Universität Karlsruhe (TH)
Research University · founded 1825

Carnegie Mellon

Reordering Strategies for Statistical Machine Translation

Diplomarbeit
Interactive Systems Laboratories Prof. Dr. Alex Waibel
Universität Karlsruhe (TH)
Carnegie Mellon University, Pittsburgh, PA, USA

von

cand. inform.
Kay Rottmann

Betreuer:
Prof. Dr. Alex Waibel
Dr. Stephan Vogel

Tag der Anmeldung:	1. April 2007
Tag der Abgabe:	27. September 2007

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Karlsruhe, den 27. September 2007

K. Rott

Kay Rottmann

Acknowledgments

This thesis was made possible by the help of a lot of people I want to thank for their support.
First of all I thank Prof. Alex Waibel for giving me the opportunity of writing this thesis in the interesting field of machine translation and for allowing me to write parts of this thesis at the Interactive System Labs at the Carnegie Mellon University in Pittsburgh. That would not have been possible without the interACT scholarship.
Many thanks also go to my advisor Dr. Stephan Vogel. Without all the discussions, hints and ideas this thesis would not have been possible.
Furthermore I want to thank everyone at the machine translation group at the CMU and at the University of Karlsruhe. Every discussion about my work resulted in new ideas.
Finally, I thank Christina and of course my parents for their patience with me and all their support throughout the course of my studies.

Abstract

This thesis focuses on the word reordering problem in a statistical machine translation system.
The approach I use is based on a reordering of the words on the source side thus allowing a monotone decoding during translation. For not making any hard decisions in this process a lattice structure is used that allows to use different reorderings without deciding for a single one.
The reorderings are created by using rules learned from a bilingual word aligned training corpus. For not suffering from data sparseness the source side of the corpus is annotated with additional information that is used to allow more generalized reordering rules. In this thesis I describe how parts of speech tags, chunk parse information and automatically built word clusters can be used for this task. As an alternative to already existing methods for word clustering, I propose methods constructed directly for the reordering task and therefore giving better results than traditional word clusterings used for other tasks. Also this allows the usage of the proposed techniques even for languages for which tools like parts of speech tagger and chunk parser are not available.
For the different types of annotations I describe the learning of the reordering rules and how context information is usable to predict the correct reordering more reliable.
Furthermore I show how the reordering of the source training corpus can be used to generate a phrase table that matches the word order in the reordering lattices better than the phrases in the phrasetable obtained from the original corpus. I also show that it is important how this reordered source corpus is generated and that a corpus generated the same way as the reorderings of the sources during translation has to be preferred.
Finally I present experiments on selected language pairs showing the effect of the proposed techniques on the translation quality. I also present an example for a language pair, in which the difficulties are shown that arise when using word class based reordering rules.

Zusammenfassung

Diese Arbeit beschäftigt sich mit dem Problem, die korrekten Wortumstellungen in einem statistischen maschinellen Übersetzungssystem zu erhalten.
Der Ansatz beruht auf einer Umordnung der Wörter in der Quellsprache um eine monotone Übersetzung dieser Wortfolge in die Zielsprache zu ermöglichen. Damit keine harten Entscheidungen getroffen werden, die im Nachhinein die Übersetzungen zu stark einschränken, wird eine Lattice-Struktur benutzt, die alle gefundenen möglichen Wortumstellungen enthält.
Die Wortumstellungen werden durch Regeln, die aus einem bilingualen, wortalinierten Korpus gelernt werden, erzeugt.
Um dabei nicht das Problem von zu wenig Trainingsdaten aufkommen zu lassen, wird eine Annotation der Quellsprache benutzt um generalisierte Umordnungsregeln zu erhalten.
Ich beschreibe in dieser Arbeit, wie man als Annotation Chunk Parse Informationen, Parts of Speech Tags und automatisch gefundene Klassifizierungen der Wörter benutzen kann. Für die automatische Klassenbildung gebe ich Methoden an, die direkt auf die Aufgabe der Wortumstellungen zugeschnitten sind und damit für die Aufgabe besser geeignet sind, als bereits vorhandene Methoden. Außerdem können mit Hilfe dieser sprachunabhängigen Klassifizierungstechniken die vorgestellten Umordnungsstrategien auch für Sprachen benutzt werden, die keine Hilfsmittel wie Parts of Speech Tagger oder Chunk Parser verfügbar haben.
Für die verschiedenen Annotationstypen beschreibe ich, wie man daraus Umordnungsregeln lernen kann und wie Kontextinformation benutzt werden kann, um die korrekten Umordnungen zuverlässiger vorhersagen zu können.
Ferner zeige ich, wie man mit Hilfe einer Umordnung des Trainingskorpus zu einer Phrasentabelle gelangt, die für die Übersetzung der Umordnungslattices besser geeignet ist, als die Phrasentabelle basierend auf dem monotonen Korpus. Allerdings ist dabei die Art, wie der Korpus der Quellsprache umgeordnet wird von entscheidender Bedeutung.
Auch präsentiere ich Ansätze, wie mit langreichweitigen Wortumstellungen umgegangen werden kann. Abschließend zeige ich in Experimenten, wie sich die Techniken auf die Übersetzungsqualität auswirken und gebe auch ein Beispiel für die Schwierigkeiten, die mit der Umstellung basierend auf Wortklassen auftreten können.

Contents

1. Introduction

Spoken languages are the main tool used for communication in this world. A language provides a way to discuss, to express feelings, to explain - shortly: to exchange information. Often whole cultural groups are defined by the language they use.
The difficulty is to overcome the barriers that are imposed by different languages. As soon as two people speak different languages it becomes very hard for them to exchange information. Typical solutions are learning the other language or everyone learning a common language.
The learning of the other language, however, introduces further problems. Not every country has access to schools for its citizens as easily as in the western countries. But even for countries with easy access to educational institutions problems are not always solved.
For example, a citizen of the European Union would have to learn 23 languages to be able to speak to people in every other country in the European Union. This is quite impossible for most people.
Language is also part of culture and so the goal should not be to switch to a single language and urge people to speak other languages, but it should be to conserve the individual languages.
Of course there are professional translators. But there are situations in which translators can hardly provide all the needed translations. For example:

- In politics: the European Union has 23 official languages. Every speech held in the European Parliament has to be translated into every other language.
- In economy: more and more companies have business contacts all over the world. Every contract and document has to be available in many different languages.
- In humanitarian aid: humanitarian aid is often provided by people who do not speak the language of the people they are helping.
- In tourism: better flight connections allow us to travel into every country we would like to and thus connect more and more different languages.

Obviously there is a need for other techniques in addition to human translators to create the needed translations.
One approach that seems reasonable is to use machine translation systems. This means systems that are able to automatically generate translations for spoken text without any further human assistance.
These systems are still science fiction. However the translation quality of machine translation systems have improved over the years so much that today's systems generate translations which are usable to get an understanding of the foreign text even though the translations are not perfect so far.

1.1 History of machine translation

Machine translation efforts already had already begun in the 1950s. The first prominent efforts started with Warren Weaver's idea of correspondences between translation and cryptography. In that time the interest in MT rose and massive funding took place.
In 1954 a first system was presented that was able to translate between Russian and English. However this was a very small and simple system that had a vocabulary of 250 words and 6 rules of grammar.
In the 1960s the skepticism rose again mainly caused by the ALPAC [PiJBC66] report which stated "MT was slow, less accurate and twice as expensive as human translation." and "there is no immediate or predictable prospect of useful machine translation." This report caused the stoppage of many projects. Only a few groups (mainly in Europe and Canada) continued their research.
In the 1970s a revival of machine translation happened because of the success of the *METEO* system created by a Canadian research group for translating weather forecasts between English and French.
Since then a variety of different approaches for machine translation have been constructed and the systems have gotten more useful. First for restricted domains, later even for large vocabulary unlimited domains the translation quality became better and better, until today where unlimited domain translations are often understandable even though a lot of errors still happen in the output.

1.2 Statistical machine translation

Statistical machine translation (SMT) is currently one of the most promising approaches to large vocabulary text translation.
In the spirit of the Candide system developed in the early 90s at IBM [BPPM93], a number of SMT systems have been presented in the last few years: [WaWa98], [OcNe00], [YaKn00].
The underlying principle of SMT is the use of Bayes rule. Let e_1^I be an English[1] sentence of length I and f_1^J a French sentence of length J. The best translation of the French sentence is

$$\tilde{e}_1^I = argmax_{e_1^I}\{\ p(e_1^I|f_1^J)\ \} = argmax_{e_1^I}\{\ p(e_1^I)\ \cdot\ p(f_1^J|e_1^I)\ \} \tag{1.1}$$

[1]For better readability I will use French as the source language from which is translated and English as the Target language that is translated into

By splitting up the translation task this way it is possible to use different sources of information. On the one hand there is the *Language Model* (LM) with $p(e_1^I)$, and on the other hand there is the *Translation Model* (TM) with $p(f_1^J|e_1^I)$. The LM gives the probability of a word sequence in the target language while the TM gives the probability how likely this sequence would have produced the given French sentence. For language models often stochastic models (n-grams) are used which estimate the probability of a word given its history by restricting the history to n preceding words. The translation model is split up into further parts, so that the length of the sentences, the lexical translation probabilities and the alignment quality also come into account.
The translation task then consists of the preprocessing which generates an appropriate input for the translation system for each sentence to translate. Then this input is handed over to the decoder, which searches for the best translation in the search space. As a result of the search performed by the decoder, various outputs are possible, such as the best hypothesis or the n best list that contains the n best translation hypothesis according to the scoring in the decoder.
The output can then be processed in a postprocessing step like n best list rescoring or handing it over to a text to speech system, or it is simply outputted as the final translation.

1.3 Problem of word order

One of the most difficult problems still remaining in statistical machine translation is dealing with the different word order in different languages.
Often the languages that are translated into another differ in their syntactic structure and because of that require reorderings of the words when translated.
To make things even worse these differences in the word order can be locally, just switching two adjacent words as in Table 1.1 or much tougher to solve: global reorderings, moving words over longer blocks of words that belong together (and also may contain further reorderings) as in Table 1.2. In the latter example the '...' indicate that other information could be embedded pushing the auxiliary verb and the infinitive verb even further apart. But whatever information is inserted there, the infinitive verb has to be moved to the auxiliary verb.

Example: ADJ NN $\rightarrow$ NN ADJ
An important agreement
Un acuerto importante

Table 1.1: example for local reordering in translating Spanish to English

Example: auxiliary verb and infinite verb
Ich werde morgen nachmittag ... ankommen
I will arrive tomorrow afternoon ...

Table 1.2: example for global reordering in translating German to English

Example: detached verb prefix
He lights a candle
Er zündet eine Kerze an.

Table 1.3: example for a reordering based on a detached verb prefix that takes place on the target side

And finally the example in Table 1.3 illustrates a reordering that is hard to cover for a phrase based system.
The reason why this is difficult for a phrase based system is that disjointed phrases are needed for that, otherwise the *lights* always gets translated into *zündet an* without being split. That results in either a result like "Er *zündet an* eine Kerze.", or another error that is often seen in phrase based translations: "Er *zündet* eine Kerze.".

1.4 Goal of this thesis

Even with those difficulties, state-of-the-art SMT systems are phrase based systems. By using phrases the system is already able to capture some local reorderings in a phrase pair. However, this is rather limited as the average length of matching phrases is typically less then two words. So it is inescapable to use further techniques for longer ranging word reorderings. That raises four main questions:

- How can the word reordering be modeled?
- How can the parameters of the model be estimated?
- How can the model be integrated into the decoder?
- How do different languages behave concerning the model?

The goal of this thesis is to give answers to these questions and provide a set of techniques for improving the translation quality.

1.5 Related work

For creating the correct word sequence two different approaches are possible. The first one relies on hard decisions in the form of grammar rules that provide the right order of the words and do not allow other reorderings. This approach can be seen in grammar-based systems.
The other approach uses weak rules with probabilities. So, different reorderings are possible with assigned probabilities. In the translation process most, if not all, of these reorderings are tested and evaluated. This is the approach chosen in statistical translation systems.
The IBM alignment model, proposed in [BPPM93] and the HMM alignment model proposed in [VoNT96] contain a so called distortion model to capture the word order in different languages. These distortion models rely on the absolute position, like in the IBM2 models or on relative positions like in the HMM alignment model.
The problem with these models is that they are very weak and essentially penalize

longer reorderings while prefering shorter. Furthermore there is no restriction on the possible reorderings. That means every possible reordering has to be tried and evaluated in decoding time. As a result of this behavior only short reorderings are allowed that take place in a certain *reordering window*.
To overcome this shortcoming, constraints like the IBM constraints [BePP96] or the Inverse Transduction Grammar [Wu96] were introduced. The inverse transduction grammar constrains (ITG) allow only those reorderings between two languages which can be generated by swapping subtrees in a binary branching tree. The IBM constraints allow for a limited number of untranslated words during decoding.
However even these constraints leave a very huge search space and further pruning techniques are needed. A comparison of both approaches can be found in [ZeNe03].
All previously mentioned approaches have in common that no lexical or syntactical information was used in the reordering process. For getting more information into the distortion models it is possible to use lexical information gathered for the words that will be reordered like information on the word classes (parts of speech (POS) or automatically generated).
Other approaches were introduced that used more linguistical knowledge, for example the use of bitext grammars that allowed the parsing of the source and target language [Wu97].
In [ShSO04] and [OGKS+04], syntactical information was used to rerank the output of a translation system with the idea of finding the correct reorderings in this stage.
In [TiZh05] a lexicalized block-oriented reordering model is proposed that decides for a given phrase whether the next phrase should be oriented to its left or right.
The most recent and very promising approach, is a reordering based on rules learned from an aligned training corpus with a POS-tagged source side ([ChCF06], [PoNe06] and [CrMa06]). These rules are then used to reorder the wordsequence in the most likely way.
This last approach builds the basis of this thesis.

