without tears', *Journal of Quantitative Linguistics*, vol. 2, pp. 217–237.

Gamallo, P., Agustini, A. and Lopes, J.G.P. (2005) 'Clustering Positions with Similar Requirements', *Computational Linguistics*, vol. 31, no. 1, pp. 107-145.

Gamallo, P., Agustini, A. and Lopes, J.G.P. (2008) 'Automatic Acquisition of Formal Concepts from Text', *Journal for Computational Linguistics and Language Technology*, vol. 23, no. 1, pp. 61-76.

Germann, U., Jahr, M., Knight, K., Marcu, D. and Yamada, K. (2001) 'Fast decoding and optimal decoding for machine translation', *Proceedings of the 39th Annual Meeting of the Association of Computational Linguistics*.

Gomes, L., Aires, J. and Lopes, J.G.P. (2009) 'Parallel Texts Alignment', EPIA'09, Aveiro, Portugal.

Goutte, C., Yamada, K. and Gaussier, E. (2004) 'Aligning words using matrix factorization', *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pp. 502–509.

Haddow, B., Arun, A. and Koehn, P. (2011) 'SampleRank Training for Phrase-Based Machine Translation', *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pp. 261-271.

Hovy, E., King, M. and Popescu-Belis, A. (2002) 'Principles of context-based machine translation evaluation', *Machine Translation*, vol. 17, no. 1, pp. 43–75.

Ildefonso, T. and Lopes, J.G.P. (2005) 'Longest Sorted Sequence Algorithm for Parallel Text Alignment', *EUROCAST 2005*, pp. 81-90.

Jelinek, F. (1969) 'A stack algorithm for faster sequential decoding of transmitted information'.

Jelinek, F. and Mercer, R.L. (1980) 'Interpolated Estimation of Markov Source Parameters from Sparse Data', *Proceedings of the Workshop on Pattern Recognition in Practice*.

Kay, M. (1985) 'Parsing in functional unification grammar', *Natural Language Parsing*, pp. 251-278.

Knight, K. (1999) 'Decoding complexity in word-replacement translation models', *Computational Linguistics*, vol. 25, no. 4, pp. 607–615.

Koehn, P. (2004) 'Pharaoh: A beam search decoder for phrase-based statistical machine translation models', *Proceedings of the Conference of the Association for Machine Translation in the Americas (AMTA)*.

Koehn, P. (2009) *Statistical Machine Translation*, Cambridge: Cambridge University Press.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A. and Herbst, E. (2007) 'Moses: Open Source Toolkit for Statistical Machine

Translation', *Annual Meeting of the Association for Computational Linguistics (ACL), demonstration session*, June.

Koehn, P., Och, F.J. and Marcu, D. (2003) 'Statistical phrase-based translation', *Proceedings of HLTNAACL*, pp. 127–133.

Larsson, N. and Sadakane, K. (1999) 'Faster Suffix Sorting'.

Lopez, A. (2008) 'Statistical machine translation', *ACM Computing Surveys*, vol. 40, no. 3, Aug, pp. 1-49.

Mahesh, K., Gomes, L. and Lopes, J.G.P. (2011) 'Using SVMs for Filtering Translation Tables', *Proceedings of 15th Portuguese Conference in Artificial Intelligence*, pp. 690-702.

Manber, U. and Myers, G. (1990) 'Suffix Arrays: A New Method for On-Line String Searches', Proceedings of the First Annual ACM-SIAM Symposium on Discrete Algorithms, 319-327.

Och, F.J. (1999) 'An efficient method for determining bilingual word classes', *EACL '99: Ninth Conference of the European Chapter of the Association for Computational Linguistics*, June, pp. 71–76.

Och, F.J. and Ney, H. (2002) 'Discriminative training and maximum entropy models for machine translation', pp. 156-163.

Och, F. and Ney, H. (2003) 'A systematic comparison of various statistical alignment models', *Computational Linguistics*, vol. 29, no. 1, pp. 19-51.

Och, F.J. and Ney, H. (2004) 'The alignment template approach to statistical machine translation', *Computational Linguistics*, vol. 30, no. 4, pp. 417-449.

Och, F.J., Tillman, C. and Ney, H. (1999) 'Improved alignment models for statistical machine translation', *Proceedings of EMNLP-VLC*, pp. 20–28.

Papineni, K., Roukos, S., Ward, T. and Zhu, W.-J. (2002) 'BLEU: a method for automatic evaluation of machine translation', *Proceedings of the Association for Computational Linguistics (ACL)*, pp. 311-318.

Pecina, P., Toral, A. and Genabith, J.v. (2012) 'Simple and Effective Parameter Tuning for Domain Adaptation of Statistical Machine Translation', *COLING*, pp. 2209-2224.

Peleja, F., Silva, J.F.d. and Lopes, G.P. (2011) 'Text Categorization: An extensive comparison of classifiers, feature selection metrics and document representation', *Proceedings of the 15th Portuguese Conference in Artificial Intelligence, EPIA 2011*.

