UNIVERSITAT POLITÈCNICA DE VALÈNCIA

UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

Departament de Sistemes Informàtics i Computació

IARFID Master Thesis:

Phrase-Based ITG Decoder

Author:
Guillem Gascó i Mora

Advisor:
Joan Andreu Sánchez Peiró

November 14, 2008

Abstract

Statistical machine translation is a well studied field of computational linguistics. Phrase-based machine translation is one of the most successful approaches to this problem. Nevertheless, the results obtained for complex corpus are still much worse than human translation. Some language phenomena cannot be modelled using the lexicalized phrase-based models, such as syntactic reorderings or long term concordances. In such cases it is necessary the use of syntactic machine translation techniques. Syntax-based machine translation use synchronous grammars and other formalisms to include syntactic information in the translation process.

Our aim in this master thesis is the definition of a hybrid translation model that uses the main strenghts of both approaches: phrase-based and syntax-based. We do not want to incorporate syntactic information in a classical phrase-based system, but to integrate phrase-based techniques and syntax-based techniques in a new search decoder. Because of its simplicity and its expressivity, we have used stochastic phrasal inversion transduction grammars as translation model.

Once, the framework is well defined, we implement a machine translation system within the framework. The decoder can be divided in two parts: a parser that obtains the syntactic tree of the input sentence, and a tree-to-string method that find the most likely target-language sentence, according to the model, given the tree.

In addition, we explore the learning of some parameters needed for the new hybrid model, such as stochastic inversion transduction grammar or the learning of the phrase table. Finally, we present some experiments over two different corpora Spanish-English and German-English.

Resum

La traducció automàtica estadística és un camp de la lingüística computacional molt ben estudiat. D'entre totes les aproximacions a aquest problema en destaca la traducció basada en segments, que darrerament ha ofert els millors resultats. Tanmateix, les traduccions obtingudes per a textos complexos encara estan molt lluny dels resultats humans. Hi ha fenòmens del llenguatge que queden fora de l'enfocament purament lexicalitzat dels sistemes basats en segments: reordenaments sintàctics, concordances a llarg termini... En aquest punt és on ha d'intervindre la traducció basada en sintaxi, que utilitza gramàtiques síncrones i altres formalismes per tal d'incloure informació sintàctica en la traducció.

L'objectiu d'aquesta tesi de màster és la definició d'un model de traducció híbrid que incorpore els avantatges de les dues aproximacions: la basada en segments i la basada en sintaxi. La intenció no és incorporar informació sintàctica a un sistema de traducció basat en segments, sinó integrar tècniques de traducció basada en sintaxi i tècniques de traducció basada en segments en un mateix sistema de cerca. La seua simplicitat i al mateix temps el seu gran poder expressiu han fet que escollim com a model de sintaxi el formalisme conegut de les gramàtiques d'inversió i transducció de segments estocàstiques.

Amb el model híbrid ja ben definit, podem implementar un sistema de traducció que l'utilitze. Aquest sistema es pot dividir en dues parts: un analitzador sintàctic que obté l'arbre d'anàlisi de l'oració d'entrada, i un traductor *arbre-cadena* que cerca l'oració en llengua destí més probable, segons el model, donat l'arbre d'anàlisi.

A més a més, s'investiga l'aprenentatge dels diferents paràmetres necessaris per a l'aproximació híbrida, com són la inferència de gramàtiques estocàstiques d'inversió i transducció o l'aprenentatge de les taules de segments. Finalment es presenten experiments sobre dos corpus diferents castellà-anglés i alemany-anglés.

Contents

Chapter 1

Introduction

"A journey of a thousand miles starts with a single step."
Mao

This chapter is an introduction to the issues the master thesis deals with. First of all, we explain general concepts about machine translation, some common problems of machine translation systems and the motivations of the work. Then we make a revision of the *state-of-the-art* in machine translation, focusing on the two approaches that we have combined in this work: Phrase-based Machine Translation and Syntactical Machine Translation. We describe also some previous approaches to the problem using hybrid systems. Finally, we give a brief overview of the full thesis.

1.1 Motivation

Machine Translation (MT) is a well-established sub-field of Computational Linguistics. MT investigates how to automatically translate text from one natural language to another. There have been important advances in MT, specially since the publication in 1993 of the work of [Brown et al., 1993]. However, current systems cannot produce output of the same quality as a human translator for open domain problems. In order to improve the results obtained by MT systems, maybe it would be a good choice trying to approach to the human-like translation process. Hence, the first question that must be answered is how does human translators work. It is commonly accepted that there are two main steps in the human translation process:

1. Decoding the meaning of the source sentence: The first thing that a translator must do is understand correctly the meaning of the sentence. This step includes

complex actions such as disambiguating the correct meaning of the sentence using other sentences of the text, or other easier tasks as identifying the main action in the sentence.

2. Re-encoding this meaning in target-language: As a second step, the translator must build a correct target-language sentence from the meaning obtained in the previous one.

The first step is related to several other fields of computational linguistics, such as semantics, word sense disambiguation, syntactical parsing, part-of-speech tagging... Usually, this kind of information is ignored in the translation process and current systems are focused in the second step.

Statistical Machine Translation (SMT) is a field within MT that tries to translate texts using statistical models with parameters obtained from bilingual corpora. Nowadays, the most promising results of MT are those obtained using the so called phrase-based models.

Every system or model of Statistical Machine Translation must deal with three major subproblems:

Modeling Problem: This problem is related to how can be modeled the process of translating a sentence. First of all, a formal model must be defined for the process of translation and this model must provide a framework to calculate the translation probability $\Pr(t|s)$.

Learning Problem: Also called *Inference Problem* or *Training Problem*. Given the statistical model chosen, how can be its parameters estimated from a bilingual corpus of parallel sentences? The system must provide methods and algorithms in order to infer the parameters of the translation and language models. These methods must be robust and efficient.

Decoding Problem: Once we have fully specified (framework and parameters) the translation model, it is necessary to provide a method to obtain the most probable target -language sentence given a source-language sentence.

Phrase-based MT approach has been demonstrated to be the best approach in translation of similar structured language pairs (Spanish-English, French-Italian...) [Callison-Burch et al., 2007]. Phrase-based decoders divide the input sentence into segments of words, called *phrases*, translate them into target-language phrases and reorder such resulting phrases to get a final translated sentence. The main advantages of phrases-based systems are that the phrase pairs can be extracted easily from a word alignment and the process of decoding is, usually, easy and fast. In addition, phrase models have a high sentence coverage. However, this approach does not use the linguistic information mentioned above and reorderings cannot take into account

the syntax of the sentence. One of the weaknesses of the phrase-based models is their inability to incorporate syntactic information in the translation. For instance, most English sentences contain a subject and a verb, but there is no way of including this information in a traditional phrase-based system. Syntactic motivated reorderings are also very difficult to include in phrase-based systems.

Another common approach within the field of SMT is the so called syntax-based MT. This approach is characterized by the use of syntactic information in the process of MT. For pairs of languages with a high number of reorderings (Spanish-German, Chinese-English...) syntax-based MT seems to be the best solution. The strength of syntax-based models is that outputs tend to be syntactically well formed and the reordering can be influenced by syntactic context [DeNeefe et al., 2007] . In order to decode the meaning of the source text, one first step must be the syntactic analysis (parsing), of the sentence. However, the estimation of parameters for the syntax-based models is complex. Besides, most of them ignore the lexical context when translating a word [1].

Most of the current phrase-based systems ignore syntactic information in the search of the most probable translation. However, syntactic information has been proved to be useful [Galley et al., 2004][Chiang, 2005][Wang et al., 2007]. It is important to know how can syntax contribute to phrase-based translation. We show it using an example. Figure 1.1 shows the parse tree of a German sentence, and two possible translations.

In **a)**, a parse tree for the sentence is presented. The dashed lines indicate that the translation of the subtree must be inverted. The non-terminals of the parse tree represent some common syntactic constituents of the sentence, such as OB for the object, V for the verb, P for the predicate... **b)** and **c)** are two translations of the sentence. The first one is the correct translation of the sentence and the second one is a possible output from a phrase-based decoder. The reordering model of phrase-based systems does not take into account syntax and usually, only the n-gram language model can help to *choose* the correct order. When this language model is limited, or in the cases of long term reordering, the output tends to be out of order. On the contrary, probably, a syntax-based decoder would have put the verb before the object in an English sentence.

Hence, both approaches, syntax-based and phrase-based MT, have strengths and weaknesses. There have been some previous attempts in order to add syntactic information to the input of a phrase-based decoder. Our aim in this work is, not to incorporate syntactic information to a phrase-based decoder, but to create a hybrid approach. Thus, we present a Phrase-Based Syntactic Decoder as an attempt to unify both approaches and take the best from each one.

[1] The translation unit is usually the word, instead of the phrase.

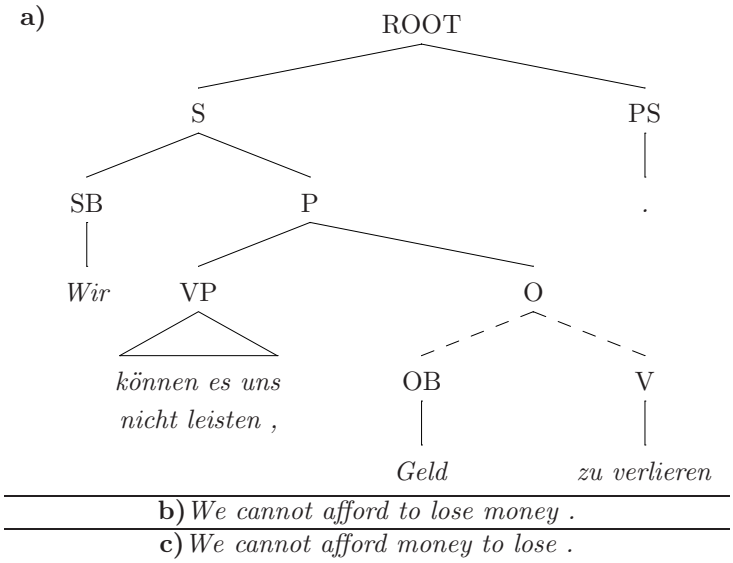


Figure 1.1: a)Parsing of a German sentence. b)Correct translation of the sentence. c)Translation obtained with a phrase-base decoder.

1.2 Previous Approaches

Machine Translation is a well studied field. From the precursors and pioneers of the first half of the 20th century, until now, lot of work have been done. Recently, important advances have been achieved, specially since the re-introduction of Statistical Machine Translation in 1993 by the researchers at IBM's Thomas J. Watson Research Center [Brown et al., 1993] , but there is still room for improvement.

There are a lot of different MT approaches: Rule-Based Systems, Interlingual MT, Example-Based Systems, Statistical Machine Translation... , but we focus this work on Statistical Machine Translation.

SMT is an approach within MT that tries to obtain a statistical model for the translation and then use that model to get the target sentence $\bar{t}^*$ from the source sentence $\bar{s}$ that maximize the probability:

$$\bar{t}^* = \underset{\bar{t}}{\operatorname{argmax}} \Pr_\theta(\bar{t}|\bar{s}) \tag{1.1}$$

where θ is the set of parameters of the model and are obtained from bilingual corpora[2]. Using Bayes rule in (1.1) and given that $\Pr(\bar{s})$ is constant for all the terms in the maximization, we obtain:

[2]From now on, we will assume that θ is implicit in the model.

$$\bar{t}^* = \underset{\bar{t}}{\operatorname{argmax}} \Pr(\bar{s}|\bar{t}) \Pr(\bar{t}) \tag{1.2}$$

where $\Pr(\bar{s}|\bar{t})$ is the probability of the translation from the target string to the source string and is called inverse translation model. $\Pr(\bar{t})$ is the probability of seeing the target string $\bar{t}$ in the target-language, and is commonly called, language model. Thus, $\Pr(\bar{s}|\bar{t})$ models the translation between the sentences and $\Pr(\bar{t})$ models the correctness of the target-language sentence.

First systems of SMT followed the model based in Equation (1.2), that is known as noisy channel model. The process of decoding of the systems that follow such model is illustrated in Figure 1.2. The source-language sentence is preprocessed[3]. Then, the decoder finds the target-language sentence that maximize the probability of translation expressed in Equation (1.2). Finally, the translated sentence is postprocessed[4].

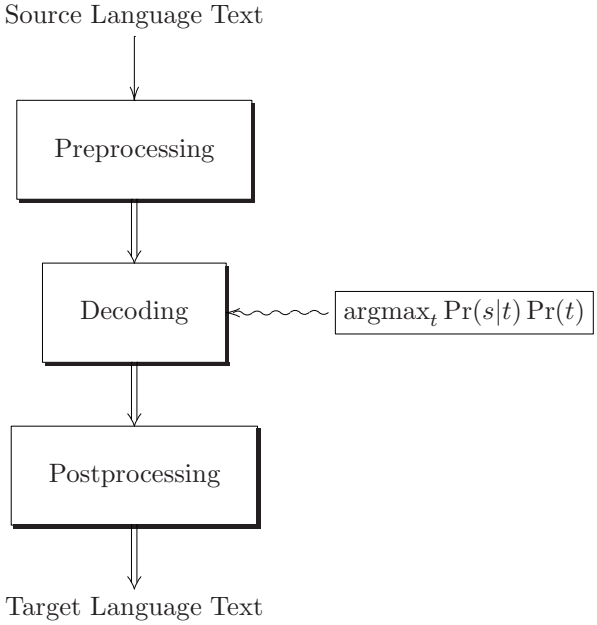


Figure 1.2: Noisy channel modelling for statistical machine translation.

Nowadays, SMT has offered very promising results. Within this approach we can find many different models: phrase-based models [Koehn, 2004], finite-state models [Picó et al., 2004], neural networks [Koncar and Guthrie, 1994], syntax-based models

[3]The preprocess usually includes tasks such as tokenization, lowercasing, named entities recognition...
[4]Some common postprocesses are re-casing, detokenization...

[Yamada and Knight, 2001] among others. The two most studied models within this field are Phrase-Based MT and Syntax-Based MT. For that reason we will explain those approaches in the following subsections.

1.2.1 Phrase Based Machine Translation and Log-Linear Models

The origin of the phrase-based model is the word-to-word model [Brown et al., 1993]. The basic assumption of this original model is that each source word is generated by only one target word. This assumption does not correspond to natural language, because, in most of the cases, it is necessary to know the context in order to find the correct translation of a word.

Phase-based models can be traced back to Och's alignment template model [Och et al., 1999] but they were formally proposed later [Tomás and Casacuberta, 2001] [Koehn et al., 2003]. Using these models, a sequence of source words is aligned with a sequence of target words, commonly named *phrases*. Hence, the statistical dictionaries of single word pairs are substituted by statistical dictionaries of bilingual phrases that usually take the form of a *phrase table*.

During the decoding, the source sentence $\bar{s}$ is segmented into a sequence of I phrases $\bar{s}_1^I$. Each of these phrases $\bar{s}_i$ is translated into a target phrase $\bar{t}_i$. This translation process is modeled by a probability distribution $\Pr(\bar{s}_i|\bar{t}_i)$. It must be noted that this approach uses the inverse translation probability following the noisy channel model, Equation (1.2). The obtained output phrases can be reordered. This reordering is modeled by a probability distribution δ.