1.6 Outline of the thesis

In the second chapter I am going to give a more detailed analysis of the reordering problem in statistical machine translation. I will discuss different types of reorderings that occur during translations and I introduce approaches on how to deal with the reordering problem. The chapter ends with a brief description of the reordering approach used in this thesis.
Chapter 3 is intended to give the details on the reordering strategies that are used for this thesis. The chapter covers techniques usable for languages with a rich linguistic toolkit but also covers strategies for languages without these tools.
The fourth chapter gives implementation details on how the proposed strategies are implemented and how to avoid pitfalls. Here the algorithmic details are presented.
The fifth chapter provides the results that were obtained when using the proposed techniques. It also contains a discussion of the results and analysis of the observed behaviors of the techniques.
Finally I give a short overview of the achievements of this work in chapter 6 and I also point out what work should be done in the future to further improve the achievements.

2. Analysis

The problem of the word order has bothered the machine translation community for a long time, and as shown above many different approaches have been introduced.
The question is, why is the reordering problem such a tough problem? The reason is that still no efficient way is found to reliably predict the best word order. That is why many current approaches result in testing many different orders using the best one among those.
But in [Knig99] it was shown that the problem of finding the correct word among arbitrarily reorderings is a NP-complete problem. So no efficient algorithm for solving this search seems to be reachable.
This results in the need for strategies that make use of heuristics and approximations to find good reorderings.

2.1 Requirements for a reordering model

As a starting point for finding strategies to solve the problem of word reordering in MT, it is important to get an idea what these strategies should be capable of. The goal is to find a way to model how words are reordered when a sentence is translated from one language into another.
The first requirement that comes into mind is the obvious one, that the resulting word order should be as close to the true word order as possible. That includes mechanisms that distinguish between different reorderings and favor those that are more probable and closer to the reference. As a side effect those reorderings that are very unlikely should be penalized.
Furthermore the model should be general. That means the model should be able to find the correct word order even when there are word sequences that have never been seen in the training corpus (however the words themselves should be known, since otherwise there would be no translation at all).
Another part that has to be thought of is the integration of the reordering model into the translation process. It has to be fast and easy to use.
The speed of the reordering comes even into more account regarding future applications of MT like simultaneous speech translation. There it is very important to

minimize the delay between the spoken sentence and the output of the translation system.
And finally, one aspect that is also very important, is the adaptation to new language pairs. The reordering model has to be adaptable to new languages without being completely rewritten. Only then is it possible to use the same system without bigger changes in different situations.

2.2 Reordering types

What are typical reorderings seen during the translation process?
This question should be used as a base to figure out which approaches can be used to deal with the reordering problem.
In general there are two different types of reorderings that occur during translations: local and long range.
Even though there is no clear line between these two types, I will use the following way to distinguish between them in this thesis.
A local reordering is a reordering in which all components are known. Usually these reorderings take place within a small window of words and the new word order applies exactly to that sequence of words. An example for such a rule is given in Table 2.1, where the word order from English to Spanish has to be swapped for the adjective noun sequence.

Example: ADJ NN $\rightarrow$ NN ADJ
An important agreement
Un acuerto importante

Table 2.1: Example for local reordering in translating Spanish to English

By long range reorderings however I am referring to reorderings that span a longer part of a sentence, and words are moved without specifying all words taking part in the reordering. This occurs very often if the underlying structure of sentences in the two languages is different, like in English which is a SVO[1] language and Japanese which has a SOV[2] structure. But even between two languages of the same type, like English and German, long range reorderings exist (Table: 2.2). In this example

Example: auxiliary verb and infinite verb
Ich *werde* morgen nachmittag ... *ankommen*
I *will arrive* tomorrow afternoon ...

Table 2.2: Example for long range reordering in German to English translation

nearly arbitrarily long word sequences can be inserted at the "...", but there is always the need to move the verb *ankommen* to the auxiliary verb *werde* on the English side.
While the first type of reorderings, the local reorderings, is well specified, the chance is high that the same underlying principle for a long range reordering is seen in many

[1]Subject Verb Object word order
[2]Subject Object Verb word order

different ways throughout the training corpus. As a result of that, the way to deal with those long range reorderings will be more difficult and needs a generalization that is not needed in the same way for the local reorderings.

2.3 Approaches for reordering

In principle there are three possibilities for choosing the moment the reordering is performed.

1. The first possibility is to reorder the source sentence in a preprocessing step before the sentence is given to the decoder for translation. Therefore, the word order in the source sentence is changed to be closer to the word order the final translated sentence will have.
   The benefits of this approach are the following: The decoding process will be monotone, so the decoding will be more efficient. Furthermore the decoder does not need to do any accounting for the reorderings. This will lead to more efficient decoding.
   But there are also counter arguments. One problem is that this requires a deeper analysis of the source sentence and usually requires other sources of information. Also there are reorderings that will not be covered by that approach. Those are reorderings that occur, for example, during translation from English to German. In the German language it often occurs that a verb prefix is detached from its verb while the same is represented by a single verb in English. Here the reordering also leads to a problem in the phrase extraction.

Example: detached verb prefix
I *arrive* ...
Ich *komme* ... *an*

   Table 2.3: Example for movement of detached verb prefix in English to German translation

   Since the system I use and most of the current systems do not extract any phrases with gaps, two translation errors will probably be seen during translation: the first is that the word is only translated into either the detached prefix or the word, but not into both. So the translation will miss the other part. The other error that may occur is that the word is translated into the phrase consisting of both, the prefix and the verb, but since it is a whole phrase nothing can be ordered between it.
   Also a strict application of this approach results in hard decisions that are made before the translation process, and so, leads to errors produced by those hard decisions.

2. The second possibility for reordering is to perform the reordering during the translation process. This is done in most current MT systems, but since every additional knowledge source used during decoding generates an additional overhead there has to be a tradeoff between speed and usage of information. Various approaches have been made for the usage of additional knowledge sources for the reordering during decoding, like the block orientation model

[Till04].
On the other hand, completely omitting additional knowledge will result in less effective pruning which results in a need for a smaller search space. Usually the pruning methods used for that restrict the reorderings to a certain windowsize within that reorderings are allowed.

3. The third reordering approach that is thinkable takes place after the search in the decoder. For example, by performing a reranking on the N-Best list or by applying syntactic motivated reorderings to get translations that are more correct [NiNe01]. One recent implementation of that was done by [ChCF06] and showed nice improvements. However the problem with this approach is that it relies on the correct choice of words during the decoding process.

Another dimension of the reordering strategy is the information that is used for the translation process. The first machine translation systems and also some recent systems, do not use any information at all. The main decisions regarding whether a reordering is good or bad are done by the language model.
The next step is to use a shallow analysis of the language via lexical information like what is done in lexicalized block reordering [Till04].
It is also possible to make a deeper analysis with more linguistic tools like POS-tagger and chunker or even dependency parsing and semantic information. In the recent publications ([ChCF06], [CrMa06], [PoNe06], ...) these were shown to give significant improvements.

2.4 Idea of lattice reordering

In this thesis I will follow the idea of using a lattice structure for the reordering itself as it is proposed in [CrMa06].
By using a lattice structure for the input of the translation process it is possible to keep different reorderings of the source side and give them as input to the search process later, without losing any information from the original sentence. Approaches that simply reorder the source side and give this new resulting source side to the translation search process suffer from the lack of reordering alternatives, since the probability for a falsely applied reordering still remains very high. As a result of encoding the different reorderings into the lattice structure the translation process becomes more robust compared to a single reordered source sentence.
So this approach can be seen as a combination of reorderings taking place before decoding and during decoding time. While the decoder does not need any additional accounting for the reorderings[3] still the reordering decisions are made during decoding time when a specific path in the lattice is chosen as a translation.

The idea of how the reordering of the source sentence should be learned is the following. For the training of an SMT system a bilingual corpus is already needed. Most

[3]of course the accounting for the translation lattice is more complex than for a simple sentence without any reordering, but since the decoder I used already works on lattices that store the different phrase translations this only results in a bigger translation lattice for which the strategies remain the same

systems use nearly no data from this corpus for the reordering process, except those reorderings that are part of a phrase pair and those reorderings that are solvable with a sufficiently strong language model. However the bilingual corpus contains alignment information which represents the mapping of words between those languages. This alignment information can be used to learn rules that cover the reordering.
The approach suffers however from the data sparseness problem. Most of the word n-grams in the translation process are not contained in the training corpus. Without further preprocessing, this would mainly result in the learning of reorderings for word sequences for which phrases exist that also cover the reordering. One solution for solving this problem is to use classes assigned to the words, instead of the words themself. These classes can be syntactic information like parts of speech (POS) information, word clusters or even information from chunk parsing can be imagined as a base for the reorderings.

The techniques described so far result in one problem. When you reorder the source side before the translation process the obtained new input sequence does not match the sequences in the original source corpus. In more detail this means for the translation process that the phrases obtained from the training corpus will not cover the reordered source sequences in the reordering lattice as well as they do in the original word sequence. This is another aspect of the reordering approach that has to be kept in mind.

3. Strategies for reordering

3.1 Lattice reordering

In this thesis I focus on the reordering of the source side. I encode the alternative reorderings in a lattice structure [CrMa06] so that there is no loss of information by making hard decisions in the preference for one reordering. The complete translation process using this reordering approach consists of different actions performed in the following steps.

- The first step that has to be completed is to find the underlying principles which determine what reordering should be used. Since I will use a rule based approach for the reorderings, I will refer to this process as the learning of the rules.
- The next step is to create the lattices for a given source sentence that encodes the different learned reorderings that can be applied to the source sentence.
- The final step is the actual translation process. In this process the best reordered source sequence in the lattice is used to translate the sentence. By deciding on a particular path in the lattice, the reorderings that are used for the translation are fixed.

3.1.1 Learning of the reorderings

I will start by describing the first aspect of the reordering using a lattice structure for the source side, which is the learning of the actual reordering rules.
The approach for learning the reorderings follows the idea in [CrMa06] to use reordering rules that can easily be learned from a word aligned bilingual corpus. Since it is a rule based approach for the reorderings, this implies a restriction over testing all possible permutations as reorderings. This results in an improved runtime behavior, because not every permutation is tested, but still allows for longer ranging reorderings.

The word aligned bilingual corpus is used to identify typical reorderings that occur in the translation process between these languages. However it makes no sense to just rely on the corpus with the actual words and sentences, since this corpus will hardly contain every word sequence of the source sentence that will be seen in the translation process later. The idea that was proposed in [CrMa06] is to use parts of speech (POS) information on the source side in addition to the words in the source corpus. With the use of the alignment information of the bilingual corpus, it is also possible to align the word classes to the target side. The same applies for the case of chunk parse information, but since this introduces other problems I will deal with it in a later section.

As a result, the reorderings observed from the training corpus can now be identified based on lexical word information or word class information and the permutation that takes place.

Formally this means that for a single sentence of the aligned bilingual corpus, there is a sequence of words $f = f_1 f_2 \dots f_J$ in the source language and the corresponding POS tags of the source sequence $\tilde{f} = \tilde{f}_1 \tilde{f}_2 \dots \tilde{f}_J$. In addition to that, the corresponding translation of the sentence $e = e_0 e_1 e_2 \dots e_I$, where e_0 represents the *NULL* word and the alignment function $a : I \rightarrow J$ with $i \mapsto a[i]$ is available.

One thing that should be kept in mind, is that using the POS information requires the use of a reliable tagger to generate it. These taggers are already available for many common languages, but when working on these common languages, a question arises why not using even more information like chunk parse information? In the chunk parse for a sentence there are even more complex structures represented besides just the function of a single word like complete verb phrases or noun phrases. The use of chunk parse information leads to even more general reordering possibilities.

On the other hand one can argue that the use of POS information also gives a restriction on the translation by reducing the reordering techniques to well explored languages, for which these tools exist. But in one application of machine translation, humanitarian aid, you very often have to deal with languages spoken only by a few people without any linguistic tools available for that language.

For this case I will present techniques to cluster the source language words, showing that the same techniques are usable as those for languages in which tools like POS tagger exist.

The question that arises is how to use these resources to find the reordering rules. This is done via the observation that a reordering of words takes place whenever there is a crossing in the alignment information. That means whenever there is an i and a j with $i < j$ and $a[i] > a[j]$. An example for such a situation is shown in Figure 3.1.