Ribeiro, A. (2002) 'Parallel Text Alignment for Extraction of Translation Equivalents', *Ph.D. Thesis*.

Ribeiro, A., Dias, G., Lopes, J.G.P. and Mexia, J.T. (2001) 'Cognates Alignment', *Proceedings of the Machine Translation Summit VIII (MT Summit VIII)*, pp. 287-292.

Ribeiro, A., Lopes, J.G.P. and Mexia, J.T. (2000a) 'A self Learning Method of Parallel texts Alignment', *Envisioning Machine Translation in the Information Future: AMTA*, pp. 30-39.

Ribeiro, A., Lopes, J.G.P. and Mexia, J.T. (2000b) 'A self Learning Method of Parallel Texts Alignment', *Envisioning Machine Translation in the Information Future, 4th Conference of the Association for Machine Translation in the Americas, AMTA 2000*, pp. 30-39.

Ribeiro, A., Lopes, J.G.P. and Mexia, J.T. (2000c) 'Aligning Portuguese and Chinese Parallel Texts Using Confidence Bands', *PRICAI 2000: Topics in Artificial Intelligence, Sixth Pacific Rim International Conference on Artificial Intelligence*, pp. 627-637.

Ribeiro, A., Lopes, J.G.P. and Mexia, J.T. (2000d) 'Extracting Translation Equivalents from aligned parallel texts: comparison of measures of similarity', *Advances in Artificial Intelligence: IBERAMIA-SBIA*, pp. 339-349.

Ribeiro, A., Lopes, J.G.P. and Mexia, J.T. (2000e) 'Linear Regression Based Alignment of Parallel Texts Using Homograph Words', *ECAI 2000: Proceedings of the 14th European Conference on Artificial Intelligence*, pp. 446-450.

Ribeiro, A., Lopes, J.G.P. and Mexia, J.T. (2000f) 'Using Confidence Bands for Alignment with Hapaxes', *Proceedings of the International Conference on Artificial Intelligence (IC-AI 2000)*, pp. 1089-1095.

Ribeiro, A., Lopes, J.G.P. and Mexia, J.T. (2000g) 'Using Confidence Bands for Parallel Texts Alignment', *Proceedings of the 38th Ann. Meeting of ACL*, pp. 432-439.

Russo, L., Navarro, G. and Oliveira, A.L. (2008) 'Fully-Compressed Suffix Trees', *Proceedings LATIN*, pp. 362-373.

Sadakane, K. (2007) 'Succinct data structures for flexible text retrieval systems', *Journal of Discrete Algorithms*, vol. 5, no. 1, pp. 12-22.

Setiawan, H., Li, H. and Zhang, M. (2005) 'Learning phrase translation using level of detail approach', *Proceedings of the Tenth Machine Translation Summit (MT Summit X)*.

Shannon, C.E. (1948) 'A mathematical theory of communication', *Bell System Technical Journal*, vol. 27, no. 3, pp. 379-423.

Shannon, C.E. (1951) 'Prediction and entropy of printed English', *Bell Systems Technical Journal*, vol. 30, pp. 50-64.

Shin, J.H., Han, Y.S. and Choi, K.-S. (1996) 'Bilingual knowledge acquisition from Korean-English parallel corpus using alignment', *Proceedings of the 16th International Conference on Computational Linguistics (COLING)*.

Silva, J.F., Dias, G., S, G. and Lopes, J.G.P. (1999) 'Using LocalMaxs Algorithm for the Extraction of Contiguous and Non-contiguous Multiword Lexical Units', *Progress in Artificial Intelligence: EPIA*, pp. 113-132.

Silva, J.F.d. and Lopes, J.G.P. (2006) 'Identification of Document Language is not yet a completely solved problem', *Proceedings of International Conference on Computational Intelligence for Modelling, Control and Automation (CIMCA)*.

Silva, J.F.d. and Lopes, J.G.P. (2009) 'A Document Descriptor Extractor Based on Relevant Expressions', *Progress in Artificial Intelligence*.

Silva, J.F.d., Mexia, J.T., Coelho, C.A. and Lopes, J.G.P. (2001) 'Document Clustering and Cluster Topic Extraction in Multilingual Corpora', *Proceedings of the IEEE 2001 International Conference on Data Mining (ICDM'01)*, pp. 513-520.

Teixeira, L., Lopes, G.P. and Ribeiro, R. (2013) 'Language Independent Extraction of Key Terms: An Extensive Comparison of Metrics', *Agents and Artificial Intelligence. Communications in Computer and Information Science Series.*, vol. 358, pp. 69-82.

Tillmann, C. (2003) 'A projection extension algorithm for statistical machine translation', *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pp. 1–8.

Tillmann, C. (2004) 'A unigram orientation model for statistical machine translation', *HLT-NAACL*.

Viterbi, A.J. (1967) 'Error bounds for convolutional codes and an asymptotically optimum decoding algorithm', *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 260–269.

Wang, Y.-Y. and Waibel, A. (1997) 'Decoding algorithm in statistical machine translation', *Proceedings of ACL-EACL*, pp. 366–372.