The language model use to be a n-gram language model and sometimes, is introduced a factor ω to calibrate the output length. In summary, the Equation (1.2) becomes:

$$\bar{t}^* = \underset{\bar{t}}{\operatorname{argmax}} \Pr(\bar{s}|\bar{t}) \Pr(\bar{t}) \omega^{|\bar{t}|} \tag{1.3}$$

where $\Pr(\bar{s}|\bar{t})$ is decomposed into

$$p(\bar{s}_1^I|\bar{t}_1^I) = \prod_{i=1}^{I} \Pr(\bar{s}_i|\bar{t}_i)\delta \tag{1.4}$$

In [Och et al., 1999] the authors observed that the inverse translation model ($Pr(\bar{s}_i|\bar{t}_i)$) could be replaced by the directed translation model ($Pr(\bar{t}_i|\bar{s}_i)$) without a significant loss in translation quality. This fact is hard to justify within the noisy channel translation framework. For that reason, in [Och and Ney, 2002] is presented a new approach (1.5) that can take into account the inverse translation model, the direct translation model and many other. Following a Maximum Entropy approach, the translation can be viewed as a log-linear combination of models:

$$\bar{t}^* = \underset{\bar{t}}{\operatorname{argmax}} \frac{\exp[\sum_i \lambda_i h_i(\bar{t}, \bar{s})]}{\sum_{\bar{t}'} \exp[\sum_i \lambda_i h_i(\bar{t}', \bar{s})]} = \underset{\bar{t}}{\operatorname{argmax}} \sum_i \lambda_i h_i(\bar{t}, \bar{s}) \tag{1.5}$$

Each of the models, $h_i(\bar{t}, \bar{s})$, called feature is adjusted by a weight, λ_i, computed in the learning phase. It can be seen that the Equation (1.2) is a special case of the log-linear equation with two feature functions (h), the inverse translation model ($\Pr(\bar{s}|\bar{t})$) and the language model ($\Pr(\bar{t})$).

The introduction of such models, usually called log-linear models, opened a new stage full of possibilities. This allows the introduction of new models. The most commonly used models in the literature are:

- Inverse translation model: $\Pr(\bar{s}|\bar{t})$ Probability of the source phrase, given the target phrase.
- Direct translation model: $\Pr(\bar{t}|\bar{s})$ Probability of the target phrase given the source phrase.
- Language model: $\Pr(\bar{t})$ Probability of the resulting target-language sentence.
- Lexical models: Translation probability using a word-to-word model, usually IBM1.
- Phrase penalty: A constant ρ equal for all the phrases. This constant penalizes the translations composed by a higher number of phrases.
- Word penalty factor: $|\bar{t}|$ This feature penalizes short or long[5] sentences.

1.2.2 Syntactic Machine Translation

Syntactical MT systems, or commonly syntax-based MT systems, follow several different methodologies. It is for that reason that we cannot present an unified framework. Thus, we only present an informal description of some of those systems.

Some of these approaches use a parse tree as input, apply transformations to it and get the resulting string from the transformed tree [Yamada and Knight, 2001] [Liu and Gildea, 2008][Nguyen et al., 2003]. For example, Yamada and Knight presented in 2001 a system that transforms a source-language parse tree into a target-language string. This kind of systems are usually called *tree-to-string decoders*. The input sentence is preprocessed by a syntactic parser. The decoder performs some operations over each of the nodes of the tree:

- Reordering of the child nodes.
- Inserting extra words at each nodes.
- Translating leaf words.

[5] Depending on the sign of its weight λ.

Finally it gets the sentence that yields the transformed tree. We can see an example of the decoding process in Figure 1.3[Yamada and Knight, 2001].

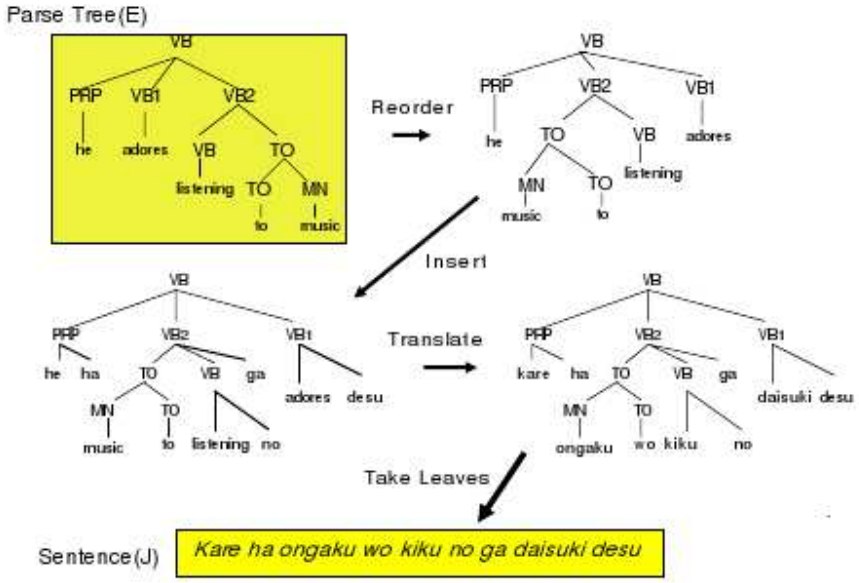


Figure 1.3: Example of Yamada and Knight tree-to-string decoding.

Other approaches obtain a Syntax Directed Transduction Scheme (SDTS) [Aho and Ullman, 1972] , also called Synchronous Grammar, from the training corpora. A SDTS is a grammar that yields simultaneously strings of 2 languages. Then, in order to produce the translation of a sentence, parse the input string and obtain the output string at the same time. In [Galley et al., 2004], the authors present an algorithm to obtain the minimal set of syntactically motivated transformation rules (synchronous grammar rules) from parallel corpora. Using the parse tree of the output and the alignments input-output of the training set, they obtain a derivation of a Synchronous Grammar. Finally they extract the rules from the derivation and test their system proving that it can explain the transformation of any parse tree of the source into a string of the target-language.

Finally, in [Chiang, 2005] is presented a work that uses a synchronous context-free grammar obtained from an aligned corpus following a two-step method:

1. Identify the initial phrase pairs using common phrase extraction techniques [Och and Ney, 2004].

2. Then, obtain the rules from the phrases looking for phrases that contain other phrases and replacing the subphrases with non-terminal symbols.

The result of this algorithm is a very large set of rules, hence, it must be filtered following some constraints and in order to smooth the system other rules are inserted. Rules are scored using a log-linear model. Finally, it is used, as a decoder, a CKY parser with beam search together with a postprocessor to map the input derivations into target sentences. It must be noted that the rules use only one non-terminal symbol and it has no linguistic motivation. An extension to this work using more non-terminal symbols with a linguistic motivation is presented in [Zollmann and Venugopal, 2006].

1.2.3 Combined Approaches

We say that a MT system is a combined approach between syntax and phrase-based when it uses techniques and information of both approaches in order to improve the translation. In most of the cases, such systems use a standard phrase based decoder with some kind of syntactic information or use syntax modelling as a pre or postprocess.

The approach presented in [Wang et al., 2007] uses syntactic reordering as a preprocess for a phrase-based system (training and decoding) for a Chinese-English translation. A similar approach is used in [Collins et al., 2005] and [Nießen and Ney, 2004].

Another kind of combined approaches are those that attach syntactic information to the input source-language sentence (for instance the chunks of the sentence or the shallow parsing) and then use a phrase based decoder [Hassan et al., 2007].

The decoder presented in this master thesis differs from all these previous approaches in an important aspect. The system presented here uses syntax-directed decoding together with phrase-based techniques (phrase tables, log-linear models, beam search...). In addition, the decoder uses syntax based reordering models during the decoding besides a n-gram language model. Therefore, we take advantage from the benefits of both, syntactic and phrase-based machine translation systems.

1.3 Overview

The rest of the contents of this Master Thesis are structured following the three subproblems explained in Section 1.1.

Chapter 2 deals with the modeling problem. All the formalisms used in the thesis are explained in that chapter. First, Stochastic Inversion Transduction Grammars (SITG) are presented together with a bilingual parse algorithm. Then, we explain a modified version of SITG which translation unit is the phrase instead of the word: Stochastic Phrasal Inversion Transduction Grammars (SPhITG). We also specify the theoretical

model beyond the decoder and the features used in the loglinear combination used to score the translations.

Chapters 3 and 4 are related with the learning problem and the estimation of the parameters for the models. In the former, three methods to obtain SITG are presented, and in the later we describe two algorithms that provide phrase tables from a bilingual parallel corpus.

The decoding problem is dealt in Chapter 5. In order to use the syntax for translation, we obtain, first, the most probable syntax tree from the input sentence. Then, we translate this tree into a target-language sentence. We consider the parsing process as a step of the translation process so, in this chapter, is explained also the fast parsing algorithm used. After that, is presented the recursive tree-to-string algorithm, and some acceleration techniques implemented such as hypothesis recombination of beam pruning.

In Chapter 6 we present some experiments performed using the decoder. These experiments have been carried out over the `EuTrans` and `Europarl` corpora. In that chapter we describe the main features of the corpora, the experiments carried out and a short discussion of the results obtained.

Finally, the conclusions of the thesis and some future work suggestions are presented in Chapter 7.

Chapter 2

Translation Model Description

"There is nothing more practical than a good theory."
Leonid Ilich Brezhnev

This chapter deals with the problem of modeling, the explanation of the translation model. A statistical translation model is a mathematical model in which the process of human translation is statistically modeled. As have been stated in the introduction, syntactic information is important in order to obtain good translations. Hence, we propose a model that use syntax formalisms. Synchronous Grammars [Aho and Ullman, 1972], also called Syntax Directed Transduction Schemes (SDTS), are a well-studied formalism to syntactically modelate translation processes. A major problem of such grammars is the high cost of their parsing algorithms. Inversion Transduction Grammars are a binary category of Synchronous Grammars. In this work, we have chosen the use of ITG because of the reduced complexity of their algorithms. Hence, we first explain the formalisms and methods that are needed in the development of the translation model: Stochastic Inversion Transduction Grammars, its phrasal phrasal extension and bilingual parsing algorithms. Finally, we explain the translation model with an informal explanation of the decoding algorithm.

2.1 Inversion Transduction Grammars

Synchronous grammars provide a generative process to produce a sentence and its translation simultaneously. Due to the exponential complexity of their parsing algorithms, its use is restricted for short sentences when using a big grammar. Inversion Transduction Grammars (ITG) [Wu, 1997] are a special kind of Synchronous Grammar whose parse algorithms have polynomial complexity. This fact leads us to the use of ITG as main model for the translation process. An ITG is a tuple $(\mathcal{N}, \Sigma, \Delta, S, \mathcal{R})$ where $\mathcal{N}$ is the set of non-terminals, $S \in \mathcal{N}$ is the root non-terminal, Σ is the input alphabet, Δ is the output alphabet, and $\mathcal{R}$ is a set of rules. Rules can be divided in two sets[1]:

- Syntactic Rules: These rules have the form: $A \rightarrow [BC]$ or $A \rightarrow \langle BC \rangle$, where A, B and C are non-terminals and the brackets enclosing the right part of the rule mean that the two non-terminals are expanded in the same order in the input and output languages, whereas the rules with pointed bracketing expand the left symbol into the right symbols in the straight order in the input language and in reverse order in the output language.

- Lexical Rules: $A \rightarrow x/y$ where $x \in \{\Sigma \cup \epsilon\}$ and $y \in \{\Delta \cup \epsilon\}$. It must be noted that x or y can be the empty string, denoted by ϵ, but not both in the same production.

For any bilingual strings $u = (u^s/u^t)$, $v = (v^s/v^t)$, where $u^s, v^s \in (N \cup \Sigma)^*$ are the source part, and $u^t, v^t \in (N \cup \Delta)^*$ are the target part, we say that u directly yields v, written as $u \Rightarrow v$ if exists a rule $(\alpha \rightarrow \beta^s/\beta^t)$ in $\mathcal{R}$ such that $u = (u_1^s \alpha u_2^s / u_1^t \alpha u_2^t)$ and $v = (v_1^s \beta^s v_2^s / v_1^t \beta^t v_2^t)$. We write $u \Rightarrow^r v$, when the application of the rule $r = \alpha \rightarrow \beta^s/\beta^t$ produces v from u.

In order to get an unifyed framework we substiute always the most left non-terminal. For any u, v, $u \Rightarrow^* v$, and we say that u yields v, if exist $u_1, u_2, \ldots u_k$ such that $u \Rightarrow r_1 u_1 \Rightarrow^{r_2} u_2 \Rightarrow^{r_3} \cdots \Rightarrow^{r_k} u_k \Rightarrow^{r_{k+1}} v$. Each of the ordered sets of rules $\{r_1, r_2, \ldots r_{k+1}\}$ used to produce v from u is called derivation. If u yields v following the derivation d, we express it as $u \Rightarrow^d v$. Commonly, we say that a non-terminal A yields a bitext x/y when $A \Rightarrow^* x/y$. The bilingual language generated by the ITG is the set of bilingual sentences x/y such that $S \Rightarrow^* x/y$.

2.1.1 Stochastic ITG

A Stochastic ITG (SITG) is the natural stochastic extension of an ITG where each one of the rules has been augmented with a probability. For each non-terminal A, the production probabilities are subject to the constraint:

[1] We limit the explanation to the Normal Form presented in[Wu, 1997].

$$\sum_{B,C \in \mathcal{N}} (\Pr(A \rightarrow [BC]) + \Pr(A \rightarrow \langle BC \rangle)) + \sum_{x \in \Sigma, y \in \Delta} \Pr(A \rightarrow x/y) = 1 \tag{2.1}$$

We define the probability of a derivation d from the non-terminal A to the bitext x/y as the product of the probabilities of each of its rules:

$$\Pr(d) = \Pr(A \Rightarrow^d x/y) = \Pr(r_1) \cdot \Pr(r_2) \cdot \cdots \cdot \Pr(r_k) \text{ when } d = r_1, r_2, \ldots r_k \tag{2.2}$$

and

$$\Pr(A \Rightarrow^* x/y) = \sum_d \Pr(A \Rightarrow^d x/y) \tag{2.3}$$

Hence, the probability of a bilingual string (x, y) given a SITG is the sum of the probabilities of all derivations that yield the string from the initial symbol.

$$\Pr(s, t) = \sum_d \Pr(d) \ , \ d \in S \Rightarrow^* s/t \tag{2.4}$$

However, in some cases, it can be assumed that the probability of a bilingual string is similar to the probability of its most likely derivation.

$$\Pr(s, t) \approx \max_d \Pr(d) \ , \ d \in S \Rightarrow^* s/t \tag{2.5}$$

2.1.2 Stochastic Billingual Parsing with ITG

Inside Algorithm

Once we have described the formalism, we must propose an algorithm to compute the probability of a bilingual string (x, y). This algorithm is similar to the Viterbi SITG parsing algorithm proposed in [Wu, 1997]. This probability is computed using a dynamic programming algorithm similar to the *inside* algorithm for CFG [Hopcroft and Ullman, 1979]. For simplicity, we denote $\Pr(A \Rightarrow^* x_i^j / y_k^l)$ as $\delta_{i,j,k,l,A}$ and the lengths of x and y are defined as $|x| := T$ and $|y| := V$. Then, the probability of the bilingual string (x, y), $\Pr(x, y)$, is denoted by $\delta_{0,T,0,V,S}$. Figure 2.1 shows the *inside* algorithm for SITG.