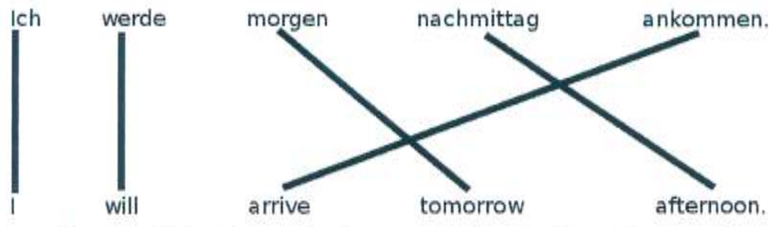

Figure 3.1: Example for a crossing in the alignment

He will come.
Er wird kommen.
He says that *he will come.*
Er sagt, dass *er kommen wird.*

Table 3.1: Example for different word order in different contexts

The reordering that can be learned from such a situation is the reordering including the words from f_i to f_j. The essential question is what exactly is stored as a trigger for the reordering rule? In [CrMa06] the exact POS sequence is used. But in some cases this does not cover the situation well. The problem with this approach is that the word order in some languages depends on the context situation in which the word sequence occurs. In Table 3.1 an example is shown, where exactly the same English word sequence is translated into different word orders on the German translation side. The problem that arises from the approach in [CrMa06] is that in both cases the same word order for the reordered sequence would be preferred, even though the target language requires different word orders.
When looking at the example, a very simple solution comes to mind. The idea is to use the additional information that is provided by the sentence. It will happen very rarely, that the sequence *He will come* becomes reordered in the target language German, when the sequence occurs at the beginning of a sentence. On the other hand, the reordering becomes very probable when the sequence *he will come* is seen directly behind the word *that*.
This observation leads to the idea that an already very small context of words used in addition to the reordering sequence should provide the information of which reordering is more useful in that situation. Therefore, in addition to the rules as they were proposed in [CrMa06], I decided to use additional context information to indicate the situation of a reordering.
That results in the following building blocks of a reordering rule learned from the bilingual word class annotated training corpus. The first part - the left hand side[1] - of a reordering rule describes the situation on the source side which triggers the reordering. This situation is not necessarily described only by the POS sequence of the words that are reordered. I extended the left hand side to also be able to consist of the context in which this sequence is observed. The following contexts were used in this work, as they already reflect many different language properties.

- Sequence of POS tags that get permutated: This is exactly the same rule description as it was originally proposed. Rules of this type cover those reorderings that are very general like the inversion of word order for the "ADJ NN" sequence when translating between English and Spanish.

- Sequence of words that gets permutated: This rule type is a very specific rule type. The idea behind the reordering of exact word sequences is to get those reorderings of words that occur very often correct.

[1] I will use the same naming convention as in [CrMa06]

- Sequence of POS tags with one or two additional POS tags or words to the left or right: This reflects the situation of the problem shown in Table:3.1 and tries to cover it
- Sequence of parts-of-speech tags that gets permutated with additionally the tag to the left and to the right: The idea behind this is to reorder those parts that are enclosed in a bracket structure like subsentences.

The second building block of a reordering rule is the permutation that is performed on the sequence that gets reordered. This is nothing special since this part, the right hand side, directly stores the output order of the reordered word sequence.
And finally, the third building block is a score that is assigned to every rule. This score is used as an indicator of how probable the particular reordering rule is for the translation. The way the rules are scored can be done in many different ways. In this work I tried relative frequencies and the language model probabilities of a reordered source language.
To conclude with the description of the reordering rules I give a small example for how the reordering rules with different types of context look like (Table: 3.2).

source sequence	rule	freq.
PDAT NN VVINF	3 1 2	0.40
VVFIN :: PDAT NN VVINF	3 1 2	0.71
moechte :: PDAT NN VVINF	3 1 2	0.92

Table 3.2: Example rules for German to English translation with no context, with one tag of context to the left and one word of context to the left

3.1.2 Lattice construction

After the learning of the rules, they have to be applied to the source sentence and the alternative word orders have to be stored. As already mentioned before, it is not sufficient to just generate the most probable reordered source sentence, since this would include hard decisions that are not reversable later in the translation process. This is why it makes sense to keep the alternative source word orders and use the information contained in them in the translation process.
The way I chose to solve this problem, is to encode the different reorderings into a lattice structure as proposed in [CrMa06]. This lattice structure is then usable as the input for the actual translation process.

The process of the lattice construction works in the following way: First of all, since the reordering rules are based on additional information (POS tags, chunk parsing or word clusters), the sentence that is to be translated is analyzed to obtain that information for the sentence. Then the sentence is used as a first monotone path in the lattice. Every word in the sentence represents an edge, and the first node of the first edge represents the start. Every possible reordering is reflected by a path from the first node to the last node. An example for such a starting

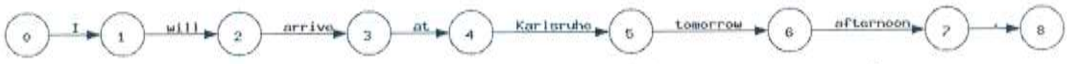

Figure 3.2: A lattice consisting of the monotone path

lattice is given in Figure: 3.2. Then the reordering rules are used to create the full reordering lattice. This is done by testing for every subsequence of the sentence, whether there is a rule that applies to the situation that is seen in that subsequence. If such a subsequence is found, a new path for every reordering on that subsequence is inserted at the corresponding nodes (see Figure 3.3). After doing this for every

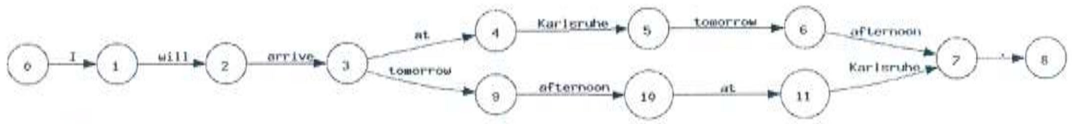

Figure 3.3: A lattice consisting of the monotone path and one Reordering

possible subsequence of the sentence, the final reordering lattice is complete. One possible result for such a lattice is shown in Figure 3.4. This lattice however is a

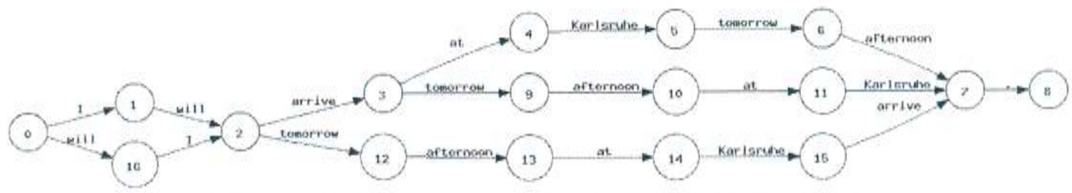

Figure 3.4: Lattice containing all possible reorderings

very small example. The lattices created in this process can grow very large and they can get much more complex than this example.
From the above presentation the question arises whether it makes sense to perform a recursive application of rules - applying reorderings on newly inserted reordered paths. This is dependent on the type of information that is used for the learning of the rules. When the classes that are used as a generalization for the words do not contain any hierarchical structure and always cover only a single word it makes no sense, since the reorderings include every part of a word sequence and so longer reorderings always cover the smaller reorderings.
If, however, there is a hierarchical structure in the classes as is the case when chunk parse information is used, then there is a use of recursive application of the rules on already reordered paths.

3.1.3 Assigning scores to the edges in a lattice

In the lattices constructed so far, there is no preference for one path over another, except for the preferences induced by those resources used during the translation process. This means that the decoder for the translation process has to rely on the information that is provided to it, without further information concerning the actual reorderings.
But I learned for every single rule its relative frequency and this information can be used in the lattice construction to provide the decoder with information about the

reorderings. This is done by assigning these scores to the corresponding edges of the lattice. It is not sufficient however, to just use the new scores and assign them to the first edge of every reordered path. The score of the reordered path and the score of the original monotone path both have to be modified to make them comparable to each other.
Since we are using the relative frequencies, the sum over the scores of all outgoing edges has to be 1.0. That means that the score of a reordering in the lattice is its relative frequency seen in the training corpus weighted with the remaining probability mass of the subpath where it takes place. More formally: for a reordering r via the m'th of n applied rules from node l to node r for this subpath:

$$Score(r_m^{l,r}) = ProbabilityMass^{l,r} \cdot P(r_m) \tag{3.1}$$

Where $ProbabilityMass^{l,r}$ is the probability mass that is remaining for the monotone subsequence from node l to node r. The effective score for the monotone path then computes

$$Score(monotone^{l,r}) = ProbabilityMass^{l,r} - \sum_{i=1}^{n} Score(r_i^{l,r}) \tag{3.2}$$

It is important to note, that the order of the rules according to their length is substantial. Since smaller reorderings take place on a subpath of a longer reordering, these reorderings have to be applied after the longer reorderings.

3.1.4 Translation using the reordering lattice

After the creation of the reordering lattices, they have to be translated. Since the original translation system already uses a lattice structure to encode the different translation possibilities for the phrases, the use of lattices did not introduce a big difference, except that with the reordering lattices, a lattice structure already existed, in which the phrase translations had to be inserted.
Since this functionality already exists, there was no further change needed and the reordering lattices were usable as they were.

3.2 Reordering of the training corpus

The reordering method presented so far poses a problem that did not exist before. When the source sentences are reordered and they are encoded into a lattice structure with assigned scores, the translation process will prefer paths in the lattice that tend to be in different word order than the original sentence. This is desired behavior, but it should be kept in mind that the translation process uses resources that do not fit this situation very well. Namely, the phrases for the translation process will provide some problems since they are extracted using the original bilingual corpus. This original corpus, however, differs in the word order to the more likely paths through the lattice, since it is not reordered.
One idea that was originally mentioned by [PoNe06] is to reorder the training corpus of the source language and then use phrases extracted from this better matching corpus. In the situation described in [PoNe06] the reorderings mainly consisted of

one single rule, and there was no lattice used for the encoding of different reorderings. In my system there is quite a big number of different reorderings that represent different types of reorderings. Therefore it is not possible to give a unique reordering for every sentence.
Two ideas come into mind concerning this reordering problem. The first is not to use the reordering rules at all, since a much more monotone alignment for the bilingual corpus should be achievable when using the already existing alignment information. So the approach is to use the alignment information and output the words of the source in an order that would monotonize the existing alignment.
The other idea is to use all the different reordering rules and apply them to every source sentence to create normal reordering lattices. But then instead of translating the lattice, the best paths according to the scores in every lattice are used to generate a new source corpus.
Both approaches should generate a source corpus that is more monotonously aligned to the target side than before, and so it should better match the situation seen during the translation process with the reordering lattices. The question that arises is: which of these described techniques works better? The reordering using the alignment information should create more monotone alignments, while the reordering using the reordering rules should generate a source corpus that is more similar to the word sequences seen during the translation process.

3.3 Using a language model on the source side for reordering

Another strategy to find a reordering on the source side that generates a better translation is to use a source language model for the reordering process. This language model is not based on the original source corpus, but on a reordered one.
So, as described in the previous section, there are different ways to reorder the source corpus. The first way is to reorder it using previously extracted reordering rules, and the other way is to reorder it using the alignment information.
The technique of how to use this new language was shown in [ZhZN07]. There the source language model was used to score the reorderings obtained by reordering rules encoded in a lattice structure as described above. Another approach to making use of such a language model is to use it for scoring all possible reorderings itself, without prior extracted rules that generate the reorderings. In both cases the technique is nearly the same.
The idea is to use a language model and score every edge on every path in the lattice that contains the reorderings. While in the first case the lattice only consists of those reorderings for which a rule exists, in the second case every possible reordering (up to a given length, otherwise the runtime would not allow a complete search) is inserted into the lattice.
This approach introduces one problem. Since the scores are assigned to lattice edges, it is possible that there are different paths leading to the node from which the edge starts. To make matters worse, this happens very often. A small sample lattice where this happens is given in Figure 3.5.

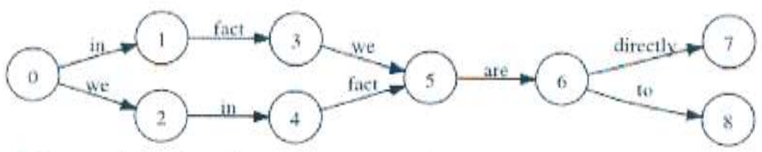

Figure 3.5: Lattice with more than one history for edges

In the given example, the history of the word *we* on the edge from node 3 to node 5 is well defined, since it is *in fact*. On the other hand, the histories for the word *are*, *directly* and *to* are not well defined and depend on the path you came to node 5. So the source language model score, even when a bi-gram is used, is not well defined. This problem does not exist when integrating a source language model into the translation process, since during decoding time the history is well known.
One possible way of dealing with this is to use the minimum score over all preceding edges as history. By using the minimum instead of the maximum, the hope is that in the later search the maximization of the language model probability results in using the best edge, even if the worst history that is possible for that edge is chosen on this path.
Another way to deal with that is to abandon the use of lattices and only deal with single reordered sentences (of course this approach can be extended to dealing with the n best possible reorderings [SaCa07], but results in increased translation time). In that case the best way according to the reordered source language model through the reordering lattice is searched. During search the exact history of a word is always known and so the best reordering according to the reordered source language model can be generated. This reordered sentence can then be used as input to the translation process and should also be translatable in a more monotone manner than without that reordering.

3.4 Reordering for rare languages

In this section I want to propose strategies on how to deal with languages for which tools like POS tagger and chunk parser are not available. This is especially the case for languages only spoken by a minority. The interest in dealing with these languages stems from the fact that they are often needed for humanitarian aid projects.

3.4.1 Reordering for rare source languages and common target languages

The first approach I want to introduce is based on the observation that while the source language is rare, often the target language is a common language and offers the possibility to make a deeper analysis with linguistic tools like a POS tagger.
The idea is to use the alignment which exists and a POS tagged target side to project the POS tags onto the source language. After that, a corpus of the source language with assigned POS tags is obtained. This can be used to build a new tagger for the source side.
The problem that comes with this approach is the underlying assumption that the

translation of words also carries the function of the words over into the other language. But even if that is not the case, there is still the hope that the constructed tagger still assigns words that have the same reordering behavior to the same class.