Weaver, W. (1955) 'Translation (1949)'.

Yamamoto, M. and Church, K. (2001) 'Using Suffix Arrays to Compute Term Frequency and Document Frequency for All Sub-Strings in a Corpus', *Computational Linguistics*, vol. 27, no. 1, pp. 1-30.

Yamamoto, K., Kudo, T., Tsuboi, Y. and Matsumoto, Y. (2003) 'Learning sequence-to-sequence correspondences from parallel corpora via sequential pattern mining', *HLT-NAACL 2003 Workshop: Building and Using Parallel Texts: Data Driven Machine Translation and Beyond*.

Zhang, Y. and Vogel, S. (2005) 'Competitive grouping in integrated phrase segmentation and alignment model', *Proceedings of the ACL Workshop on Building and Using Parallel Texts*, pp. 159–162.

Zhang, Y., Vogel, S. and Waibel, A. (2003) 'Integrated phrase segmentation and alignment algorithm for statistical machine translation', *Proceedings of International Conference on Natural Language Processing and Knowledge Engineering (NLP-KE'03)*.

Zhao, B. and Vogel, S. (2005) 'A generalized alignment-free phrase extraction', *Proceedings of the ACL Workshop on Building and Using Parallel Texts*, pp. 141–144.

ANNEX

In order to express a direct relation between the formulas and the right-to-left implementation of the decoding algorithm, a sentence divided into n non-empty phrases will be represented as $e_n, e_{n-1}, \dots, e_1$, where e_n represents the first phrase, e_{n-1} represents the second phrase, e_{n-2} represents the third phrase, and so on, with e_1 representing the last (n^{th}) phrase. This way, $stm(e_n)$ represents the *stm* score of a sentence $e_n, e_{n-1}, \dots, e_1$, with its formula shown below.

$$stm(e_n) = \sum_{i=1}^{n-1} \frac{es(e_{i+1}, e_i)}{n^{\alpha}}$$

Following the same logic, $rstm(e_n)$ represents the *stm* score for the same sentence which, being recursive, will depend on $rstm(e_{n-1})$. The base case of the recursive function will be the *rstm* value of e_2, since there has to be at least a pair of phrases. The *rstm* formula is shown below.

$$rstm(e_n) = \begin{cases} 0, & n = 1 \\ \dfrac{es(e_n, e_{n-1}) + rstm(e_{n-1}) \cdot (n-1)^{\alpha}}{n^{\alpha}}, & n > 1 \end{cases}$$

The equivalence between *stm* and *rstm* is demonstrated by induction, as shown below.

The Base Case

Since the formulas are applied for n>1, the base case is n=2. Applied to the original form, *stm*:

$$stm(e_2) = \sum_{i=1}^{2-1} \frac{es(e_{i+1}, e_i)}{2^{\alpha}}$$

$$= \sum_{i=1}^{1} \frac{es(e_{i+1}, e_i)}{2^{\alpha}}$$

$$= \frac{es(e_2, e_1)}{2^{\alpha}}$$

Applied to the recursive form, *rstm*:

$$rstm(e_2) = \frac{es(e_2, e_1) + rstm(e_1) \cdot (2-1)^\alpha}{2^\alpha}$$

$$= \frac{es(e_2, e_1) + 0 \cdot 1^\alpha}{2^\alpha}$$

$$= \frac{es(e_2, e_1)}{2^\alpha} = stm(e_2)$$

So the equivalence for the base case is confirmed.

The General Case

Proceeding with the induction proof, *n*+1 is verified for *stm*

$$stm(e_{n+1}) = \sum_{i=1}^{n+1-1} \frac{es(e_{i+1}, e_i)}{(n+1)^\alpha}$$

$$= \sum_{i=1}^{n} \frac{es(e_{i+1}, e_i)}{(n+1)^\alpha}$$

$$= \sum_{i=1}^{n-1} \frac{es(e_{i+1}, e_i) + es(e_{n+1}, e_n)}{(n+1)^\alpha}$$

And also for *rstm*

$$rstm(e_{n+1}) = \frac{es(e_{n+1}, e_n) + rstm(e_n) \cdot n^\alpha}{(n+1)^\alpha}$$

Applying the rule, assuming $rstm(e_n) = stm(e_n)$

$$rstm(e_{n+1}) = \frac{es(e_{n+1}, e_n) + rstm(e_n) \cdot n^\alpha}{(n+1)^\alpha}$$

$$= \frac{es(e_{n+1}, e_n) + \sum_{i=1}^{n-1} \frac{es(e_{i+1}, e_i)}{n^\alpha} \cdot n^\alpha}{(n+1)^\alpha}$$

$$= \frac{\sum_{i=1}^{n-1} es(e_{i+1}, e_i) + es(e_{n+1}, e_n)}{(n+1)^\alpha}$$

$$= \sum_{i=1}^{n} \frac{es(e_{n+1}, e_n)}{(n+1)^\alpha} = stm(e_{n+1})$$

this way demonstrating, by induction, the equivalence between *stm* and *rstm*.