Viterbi Algorithm

A similar algorithm can be used to compute the most likely derivation and its probability instead of the total probability. The resulting algorithm is the same that the one in Figure 2.1 but using maximizations instead of sums in expressions (2.6), (2.7) and (2.8). Such algorithm is described in [Wu, 1997] together with a method

1. **Initialization**

$$\delta_{t-1,t,v-1,v,A} = \Pr(A \to x_t/y_v) \text{ for } 1 \le t \le T \text{ and } 1 \le v \le V$$

$$\delta_{t-1,t,v,v,A} = \Pr(A \to x_t/\epsilon) \text{ for } 1 \le t \le T \text{ and } 0 \le v \le V$$

$$\delta_{t,t,v-1,v,A} = \Pr(A \to \epsilon/y_v) \text{ for } 0 \le t \le T \text{ and } 1 \le v \le V$$

2. **Recursion**

For all A, s, t, u, v such that $\begin{cases} A \in \mathcal{N} \\ 0 \le s \le t \le T \\ 0 \le u \le v \le V \\ t - s + v - u > 2 \end{cases}$

$$\delta_{s,t,u,v,A} = \delta^{[]}_{s,t,u,v,A} + \delta^{\langle\rangle}_{s,t,u,v,A} \tag{2.6}$$

where

$$\delta^{[]}_{s,t,u,v,A} = \sum_{\substack{B \in \mathcal{N} \\ C \in \mathcal{N} \\ s \le s' \le t \\ u \le u' \le v \\ (s'-s)(t-s')+ \\ +(u'-u)(v-u') \ne 0}} \Pr(A \to [BC]) \cdot \delta_{s,s',u,u',B} \cdot \delta_{s',t,u',v,C} \tag{2.7}$$

$$\delta^{\langle\rangle}_{s,t,u,v,A} = \sum_{\substack{B \in \mathcal{N} \\ C \in \mathcal{N} \\ s \le s' \le t \\ u \le u' \le v \\ (s'-s)(t-S)+ \\ +(u'-u)(v-u') \ne 0}} \Pr(A \to \langle BC\rangle) \cdot \delta_{s,s',u',v,B} \cdot \delta_{s',t,u,u',C} \tag{2.8}$$

Figure 2.1: Inside algorithm for SITG.

to obtain the most likely derivation, also called *Viterbi derivation*, from the dynamic probramming chart. The probability of the Viterbi derivation is a good approximation of the total probability when

$$\Pr(A \Rightarrow^* x/y) \approx \max_d \Pr(A \Rightarrow^d x/y) \tag{2.9}$$

Outside Algorithm

Another possible algorithm in order to compute probabilities of strings is the outside algorithm. Whereas inside algorithm follows a bottom-up algorithm, outside algorithm uses a top-down approach. Thus, we define the probability $\beta_{i,j,k,l,A}$, called outside probability, as the probability of the derivation $S \Rightarrow x_0^i A x_j^{|x|}/y_0^k A y_l^{|y|}$. Then, the probability of the complete bilingual string (x/y) is $\beta_{i,i,k,k,A}$. Figure 2.2 show the outside algorithm for SITG. Again, we take T and V as the lenghts of x and y respectively, and δ correspond to the estimations of the inside described above.

1. **Initialization**

$$\beta_{0,T,0,V,S} = 1$$
$$\beta_{0,T,0,V,A} = 0 \quad \forall A \neq S$$

2. **Recursion**

For all A, s, t, u, v such that $\begin{cases} A \in \mathcal{N} \\ 0 \leq s \leq t \leq T \\ 0 \leq u \leq v \leq V \end{cases}$

$$\beta_{s,t,u,v,A} = \beta^{[1]}_{s,t,u,v,A} + \beta^{[1]}_{s,t,u,v,A} + \beta^{\langle 2 \rangle}_{s,t,u,v,A} + \beta^{\langle 2 \rangle}_{s,t,u,v,A} \tag{2.10}$$

where

$$\beta^{[1]}_{s,t,u,v,A} = \sum_{\substack{B \in \mathcal{N} \\ C \in \mathcal{N} \\ t \leq s' \leq T \\ v \leq u' \leq V \\ s'-t+u'-v \neq 0}} \Pr(B \to [AC]) \cdot \delta_{t,s',v,u',C} \cdot \beta_{s,s',u,u',B} \tag{2.11}$$

$$\beta^{[2]}_{s,t,u,v,A} = \sum_{\substack{B \in \mathcal{N} \\ C \in \mathcal{N} \\ 0 \leq s' \leq s \\ 0 \leq u' \leq u \\ s-s'+u-u' \neq 0}} \Pr(B \to [CA]) \cdot \delta_{s',s,u',u,C} \cdot \beta_{s',t,u',v,B} \tag{2.12}$$

$$\beta^{\langle 1 \rangle}_{s,t,u,v,A} = \sum_{\substack{B \in \mathcal{N} \\ C \in \mathcal{N} \\ t \leq s' \leq T \\ 0 \leq u' \leq u \\ s'-t+u-u' \neq 0}} \Pr(B \to \langle AC \rangle) \cdot \delta_{t,s',u',u,C} \cdot \beta_{s,s',u',v,B} \tag{2.13}$$

$$\beta^{\langle 2 \rangle}_{s,t,u,v,A} = \sum_{\substack{B \in \mathcal{N} \\ C \in \mathcal{N} \\ 0 \leq s' \leq s \\ v \leq u' \leq V \\ s-s'+u'-v \neq 0}} \Pr(B \to \langle CA \rangle) \cdot \delta_{s',s,v,u',C} \cdot \beta_{s',t,u,u',B} \tag{2.14}$$

Figure 2.2: Outside algorithm for SITG.

Complexity

The dynamic programming algorithm works in the three estimations described above by analyzing each span only once and storing its value in a chart for a future use. But there is a set of $O(|s|^2|t|^2|N|)$ spans to be solved and the cost of the analysis of each of them is $O(|s||t||G|)$, being $|s|$ and $|t|$ the lengths of the source and target sentence respectively, $|N|$ the number of non-terminal symbols of the ITG and $|G|$ the size of the grammar. Assuming the length of input and output proportional (n), the number of problems is $O(n^4 \cdot |G|)$ and the cost of each one, $O(n^2 \cdot |G|)$. Hence, the total time of computing it is $O(n^6 \cdot |G|)$.

Viterbi Algorithm with Bracketing Information

Thus, the cost of those algorithms is, thought polynomial, still very high. It is too slow when using large data sets with large sentences. However, in [Sánchez and Benedí, 2006] the authors suggest a new algorithm to compute the probability of the Viterbi when both parts of the bilingual string are bracketed. A string is bracketed if it is annotated with parenthesis that mark constituent frontiers. We represent a bracketed bilingual string with the tuple (x, B_x, y, B_y) where x and y are the strings, B_x is the bracketing of x and B_y is the bracketing of y. A derivation of x/y is compatible with B_x and B_y when all the spans defined by it are compatible with B_x and B_y. This compatibility is defined by the function:

$$c(i,j,k,l) = \begin{cases} 1 & \text{when } (i,j) \text{ does not overlap any } b \in B_x \text{ and} \\ & \quad (k,l) \text{ does not overlap any } b \in B_y \\ 0 & \text{otherwise} \end{cases} \tag{2.15}$$

This function prunes all the derivations whose parsing is not compatible with the bracketing of the input string.

The dynamic programing algorithm can be modified in order to explore only those spans compatible with the bracketing. Thus, complexity of this algorithm is $O(n^2 \cdot |G|)$ when both sentences are fully bracketed. This allows us to work with real tasks with longer strings.

2.1.3 Phrasal ITG

There are two major problems in the use of ITG:

1. As stated by Wu, ITGs cannot represent all possible permutations of words that may occur during translation. However, in [Cherry and Lin, 2006], the authors state that only a small percentage of human translations cannot be represented by an ITG transduction .

1. **Initialization**

$$\delta_{t-1,t,v-1,v,A} = \Pr(A \to x_t/y_v) \text{ for } 1 \leq t \leq T \text{ and } 1 \leq v \leq V$$

$$\delta_{t-1,t,v,v,A} = \Pr(A \to x_t/\epsilon) \text{ for } 1 \leq t \leq T \text{ and } 0 \leq v \leq V$$

$$\delta_{t,t,v-1,v,A} = \Pr(A \to \epsilon/y_v) \text{ for } 0 \leq t \leq T \text{ and } 1 \leq v \leq V$$

2. **Recursion**

For all A, s, t, u, v such that $\begin{cases} A \in \mathcal{N} \\ 0 \leq s \leq t \leq T \\ 0 \leq u \leq v \leq V \\ t - s + v - u > 2 \end{cases}$

$$\delta_{s,t,u,v,A} = c(s+1,t,u+1,v)\max(\delta^{[]}_{s,t,u,v,A}, \delta^{\langle\rangle}_{s,t,u,v,A}) \tag{2.16}$$

where

$$\delta^{[]}_{s,t,u,v,A} = \max_{\substack{B \in \mathcal{N} \\ C \in \mathcal{N} \\ s \leq s' \leq t \\ u \leq u' \leq v \\ (s'-s)(t-s')+ \\ +(u'-u)(v-u') \neq 0}} \Pr(A \to [BC]) \cdot \delta_{s,s',u,u',B} \cdot \delta_{s',t,u',v,C} \tag{2.17}$$

$$\delta^{\langle\rangle}_{s,t,u,v,A} = \max_{\substack{B \in \mathcal{N} \\ C \in \mathcal{N} \\ s \leq s' \leq t \\ u \leq u' \leq v \\ (s'-s)(t-s')+ \\ +(u'-u)(v-u') \neq 0}} \Pr(A \to \langle BC\rangle) \cdot \delta_{s,s',u',v,B} \cdot \delta_{s',t,u,u',C} \tag{2.18}$$

Figure 2.3: Viterbi algorithm for SITG with bracketing.

2. This kind of grammars does not take into account direct multi-word (or phrasal) transduction. An ITG gets the transduction of a segment of words only by the combination of the transduction of its constituents.

In order to overcome, at least partially, such problems, we present a phrasal extension of ITG: the so called Phrasal ITG (PhITG). PhITG has been already used in the literature [Cherry and Lin, 2007][Zhang et al., 2008] but, as far as we know, does not exist a formal definition.

PhITG are very similar to ITG, the only difference is that the lexical rules can produce directly strings instead of a single word in each of the languages. This changes the definition of lexical rules. A lexical rule have now the form $A \to x/y$ with $x \in (\Sigma)^*$ and $y \in (\Delta)^*$. Figure 2.1.3 shows two different PhITG-parsings for a given pair of sentences in English and Spanish. In the part **b)** the non-terminal NP produce a *phrase* translation.

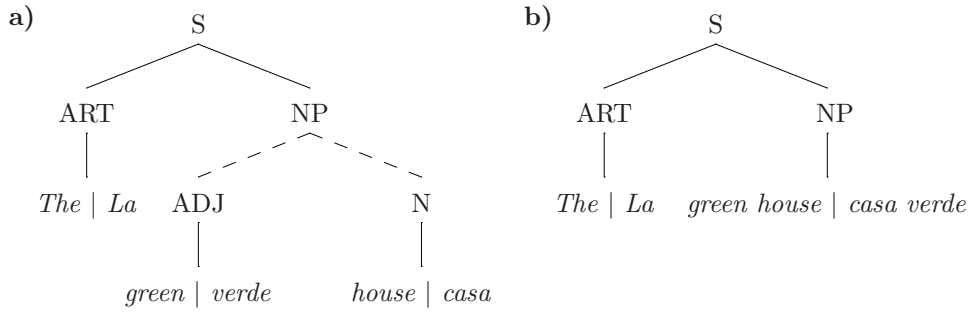


Figure 2.4: Two different Phrasal ITG parsings for a pair of sentences English-Spanish.

Analogously to the definition of SITG, we can define the Stochastic PhITG formalism. The constraint for the assignation of probabilities is the same that for SITG, showed in Equation (2.1). The inside algorithm for SITG can be adapted for SPhITG by changing Equation (2.6) for

$$\delta_{s,t,u,v,A} = \delta^{[]}_{s,t,u,v,A} + \delta^{\langle\rangle}_{s,t,u,v,A} + \Pr(A \rightarrow x_s^t / t_u^v) \tag{2.19}$$

In the case of the Viterbi algorithm with bracketing information, we change Equation (2.16) for the equation:

$$\delta_{s,t,u,v,A} = \max(\delta^{[]}_{s,t,u,v,A}, \delta^{\langle\rangle}_{s,t,u,v,A}, \Pr(A \rightarrow x_s^t / t_u^v)) \tag{2.20}$$

Note that the complexity of both algorithms is the same we explained in Subsection 2.1.2: $O(n^6 \cdot |G|)$ for the inside/viterbi algorithm and $O(n^2 \cdot |G|)$ for the algorithm that uses bracketing when the bilingual string is fully bracketed.

2.1.4 Collapse of a SITG into a SCFG

The parsing algorithms described above find the probability (or the optimal derivation) of a bilingual string given a SITG or SPhITG. The problem that appear when we want to define the translation model is that we don't have the bilingual string, but only its source-language part and our aim is to find the target-language part that maximizes the parsing probability.

The decoder explained in Chapter 5 uses a SCFG collapsed from a SITG in order to parse the input source-language sentence and then transform (see next section) the SCFG trees into SITG derivations.

Therefore, we must define a method to get a SCFG from a SITG. Using this CFG we will parse the input source-language string. We say that a SITG $\{\mathcal{N}, \Sigma, \Delta, S, \mathcal{R}\}$ *collapse* into a SCFG $\{\mathcal{N}, \Sigma, S, \mathcal{R}'\}$ when $\forall r' \in \mathcal{R}'$:

$$Pr(r') = \begin{cases} \Pr(A \to [BC]) + \Pr(A \to \langle BC \rangle) & \text{if } r' = A \to BC \text{ and} \\ & \quad A \to [BC], A \to \langle BC \rangle \in \mathcal{R} \\ \sum_{y \in \Delta} \Pr(A \to x/y) & \text{if } r' = A \to x \text{ and } A \to x/y \in \mathcal{R} \end{cases}$$

It can be easily proved that the grammar resulting from the collapse of a SITG is a CFG. The same problem can be done with a SPhITG but the probabilities of the rules of the resulting grammar must be renormalized after the collapse. Figure 2.5 shows the SCFG resulting from the collapse of a SPhITG.

$$\left.\begin{array}{l} \Pr(A \to [BA]) = 0.4 \\ \Pr(A \to [CD]) = 0.3 \\ \Pr(A \to \langle CD \rangle) = 0.1 \\ \Pr(A \to \text{big dog/perro grande}) = 0.1 \\ \Pr(A \to \text{big dog/gran perro}) = 0.1 \\ \Pr(B \to \text{a/un}) = 1 \\ \Pr(C \to \text{big/grande}) = 0.7 \\ \Pr(C \to \text{big/gran}) = 0.3 \\ \Pr(D \to \text{dog/perro}) = 1 \end{array}\right\} \text{collapses in} \left\{\begin{array}{l} \Pr(A \to BA) = 0.5 \\ \Pr(A \to CD) = 0.5 \\ \Pr(B \to \text{a}) = 1 \\ \Pr(C \to \text{big}) = 1 \\ \Pr(D \to \text{dog}) = 1 \end{array}\right.$$

Figure 2.5: Collapse of a SPhITG into a SCFG.

2.1.5 Transformation of a SCFG derivation into a SITG derivation

In the translation model, once we have collapsed a SITG into a SCFG, we can get the monolingual parsing of the input source language sentence. However, several diferent translations can be applied over a input parse tree. We define the concept of *transformation* in order to formalize that multiple choice concept.