3.4.2 Using clustering techniques

Although with the projection of the POS tags from the target language onto the source language a way of annotating the source language is given, it still makes sense to search for further techniques without the need for a tagger on the target side. The main reason for this is, if there is a way to deal with the reordering problem without the need for such tools for the analysis like a POS tagger, the system becomes more and more independent of the actual speech pairs that it has to translate [2]. It is not clear yet which approach is better.
Also, the tagging nearly always means a bottleneck for the speed of the translation system. Many taggers need a significant amount of time for starting and some taggers are also very slow at tagging, so the use in a time sensitive application like simultaneous speech translation is virtually impossible.

A solution for this problem is to use a clustering technique that takes the reordering behavior of the words into account. With clusters extracted during training, it would be very easy and fast to simply convert a sentence into the clustered sequence and then applying the techniques that work for POS based reorderings to the clustered text.
The challenge with this approach is how to cluster the words. Fortunately, clustering of data is a very famous problem in the machine learning theory and many techniques have been introduced over the years. Already existing for clustering of the words in a corpus is the MKCLS clustering [Och99] that minimizes the language model perplexity. It is also possible to use k means clustering while using the observed reordering behavior of the words. And finally it is possible to search a clustering motivated by the minimization of a criterion function for the reordering, similar to the approach used in [Och99].

3.5 Long range reorderings

The techniques described so far all deal with relatively small reorderings. The reorderings that are learned are restricted by the fact that the whole sequence of the reordering has to be known. This is a problem because of data sparseness. As a result of that it is desireable to learn more generalized reorderings that allow reorderings of sequences not seen in the training corpus. This is especially the case when dealing with long range reorderings.
One way of being able to cover those reorderings has already been mentioned, although not in that context. Using chunk parse information automatically allows longer reorderings because of the more general content a single chunk can have.
On the other hand, one can think of a generalized tag that reflects more than one POS tag at once. For example, a tag with the meaning of a kleensche star which represents arbitrary sequences of tags. So rules like the one in Table: 3.3 are possible. The intention of that rule is that whenever there is a subsequence of POS

[2]Of course that does not mean that the training for specific language pairs becomes obsolete

VBAUX * VBINF	0 2 1	0.2

Table 3.3: Example for a rule with a kleensche star

tags that start with an auxiliary verb, then an arbitrary sequence follows, and at the end of the subsequence there is a VBINF, then the VBINF is moved over the whole sequence represented by the "*"

3.5.1 Integration of long range reorderings into the lattice

The problem with the approaches for long range reorderings is that they need additional reordering after the long range reordering, since no local reordering has been performed. As a result of this the application of these reorderings onto the source sentence should be performed in two steps.
In the first step, the long range reordering rules are used to insert those paths representing long range reorderings into the lattice. In the next step all other reordering rules are applied to every path in the lattice.

3.5.1.1 Reordering using information from chunk parsing

The approach of using chunk parse information for the reordering of the source side and creating reordering lattices has been proposed by [ZhZN07]. The idea is to use the parse tree generated by a chunk parser and learn reorderings based on the branches of that tree. Then these learned reorderings can be applied to the text that has to be translated.
The reorderings are learned nearly the same way as described above for the word class based reorderings. But in this case a chunk gets the union over all alignment points of its words assigned. Again a reordering gets learned whenever there is a crossing in the alignments. This time a crossing occurs whenever there is a chunk c_i consisting of the words $w_1^i \dots w_m^i$ with alignments $a[w_1^i] \dots a[w_m^i]$ appearing before a chunk c_j consisting of the words $w_1^j \dots w_n^j$ with alignments $a[w_1^j] \dots a[w_n^j]$ on the source side and $\forall r = 1..m : \ \forall s = 1..n : \ a[w_r^i] > a[w_s^j]$.
The tree structure, however, already leads to an extension over the idea proposed in [ZhZN07], since they did not allow for the hierarchical application of reordering rules. The problem with this is that the nearer to the root a reordering of subtrees takes place, the fewer local reorderings will be performed. This is counterintuitive, since a subtree that is nearer to the root usually represents a longer subsequence.
Therefore a hierarchical application of the rules is also used in this thesis, and this allows further reorderings in subtrees that are part of a reordering on a higher level in the tree. The difference in the resulting lattice structure can be seen in 3.6 and 3.7.

One problem with this approach is that the hierarchical application should be stopped if the depth of recursion is too deep. Otherwise the lattice would grow to unusable proportions and would result in excessive memory usage and long translation times. An example for such a huge lattice can be seen in 3.8. Therefore, it makes sense to restrict the depth of recursion during the construction of the lattices.

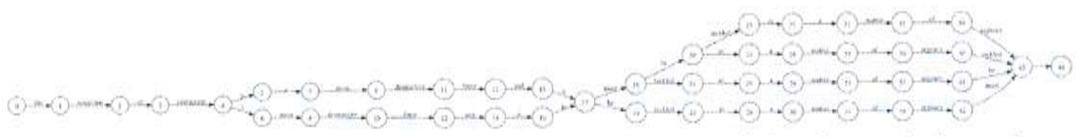
Figure 3.6: Lattice without hierarchical application of chunk-reordering rules

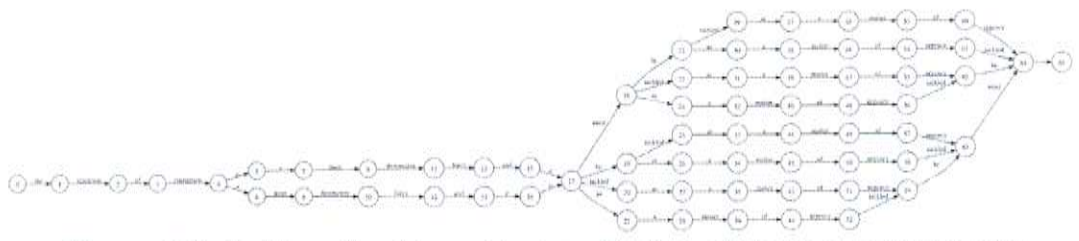
Figure 3.7: Lattice after hierarchical application of chunk-reordering rules

3.5.1.2 Combination of chunk based reordering with POS based reordering

The idea of combining both reorderings of chunks and reorderings with POS based rules is to overcome the weaknesses of each approach. The main weakness of POS based reorderings is the lack of generality. POS based reorderings will only work well on short range reordering since they have enough training data for that. For long range reorderings, however, they will not be very useful, since the data sparseness does not allow training for this. On the other hand, the chunk parse information based reordering approach does not allow reorderings which mix up two different chunks. Here the reorderings will only take place in chunks, not over any border. But sometimes it is important to have exactly these reorderings.
This should be solved, or at least minimized, when in addition to the chunk reordering, a POS based reordering takes place.
The way this can be done is to construct a chunk based reordering lattice as a first step. As the second step every path in the lattice is considered and the reorderings based on the POS rules are applied onto the path.
It is important to remember that the reorderings caused by the chunk reorderings will multiply the subsequences where the POS reorderings can be applied. So the lattice structure will grow very large with this approach. Also important is the method of scoring the lattice edges, since POS based reordering scores are not comparable to those caused by the chunk based reorderings.

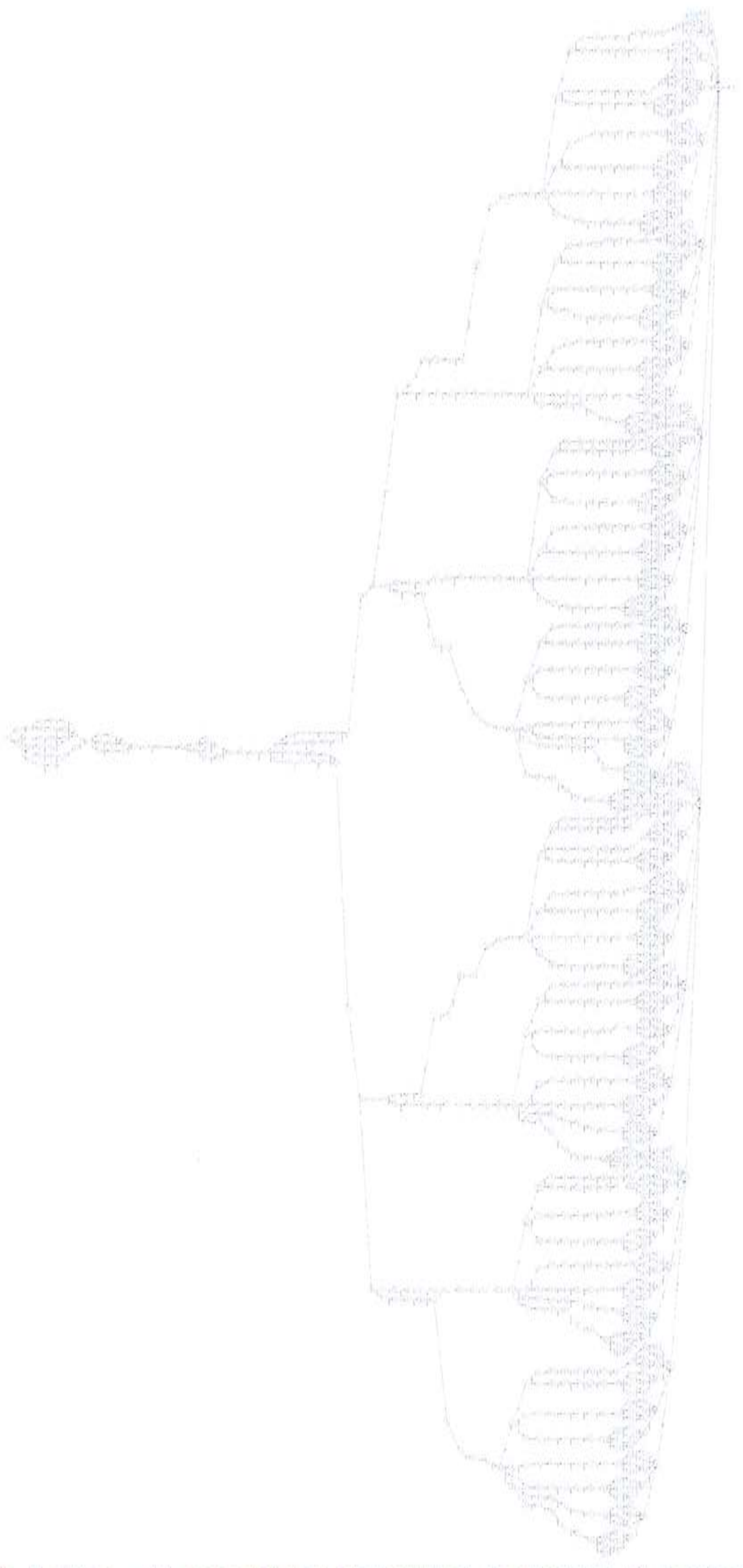
Figure 3.8: Lattice with unrestricted hierarchical application of chunk-reorderings

4. Implementation details

This chapter is intended to give the details and the problems that occur when implementing the previous described techniques. I will present pseudocode to make the way I implemented the techniques more understandable.

4.1 Reorderings based on word classes

As a starting point I will describe the approach, how the reordering works, when a class is used for every word and when there is no grouping of words into larger chunks.
This is the case when parts of speech tags or classes for the words are used, and every word in the source gets assigned to exactly one tag or class.
That already introduces the first step for the learning of the rules, that is to annotate the source corpus. It is crucial to have exactly one annotation for every source word, otherwise shifts in the rules would occur and lead to wrong reorderings.

4.1.1 Learning word class based reorderings

The learning of the reordering rules then starts with an iteration over the corpus and testing for every sentence and every subsequence, whether it contains an alignment crossing. As already mentioned above that is the case for a source sentence $f = f_1 f_2 \dots f_J$ with corresponding target sentence $e = e_0 e_1 e_2 \dots e_I$ and the alignment function $a : I \to J$ with $i \mapsto a[i]$, whenever there is an i and j with $i < j$ and $a[i] > a[j]$. This is the same way to learn the rules as in the previous work by [CrMa06]. I decided however to modify this approach a little bit, since it results in a lot of rules that are unnecessary for the translation process. I store only those rules, that are seen without being part of a larger reordering. Every occurrence - that means also those that occur as part of a larger reordering - is counted only for the computation of the relative frequencies.
That results for the learning of the different rules in the following algorithm:

```
For every sentence in aligned bilingual corpus
    For i = 0 to source sentence length do
        For j = source sentence length down to i do
            if a[i] > a[j]
                 store rules for the sequence between i and j
                 set i = j
            end if
```

After this iteration over the bilingual corpus, the different rules of the different context types are stored. However the problem is, that with that first iteration you do not know the exact numbers of the occurrences for every rule, since it is only seen, when it was not part of a larger reordering. So another iteration over the corpus is performed, that counts for every reordering rule the number of occurrences. This is done by a very small modification of the above given algorithm:

```
For every sentence in aligned bilingual corpus
    For i = 0 to source sentence length do
        For j = source sentence length down to i do
            if a[i] > a[j]
                 if corresponding rule for sequence i to j exists
                      increase counter for that rule
            end if
```

It is not useful however to keep all those rules that were seen only a few times. For these rules it is more likely that they were caused by misalignments or wrong class associations of the words. So I decided to throw away those rules seen only below a given threshold.
This results in a table with all reordering rules and their absolute frequencies. The next step is to compute the relative frequency for every rule.
This is possible in a very time efficient manner if a suffix array is used for indexing the corpus [ZhVo06]. The suffix array allows for requests that give the absolute number of occurrences of an n-gram in the corpus. So for every rule the occurrence count of the triggering part is requested from the suffix array and used for the computation of the relative frequencies.

```
For every rule R
    N = # occurrences of left hand side
    A = # absolute occurrences of R
    relative frequency of R = A / N
```

One problem that comes into account due to the use of different contexts is that the left hand side does not necessarily come from one single corpus, but also consists of words from the source corpus and the corresponding tags. So there is no suffix array available for that information.
A quick way of solving that problem is to combine both suffix arrays and compare the results of requests for the corresponding parts in both of the suffix arrays. This

however leads to a higher runtime. An alternative to that is to use a new modified suffix array that uses both corpora for the indexing and sorts the sequences according to the context that is used. Since the counting is only used once for every language pair, I decided to use the first approach even though it has a worse runtime behavior.

4.1.2 Applying word class based reorderings

The application of the class based reordering rules learned from the aligned bilingual corpus is straightforward. Starting with a lattice that only contains the monotone path of the word sequence every edge on that path gets a score of 1.0. In a previous step the source sentence was annotated with the word classes corresponding to its words.
Then, beginning with the largest subsequences, on that monotone path every subsequence is tested if an reordering rule exists for it (using the additional information from the corresponding annotation of the words). If so, a reordered path is inserted, while the score of this new inserted path equals the score of the reordering rule weighted with the score of the monotone path.
After applying all rules of one length, the monotone path score gets adjusted with the remaining score on that path and it is continued with the next smaller subsequence. The base algorithm for that is:

```
For i = 0 to source sentence length
    For j = source sentence length downto i
        NewMonotoneScore = oldScore(MonotonePathEdge(i,i+1))
        For every rule r, that reorderes source sequence from i to j
            Insert reordered sequence as new path
            Score for new Path = Score (r) * oldScoreEdge(i,i+1)
            NewMonotoneScore = NewMonotoneScore
                                 - (Score(r) * oldScoreEdge(i,i+1))
        oldScore(MonotonePathEdge(i,i+1)) = NewMonotoneScore
```

The problem with rules using different kinds of contexts however is, that the scores do not sum up to 1.0. The question is how to combine rules of different types into one system. The first approach of using the scores for every type as an individual score is not useful at all. There are situations, when one kind of context prefers a reordering and another kind of context assigns a very low score to that reordering. For example as in Table: 4.1. The problem is that the rule that only uses the tag annotation as information assigns a very low score to the reordering compared to the rule that uses a word to the left as context information. So in this case that would result in a preference for the rules that make use of one context tag. On the other hand there are situations, where the distribution of scores for the rules is in exactly inverted order. Then the preference would be for those rules without context.
This combination would lead to the problem that it is not clear how optimize the scaling factors for the rules and probably it would result in a distribution that prefers only one single rule type, the one that improved the translation the most.

But there is a way to solve that dilemma. It is a combination of all the different scores of the rule types into one single score by taking the maximum of the scores

source sequence	rule	freq.
PDAT NN VVINF	3 1 2	0.40
VVFIN :: PDAT NN VVINF	3 1 2	0.71
moechte :: PDAT NN VVINF	3 1 2	0.92

Table 4.1: Example rules for German to English translation with no context, with one tag of context to the left and one word of context to the left

for the same reordering.
At the same time the different scores for the monotone path are computed for every different rule type, as if only that rule type existed. But when the final score for the monotone path is to be stored, the minimum over all those computed for the monotone path is used.
This results in the desired behavior that a reordering that is very likely considering one rule type gets a higher overall score, while those reorderings that have only a low relative frequency for every type of context get a lower score.

4.2 Chunk based reordering

The difference between word classes and chunks is that in a chunk more than one word can be combined, while a class token in a sentence always represents a single word. The problem with that is that the above described learning of reorderings has to be modified.
The situation gets even more difficult when there is a hierarchical structure in the chunks.

4.2.1 Learning of chunk based reorderings

The first step of dealing with chunk parse information for reordering is again the task of identifying the reordering rules. This time however the problem is a bit more complex than before, since every chunk can contain more than one word at once and to make things worse, it can also contain further chunks. The easiest way of dealing with chunk parse information is to interpret it as a tree structure. One example parse obtained from the corpus of the source side is given in Figure 4.1.

With the knowledge of the structure of a chunk parse, it is now possible to learn reordering rules from that. Again the bilingual corpus is already word aligned. Now the task is to transform the alignment information for the use in the parse tree. Therefore every node in the tree gets the alignment points corresponding to all of the words in the subtree under the node.
A rule is learned in that case, when on one level of the tree, there are nodes, for which a complete swap exists. That means a reordering takes place between the set of nodes $n_1 \dots n_k$, with every node n_i is aligned to $a[i_1] \dots a[i_{max}]$, when $l < k$ and

$$\forall i \in l_1 \dots l_{max} : \forall j \in k_1 \dots k_{max} : a[i] > a[j] \qquad (4.1)$$

No reordering is learned, where subtrees under one node in the parse tree are mixed together. But on the other hand the reorderings are learned for every level of the

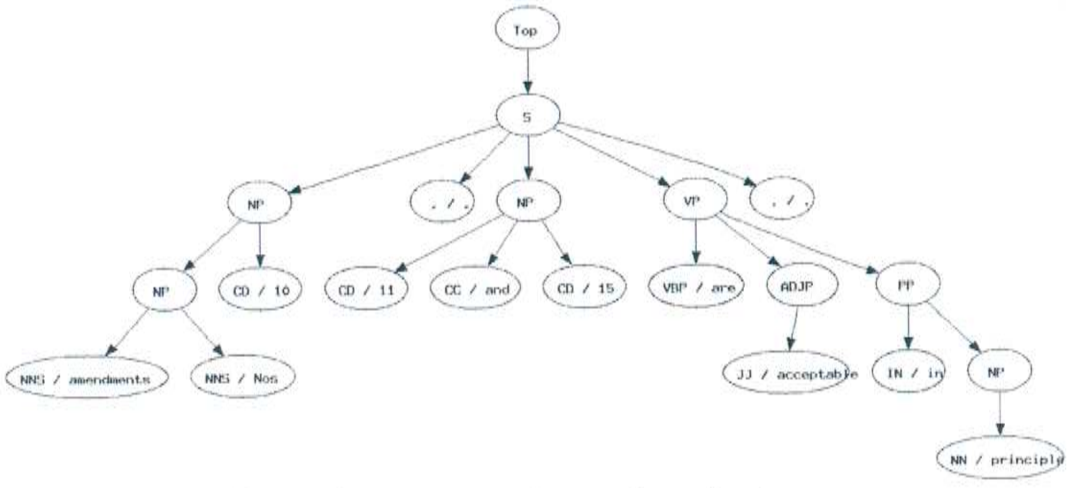

Figure 4.1: Example of a tree for a chunk parse

tree.
To speed things up, it is not necessary to store every alignment point for every node, but every node only needs to know its largest and lowest alignment point. So the comparisons in the above equation are easily reduced to 1 comparison. Then the pseudocode for the learning of the reordering rules can be written down as follows:

```
learnRulesUnderNode (n)
  For i = 0 to MaxChildNode(n)
    For j = MaxChildNode(n) downto i
      If MaxAlignmentPoint (n.child[j]) <
                             MinAlignmentPoint (n.child[i])
            Store reordered sequence of the Nodes from i to j
      EndIf
    EndFor
    learnRulesUnderNode(n.child[i])
```

After learning the rules, again the relative frequencies have to be computed. This time a suffix array approach is not useful, since every layer of the parse tree would have to be indexed what results in a really huge corpus and already the creation of that corpus would be time consuming. But on the other hand, there is only one type of reordering and the reorderings tend to be describable through smaller patterns than before. So it is no problem at all to just iterate once more over the corpus and count the reorderings.

4.2.2 Applying chunk based reorderings

As before for word class based reorderings the question is how to apply the reorderings onto the sentences that have to be translated to create a lattice. For that task, the source sentence is chunk parsed and internally a parse tree is created. Then for every level in this tree it is tested, whether there exists a rule for any of the subsequences of connected nodes. If so, the corresponding reordered word sequence is inserted into the lattice. The modification of the scores on the monotone paths

and on the reordered paths remains the same as before.
Another difference to the lattice creation for word class based reorderings is the fact that now it makes sense to have a hierarchical construction of the reordering lattices. The use of the parse tree however already provides the functionality, that is needed for that. So the pseudocode for the lattice creation is pretty easy to describe.

```
findReorderingsUnderNode (n)
  For i = 0 to MaxChildNode(n)
    For j = MaxChildNode(n) downto i
      If exists reordering rule for n.child[i] to n.child[j]
         Insert reordered path into lattice
         modify score of reordering according to monotone path
         For every node k in i to j
          findReorderingsUnderNode(k) on new path
      EndIf
      Modify score of monotone path
    EndFor
    findReorderingsUnderNode(n.child[i])
```

4.2.3 Combination of chunk based reordering with word class based reorderings

In the description of the rule learning process for chunk based reorderings I already mentioned, that no reordering rules are learned, where a mix of different subtrees happens. Only reorderings of complete subtrees are allowed. This however does not reflect the reality very well, since this assumption is only true if the two languages share the same underlying building blocks like noun or verb phrases. Often that is not the case and so reorderings into the block of another chunk are needed.
One solution to overcome this weakness is to combine the chunk based reorderings with POS-based reorderings. This is done by the following algorithm:

```
build chunk reordering lattice
for every path in the lattice
    for every subpath smaller given length
        apply POS-based reorderings
```

Since both reordering techniques reflect very different assumptions of the reorderings I decided in this case to use individual scores of both types and not to combine the scores of both into a new score.
Another problem that is introduced by the combination of chunk based reorderings with class based reorderings is the fact, that the number of paths on which the class based reorderings are inserted are very high. That results in an immense translation lattice and in bad runtime behavior. But there is a very simple way to prevent that and it is also a way that reflects the purpose of the different rules. The idea is to use only those class based reorderings shorter than a given length and to not apply the chunk based reorderings hierarchical.

So there is a division in the purpose of both reordering approaches. The chunk based reorderings are used to reflect the long range reorderings, while the word class based reorderings are used to get the local reorderings right.

4.3 Word Class based long range reorderings

So far the only way to get long range reorderings is to use chunking information or to hope that a long sequence was seen very often in the training corpus, and therefor the reordering gets learned.
I also tried the approach of using long range reordering simply based on the class based information. The rules that I used for that were of the form "VBAUX * VBINF - 0 2 1".
The learning of the rules is very similar to the learning of rules with one tag to the left as context information. However in this case I only stored the context tag to the left and the most right tag sequence that is reordered in the same order as on the source side to the front in the permutation.
The part in the tag sequence that gets jumped over by the most right part is substituted by the kleensche star, without regard to further reorderings that take place in this subsequence.

4.4 Clustering as an alternative to parts of speech

The previous sections in this chapter all used the fact that there are tools that provide the clustering or chunking of the data. However this is not always the case and it is especially in those languages not the case that are spoken by only a few people. As mentioned above, one of my goals in this thesis was also to provide some reordering techniques for those languages, that do not have this huge toolbox of analysis programs.
Since the above described techniques refer to word classes there is no difference in them whether those classes are parts of speech information or whether the classes were obtained another way. So what was said before can be directly used with the following approaches for the clustering of the words.

4.4.1 POS-projection from target to source language

The first approach to deal with languages that do not have a parts of speech tagger available is motivated by the fact, that often the target language is more famous and therefore at least for the target language such a tool is available.
The idea is to use the target language parts of speech information and use it on the source side via a projection using the alignment information.
Using this approach it is possible to obtain a sequence of parts of speech tags for the source side of the corpus. Now this can be used as training data for a new parts of speech tagger. A very easy approach for that is to train a translation system with that data, that translates the source words into the corresponding parts of speech tags, without any reordering enabled.
When the languages share the same underlying concepts of the parts of speech, then the hope is that the so constructed parts of speech tagger will provide the needed

information that can be used for the reordering process.
The approach however is not as easy as it sounds by now, because there are some things that have to be kept in mind. The one thing is that there are unaligned words on the source side. That will lead to the situation, that not every word in the source language gets a tag assigned to it. I deal with this problem by assigning to those not aligned words the NULL tag, in the hope that this information is somehow useful in the later translation process.
Another problem is that often there are tags on the target side aligned to more than one tag on the source side. So should that tag be assigned to only one of those words - the most probable, or should it be assigned to all of the corresponding tags? In [YaNg01] there is a discussion about that concerning the construction of a tagger. Since every untagged word will probably decrease performance, I decided to multiply the tag on the source side and add the count of the word in that tag to the new tag as it was proposed in [YaNg01].
But even more difficult is the situation where one word on the source side is aligned to more tags on the target side. In this situation I decided to just keep the first corresponding tag. But it is really important to keep in mind, that this decision is very strict and it makes sense to think about better ways of dealing with that situation. One could be to combine the set of tags that correspond to one word into a supertag, but I did not test the effect of that.
With the creation of such a tagger it is now the same procedure for the reordering as it was when a tagger already was available.