Being d' a SCFG derivation resulting from the collapse of an SITG $G = \{\mathcal{N}, \Sigma, \Delta, S, \mathcal{R}\}$, we say that d, derivation of G, is a *transformation* of d', when each of the ordered rules r'_i of d' is substituted in d by r_i such that:

$$r_i = \left\{\begin{array}{c} A \to [BC] \\ \text{or} \\ A \to \langle BC \rangle \end{array}\right\} \text{ when } r'_i = A \to BC \text{ and } \begin{array}{c} A \to [BC] \\ A \to \langle BC \rangle \end{array} \in \mathcal{R} \qquad (2.21)$$

$$r_i = \left\{ \begin{array}{c} A \rightarrow x/y^1 \\ \text{or} \\ \vdots \\ \text{or} \\ A \rightarrow x/y^K \end{array} \right\} \text{ when } r'_i = A \rightarrow x \text{ and } \begin{array}{c} A \rightarrow x/y^1 \\ \vdots \\ A \rightarrow x/y^K \end{array} \in \mathcal{R} \tag{2.22}$$

being $y^1 ... y^K \in \{\Delta \cup \epsilon\}$.

A similar procedure can be carried out with SPhITG instead of SITG but Equation (2.21) becomes then:

$$r_i = \left\{ \begin{array}{c} A \rightarrow [BC] \\ \text{or} \\ A \rightarrow \langle BC \rangle \\ \text{or} \\ A \rightarrow x_{i..j}/y^1 \\ \text{or} \\ \vdots \\ \text{or} \\ A \rightarrow x/y^K \end{array} \right\} \text{ when } r'_i = A \rightarrow BC \text{ and } \begin{array}{c} A \rightarrow [BC] \\ A \rightarrow \langle BC \rangle \\ A \rightarrow x_{i..j}/y^1 \\ \vdots \\ A \rightarrow x/y^K \end{array} \in \mathcal{R} \tag{2.23}$$

being $y^1 ... y^K \in (\Delta)^*$ and the rule r'_i covers the segment $x_{i..j}$ in d'.

It must be noted that one CFG derivation d' can produce several transformations d_i that can yield different bitexts. Figure 2.6 shows all the transformations of a derivation d' using the SCFG and the SPhITG of Figure 2.5.

We say that two transformations d_1 and d_2 (derivations of a SITG) are *equivalent* when both yields the same bitext. For instance, in Figure 2.6, d_1 and d_5 are equivalent because both yields the bitext (a big dog / un perro grande),i.e.

$$A \Rightarrow^{d_1} \text{a big dog / un perro grande} \equiv A \Rightarrow^{d_5} \text{a big dog / un perro grande}$$

For simplicity, from now on, we will write $\tau(d')$ to refer to the set of all the transformations of d'.

2.2 Translation Model

Once all the formalism have been presented, we can define the translation model. Then, given a SPhITG and given an input source-language sentence s, we want to find the target-language sentence t^* such that maximizes the joint probability:

$$t^* = \underset{t}{\operatorname{argmax}} \Pr(s,t) \tag{2.24}$$

The parsing algorithms for SPhITG need both parts, source-language and target language, of the bilingual string. For some very small tasks we could explore all the possibilities over t, get their probabilities and find the most likely t. However, in real tasks, the search space over t is too large and such exhaustive search is not possible. In such cases, we only can do an approximate search by collapsing the SPhITG into a SCFG and parsing the input source-language string.

$$t^* = \underset{t}{\operatorname{argmax}} \Pr(s,t) = \underset{t}{\operatorname{argmax}} \Pr(t|s) \Pr(s) = \underset{t}{\operatorname{argmax}} \Pr(t|s) \tag{2.25}$$

Now we marginalize over all the derivations δ of the collapsed SCFG that yields the input source-language sentence ($S \Rightarrow^{delta} s$):

$$t^* = \underset{t}{\operatorname{argmax}} \sum_{d'} \Pr(t,\delta|s) = \underset{t}{\operatorname{argmax}} \sum_{d'} \Pr(t|\delta,s) \Pr(\delta|s) \tag{2.26}$$

Now, instead of taking the sum over all the possible derivations, we only pick the most likely derivation δ^* and divide the search in two steps: obtaining the most probable derivation from the source-language sentence (parsing), and obtaining a *translation* from δ^* and s :

$$\delta^* = \underset{\delta}{\operatorname{argmax}} \Pr(\delta|s) \quad , \forall \delta | S \Rightarrow^{\delta} s \tag{2.27}$$

$$t^* = \underset{t}{\operatorname{argmax}} \Pr(t|\delta^*, s) \tag{2.28}$$

Parsing is a well-studied problem, so we can focus on the translation or *tree-to-string* part. Now we marginalize over all the possible transformations , Section 2.1.5, of δ^* ($\tau(\delta^*)$) to the original SPhITG:

$$\Pr(t|\delta^*, s) = \sum_{d \in \tau(\delta^*)} \Pr(d, t|delta^*, s) \tag{2.29}$$

In this point we can continue in two different directions: we can take the sum just as it is or we can assume that it is similar to the probability of the most likely derivation:

$$\Pr(t|\delta^*, s) \begin{cases} = \sum_{d \in \tau(\delta^*)} \Pr(d, t|\delta^*, s) \\ \\ \approx \max_{d \in \tau(\delta^*)} \Pr(d, t|\delta^*, s) \end{cases} \tag{2.30}$$

This decision makes no difference for the later development and we have used both approaches in the experimentation. From now on we use the first equation in (2.30) but a similar development could be done using the second one.

Hence, we must give a model for the probability of a target sentence and a derivation given a parse tree. We have merged several models using a log-linear combination as it is made in [Och and Ney, 2002].

$$\Pr(d,t|\delta^*,s) = \frac{\exp[\sum_i \lambda_i h_i(t,d,\delta^*,s)]}{\sum_{d',t'} \exp[\sum_i \lambda_i h_i(t',d',\delta^*,s)]} \tag{2.31}$$

Summing up, we have:

$$t^* = \underset{t}{\operatorname{argmax}} \sum_{d\in\tau(\delta^*)} \frac{\exp[\sum_i \lambda_i h_i(t,d,\delta^*,s)]}{\sum_{t',d'} \exp[\sum_i \lambda_i h_i(t',d',\delta^*,s)]} = \underset{t}{\operatorname{argmax}} \sum_{d\in\tau(\delta^*)} \sum_i \lambda_i h_i(t,d,\delta^*,s) \tag{2.32}$$

That means that we must find the target language sentence t^* that maximizes the sum over all the equivalent transformations of δ^* that yields the bitext s/t. If we take the second option in Equation (2.30), we arrive to the expression:

$$t^* = \underset{t}{\operatorname{argmax}} \max_{d\in\tau(\delta^*)} \frac{\exp[\sum_i \lambda_i h_i(t,d,\delta^*,s)]}{\sum_{d',t'} \exp[\sum_i \lambda_i h_i(t',d',\delta^*,s)]} = \underset{t}{\operatorname{argmax}} \max_{d\in\tau(\delta^*)} \sum_i \lambda_i h_i(t,d,\delta^*,s) \tag{2.33}$$

On the contrary, if we choose the second option, we must maximize instead of summing over all the transformations.

In both cases, we have a loglinear combination of models, also named features, $h_i(t,d,\delta^*,s)$. Such kind of combinations has been widely used in phrase-based MT. As we said in Section 1.2.1 there are some models that has been already used in the literature. However, our approach allows us to use, besides the usual models, other models derived from the syntactic information of $\delta*$ and d. Next, we show some of the features we have used in this work:

Direct Translation Probability: Probability of the target sentence given the source sentence: $h_1 = Pr(t|s)$

Inverse Translation Probability: Probability of the source sentence given the target sentence: $h_2 = Pr(s|t)$

Lexical Direct Probability: Probability of the source sentence given the target sentence using an IBM1 translation model: $h_3 = Pr_L(t|s)$

Inverse Direct Probability: Probability of the source sentence given the target sentence using an IBM1 translation model: $h_4 = Pr_L(s|t)$

N-gram Language Model: Probability of the source sentence given the target sentence: $h_5 = Pr_{LM}(t)$

Syntactic Probability: Probability of the rules of the derivation: $h_6 = Pr(d)$. This feature models the use of inverse and direct productions and with that, the order of the target sentence.

Word Penalty Factor: This feature is used to model the length of the output: $h_7 = exp(|t|)$, being $|t|$ the number of words of the target sentence.

Phrase Penalty Factor: This feature is used to control the number of phrases used in the translation.

Besides all the models described above, we plan to use in the future some other such as syntactical probabilities described in [Sanchis-Trilles and Sánchez, 2008] or complex reordering probabilities based on maximum entropy models or support vector machines.

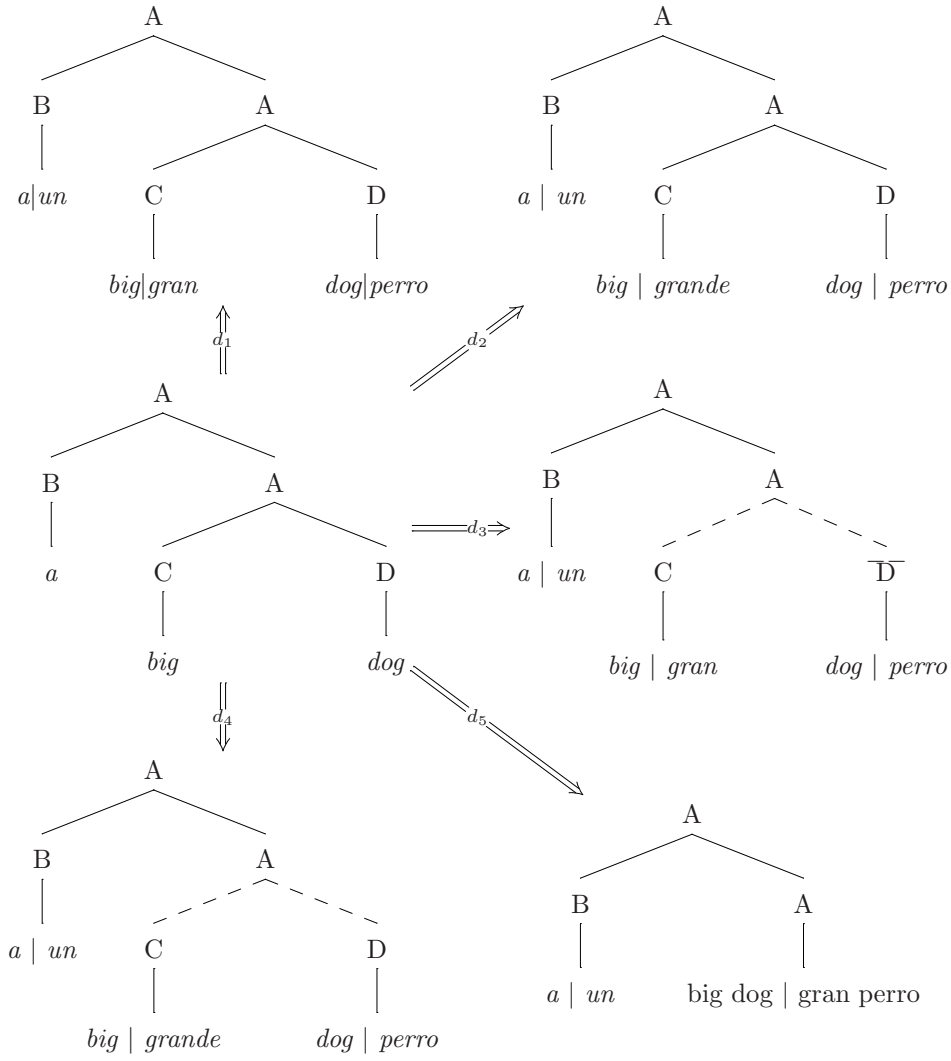


Figure 2.6: Transformations for a CFG derivation.

Chapter 3

Inference of Stochastic Inversion Transduction Grammars

" Grammar is a piano I play by ear.
All I know about grammar is its power."
Joan Didion

Grammar inference is the process of inducing a formal grammar from a text. That is obtaining a model that accounts the characteristics of the text. In the case of stochastic grammars, the inference have two main parts: learning which rules *need* the grammar (inference of the structure) and the estimation of the probabilities (stochastic estimation).

We use, in our approach, Stochastic Inversion Transduction Grammars, so we must infer the SITG (structure and probabilities) from a bilingual corpus. In this chapter, we explain three different methods to learn the structure (and initial probabilities) of a SITG. Two of such methods use, besides the alignments of the bilingual corpus, a monolingual parse in order to give a linguistic motivation to the grammar. The grammars obtained by the use of these methods will be the initial models for the estimation of probabilities. In addition, we explain two different algorithms for the stochastic estimation: Viterbi-like and Inside-Outside estimations.

3.1 Inference of the Structure of SITG

We propose, in this section two methods in order to infer the structure for SITG. Such methods estimate, also, initial probabilities for the rules inferred. Thus, the resulting grammars can be used as a initial model in the stochastic estimation of the probabilities described in Section 3.2. The first method was proposed in the article [Sánchez and Benedí, 2006] and gets a non linguistically motivated SITG using the alignments of the bilingual corpus. The other algorithm incorporate linguistic syntactic information by the use of the monolingual linguistic parse tree of the input.

In both cases, SITG obtained are restricted only to the words and alignments seen in the training corpus. For that reason, the sentences that contain out of vocabulary words or alignments that have not been seen in the training corpus cannot be parsed. In order to smooth the grammars and get a full coverage of all possible sentences, we have added two new sets of rules: $A \rightarrow s \mid \epsilon$ and $A \rightarrow \epsilon \mid t$ that have not been inferred from the alignment, with a low probability; the problem with the out of vocabulary words is solved by the incorporation of three new lexical rules ($A \rightarrow$ UNK/UNK , $A \rightarrow$ UNK/ϵ and $A \rightarrow \epsilon$/UNK). During the parse of a sentence, if we find an out-of-vocabulary word we substitute it by the word UNK and one of these productions is used.

3.1.1 Ergodic SITG

This method divide the inference of the rules in two parts: inference of lexical rules and inference of syntactic rules.

The probability of a lexical rule is computed using the alignments. From these alignments we obtain the probability $\Pr(a|b)$ where $a \in \{\Sigma \cup \{\epsilon\}\}$ and $b \in \{\Delta \cup \{\epsilon\}\}$. Hence, we can use $\Pr(a|b)$ to compose the rule $A \rightarrow a/b$, being $A \in N$ with $\Pr(A \rightarrow a/b) = \Pr(a|b)$. In order to get a less complex model, and following [Wu, 1997], only one non-terminal symbol is used to produce all the lexical rules.

In order to obtain the probabilities of the syntactic rules we add the rules to obtain an ergodic grammar, i.e. all the possible syntactic rules (direct and inverse) that can be formed using all the non-terminals of the grammar. Then a similar random probability is assigned to each syntactic rule.

Finally we smooth the grammar, as have been proposed above, and normalize the probabilities of all the rules to fulfil constraint (2.1).

3.1.2 Adding Linguistic Information to a SITG

Adding non-terminals to the grammar, increases its complexity and hence, its expressive power. However an increasing number of non-terminals increase also the temporal

cost of the parsing algorithm. With the ergodic model, with N non-terminals, the number of syntactic rules is $2N^3$. Therefore, with a high number of non-terminals the number of resulting syntactic rules slow highly the parsing algorithms. In addition, the ergodic SITG does not take benefit of the usual linguistically motivated reorderings. We can add syntactic motivation to the SITG by using the linguistically motivated monolingual parse trees of the source language part of the corpus[1].

Figure 3.1 shows an example of this first method for a given bilingual sentence and the algorithm is also illustrated in the pseudo code of the Algorithm 1. First, we get the monolingual parsing of the source language part of a sentence (**a**). The non-terminals of this tree represent some linguistic constituents of the sentence; for instance P (Predicate), AT (Attribute)...

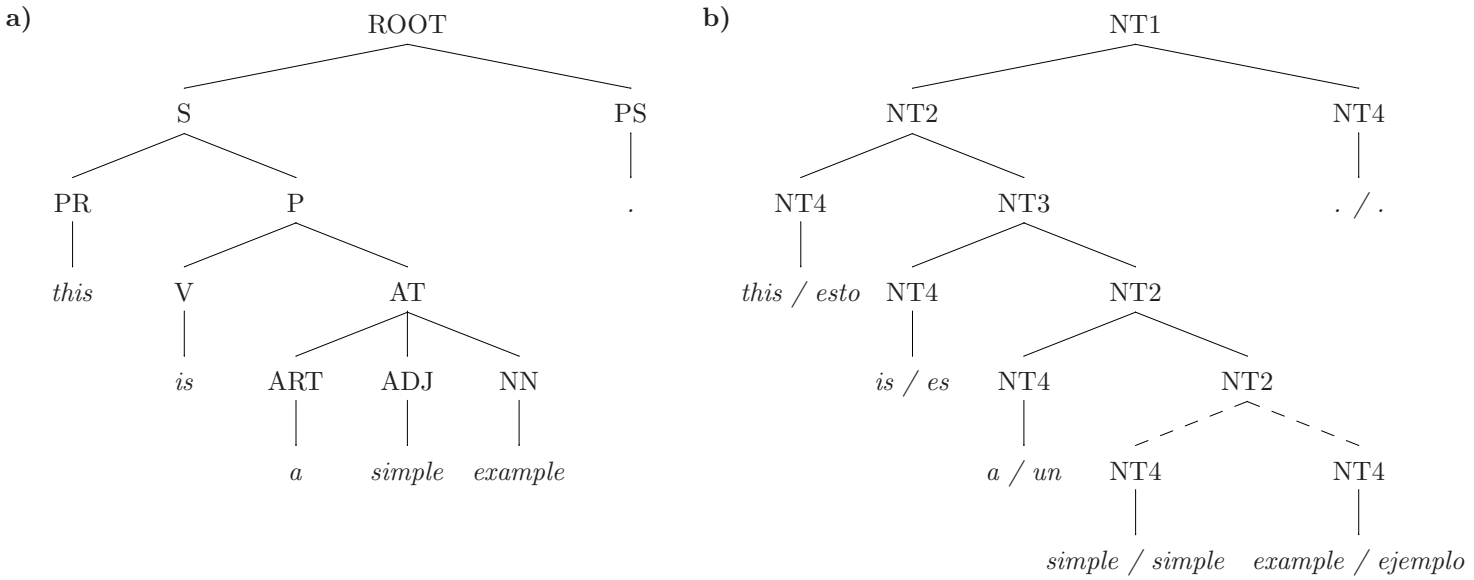
Then, we extract the bracketing information of the monolingual parse tree and use it in order to get a bilingual parse tree (**b**) using a SITG with the algorithm described in Figure 2.3. Since we have used the bracketing given by the monolingual tree, for each node of the monolingual tree exists a node in the bilingual tree that covers exactly the same part of the sentence. Note that this relation is not symmetric, because there can be nodes in the bilingual tree that do not have an equivalent in the monolingual tree, the syntactic monolingual tree is not necessarily a binary tree. For instance, in the figure, the node with the non-terminal NT2 that yields the string "*simple/simple example/ejemplo*" has not an equivalent in the monolingual tree. In order to obtain a complete equivalence between both trees we binarize the parse tree using the information of the SITG tree.

Now, we can associate the reordering information of each SITG tree node with its equivalent in the parse tree (line 7 in Algorithm 1). We do the same with the lexical productions of the SITG tree. Thus, we have a SITG tree with linguistic syntactic information. This process is repeated for all the bilingual sentences of the corpus. Finally the probabilities of the rules are computed by counting over all the resulting trees and normalizing.

3.2 Stochastic estimation for SITG

Once the structure of the grammar and the initial probabilities of its rules is inferred by means of one of the methods proposed in previous section, we must re-estimate the probabilities given a bilingual corpus. The basic assumption is that a good grammar is one that makes the sentences in the training corpus likely to occur. This process consist then, in finding the probabilities of the rules that maximizes likelihood of the corpus.

[1] From now on, monolingual parse trees



Rules obtained:

ROOT → [S PS]	S → [PR P]	PRD → [V AT]	AT → [ART ADJ__NN]
ADJ__NN → ⟨ ADJ NN ⟩	PR → *this / esto*	V → *is / es*	ART → *a / un*
ADJ → *simple / simple*	NN → *example / ejemplo*	PS → . / .	

Figure 3.1: Adding linguistic information to a SITG.

Algorithm 1 InferenceOfLinguisticSITG(bCorpus,ITG,CFG)

```
1: newITG ← ∅
2: for all bSentence ∈ bCorpus do
3:    synTree ← monolingualParse(CFG,bSentence.input)
4:    bracketedSent ← obtainBracketing(synTree)
5:    itgTree ← bilingualParse(ITG,bracketedSent)
6:    binarize(synTree)
7:    combTree ← combineTrees(parseTree,itgTree)
8:    addRules(newITG,combTree)
9: end for
10: countAndNormalize(newITG)
11: return newITG
```

In this section, we propose two SITG re-estimation methods similar to the classical Viterbi and Inside-Outside for SCFG.

3.2.1 Viterbi-like Estimation

To determine the probabilities of the rules we compute:

$$\Pr'(A \to \alpha) = \frac{\mathrm{C}(A \to \alpha)}{\sum_{\gamma} \mathrm{C}(A \to \gamma)} \qquad (A \to \alpha), (A \to \gamma) \in \mathcal{R}, \tag{3.1}$$

where $\mathrm{C}(\cdot)$ is the number of times that a rule is used.

When we have parsed corpora available, this estimation can be done directly. In this case we only have the bilingual corpus, not its parse trees, but we can use the Viterbi parsing algorithm described in Chapter 2 to parse the corpus.

The iterative algorithm of Viterbi re-estimation is presented in Algorithm 2. Each sentence of the corpus is parsed and the probabilities of the SITG rules are updated by counting and normalize over the number of occurrences in the parse trees. This process is repeated until the differences in the SITG of two iterations are small.

Algorithm 2 ViterbiReestimation(SITG,bCorpus)

```
1: repeat
2:    RuleSet← ∅
3:    for all bSentence ∈ bCorpus do
4:       SITGparse ← ViterbiParse(bSentence,SITG)
5:       addRules(RuleSet,SITGparse)
6:    end forSITG←countAndNormalize(RuleSet)
7: until Convergence
8: return SITG
```

The complexity of this algorithm is the same as the Viterbi parsing algorithm: $O(n^6 \cdot |G|)$. However, if we have a fully bracketed corpus, we can use the version that uses bracketing information in order to decrease the complexity until $O(n^2 \cdot |G|)$.

3.2.2 Inside-Outside Estimation

In the Inside-Outside method, we try to determine the probabilities of the rules but we only see directly the probability of the whole sentence (using Inside, δ, and Outside ,β, estimations, Section 2.1.2).

For brevity, we do not explain deeply the algorithm that is similar to the Inside-Outside algorithm for SCFG [Baker, 1979] but we give the re-estimation expressions for each bilingual sentence x/y such that $|x| = T$ and $|y| = V$:

$$\Pr'(A \to a/b) = \frac{\sum_{s=1}^{T}\sum_{u=1}^{V}\delta_{s-1,s,u-1,u,A}\Pr(x_s = a)\Pr(y_u = b)\beta_{s-1,s,u-1,u,A}}{\sum_{s=0}^{T}\sum_{t=s}^{T}\sum_{u=0}^{V}\sum_{v=u}^{V}\delta_{s,t,u,v,A}\beta_{s,t,u,v,A}} \quad (3.2)$$

$$\Pr'(A \to [BC]) =$$

$$\frac{\sum_{s=0}^{T-1}\sum_{t=s}^{T}\sum_{u=0}^{V-1}\sum_{v=u}^{V}\sum_{s'=s}^{t}\sum_{u'=u}^{v}\delta_{s,t,u,v,A}\Pr(A \to [BC])\beta_{s,s',u,u',B}\beta_{s',t,u',v,C}}{\sum_{s=0}^{T}\sum_{t=s}^{T}\sum_{u=0}^{V}\sum_{v=u}^{V}\delta_{s,t,u,v,A}\beta_{s,t,u,v,A}} \quad (3.3)$$

$$\Pr'(A \to \langle BC\rangle) =$$

$$\frac{\sum_{s=0}^{T-1}\sum_{t=s}^{T}\sum_{u=0}^{V-1}\sum_{v=u}^{V}\sum_{s'=s}^{t}\sum_{u'=u}^{v}\delta_{s,t,u,v,A}\Pr(A \to \langle BC\rangle)\beta_{s,s',u',v,B}\beta_{s',t,u,u',C}}{\sum_{s=0}^{T}\sum_{t=s}^{T}\sum_{u=0}^{V}\sum_{v=u}^{V}\delta_{s,t,u,v,A}\beta_{s,t,u,v,A}} \quad (3.4)$$

Even when we precompute the Inside and Outside estimation charts and store them for a direct access, the complexity of this algorithm is still $O(n^6 \cdot |G|)$. Such cost is excessive for real tasks.

Chapter 4

Obtaining Phrase Tables

"Learning is finding out what you already know.
Doing is demonstrating that you know it.
Teaching is reminding others that they know just as well as you."
Richard Bach

In the previous chapter some methods for inferring the stochastic inversion transduction grammars have been presented. In our model described in Chapter 2, the grammar drives the segmentation of the input sentence and the reordering of the translated phrases[1]. Hence, besides the grammar, it must be inferred a new structure in order to model the translation between source language phrases and target language phrases. Such structure usually takes the form of a probabilistic bilingual phrase dictionary called *phrase table*.

Most recent methods on extracting a phrase table from a bilingual corpus start with a word alignment. In this chapter, we explain the most commonly used of such methods, the one proposed in [Och and Ney, 2003]. However, there are other methods that infer the phrase tables using syntactic approaches. Due to its strong relation to the issues dealt in this thesis, we explain also the approach presented in [Sánchez and Benedí, 2006] that uses SITG in the construction of the phrase table.

[1] Such reordering is also driven by the n-gram language model.

4.1 Obtaining Word Phrases with SITG

This mthod was proposed in [Sánchez and Benedí, 2006]. The structured space of a stochastic synchronous grammar is a natural fit to phrase pair probability estimation. Due to the large search space, the estimation of such probabilities must be limited to a binary stochastic synchronous grammar, a SITG. In the paper, authors addressed the problem of obtaining translation tables from bilingual corpora by means of SITG. First, a SITG is obtained from the corpus using one of the methods described in Chapter 3. Then, each bilingual sentence is parsed using the SITG obtaining the most probable SITG tree.

Each of the nodes of the parse tree yields a bilingual phrase. All such bilingual phrases are introduced in the phrase table. Figure 4.1 shows an example of this kind of phrase extraction. The probabilities for the inverse and direct translation models are computed for each segment pair (s, t) according to the formulae:

$$\Pr(s|t) = \frac{C(s,t)}{\sum_{s'} C(s',t)} \qquad \Pr(t|s) = \frac{C(s,t)}{\sum_{t'} C(s,t')} \tag{4.1}$$

being $C(s, t)$ the number of times segments s and t were extracted together.

The probabilities for the lexical models can be computed using the IBM1 model over the extracted phrases. It must be noted that using this phrase extraction, we can compute also syntax based translation models like it is proposed in [Sanchis-Trilles and Sánchez, 2008].

4.2 Obtaining Word Phrases from Bilingual Alignments

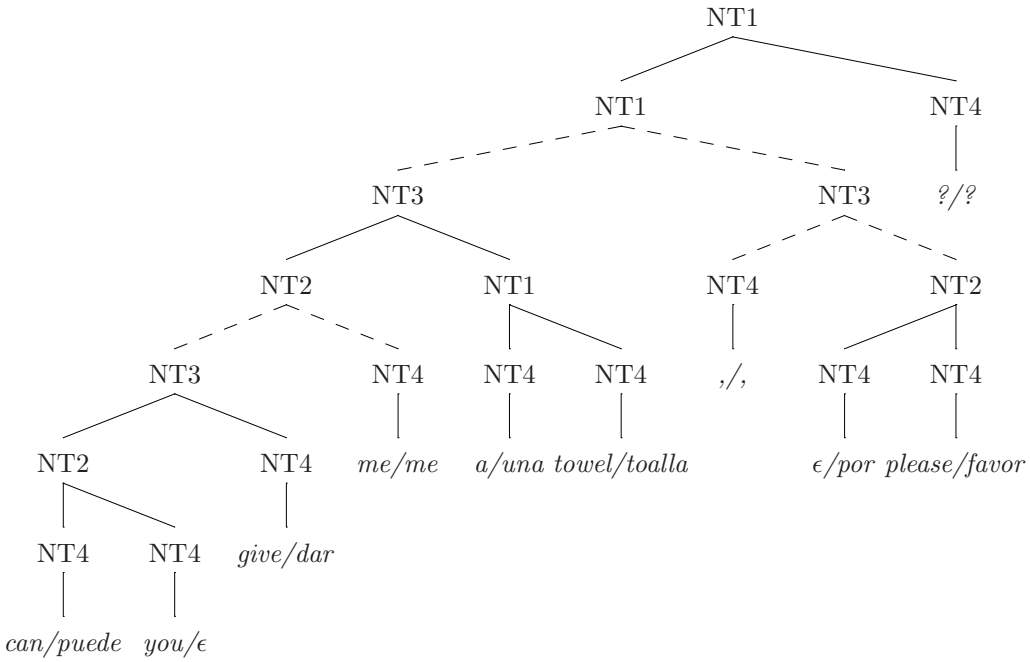
From all the methods that use bilingual alignments in order to obtain phrase tables, we explain the one proposed in [Och and Ney, 2003] because nowadays is the most used in phrase-based systems. In such article, the authors propose to use the intersection of the alignments in both directions as a initial point. To such alignment points, some other are added from the union of the initial alignments depending on a number of criteria:

- From which aligned has been extracted the new alignment post (Source-Target or Target-Source).
- Wether the new point has neighbours in the intersection or not.
- Wether the target word of the new point is unaligned so far or not.
- The lexical alignment probability for the potential point.

Bilingual Sentence:

can you give me a towel , please ? ||| por favor , me puede dar una toalla ?