4.4.2 Heuristic for finding clusters

When even for the target language no tools are available or when the use of them did not show the desired results, the question is, if there is a way to find another clustering of the source words, that reflects the reordering behavior and therefor can be used for the reordering task.
The first way I want to introduce is a clustering using the k means algorithm [Lloy82] for this task.
The clustering uses the information that is in the word alignment of the bilingual corpus and clusters the words corresponding to the observed reordering behavior.
Therefore in the first step for every word a vector is computed, that describes the reordering behavior of that word. This is done via observing all the reorderings in the corpus, as if the rules are extracted. This permutation is than stored and in addition to that the position of the current word in this permutation.
The combination of the permutation and the position of the word in this permutation then is used as one feature in the vectors for every word.
That means when the distribution of the occurrences of one word is similar to the distribution of the occurrences of another word, both words seem to share the same reordering behavior.
The next step, after extracting the vector for every word, is to cluster the data points into classes. This can be done directly by applying the k-means algorithm [Lloy82]. The k-means algorithm needs a distance measure for classification. The obvious distance measure, the Euclidean distance, however is not useful without preprocessing. The problem with the Euclidean distance is that words that occur

more often in the corpus are in completely different areas of the space than those words occurring only a few times. One way to deal with that is to normalize the vectors to a length of 1. On the other hand there is an easier way, by just using the angle between two vectors to discriminate between the classes. So the process of clustering the words can be described the following way:

```
for every word
determine feature vector
initialize k Means
iterate for X times
    for every word w
        assign word w to class c if angle(w,c) minimal
    determine new k Means
```

4.4.2.1 Improving clustering with principal components analysis

As an improved version of this clustering I tried to use the principal components analysis [DiHe04] to find those directions in space in which the variance between the classes is largest and use this information for discriminating between the classes.
The algorithm is nearly the same as above, with the difference, that after the extraction of the feature vectors for the words these vectors are normalized and then transformed by using the principal components analysis to identify the main directions with the most variance.
After this transformation the k-means algorithm as above is used to find a clustering of the data.

4.4.3 Finding optimal clusters

The so far described methods for finding clusters are similar in using heuristics to find a clustering for the data. However it is interesting whether a way exists that generates a clustering of the data, that uses a mathematical analysis of the data and tries to find the optimal clusters according to the final reorderings.
One way how that can be achieved is to do something similar to the clusterings that try to minimize the language model perplexity, but trying to minimize a "reordering perplexity". In [Och99] a way is proposed how to minimize the language model perplexity of a clustering. I will use the same strategy to minimize a measure for the "reordering perplexity".

The idea of finding an optimal clustering relies heavily on an optimization criterion for clusterings. The way I built such a criterion followed the idea of the monolingual word clustering described in [Och99]. In this case the idea is to find the best clustering $\hat{\mathcal{C}}$ that maximizes the probability of the correct reordering of a corpus with N words, over the different clusterings $\mathcal{C}$:

$$\hat{\mathcal{C}} = \arg\max_{\mathcal{C}} p(CorrectReordering_1^N|\mathcal{C}) \tag{4.2}$$

$CorrectReordering_1^N$ reflects the correct word order on the complete corpus. Of course it is not possible, to estimate the probability $p(CorrectReordering_1^N|\mathcal{C})$ in a single step, therefore it makes sense to decompose it into smaller parts. Therefore I made the assumption, that the reordering of the whole corpus consists of $i = 1..\tilde{N}$ smaller reorderings r_i performed on the wordsequences w_i. In addition to that I made the assumption, that the word sequence w_i only depends on the corresponding classes $\mathcal{C}(w_i)$ and not on the reordering on that sequence r_i.

$$p(CorrectReordering_1^N|\mathcal{C}) := \prod_{i=1}^{\tilde{N}} p(r_i|\mathcal{C}(w_i)) \cdot p(w_i|\mathcal{C}(w_i)) \tag{4.3}$$

The estimation of the probabilities in this equation can be estimated as follows:

$$p(r_i|\mathcal{C}(w_i)) := \frac{n(r_i|\mathcal{C}(w_i))}{n(\mathcal{C}(w_i))} \tag{4.4}$$

$$p(w_i|\mathcal{C}(w_i)) := \frac{n(w_i)}{n(\mathcal{C}(w_i))} \tag{4.5}$$

Inserting this into the equation leads to the following new equation:

$$p(CorrectReordering_1^N|\mathcal{C}) = \prod_{i=1}^{\tilde{N}} \frac{n(r_i|\mathcal{C}(w_i))}{n(\mathcal{C}(w_i))} \cdot \frac{n(w_i)}{n(\mathcal{C}(w_i))} \tag{4.6}$$

Applying the negative logarithm leads to:

$$\begin{aligned} Score(\mathcal{C}) &= -\sum_{i=1}^{\tilde{N}} \log p(r_i|\mathcal{C}(w_i)) + \log p(w_i|\mathcal{C}(w_i)) \\ &= -\sum_{i=1}^{\tilde{N}} \log \frac{n(r_i|\mathcal{C}(w_i))}{n(\mathcal{C}(w_i))} + \log \frac{n(w_i)}{n(\mathcal{C}(w_i))} \end{aligned}$$

Now it comes into account, that the goal after using the negative logarithm is to minimize the value of the function since the count of the word sequences is a constant that has for every clustering the same value it can be omitted.

$$\hat{Score}(\mathcal{C}) = -\sum_{i=1}^{\tilde{N}} (\log n(r_i|\mathcal{C}(w_i)) - 2 \log n(\mathcal{C}(w_i))$$

It is still necessary to loop over the complete training corpus, to get the reorderings one after another. This however, is not necessary after a reordering of the summation order. R denotes all reorderings that have been observed in the corpus and C denotes all sequences of clusterings that are reordered for a given clustering $\mathcal{C}$. That leads to the final criterion function.

$$\begin{aligned} LP_1(\mathcal{C}) &= -\sum_{r \in R, c \in C} n(r|c) \cdot \log(n(r|c)) \\ &\quad +2 \sum_{c \in C} n(c) \cdot \log(n(c)) \\ \hat{\mathcal{C}} &= \arg\min_{\mathcal{C}} LP_1(\mathcal{C}) \end{aligned}$$

The task of finding the best clustering then became the task of minimizing the function LP_1, without the need of iterating over the whole corpus again, but using information that is extractable beforehand.

```
initialize random clustering C
altScore = value of criterion function for C
iterate for X times
   For all v in vocabulary
       For all i in Classes
           move v into class i
           Score = value of criterion function
           if (Score < altscore)
               C = new Clustering
               altScore = Score
```

One problem with this approach however is that it is very time consuming and only makes sense for a very small number of classes, otherwise the time multiplies by the number of classes used.

4.4.4 Problems with automatic clustering

A problem that always comes with the task of clustering on the training corpus is the question, how to deal with words not seen in the training corpus. In this thesis I decided to build an own class for rare words. That means I introduced a new class used for those words only seen once during training. When the testset is clustered all never seen words in the testset are also assigned to this class. So as a result, those unknown words will still take part in reorderings.
One question remains unanswered. That is the question, how many classes should be used for the clustering. As a first indication the number of classes can be the same as used by POS tagging. This leads to around 30 classes. However one can argue that those classes are very speech dependent, and that the number of classes differs from language to language[1].
In contrast to that is the argument, that the more classes you have the more data sparseness will come into account. So one general goal should be to minimize the number of classes to restrict the effect of data sparseness to a minimum. The minimal number of classes however should not be too small, so having less than 5 classes will have the result, when you try to distinguish between all reorderings of the length of 10 you will have 10! possible permutations, but only 4^{10} different class combinations. This may be sufficient for languages that mainly share the same word order, but not for the general case.

4.5 Reordering of the source corpus

The reordering of the source corpus when using the rule based reordering is directly comparable to the creation of the translation lattices. But the difference is that after creation of the reordering lattice the best path according to the scores assigned to the edges is computed and the corresponding word order is output as new source sentence. The pseudocode for this task is the following:

[1]morphological rich languages tend to have more POS tags defined as other languages

```
reorderSourceSentence(Sentence s, Rules r)
   createReorderingLattice for Sentence s using Rules r
   find best way through the lattice with maximum score
```

For the finding of the best way through the lattice with the maximum score the Dijkstra algorithm is used [Dijk59]. This algorithm is a greedy algorithm that always expands the current best node and finds the optimal path through the lattice.
The reordering of the source side according to the alignment information works different, since it does not need a lattice to be created. One difficulty however is that there are unaligned words. I decided to move these unaligned words according to the next following aligned word. This decision was based on a review of the alignment data and it may be improved by some statistically motivated model. Also this may differ for different languages.
Another difficulty is to decide for one alignment, since a source word may be aligned to more than one word on the target side. Here I decided to rely on the first aligned target word.
The pseudocode for the alignment based reordering however is the following.

```
for i = 0 to source sentence length
   if word i has alignment information a[i]
      for all k in unalignedWords
          push pair(a[i], k) into alignmentArray
      clear unalignedWords
      push pair(a[i], i) into alignmentArray
   else
      push i into unalignedWords

sort alignmentArray using first pair element first
for all pairs in sorted alignmentArray
    print word corresponding to second part of pair
```

Both techniques reorder every sentence of the source corpus so that it can be used in the further steps as a new knowledge source.

5. Experiments

5.1 Experimental setup

This chapter is intended to provide the experimental results of the proposed techniques. Therefore I repeated the evaluations on different language pairs to get a better picture of the generality and the significance of the proposed strategies.
The evaluated language pairs are English $\rightarrow$ Spanish, English $\leftrightarrow$ German, Japanese $\rightarrow$ English and Farsi $\rightarrow$ English. The corpora and test sets used are based on different evaluation campaigns. For the English $\leftrightarrow$ German and English $\rightarrow$ Spanish tests, the European parliament speeches were used for training and evaluation. For Japanese $\rightarrow$ English task the same setting was used as in the IWSLT evaluation 2006. The statistics of the corpora and used testsets are given in Table 5.1

Additional to the corpus statistics I also computed statistics over the reorderings that are encoded in the word aligned bilingual corpus. Figure 5.1 shows the movement of words according to their relative position in the source and target sentence. So a value of -1.0 would mean that a word in the source sentence on the last position is aligned to the target word on the left position, whereas a value of 1.0 would mean the first source word is aligned to the last target word. The value that is assigned to the relative movement gives the relative frequency of this movement in comparison with all seen alignments.
These two sources of information already give some insight into the difficulties for the reordering process. Figure 5.1 indicates that the languages English, German and Spanish are very similar in the word order and only a few words have to be reordered, and those words that have to be reordered are only moved over small parts of the sentence. On the other hand, the translations from Farsi to English or Japanese to English have a lot more reorderings, and they tend to be longer than those in the other language pairs. Using only the information of this figure, one could get the idea that Farsi to English translation is as hard as Japanese to English translation. Perhaps the Japanese to English a little bit more difficult, because there are slightly more reorderings that move over larger parts of the sentence.

System	Set	Sentences	Words	Vocabulary Size/OOV
en $\leftrightarrow$ de	corpus en	1.2M	35M	97K / -
	corpus de	1.2M	33M	298K / -
	development en	2K	58K	6103 / 62
	development de	2K	54K	8762 / 306
	test en	2K	59K	6294 / 407
	test de	2K	55K	9040 / 604
en $\rightarrow$ *es*	corpus en	1.2M	33M	94K / -
	corpus es	1.2M	34M	135K / -
	development en	1.2K	30K	4084 / 79
	test en	1.2K	30K	4100 / 105
jp $\rightarrow$ en	corpus jp	190K	2.1M	33K / -
	corpus en	190K	2.1M	22K / -
	development jp	489	7445	1158 / 50
	test jp	500	8061	1308 / 52
irn $\rightarrow$ en	corpus irn	131K	822K	28K / -
	corpus en	131K	959K	14K / -
	development irn	463	3570	1184 / 75
	test irn	433	5162	1240 / 237

Table 5.1: Corpus statistics for the various tasks.

But in combination with Table 5.1 a better interpretation of the situation is possible. It can be inferred from that table, that the average sentence length for the Farsi input is of length 6.23 words, whereas the average sentence length for the Japanese input is 11.3 words. This in combination with the fact that there are still a lot of reorderings that have a relative movement in the sentence of more than $|0.6|$ results in the fact that in Japanese for the same reordering coverage as in Farsi with a rules up to a reordering length of 4, rules up to a length of 7 words are needed. This is an indication for the Japanese to English translations being harder than the Farsi to English translations.
For the system I used various tools. The Baseline system is the statistical machine translation system as it is described in [LZNB+07]. This system uses internal reordering, where a reordering window is used. The reordering window gives the maximum span of source words, how far a word is allowed to be moved. The reorderings in this system are modeled by a distortion model, that penalizes longer movements. So it does not use any further linguistic knowledge and is mainly guided by the language model.
For the alignment of the bilingual training corpus the GIZA++ toolkit [OcNe03] was used, which gives a word alignment for a bilingual corpus. To get a more dense and useful alignment it was postprocessed with the grow-diag-final algorithm described in [KAMCB+05]. The phrases used in the systems were extracted from the word aligned bilingual corpus, using the Pharaoh phrase extraction toolkit as described in [KoOM03]. In a second step the scores of the phrases were smoothed by the Kneser Ney smoothing [KnNe95].
For the linguistic annotation different tools were used. The parts of speech for the

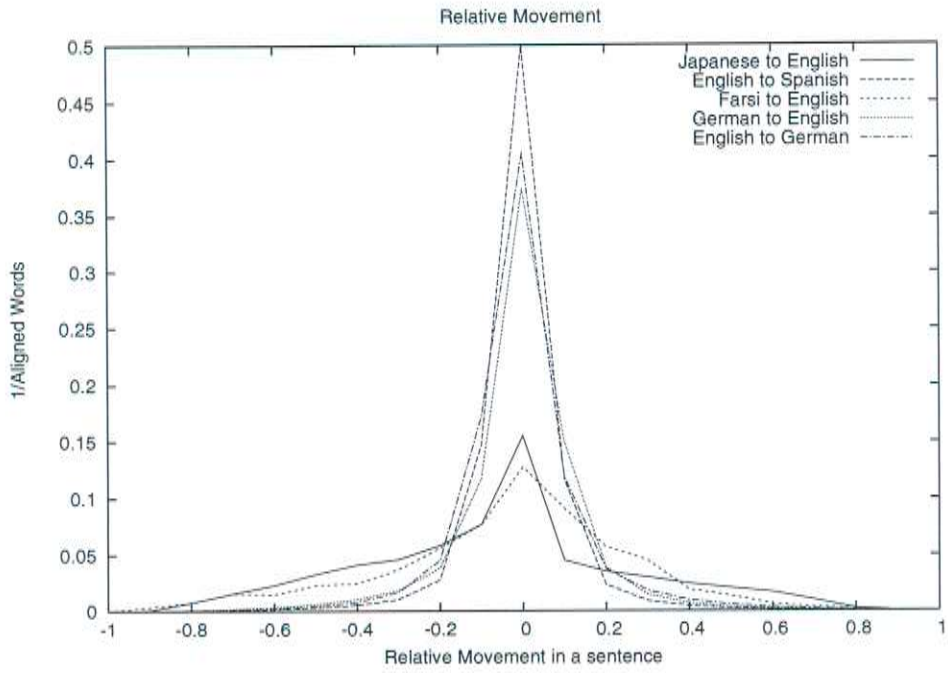

Figure 5.1: Frequencies of relative movements for the different language pairs

English language were obtained via the *Brill Tagger* that uses a tagset size of 36 tags. The chunk parsing on the English language was performed by the chunk parser described in [TsTs05].
The German language parts of speech tagging was done by the *Stuttgart Tree Tagger* using a tagset of 57 tags [Schm94].
For the evaluation I mainly used the Bleu [PRWZ01] evaluation metric. For some evaluations I also have Meteor [BaLa05] scores. When available the Meteor score is given in round brackets behind the Bleu score.