ITG Parse Tree:



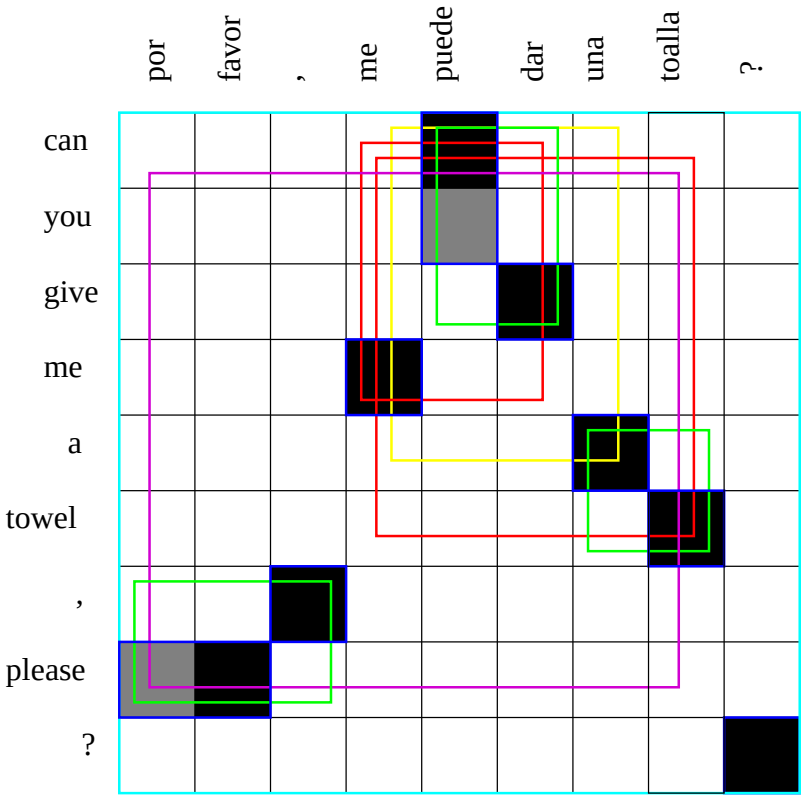
Phrases Obtained:

(can you give me a towel , please / por favor , me puede dar una toalla) , (can you give me a towel , please ? / por favor , me puede dar una toalla ?) , (can you give me a towel / me puede dar una toalla) , (, please / por favor ,) , (please / por favor) , (can / puede) , (you / ϵ) , (give / dar) , (me / me) , (a / una) (towel / toalla) , (ϵ / por) , (please / favor) , (a towel / una toalla) , (can you / puede) , (can you give / puede dar) , (, / ,) , (? / ?) , (can you give me / me puede dar)

Figure 4.1: Example of SITG phrase extraction.

Using the resulting alignment, all the consistent phrases are collected. We say that a bilingual phrase is consistent with an alignment when the words of the source part of the phrase are only aligned to words of the target part and vice versa. Figure 5.3 shows an example of the bilingual phrases extracted from a bilingual alignment.

The probabilities of the translation models are computed as have been explained in the previous section. In the limited example showed in Figures 4.1 and 4.2, the phrases obtained are almost the same. But usually, the number of phrases obtained with this method is higher.



Phrases Obtained:

(can you give me a towel , please ? | me puede dar una toalla , por favor ?) , (can you give me | me puede dar) , (can you give me a | me puede dar una) , (can you give me a towel | me puede dar una toalla) , (a towel | una toalla) , (can you give me a towel , please | me puede dar una toalla , por favor) , (me | me) , (a | una) , (towel | toalla) (, | ,) , (can you | puede) , (please | por favor) , (, please | please ,) , (? | ?)

Figure 4.2: Consistent phrases given a bilingual alignment.

Chapter 5

Decoding Process

The best structure will not guarantee the results nor yield.
But the wrong structure is a failure guarantee.
Peter Drucker

As have been stated in Chapter 2, our model divides the decoding process in two parts. The first one is the process of parsing the input sentence in order to obtain the most likely analysis tree given a formal grammar. The second part is a tree-to-string decoding that obtain the most likely target language sentence given the features of the translation model. The structure of the decoder is illustrated in Figure 5.1.

This separation allows us to use a monolingual parser integrated with the decoder or obtain the parse tree with an external parser. In this chapter, we present, first a new fast A^* parsing algorithm that can be use as a first step for the decoder. After that, we explain the core algorithm of the tree-to-string decoder giving some implementation and optimization details.

Stochastic parsing is the process of obtaining a parse tree for a given input sentence. This parse tree has been widely used in Machine Translation [Yamada and Knight, 2002, Charniak et al., 2003]. One of the formalisms that have been widely used for stochastic syntactic parsing are the Stochastic Context Free-Grammars (SCFGs) [Stolcke, 1995, Roark, 2001, Collins, 2003]. Most of the current syntactic parsing algorithms [Collins, 2003] that parse a sentence given a SCFG are based on the classical Cocke-Younger-Kasami [Hopcroft and Ullman, 1979] (CYK) and Earley [Earley, 1970] algo-

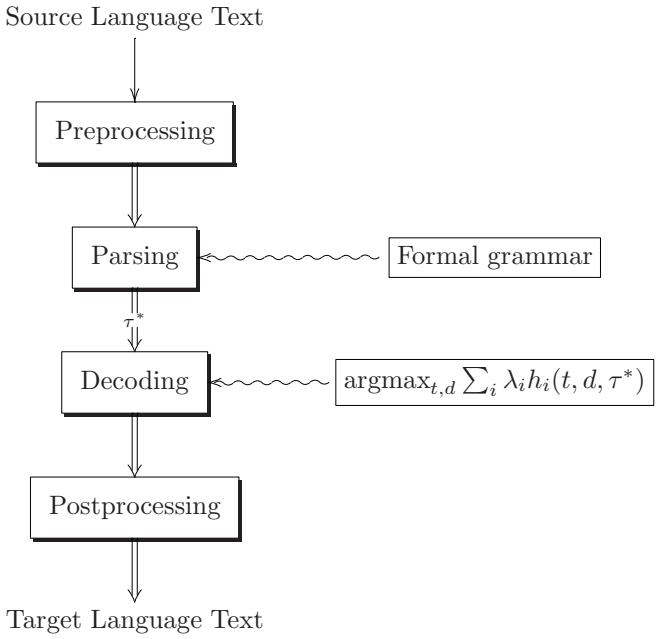


Figure 5.1: Architecture of our syntactical translation approach.

rithms. An important problem that is related to those algorithms is their cubic time complexity.

In [Gascó and Sánchez, 2007], an A* algorithm is presented to compute the exact, most probable parse tree (also known as Viterbi tree) of a string in the case of grammars with large vocabularies. This is the case of machine translation tasks. Several bounds are considered for the A* search and very good results are reported. Hence, we have used this parser to obtain the parse tree of the input sentence. In this section, we explain briefly the A* parsing algorithm presented in [Gascó and Sánchez, 2007] and its incorporation to the decoder.

5.1 Fast A* Parsing

The goal of stochastic syntactic parsing is to obtain a parse tree for a given input sentence. For this purpose, a stochastic grammatical model is used together with a parsing algorithm. SCFGs are grammatical models that are commonly used for stochastic parsing.

Some of the current syntactic parsing algorithms are based on the classical CYK and Earley algorithms. Both are dynamic programming algorithms. The cubic time

complexity of these algorithms restricts their use when dealing with wide-coverage grammars and long sentences. Therefore, a method for accelerating parse selection must be considered.

One of these methods consists of using a beam-search strategy together with a greedy algorithm [Roark, 2001]. However, the nature of these algorithms sometimes implies the loss of the most probable parse tree because the global optimum is not necessarily optimal at an intermediate stage. Therefore, the optimum at this stage could be pruned from the list of hypotheses.

Another possible method to accelerate parse selection consists of using a chart together with an agenda. The agenda is used to store the items to be processed. The items are chosen from the agenda according to some *figure of merit*. If the *figure of merit* is appropriately chosen, the number of items that are processed before obtaining a possible parse tree is notably lower than the maximum number of items that should be processed in an exhaustive search.

In [Klein and Manning, 2003a], an A* algorithm is presented to compute the exact most probable parsing of a string. In this parser, the search is driven by a function that guarantees that the best parse string is not lost. In that work, several bounds are proposed for the A* search, and experimental results are reported for delexicalized strings of the Penn treebank corpus. The space complexity is very large for some the bounds proposed in [Klein and Manning, 2003a], which hampers their application in tasks with large vocabularies.

Given the large space complexity associated to some of the bounds proposed in [Klein and Manning, 2003a], we propose new bounds for real tasks with large vocabularies. We studied experimentally which of them can be used in a real scenario. In addition, we propose using some bounds that can decrease the number of processed edges notably even though they do not guarantee the optimality of the solution.

5.1.1 Lexicalized Bounds for A* Parsing

An A* search is a guided search across the problem solution space which is a special case of *Best-first search*. This solution space is composed by *edges*. An edge represents a non-terminal symbol of the grammar over a span. The search procedure uses a function $f(e)$ in every edge e of the solution space in order to decide if it will be the next edge to be explored. The $f(e)$ function is the combination of two functions: $g(e)$ and $h(e)$. Function $g(e)$ is the probability of the edge e, that is, the probability of the most probable parse tree that starts from the non-terminal symbol of the edge. The function $h(e)$ is an estimation of the future probability of obtaining a goal edge (a solution of the problem) from e. The function $h(e)$ is based on the outside part of the span of e, i.e., on its outside context. The closer that $g(e)$ is to the real cost (probability),

the more edges will be pruned. If the $h(e)$ function does not overestimate the cost to reach the goal, the search will be complete and optimal [Russell and Norvig, 2003].

This search method can be applied to the problem of finding the best parsing for a sentence, given a stochastic context-free grammar. The most promising edge at each moment is the one that is chosen edge to be expanded. Thus, an agenda with all the hypotheses (edges) ordered by their estimated cost (probability) is needed. From now on, we will assume that the probability of an edge is represented by its logarithm, so the product of probabilities is, in fact, the addition of their logarithms. The edge with the highest probability in the agenda is the most promising one.

In [Klein and Manning, 2003a], some context summary estimates are proposed with good results. A summary of the context of an edge in a sentence is taken (for example, we only take into account the word on the left of e as its context). There are probably many sentences that fits the same context, so, we take the maximum probability of all the edges e' that fit that context, i.e., the minimum cost. Hence, the real cost of getting a goal from e is always lower or equal to this estimate. Context summary estimates are always admissible functions since they never overestimate the cost to reach the goal. Their value is the maximum probability over all the derivations that fit the context and the real probability cannot be larger. If the summary function is carefully chosen and the number of contexts is not excessively large, we can precompute them and access to each one in constant time per edge in parse time.

Some of these summary estimates are described below. The simplest of all is the *NULL* context estimate: all the possible contexts have the same probability. The *SX* context estimate takes into account the number of terminals on the left and on the right of the edge. *SXL* also takes into account the terminal that is on the left of the edge. Other more complex context summary estimates have been considered in that work. Finally, the non-practical context estimate is the real cost of the outside part of the edge, i.e., the *TRUE* estimate.

Most of the context summary estimates proposed cannot be computed when dealing with a big SCFG. For example, the space complexity for the SXL estimate (one of the simplest estimates) is $\Theta(l^2 \cdot N \cdot T)$ where l is the maximum length of the sentences parsed, N is the number of non-terminal symbols, and T is the number of terminal symbols. For a 70-length sentence using a grammar with 97 non-terminal symbols and more than 40,000 terminal symbols, there are more than $1.9 \cdot 10^{10}$ possible contexts. Hence, assuming a space cost of 4B per context, more than 70GB is needed to store them.

Therefore, of the context summary estimates proposed in [Klein and Manning, 2003a], the most informative one that can be used with grammars of this kind, is *SX*.This context estimate ignores the lexical context of the edge. For grammars with many of lexical rules, *SX* is too optimistic and prunes very few edges. To solve this problem, a new

estimate *SXLex* (a lexicalization of the *SX* estimate) can be used. This estimate uses more contextual information in order to improve the parsing process. *SXLex* can be divided into two parts. The first part is similar to the SX context summary estimate but does not take into account the probability of generating terminals from preterminal symbols, that is, the maximum probability of all the derivation trees that produce L preterminals symbols to the left of the edge symbol and R to the right. As with *SX*, this part can be precomputed. The second part of the SXLex estimate, the lexical part, must be computed one time for each sentence. The lexical part of a sentence S, $\text{Lex}(S, L, R)$, taking into account L symbols to the left and R to the right, is the maximum probability over all the derivations from preterminal symbols to vocabulary words. That is:

$$Lex(S, L, R) = \sum_{r=1}^{R} \max_{N} P(N \rightarrow S_r) + \sum_{l=|S|-R}^{|S|} \max_{N} P(N \rightarrow S_l)$$

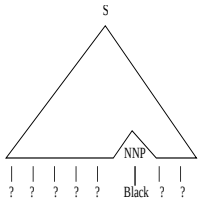
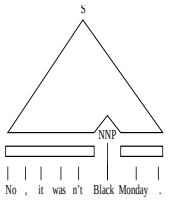
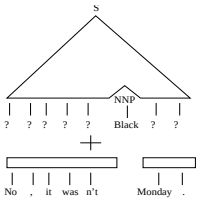
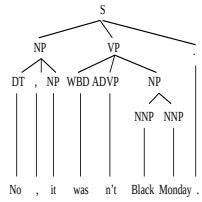
Estimate	a) SX	b) SXlex	c) SXlex2	d) TRUE
Context				
Score	-27.33	-32.83	-41.75	-49.16
Edges	679	346	85	-

Figure 5.2: Estimates for a given sentence: Context information needed, score obtained for each of the bounds and number of edges needed to finish the parsing.

The *SXLex* estimate of an edge is the sum the two parts described above. The optimality of *SXLex* can be easily proved. The lexical part is the probability of an optimistic derivation from preterminals to the terminals of the string being parsed. The *SX* part is a context summary estimate for the preterminals before and after the edge. The sum of the parts is always greater or equal to the real future cost.

In order to reduce the number of edges to be processed even more, a new bound closer to the true cost is proposed. This bound, *SXLex2*, is a combination of *SX* and the lexical part of the *SXLex* estimate. The main difference between *SXLex* and *SXLex2* is that *SXLex* uses the *SX* estimate from the initial symbol of the grammar to the preterminals and *SXLex2* uses the complete *SX* estimate from the initial symbol to the terminals (vocabulary words). It should be noted that this bound is not always optimistic. Hence, the most probable parsing is not guaranteed and is a 'non-optimal'

bound. However, in spite of this fact, the derivation tree obtained is the optimal one in most of the cases.

Figure 5.2 shows an example of the value of the different estimates for a given edge in the parsing process of a sentence from Penn Treebank. If no bound is used (*NULL* estimate), the number of edges processed is 1732. The *SX* estimate only takes into account the non-terminal of the edge (NNP) and the number of terminals before and after it. As can be observed, *SX* is too optimistic, so the number of processed edges to parse the sentence is still large. Taking into account lexical information, *SXLex*, the number of edges decreases to 346. Finally, with the *SxLex2* bound, the value obtained is closer to the true value, and is still optimistic. The savings produced with this bound are considerable: only 85 edges are needed.

5.1.2 Results and impact

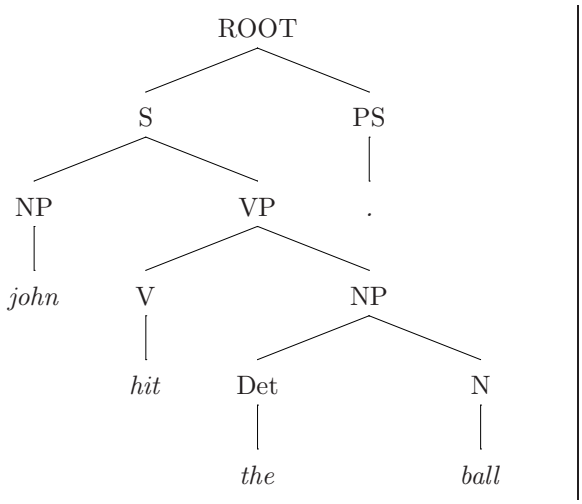
Lexicalized bounds for A* parsing have been explored in this section. These new bounds consider lexical information in order to get a more realistic estimation of the future cost of the parsing. The use of lexical information has been proved to be useful in the search for the optimal solution. Other bounds that do not guarantee the optimality of the solution have also been proposed because they can be practical under the following conditions: if they produce more savings than the 'optimal' bounds and prune the most probable parsing in only few cases. These new proposed bounds are very useful for parsing with large vocabularies.

5.2 Tree-to-String Decoder

The second part of the translation system is the tree-to-string decoder. The decoder should find the most likely target-language string from the parse tree of a source language sentence. Formally, and as have been stated in Chapter 2 (see expression (2.28)) :

$$t^* = \underset{t}{\operatorname{argmax}} \Pr(t|\delta^*, s)$$

The probability of the *translation* is modelled using a log-linear model that uses several different probabilities that have been explained in Section 2.2. Following the phrase-based MT approach, we use phrases as the basic translation unit. However, we only allow those segmentations that are consistent with the input parse tree. A segmentation is consistent when each of the phrases that form it yield exactly from a node in the parse tree. In Figure 5.3 we show the consistent segmentations of a sentence given its parse tree. We call this constraint over the possible segmentations, *consistency constraint*.



Consistent Segmentations

(john) (hit) (the) (ball) (.)
(john) (hit) (the ball) (.)
(john) (hit the ball) (.)
(john hit the ball) (.)
(john hit the ball .)

Figure 5.3: Parse tree and its consistent phrases.

5.2.1 Core Algorithm

The decoder implements a beam search over the parse tree. Thus, the basic algorithm uses a recursive tree-traversal in order to generate partial translations and combine them until it gets a complete translation. Each of such partial translations is called *hypothesis*. The hypotheses obtained in each node of the tree are stored in an *agenda*. An agenda is a set of hypotheses ordered by their probabilities. Given a segmentation consistent with the input parse tree, many phrase translations can be applied, i. e. we can find several phrases in the phrase table, whose source part is equal to some of the segments . We call each of these translations a *direct hypothesis*. Such direct hypotheses correspond to the phrasal productions of the SPhSITG described in Chapter 2.

Each hypothesis that covers the whole sentence is a possible translation of the input sentence. However, usually do not exists a phrase that covers the whole input sentence, so direct hypotheses must be merged in order to get *combined hypotheses*. Every hypothesis h must be stored with its corresponding target language phrase (target phrase) and its translation probability (score). Fore each node of the input parse tree an agenda is created in order to store the translation hypotheses corresponding to the node. Note that we do not need to store the source language words covered by the hypothesis because this information is implicit in the node of the tree.

Thus, the target language translation is generated following a bottom-up traversal of the tree. This process is illustrated in the pseudo code algorithm in Algorithm 3.

Algorithm 3 treeToString(tree)

```
1: phrases ← obtainPhrases(tree.input)
2: addHypothesis(agenda,phrases)
3: if isNotLeaf(tree) then
4:    leftAgenda ← treeToString(tree.leftSon)
5:    rightAgenda ←treeToString(tree.rightSon)
6:    for all h1 ∈ leftAgenda do
7:       for all h2 ∈ rightAgenda do
8:          addHypothesis(agenda,combine(h1,h2))
9:          addHypothesis(agenda,combine(h2,h1))
10:      end for
11:   end for
12: end if
13: return agenda
```

First, we store in the agenda all the direct hypotheses correspondent to part of the input sentence direct that yields the node of the tree (lines 1 and 2). When such part can be divided, i. e. the node of the tree has sons, it is not a leaf, we make a recursive call to the algorithm in order to get the agendas of each of the sons (lines 4 and 5). Then, we get the combined hypotheses from the direct (line 8) and inverse (line 9) merge of the agendas of both sons and finally, we return the resulting agenda(line 13). At the end of the tree-to-string process, we get an agenda that contains an ordered set of complete translation hypotheses.

Figure 5.4 shows the generation of the 2 kinds of hypotheses: direct and combined. Direct hypotheses are extracted from the phrase table, whereas combined hypotheses are the result of combining 2 hypotheses from the sons' agendas.

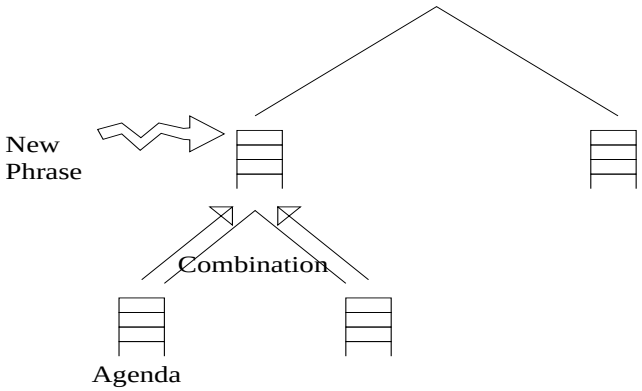


Figure 5.4: Two different kinds of hypothesis: direct (new phrase) and combined.

Now we describe in detail the process of combining two hypotheses h_1 and h_2 to get two new hypotheses h_d and h_i. Let $\mathcal{N}$ be the non-terminal symbol of the node of the tree where h_d and h_i must be generated and $\mathcal{N}_1, \mathcal{N}_2$ the nodes of h_1 and h_2 respectively. The target phrase of h_d is the concatenation of the target phrase of h_1 and h_2 in direct order, and the target phrase of h_i is the concatenation of h_1 and h_2 in inverse order. The score of h_d and h_i is the result of combining the scores of h_1 and h_2 and the probability of the rule $\mathcal{N} \rightarrow [\ \mathcal{N}_1\mathcal{N}_2\ ]$ and $\mathcal{N} \rightarrow \langle\ \mathcal{N}_1\mathcal{N}_2\ \rangle$ respectively. It must be noted that the n-gram language model must be recomputed for the new target phrase of h_d and h_i. This process is illustrated with an example in Figure 5.5.

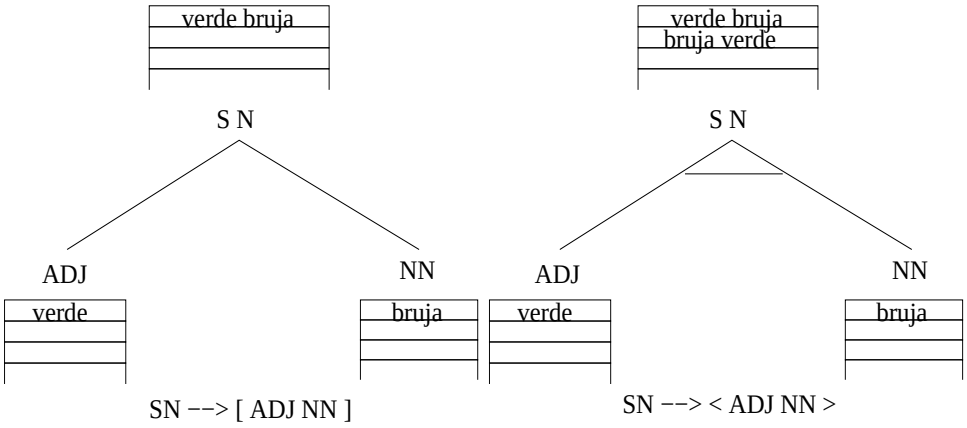


Figure 5.5: Direct and inverse combination.

All the hypotheses of each node of the tree are stored in the an agenda ordered by their probability. When the decoder comes back to the root node of the tree, the hypotheses of the agenda are complete translations. Among these complete translations, the one with highest probability (score) is selected as best translation.

5.2.2 Implementation details

During the search it is possible that two hypothesis with the same target phrase appear in the an agenda. Recombining those hypothesis is a risk-free way to reduce the search space. Following what we said in Section 2.2, the probability of a target sentence given a parse tree can be computed *as is* or assuming that is equal to the most likely of all its possible transformations (See Expression (2.30) of Section 2.2).

$$\Pr(t|\delta^*, s) \begin{cases} = \sum_{d \in \tau(\delta^*)} \Pr(d, t|\delta^*, s) \\ \\ \approx \max_{d \in \tau(\delta^*)} \Pr(d, t|\delta^*, s) \end{cases}$$

If we take the first option, when the decoder recombine two hypotheses, merge them in a new hypothesis with a score that is the sum of their scores. We call to this recombination option *sum*. If we choose the second option, the decoder only keeps the most likely (the one with a greater score). This recombination strategy is called *unicity*.

In the implementation of the decoder, we give to the user the election of the recombination method. Another option could be not to recombine hypotheses (*none*) in order to get a faster search when the number of hypothesis in each agenda is limited (as is explained next)[1].

Even though the tree consistency constraint prunes the possible segmentations of the input sentence, and with that, also reduces the number of hypotheses, the number of possibilities is still exponential [Knight, 1999]. Although the algorithm described so far can search through all possible translations, the time that it would take is excessive.

In order to prevent the exponential explosion in the number of hypotheses, the decoder uses a beam search algorithm. In each agenda (that means in each node of the tree) we keep only a limited set of the most likely hypotheses. Since each agenda stores hypotheses that cover the same set of source language words, the search is not biased towards the hypotheses that cover the easier part of the sentence, i.e. the words with a higher translation probaility, like happen in the decoder presented in [Koehn, 2004]. Thus, it is no necessary to use a future cost estimation to prevent that biasing.

The pruning of the beam search can be directed using two different strategies: histogram pruning or threshold pruning. Histogram pruning uses a fixed size n for the agenda, so we only keep the n best hypotheses. On the contrary, threshold pruning fixes a threshold α and prunes out all the hypothesis with a score lower than a factor of α the score of the best hypothesis in the agenda.

Pruning out the less likely hypotheses could delete a hypothesis that would have been part of the best translation. Thus, we say that this kind of pruning is not risk free.

Usually the main interest of the translation decoders is to get the best translation hypothesis given an input source language sentence. However, for some applications it is necessary to get, not only the best translation, but also the second, the third and so on.

[1] It is no necessary to check the unicity of the hypotheses in the agendas.

The decoder described in this work can obtain the n-best translation hypotheses. It must only get the n-best hypotheses of the root node agenda. A common method used in speech recognition, that has also appeared in machine translation [Koehn and Knight, 2003][Och et al., 2003] is to get the n-best translations and then rescore them using additional features (new complex language models, new translation models...). For instance, we propose as a further work the use of a phrasal SITG to rescore the n-best translations obtained using the decoder.

As a future work we plan to obtain the word graph [Ueffing et al., 2002] instead of the n-best translations. It is possible to obtain the n-best list from the word graph, but word graphs are also commonly used in computed assisted translation tools [Och and Ney, 2003][Sanchis-Trilles et al., 2008].

Chapter 6

Experimental Results

"There is no such thing as a failed experiment,
only experiments with unexpected outcomes"
Richard Buckminster

In this Chapter we present the experiments carried out in order to test the performance of the decoder. We evaluate the decoder using two different corpus: `EuTrans-I` corpus [Amengual et al., 2000] (Spanish-English) and `Europarl` V2 corpus [Koehn, 2005] (German-English).

As a preprocess for the corpus, all the pairs of languages were lowercased and tokenized. The results were measured also using the lowerized and tokenized version of the test. The quality of the translation was measured using Bleu [Papineni et al., 2001].

We carried out experiments to test the SITG inference methods explained in Chapter 3. In order to increase the speed of the SITG inference algorithms, we have used the Viterbi algorithm that takes benefit of the bracketing information. The bracketed corpus and the linguistic parse tree was obtained using the freely available parser described in [Klein and Manning, 2003b]. The alignments of the bilingual corpus were computed using the toolkit GIZA++ [Och and Ney, 2003].

The phrase tables were computed using the methods presented in Chapter 4. The baseline system was the phrase-base system Moses [Koehn et al., 2007]. The baseline system and the hybrid decoder used in each experiment the same phrase table.

The language model used in all the experiments was a 5-gram computed using the toolkit SRILM [Stolcke, 2002]. And in order to adjust weights λ_i of the log-linear model we used the Downhill Simplex Algorithm, also called Nelder-Mead method [Nelder and Mead, 1965], over a tuning set.

6.1 EuTrans-I Corpus.

`EuTrans-I` corpus [Amengual et al., 2000] is an aligned bilingual corpus Spanish-English about typical dialogues in the desk of a hotel. Table 6.1 shows the statistics of the corpus.

		Spanish	English
Training	Sentences	10,000	
	Words	97,131	99,292
	Vocabulary Size	686	513
Test	Sentences	2,996	
	Words	35,023	35,590
	Out of Vocabulary Words	0	0

Table 6.1: `Eutrans-I` corpus statistics.

This corpus, that has been semiautomatic generated, is considered as an easy translation task. That is why we have used this corpus only to check the correctness of the implementation and to tune some of the decoder's parameters.

6.1.1 Experiments

The experiments carried out over this corpus consist in the translation Spanish-English using the decoder and the phrases obtained with the use of SITG, Section 4.1. We used an ergodic SITG with one iteration of the Viterbi stochastic estimation. The three recombination methods explained in Section 5.2.2 were tested: *none*, *unicity* and *sum*. Table 6.2 shows the results such experiments.

Experiment	BLEU
Baseline	90.5
None	90.2
Unicity	90.2
Sum	90.3

Table 6.2: Results with the `EuTrans-I` corpus.

The results obtained are very similar to the baseline and the *sum* recombination strategy seems to be slightly better than the other strategies. Experiments over `EuTrans-I` are not conclusive because the corpus is too much simple. Sentences are short and the reorderings can be modelled in most of the cases, by the n-gram language model. In addition, the languages of the corpus, Spanish and English, have a similar structure and the number of reorderings is low.

6.2 Corpus Europarl V2.

The `Europarl` corpus [Koehn, 2005] has been extracted from the proceedings of the European Parliament and has versions in 11 European languages. We have used in this work only the German-English section of the corpus. The main statistics of this section of the corpus is shown in Table 6.3.

		German	English
Training	Sentences	751,088	
	Words	15,256,793	16,052,269
	Vocabulary Size	195,291	65,889
Test	Sentences	2,000	
	Words	55,533	59,307
	Out of Vocabulary Words	141	387

Table 6.3: `Europarl` corpus statistics.

It can be seen that this is a real and complex corpus. Actually, it has been used as the main task in several current MT competitions [Callison-Burch et al., 2007]. This pair of languages is specially difficult due to the difference between the vocabulary sizes and the high number of reorderings.

6.2.1 Experiments

We have carried out several experiments over this corpus changing the different parameters and options of the decoder: SITG used, bracketing information in the input, phrase table...

Ergodic Grammar

The first experiment consisted on testing the influence of the estimation procedures over the SITG. Therefore, we used in this experiment an ergodic grammar with 4 non-terminals and we varied the number of iterations of the Viterbi estimation algorithm. The phrase table used was the same SITG extraction phrase table for baseline and hybrid decoders. Again, we tested the three hypothesis recombination techniques. Table 6.4 shows the results of the experiments.

	BLEU		
	No Estimation	1 Iteration	2 Iterations
Baseline	22.53		
None	16.32	17.10	17.29
Unicity	16.56	17.44	17.45
Sum	16.63	17.41	17.33

Table 6.4: Test results for the `Europarl` corpus with an ergodic grammar.

It can be seen, that the results, thought being still far from the baseline, improve with the use of the estimation algorithm. The choice of the SITG has a real influence in the results of the decoder because besides of modelling the reorderings, it also define the segmentation of the input sentences.