5.2 Results

5.2.1 Parts of speech based reordering rules

The first series of experiments was intended to evaluate the use of parts of speech based reordering rules for the translation. Therefore we performed the experiments on the English → Spanish and English ↔ German translation tasks.

5.2.1.1 Optimal threshold for the rules

The first question that has to be answered is: is there a threshold value which marks that below this value there is no further improvement in translation quality. Finding such a threshold is a very time consuming task, since those thresholds can differ from

	System	en → es	en → de	de → en
dev	Baseline(RO3)	49.98	18.92	25.64
	no context 0.05	50.36	19.48	26.69
	no context 0.1	51.09	19.55	26.46
	no context 0.2	50.66	19.30	26.01
	no context 0.3	50.59	19.22	25.73
	context 0.01	50.92	19.34	25.85
	context 0.05	50.90	19.34	25.86
	context 0.1	50.84	19.44	25.79
	context 0.2	50.74	19.30	25.67
unseen	base(RO3)	48.51	17.69	23.70
	no context	**49.57**	17.78	**24.79**
	context	**49.49**	17.79	23.87

Table 5.2: Case sensitive Bleu scores for systems without any context information for the reordering rules and systems using rules with exactly one tag of context to the left and right

the context type that a rule uses. Since this would require a series of experiments for every setting which is on the one hand a very time consuming task and would expand this thesis unnecessarily, I decided to make these evaluations only for two different types of context and use the so obtained result for the whole system.
The two rule types I decided to use were the rules based solely on the parts of speech of the reordered sequence without any context, and those rules, that used the parts of speech tag to the left and the right of the reordered sequence as context information. The hope in choosing these two rule types was that they have the biggest difference in their context use and so that the results obtained from both of them will generalize to a sufficient level.
The experiments consisted of building reordering lattices only using one type of reordering rules and only those rules with a relative frequency above a given threshold. Then the best setting for the development data was used for an evaluation on the unseen data. The results are presented in Table 5.2.

The best results were achieved when using a threshold value of ≈ 0.1. Even though these first experiments were just one step to the final system it is already remarkable that all settings that used rule based reorderings performed better than the baseline system that used internal reordering in the decoder. Another interesting point is that these first experiments already indicated that there is a big difference in the languages that are translated.
Before I made these experiments it seemed obvious that the rules with the left and right tag as context information should perform worse than those based on the reordered tag sequence solely. It would have been no big surprise if those reorderings even performed worse than the internal reordering, because those rules seem to be very specific. The results regarding those rules however were quite astonishing. For the English to Spanish system the translation quality was in both rule based reordering systems ≈ 1.0 Bleu better than the baseline. Also the English to German translations showed no big difference between both rule types. Only for the German

to English translation task the rules that did not use any context information outperformed the other rules by ≈ 1.0 Bleu.
The reason for this behavior is that this context works for English and Spanish as a source side, since there is much less variance in the word order for these languages. In the German language however, the word order differs a lot depending on the part of the sentence where the words occur.
Another result that can be inferred from the first experiment is the fact that the rule based reorderings show significant improvements over the baseline[1]. This is in harmony with previously reported results using parts of speech based rules for reorderings.

5.2.1.2 Combination of the different rule types

After determing the threshold that is usable as a lower bound for the relative frequencies of the rules that get applied, I fixed this value to the estimated value of 0.1 for the following experiments. But it has to be kept in mind, that this value may differ for further language pairs, and that it makes sense to reestimate it when switching to other languages.

The next step in the experiments was to use the full system. That means that all previously rules with the different types of contexts were used simultaneously for the creation of the source lattice.
Again the lattices for the developement set and the test set were created, but this time using the combination of all rules with a relative frequency above the given threshold of 0.1 (results in Table: 5.3). When reviewing the results obtained, it is interesting to note, that only for the translation from English to German there was an increase of ≈ 0.5 in Bleu. The reason for that increase in Bleu score is that the usage of the different context types allowed a better estimation of the function the word sequence has in the target sentence. The other scores however were still on nearly the same level as they were when only one rule type was used. The question is why is that the case?
To answer the question you have to keep in mind, that the rules that get extracted have to be seen very often in the corpus to get a high score. But when a sequence was often seen in the bilingual training corpus, it is more likely that a good phrase for exactly that sequence exists. This is not a big issue for those rules that abstract from the word level. But for the rule type that uses exactly the reordered word sequence as a trigger to insert a reordering into the translation lattice that becomes a problem. For these sequences there will probably be a whole phrase exist that translates this sequence. But after reordering the old sentence, in which this good phrase exists, the path containing this phrase has a very bad score and the new reordered sequence has never been seen before.
So the decoder has a more difficult task in translating this reordered phrase than before. However this is a problem that comes in different strengths for the different language pairs. While for the English to German and German to English translation the German language has a very loose word order, this is no big problem at all. But the translation from English to Spanish suffers from this problem, since in both

[1]translation scores significantly better on a 95% level are presented in bold letters

	System	en → es	en → de	de → en
dev	Baseline(RO3)	49.98	18.92	25.64
	no context 0.1	51.09	19.55	26.46
	combination of All Rules	51.15	19.58	26.90
	no word based rules	51.05	19.61	26.88
unseen	base(RO3)	48.51	17.69 (42.85)	23.70 (50.58)
	no context	**49.57**	17.78	**24.79**
	All Rules	**49.58**	**18.27** (43.11)	**24.85** (51.12)
	no word based rules	**49.83**	**18.21** (43.06)	**24.88** (51.10)

Table 5.3: Case sensitive Bleu (Meteor) scores for systems without any context, with all rule types used and with all rule types except for those rules exclusively using words as left hand side

languages the word order is more fixed than in German. Nearly always there is a swapping in positions for a noun and adjective in translation between those two languages. But the phrases for these short sequence cover that and thus work against the reordering rules in that case.
So the idea to overcome that problem is to only use the reordering rules that make use of parts of speech tags and to not use the rules that only use the words themselves for the reordering.
In Table 5.3 it is shown that the expected behavior is reflected in the results, although it also seems, as if the removing of the word based rules reduced performance for the English to German translation.

However there is still no big improvement for the German → English, and no further change in the English → German translation task. So that is only half of the answer. Another aspect that did not lead to an improvement in the translation score can be seen in the fact that the English word order is so strict. As a result of that, the target word order does not differ when using different kinds of contexts. Since the rules that used just the parts of speech sequence, already covered the reorderings very well, there was no further improvement to be expected.

5.2.1.3 Analysis of rule usage

To get a some insight into the process of the lattice creation and the reordering, it makes sense to look at the number of rules learned and the number of the rules that are applied in the test sets. They are given in Table: 5.4.
The first surprise is seen when only looking at the numbers for the rules that use only the reordered parts of speech sequence as left hand side. Even though English to Spanish translation is expected to have the most similar word order, the most rules were learned for this language pair. For German ↔ English translations the numbers of those rules were around 8.000 learned rules below.
The same applies for the rules that used the context to the left and the context to the right. Again for English to Spanish translation the most rules were learned. However, if you consider the number of rules that got applied the picture is completely different for these types of rules. Here the number of rules that were applied for the Spanish task were at a factor of 5 less than the numbers for the German ↔ English task.

System		# en → es		# en → de		# de → en	
Rules	> freq	Rules Learned	Rule Matches	Rules Learned	Rule Matches	Rules Learned	Rule Matches
POS	0.05	21K	12K	7K	60K	13K	72K
	0.1	6K	7K	4K	27K	8K	32K
	0.2	2K	4K	1K	8K	3K	14K
	0.3	1K	3K	469	3K	1K	7K
con-text	0.01	72K	21K	32K	89K	38K	88K
	0.05	46K	6K	22K	36K	28K	37K
	0.1	25K	4K	15K	19K	21K	17K
	0.2	15K	3K	8K	8K	14K	9K
all	0.1	122K	13K	177K	52K	206K	54K

Table 5.4: Number of reordering rules learned from the training corpus and number of rule matches on the test sentences with respect to the relative frequency threshold, just using POS sequence, using context of one tag to the left and right, and all rules together.

Even with regard to the smaller testset, a difference of 2.5 still lasts.
Considering all different rule types together the numbers are quite different. In that case the least amount of rules were learned for the English → Spanish translation, while for the German ↔ English task more than 1.5 times more rules were learned. This is also the case when only the unique reorderings are concerned, that means those reorderings that differ - without regarding to the context - in the parts of speech sequence or the permutation (Table: 5.5).
This complies with the assumption, that with the use of context in the rules, a better coverage of the reorderings is possible.
Another interesting point that can be inferred from Table 5.4 is the increase of applied rules when switching from the system that only uses one type of rule, to the system that uses all rule types together. Even though the number of rules that were learned were at least twenty times higher for all rules together, the number of rules that is applied in the test sets does not even double. That indicates once more, that there are additional helpful rules, but also there are a lot of rules that model the same reordering by using different contexts with different scores.

System	different reorderings for POS sequences	only POS rules
en → es	46K	6K
en → de	76K	4K
de → en	75K	8K

Table 5.5: Number of rules that differ in their reordering for the parts of speech sequences (without rules using words as left hand side)

5.2.2 Training corpus reordering

As it can be seen from the results above the problem with the reordering is that the reordered source sentence does not match the training corpus as good as it did

before reordering. So the next experiments dealt with the task to reorder the source side of the bilingual training corpus and use the phrases extracted from that corpus for translation.
These phrases from the reordered source corpus should match the new word order on the most likely path in the lattice better than before. Therefore an improvement in translation quality is expected.
For the reordering of the source corpus I tried two different approaches. The first approach is to use the alignment information as described above. The reordering using the word alignment information is based on the existing GIZA++ alignment for the bilingual training corpus. The words of the source corpus are output in the order they appear in the alignment on the target side first.
The other approach uses the reordering rules based on the rules extracted in a prior step from the tagged and aligned bilingual corpus. With the help of these rules, a lattice for every sentence in the source corpus is constructed and scored as if it is a normal translation lattice. But in the next step I searched for the best path through this lattice and use the word order on this path as the new source sentence in the reordered corpus.
Example sentences of the reordered corpus for both approaches are given in Table: 5.6. Then after a new phrase extraction on the new constructed bilingual corpus I

Original word order	it says that this should be done despite the principle of relative stability . this is all in accordance with the principles that we have always upheld .
Alignment based reordering	it says should that this relative done despite the principle of stability be is all this in accordance with the principles that we always upheld have .
Rule based reordering	it says that this should despite the principle of relative stability done be . this is all in accordance with the principles that we have always upheld .