Ergodic Grammar with Bracketing Information

In the previous experiments we can see the importance of a good segmentation of the sentences. One of the features of the decoder is the possibility of annotating the input sentences with bracketing information. The decoder must respect the segmentation given by the bracketing. In order to do that, we parsed the input sentences with a monolingual linguistic parser, then, the bracketing information was extracted from the parsed sentences. It must be noted that this bracketing is not necessarily complete since the parse tree is not always binary. In addition, the parsing can introduce errors given the high complexity of the `Europarl` corpus.

The rest of the parameters and the decoding options are the same as in the previous experiment. Results are shown in Table 6.5

	BLEU		
	No Estimation	1 Iteration	2 Iterations
Baseline	22.53		
None	17.11	17.93	18.08
Unicity	17.83	18.45	18.42
Sum	17.62	18.59	18.56

Table 6.5: Test results for the `Europarl` corpus with an ergodic grammar and bracketing information.

The results obtained are significantly better than in the previous experiment. That means that the segmentation done by the ergodic SITG were improved by the use of the bracketing information.

Linguistically Motivated Grammar

Hence, the syntactic model has a big influence on the translation results. The next step is then, to increase the complexity of the model. We used, for this experiment, a linguistically motivated SITG obtained by the use of the method explained in Section 3.1.2. The probabilities of such SITG have not been re-estimated. The results for this experiment are shown in Table 6.6.

Experiment	BLEU
Baseline	23.53
None	21.77
Unicity	22.12
Sum	22.16

Table 6.6: Test results for the `Europarl` corpus using a linguistically motivated SITG.

Results show a significative improvement, thought still being under the baseline. With a better SITG, maybe re-estimating the probabilities by the use of Viterbi or Inside-Outside re-estimation algorithms or using another structure inference algorithm, probably will improve the results. We will carry out those experiments in the future.

Phrase Tables from Bilingual alignments

Finally, we carried out an experiment in order to test the influence of the phrase table over the decoder's performance. In order to do that, we used the phrase table obtained using Moses training tools [Koehn et al., 2007], the outlines of this method are explained in Section 4.2. We used the same phrase table for the baseline decoder. Table 6.7 shows the results for this experiment.

Experiment	BLEU
Baseline	25.21
None	22.67
Unicity	23.13
Sum	23.02

Table 6.7: Test results over `Europarl` using Moses' phrase table.

The results show that the use of this phrase table improve the results of both, the developed system and the baseline. However, it must be noted that the phrase tables obtained by the Moses tools are considerably larger than SITG phrase tables, and the decoding process is slower.

Thought not being very far, the results obtained in the experiments proposed worse than the baseline results. As have been said in previous chapters, the parse tree of the input not only determines the reorderings, but also the segmentation of the source language sentence. When the parse of the input is not correct, the decoder cannot find the most appropriate phrases and such phrases cannot be reordered correctly. The sentences of the `Europarl` corpus are complex and the embedded parser introduce a high number of errors in the decoding process. In order to solve such problem, we are currently working in two different directions:

1. Obtain better SITG by increasing the number of non-terminals and hence, its complexity and expressivity of the grammars, or getting a better estimation of the probabilities.

2. Study the use of external parsers and the influence in the translation results.

Chapter 7

Final Remarks

In this Chapter we present the conclusions of the Master Thesis, describe coarsely the work that is currently being produced and suggest some future work proposals.

7.1 Conclusions

In this master thesis we have presented an hybrid MT model that uses uses formalisms and algorithms from syntax-based and phrase-based approaches. This new model allows us to introduce many different translation models in the log-linear combination, such as syntactic reordering, phrase translation probability models or n-gram language models.

In addition, we have created a decoder that use the presented translation model. The decoder has two main steps: first it parses the input sentence and then it *translates* the tree into a target-language sentence. Thus, the segmentation of the input language is determined by the source language parse tree and we can use its syntactic information in the search of the best translation. However, this restriction over the possible segmentations is a conditioning factor when translating a sentence. A bad parse tree gives a bad segmentation and this, usually produces a bad translation.

The results obtained show that linguistically motivated grammars produce a better performance than simple ergodic grammars and the re-estimation of the probabilities of the grammars are also important. Nevertheless, results are still under the baseline proposed so, maybe the reseach on the acquisition of more complex grammars is still necessary.

7.2 Future Work

The translation model and the decoder presented can be used as a tool in the research of multiple aspects of syntax-based and syntax-based machine translation.

Hence, the work developed for this master thesis gives us the possibility to explore in several different directions:

New methods for the inference of SITG: The experiments showed that the syntax formalism (the SPhITG) is a very important factor in the global performance of the decoder. Therefore, we must explore the possibility of learning

Use of linguistically motivated SITG in phrase extraction: The phrases used in the experiments have been extracted using an ergodic SITG. It would be interesting to test the phrase extraction algorithm of Section 4.1 with a linguistically motivated SITG.

Use of new phrase translation models: The phrase extraction method of Section 4.1 allows the use of new syntactic phrase probability models such as the one described in [Sanchis-Trilles and Sánchez, 2008]. The use of such or other new models is also a proposal for future work.

Exploration of syntax based reordering models: Currently we are working in the acquisition and use of new reordering complex models that combine lexical and syntactic information in a maximum entropy models and its use in the implemented decoder. Such models can be also modelled using neural networks or support vector machines.

Syntax based rescoring: Rescoring is a common strategy that consist in the generation of the n-best list of hypotheses, rescore them using a new model and then get the best one. We plan to test a strategy based in the use of a SITG as a new model for the rescoring.

Experimentation over different corpora: Syntax based translation obtain better results when translating between language pairs with a different structure, for example English-Chinese, or Spanish-Arabic. We propose, then, to use the decoder over corpora with such pairs of languages.

Bibliography

[Aho and Ullman, 1972] Aho, A. V. and Ullman, J. D. (1972). *The Theory of Parsing, Translating and Compiling. Vol 1 : Parsing*. Prentice-Hall, Englewood Cliffs, N.J.

[Amengual et al., 2000] Amengual, J., Benedí, J., Casacuberta, F., Castaño, A., Castellanos, A., Jiménez, V., Llorens, D., Marzal, A., Pastor, M., Prat, F., Vidal, E., and Vilar, J. (2000). The EuTrans-I speech translation system. *Machine Translation*, 15:75–103.

[Baker, 1979] Baker, J. (1979). Trainable grammars for speech recognition. In *Speech communication papers presented at the 97th meeting of the Acoustical Society of America*, pages 547–550, Cambridge, MA. MIT.

[Brown et al., 1993] Brown, P. F., Pietra, S. D., Pietra, V. J. D., and Mercer, R. L. (1993). The mathematic of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

[Callison-Burch et al., 2007] Callison-Burch, C., Fordyce, C., Koehn, P., Monz, C., and Schroeder, J. (2007). Meta-evaluation of machine translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 70–106, Columbus, Ohio. Association for Computational Linguistics.

[Charniak et al., 2003] Charniak, E., Knight, K., and Yamada, K. (2003). Syntax-based language models for statistical machine translation. In *In MT Summit IX. Intl. Assoc. for Machine Translation.*

[Cherry and Lin, 2006] Cherry, C. and Lin, D. (2006). A comparison of syntactically motivated word alignment spaces. *EACL*, pages 145–152.

[Cherry and Lin, 2007] Cherry, C. and Lin, D. (2007). Inversion transduction grammar for joint phrasal translation modeling. *Proceedings of the NAACL-HLT Workshop SSST.*

[Chiang, 2005] Chiang, D. (2005). A hierarchical phrase-based model for statistical machine translation. In *ACL*.

[Collins, 2003] Collins, M. (2003). Head-driven statistical models for natural language parsing. *Computational Linguistics*, 29(4):589–637.

[Collins et al., 2005] Collins, M., Koehn, P., and Kučerová, I. (2005). Clause restructuring for statistical machine translation. In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 531–540, Morristown, NJ, USA. Association for Computational Linguistics.

[DeNeefe et al., 2007] DeNeefe, S., Knight, K., Wang, W., and Marcu, D. (2007). What can syntax-based mt learn from phrase-based mt? *Proceedings EMNLP-CoNLL*.

[Earley, 1970] Earley, J. (1970). An efficient context-free parsing algorithm. *Commun. ACM*, 13(2):94–102.

[Galley et al., 2004] Galley, M., Hopkins, M., Knight, K., and Marcu, D. (2004). What's in a translation rule? In *HLT-NAACL*, pages 273–280.

[Gascó and Sánchez, 2007] Gascó, G. and Sánchez, J. A. (2007). A* parsing with large vocabularies. In *Proceedings of the RANLP*.

[Hassan et al., 2007] Hassan, H., Sima'an, K., and Way, A. (2007). Supertagged phrase-based statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*.

[Hopcroft and Ullman, 1979] Hopcroft, J. E. and Ullman, J. D. (1979). *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley.

[Klein and Manning, 2003a] Klein, D. and Manning, C. D. (2003a). A* parsing: Fast exact viterbi parse selection. In *HLT-NAACL*.

[Klein and Manning, 2003b] Klein, D. and Manning, C. D. (2003b). Accurate unlexicalized parsing. In *In Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430.

[Knight, 1999] Knight, K. (1999). Decoding complexity in word-replacement translation models. *Comput. Linguist.*, 25(4):607–615.

[Koehn, 2004] Koehn, P. (2004). Pharaoh: A beam search decoder for phrase-based statistical machine translation models. In *Machine Translation: From Real Users to Research, 6th Conference of the Association for Machine Translation in the Americas, AMTA*, pages 115–124.

[Koehn, 2005] Koehn, P. (2005). Europarl: A parallel corpus for statistical machine translation. In *MT Summit*.

[Koehn et al., 2007] 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. In *Annual Meeting of the Assoc. for Computational Linguistics*, pages 177–180, Prague, Czech Republic.

[Koehn and Knight, 2003] Koehn, P. and Knight, K. (2003). Feature-rich statistical translation of noun phrases. In *ACL '03: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 311–318, Morristown, NJ, USA. Association for Computational Linguistics.

[Koehn et al., 2003] Koehn, P., Och, F. J., and Marcu, D. (2003). Statistical phrase-based translation. In *HLT/NAACL*.

[Koncar and Guthrie, 1994] Koncar, N. and Guthrie, D. G. (1994). A natural language translation neural network. In *In Proceedings of the International Conference on New Methods in Language Processing (NeMLaP*, pages 71–77.

[Liu and Gildea, 2008] Liu, D. and Gildea, D. (2008). Improved tree-to-string transducer for machine translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 62–69, Columbus, Ohio. Association for Computational Linguistics.

[Nelder and Mead, 1965] Nelder, A. and Mead, R. (1965). A simplex method for function minimization. *The Computer Journal*, 7(2):308–313.

[Nguyen et al., 2003] Nguyen, P., Shimazu, A., Ho, T., Nguyen, L., and Nguyen, V. (2003). A tree-to-string phrase-based model for statistical machine translation. In *Twelfth Conference on Computational Natural Language Learning, Manchester*.

[Nießen and Ney, 2004] Nießen, S. and Ney, H. (2004). Statistical machine translation with scarce resources using morpho-syntactic information. *Computational Linguistics*, 30(2):181–204.

[Och and Ney, 2002] Och, F. J. and Ney, H. (2002). Discriminative training and maximum entropy models for statistical machine translation. In *ACL*, pages 295–302.

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

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

[Och et al., 1999] Och, F. J., Tillmann, C., and Ney, H. (1999). Improved alignment models for statistical machine translation. In *Proceedings of the Joint SIGDAT Conf. on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 20–28.

[Och et al., 2003] Och, F. J., Zens, R., and Ney, H. (2003). Efficient search for interactive statistical machine translation. In *EACL*, pages 387–394.

[Papineni et al., 2001] Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2001). Bleu: a method for automatic evaluation of machine translation. In *ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 311–318, Morristown, NJ, USA. Association for Computational Linguistics.

[Picó et al., 2004] Picó, D., Tomás, J., and Casacuberta, F. (2004). GIATI: A general methodology for finite-state translation using alignments. In *Statistical, Structural and Syntactical Pattern Recognition. Proceedings of the Joint IAPR Intternational Workshops SSPR2004 and SPR2004*, volume 3138 of *Lecture Notes in Computer Science*, pages 216–223. Springer-Verlag,, Lisboa, Portugal.

[Roark, 2001] Roark, B. (2001). Probabilistic top-down parsing and language modeling. *Computational Linguistics*, 27(2):249–276.

[Russell and Norvig, 2003] Russell, S. and Norvig, P. (2003). *Artificial Intelligence: A Modern Approach*. Prentice-Hall, Englewood Cliffs, NJ, 2nd edition edition.

[Sanchis-Trilles et al., 2008] Sanchis-Trilles, G., Ortiz-Martínez, D., Civera, J., Casacuberta, F., Vidal, E., and Hoang, H. (2008). Improving interactive machine translation via mouse actions. In *EMNLP 2008: conference on Empirical Methods in Natural Language Processing*, Waikiki, Honolulu, Hawaii.

[Sanchis-Trilles and Sánchez, 2008] Sanchis-Trilles, G. and Sánchez, J. A. (2008). Using parsed corpora for estimating stochastic inversion transduction grammars. In *6th edition of the International Conference on Language Resources and Evaluation*, Marrakech, Morocco.

[Stolcke, 1995] Stolcke, A. (1995). An efficient probabilistic context-free parsing algorithm that computes prefix probabilities. *Computational Linguistics*, 21(2):165–201.

[Stolcke, 2002] Stolcke, A. (2002). Srilm – an extensible language modeling toolkit. In *International Conference on Spoken Language Processing.*

[Sánchez and Benedí, 2006] Sánchez, J. A. and Benedí, J. M. (2006). Obtaining word phrases with stochastic inversion transduction grammars for phrase-based statistical machine translation. In *Proc. 11th Annual conference of the European Association for Machine Translation*, pages 179–186, Oslo, Norway.

[Tomás and Casacuberta, 2001] Tomás, J. and Casacuberta, F. (2001). Monotone statistical translation using word groups. In *Proceedings of the Machine Translation Summit VIII*, pages 357–361, Santiago de Compostela.

[Ueffing et al., 2002] Ueffing, N., Och, F. J., and Ney, H. (2002). Generation of word graphs in statistical machine translation. In *EMNLP '02: Proceedings of the ACL-02 conference on Empirical methods in natural language processing*, pages 156–163, Morristown, NJ, USA. Association for Computational Linguistics.

[Wang et al., 2007] Wang, C., Collins, M., and Koehn, P. (2007). Chinese syntactic reordering for statistical machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 737–745, Prague, Czech Republic. Association for Computational Linguistics.

[Wu, 1997] Wu, D. (1997). Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403.

[Yamada and Knight, 2001] Yamada, K. and Knight, K. (2001). A syntax-based statistical translation model. In *ACL*, pages 523–530.

[Yamada and Knight, 2002] Yamada, K. and Knight, K. (2002). A decoder for syntax-based statistical mt. In *ACL*, pages 303–310.

[Zhang et al., 2008] Zhang, H., Quirk, C., Moore, R. C., and Gildea, D. (2008). Bayesian learning of non-compositional phrases with synchronous parsing. In *Proceedings of ACL-08: HLT*, pages 97–105, Columbus, Ohio. Association for Computational Linguistics.

[Zollmann and Venugopal, 2006] Zollmann, A. and Venugopal, A. (2006). Syntax augmented machine translation via chart parsing. In *Proceedings on the Workshop on Statistical Machine Translation*, pages 138–141, New York City. Association for Computational Linguistics.