Table 5.6: Example sentences in the reordered source training corpora for English to German translation

use those new phrases for the translation task. Results for these experiments are given in Table: 5.7

The first thing that is interesting in these experiments is that the results show no improvement for the English to Spanish translations. However there is an improvement of ≈ 0.2 Bleu for the English to German and German to English translations. One reason for that is that there are only a few reorderings in English to Spanish translations compared to the German to English and English to German translations. And these new reordered phrases are already well matched when the lexical reorderings are not used. If you look at the results of the translations for English to Spanish that include the word based reorderings, you can also see the same improvement when switching to a reordered corpus as it was seen in the other language

	System	en → es	en → de	de → en
training	Baseline (RO3)	49.98	18.92	25.64
	All Rules	51.15	19.58	26.90
	NoWordRules	51.05	19.61	26.88
	All Rules + GIZA Reordered	51.41	19.44	26.01
	All Rules + Rule Reordered	51.42	20.00	27.06
	NoWordRules + RuleReordered	51.38	19.93	27.08
test	Baseline (RO3)	48.51	17.69 (42.85)	23.83 (50.58)
	Rules	**49.58**	**18.27** (43.11)	**24.85** (51.12)
	NoWordRules	**49.83**	**18.21** (43.06)	**24.88** (51.10)
	All Rules + GIZA Reordered	**49.78**	**18.23** (42.81)	24.09 (50.45)
	All Rules + Rule Reordered	**49.78**	**18.42** (43.25)	**25.06** (51.27)
	NoWordRules + RuleReordered	**49.75**	**18.34** (43.10)	**25.00** (51.21)

Table 5.7: Case sensitive Bleu (Meteor) scores for system with different reordering approaches of the source corpus that is used for the phrase extraction

pairs.
That allows us to say that in all three settings there was an improvement achieved by phrases from a reordered source corpus.
The next interesting result of the experiments is the effect that the different reordering methods of the source corpus showed. The English to Spanish translation showed no difference between the GIZA and the rule based reordering, for the English ↔ German language pair however, the use of the rule based reordering shows the better results in both directions. The reason for that is again the very similar word order in English to Spanish translation. So the rule based reorderings are very close to the GIZA based reorderings, since there is not such a big variance in the reordering rules as it is in the German and English task. A lot of the reordered sentences in the English to Spanish translation look exactly the same, no matter what reordering approach was used.
For the language pair with the less similar word order, the effect of the better matching phrases comes into account. The corpus reordered via the GIZA alignment information is not as close to the final translation situation seen in the lattices as the corpus reordered by the rules.
As a consequence of that, the resulting translations are better for the rule based reordering, even though the corpus reordered via the GIZA reordering should be more monotone in alignment to the target language.
Another effect that I observed when looking into the reordered source corpora, is the quite interesting fact, that a reordering based on the alignment information looks more "random" than the reordering performed by the reordering rules. In the Ta-

ble: 5.6 this is quite obvious. The reason for this behavior of the reordering is that the alignment contains errors and so a reordering based on that alignment will also contain errors. For the reordering based on the rules however, this effect is not so big. The tagging errors already occurred during the learning of the rules, so the rules interpolate them in a way. And in addition to that only those rules get applied that were seen with a high relative frequency. This again prevents those "random" looking reorderings.
An interesting aspect of this is, how the performance would be if there were absolutely no alignment errors. Then the only effect that comes into account would be the similarity of the training data to the test data.

5.2.3 Integration of a source language model

I also tested the effect of reordering the source sentences using a language model learned on a reordered source training corpus. Therefore I used therefor the alignment based reordering of the source training corpus to get a language model that reflects as good as possible the final word order in the target language. But as a side effect of that, the alignment quality plays an important role in these experiments.
In a next step I reordered the words in the source sentences using the new language model information. To restrict the search for the best reordering according to that word order I restricted the reordering to a length shorter than 6. The so obtained reordered input sentences then were used as input to the baseline translation system to evaluate whether the language model of the reordered source corpus alone is able to provide enough information to create better translations. The results however were pretty bad. The translation quality dropped already on the MER training data by more than 3 Bleu compared to the baseline system (Table: 5.8).

en $\rightarrow$ de training	17.24
de $\rightarrow$ en training	22.39

Table 5.8: Case sensitive Bleu scores for systems where the input text was reordered according to a language model generated on a reordered source corpus. No unseen data was evaluated

The reason for that is quite obvious. The new reordered input sentence does not reflect the training data anymore. Since there are not any alternatives in this sentence - because of the hard decisions that were made when creating it - this results in a pretty bad matching of the phrases, so the reason for the observed results should be the mismatch between the training data and the test data.
Since the scores were so bad for the training set, the system was optimized on, and they were even below the so far seen scores on unseen data, I decided to skip further evaluations and I directly went to the next step in using the source language model only as an additional score in the reordering lattice as in [ZhZN07]. Scores for that experiment are shown in Table: 5.9.
In this case the results show that there was no further improvement possible, and it seems as if the system that uses only the relative frequency of the rules on the lattice edges performed a bit better, but only on a low significance level.

training	en → de + LM	19.35
	de → en + LM	26.74
unseen	en → de all rules	18.27
	de → en all rules	24.85
	en → de + LM	18.13
	de → en + LM	24.67

Table 5.9: Case sensitive Bleu scores for systems where every lattice edge has an additional score corresponding to the source language model.

As a conclusion of these experiments can be seen that the usage of parts of speech information improved the system over the baseline system and the usage of phrases extracted from a reordered source corpus yields a further improvement. The language model of a reordered source corpus however did not help to improve the system any further.

5.3 Long range reorderings

As an approach for the long range reorderings I mentioned two different approaches. The first one is to use generalized reordering rules, that allow the usage of a kleensche star. However there are no results for this approach presentable, because this approach did not prove usability for real applications. The problem with this result is that it created very huge lattices. The lattices grew very big because of the multiple application of the same rule with the same parts in the sentence. So the reordering lattices grew in average to more than 10.000 nodes. This however leads in the implementation of the translation system I used in a much bigger translation lattice, since for every sequence of edges in the reordering lattice the phrases found for that source sequence are inserted as an additional edge. This lead to a memory and runtime issue which made this approach unusable.

5.3.1 Reordering with chunkparse information

The alternative approach for getting longer range reorderings integrated into the lattices is the use of chunk parse information. Since the chunks rely on the function of a word sequence in the sentence, it is more restricted than the first approach for longer range reorderings.
For the experiments I evaluated, if it makes sense to perform a recursion during the lattice creation or to just use a flat reordering on chunk base without any recursion. Furthermore I was interested, whether it helps to use parts of speech reordering rules in addition to the chunking based reorderings.
Results presentend in Table 5.10, show that the usage of chunk reordering also outperforms the baseline system with internal reordering. Also the results are better than the reordering based on a single rule for the parts of speech tags (compare Table: 5.2). This is the same result as already reported by [ZhZN07]. However the combination of all rule types for the parts of speech based reordering outperform the chunk based reordering.
No clear picture can be seen, whether the recursive building of the reordering lattices

	System	en → es	en → de
training	Baseline(RO3)	49.98	18.92
	ChunkFlat	50.79	19.72
	ChunkRecursive	50.76	19.81
	ChunkFlat + POS	50.83	19.86
unseen	Baseline(RO3)	48.51	17.69
	ChunkRecursive	**49.56**	**18.20**
	ChunkFlat	**49.67**	**18.18**
	ChunkFlat + Pos	**49.52**	**18.16**

Table 5.10: Case sensitive Bleu scores for the systems using chunk parse information based rules

for the chunk based reordering helped. It performed as good as the system, that did not allow the recursive applying of the reordering rules.
The recommendation for the usage of chunk based reorderings is to rely on the flat application of the chunk based reorderings without any recursion, since this generates the smaller latices and therefor has the better runtime and memory usage. The same applies for the combination of chunk based reorderings in combination with parts of speech based reorderings. Since no significant improvement was seen here, no combination should be done unless it proves to improve the system for another task.

5.4 Techniques without having POS information of the source

The following series of experiments is intended to prove that even without a chunk parser or POS tagger an improvement in translation quality can still be achieved by using stronger reordering models than the simple internal reordering.
The experiments in this case are mainly conducted on Farsi (the native name of the Persian language) to English translations. The observed results were also tested for Japanese to English and for having a comparison to linguistically motivated clustering on English to German and German to English translations.
Again the baseline system that was used was the original reordering internal in the decoder with a reordering window of 3.
The problem with the clustering methods is that it is hard to decide whether the classes are just random or if there is a real structure found in the data. And of course it is interesting how the system behaves when there is no clustering at all and every word is clustered into the same class. I therefore conducted experiments where I used random clusters and also a single class, into which every word was assigned. Since there are already different clustering methods used in the machine translation, I also tested how the clustering that is used for reducing the language model perplexity in the GIZA training [Och99] behaves for the reordering task.
The detailed results are presented in Table 5.11.

Before looking into the detailed results of the different settings, I want to start with the most astonishing behavior observed in this experiment. As an idea for further

System	MER Training	unseen data
Baseline(RO3)	24.28	22.35
Monotone	21.37	19.52
single Class	23.53	21.61
single Class(RO3)	23.93	20.69
30 Random Classes	22.73	20.24
30 Random Classes(RO3)	23.50	21.37
5 Random Classes	23.34	21.71
5 Random Classes(RO3)	23.80	21.31
POS-projection	23.92	22.21
POS-projection(RO3)	24.57	22.50
MKCLS 30	24.20	22.42
MKCLS 30(RO3)	24.38	21.87
MKCLS 5	23.94	22.18
MKCLS 5(RO3)	24.50	21.98
k-Means 30	24.34	22.68
k-Means 30(RO3)	24.72	22.66
k-Means 5	24.70	**23.13**
k-Means 5(RO3)	25.45	22.31
Pca5	23.80	22.53
Pca5+RO3	24.33	22.47
Optimized5	23.33	21.80
Optimized5+RO3	23.36	21.71

Table 5.11: Case sensitive Bleu scores on Farsi to English Translation using different annotation methods for the source side

improvement of the system over the baseline, and since I expected that the reordering rules will be not as good as they would be using a reliable parts of speech tagger, I also made experiments where I enabled internal reordering in the decoder during translation process in addition to the reordering lattices. In contrast to the results on the data used for the MER training the results with additional internal reordering were in all but two settings worse than without the internal reordering. The only exceptions from this behavior is the projection of the parts of speech tags from the target language to the source language and the setting with random clusters where 30 clusters were used.
For the second exception it can be described by the fact that the random clustering with a lot of clusters does not give any reliable hint how to reorder the words based on the rules. So the most reliable word order information comes during translation from the language model and the additional internal reordering simply allows to test more word orders. This is desirable, since as I said the random clustering does not provide reliable reordering by itself.
For the parts of speech projection this seems to be also the case, however the information obtained from the parts of speech projection seems to be a bit better.
The results for the different approaches are quite interesting concerning the previous thoughts. As a first positive result can be seen that every tried setting performed

better than monotone decoding. So giving the possibility for reorderings is always good. More interesting however is the comparison of the results with the baseline system. The only system that performed significantly better than the baseline system on the unseen data was the system that used the clustering based on the simple k Means algorithm with just 5 clusters. Quite astonishing is the fact, that the result was achieved for the small number of clusters and the setting with 30 clusters obtained the same way performed pretty bad compared to that. A reason for that is that the clustering into more classes automatically introduces more errors in the clustering. So the fewer clusters are used, the better the automatic clustering is. In addition to that the use of 5 clusters provides for a reordering of length n are able to cover 5^n of the $n!$ different permutations. That is theoretically sufficient up to a length of 11 however the real length will be below that length.
Another result is that the usage of the principal component analysis for a better disambiguation between the clusters does not perform as expected. A possible reason for that may be the fact that in the covariance matrix of the normalized feature vector, artifacts caused by scarecly seen words are reflected in the principal components analysis and lead to low quality clusters. But still the quality of the principal components analysis is as good as the quality of the baseline system.
Quite disappointing is the fact that the mathematical model used for an iterative optimization of the classes performed pretty bad and indicates that the underlying model does not reflect the reordering task as good as it was supposed to.
One thing that can be seen from the results is that there is an advantage of choosing clustering methods directly designed for the task of reordering over other methods designed not designed for the reordering. This can be seen from the fact that both the clustering using just the k Means algorithm and the prior usage of the principal components analysis show better results than all the other tested techniques.

These first results show that the absence of a parts of speech tagger or other linguistic tools does not mean that improvements with this reordering approach are not possible.

Encouraged by these first results I also evaluated the best clustering method observed for Farsi to English translation on the German to English and English to German translation task, for which comparable numbers for parts of speech based reorderings are available (Table 5.12).

The results obtained from that experiment are very encouraging, since the translation scores are completely in the range of the scores obtained by using parts of speech based reorderings. For the German to English translation the score of the cluster based reordering rules is even 0.2 Bleu above the score obtained for the parts of speech based rules.
That is quite nice, since this gives the opportunity to decide whether to use the parts of speech tagging or the clustering. While the parts of speech tagging is more reliable, the clustering is very fast. So the clustering allows a way to integrate the reordering techniques into a real time translation system.
The next step was to look into another language pair, that should be much more difficult concerning the reordering task. For Japanese to English translation there is

	System	en → de	de → en
training	Baseline	18.92	25.64
	POS	19.58	26.90
	k-Means 5	19.35	26.78
unseen	Baseline	17.69	23.83
	POS	**18.27**	24.85
	k-Means 5	**18.21**	**25.04**

Table 5.12: Case sensitive Bleu scores comparing the effect of k-Means clustering with parts of speech in the reordering usage

System	MER training	unseen data
Baseline (RO6)	28.28	24.89
k-Means 30	22.62	19.35
k-Means 5	22.82	19.55
k-Means 5 only Rules > 0.5 + RO6	28.55	24.20
k-Means 5 Rules > 0.5 - Lex + RO6	27.96	24.14

Table 5.13: Case insensitive Bleu scores on Japanese to English translation using Rules based on Clustering. RO6 indicates that additional internal reordering was used as in the baseline system

the problem, that the word order differs a lot just because of the underlying sentence structure. While English is an SVO - subject verb object - language, Japanese is an SOV - subject object verb - language. That means that the verbphrase from the Japanese sentence usually has to be moved over the objects to the corresponding subject. This introduces quite long reorderings. So the way to do that would be to use a reliable chunk parser for Japanese that identifies those parts. However for this thesis I had no such chunk parser available and so I evaluated how good the word class based reorderings are able to deal with the translation from Japanese to English. Results for these tests are given in Table 5.13.

In these experiments for the first time the system that used solely the internal reordering outperformed the rule based reordering. This was in a way to expect, since the reorderings have a very long span. That means to have a useful reordering rule, exactly the same sequence as in the test set has to be seen in the training corpus for a few times. And to make things even worse, the size of the training corpus is pretty small, so that this is very unlikely.
An idea to make use of the deeper analysis that is performed for the rule based reordering on the source side is to combine the rule based reordering with the internal reordering mechanism. This however still performed worse than the baseline system. Another idea is to use only those rules that have a high probability and then use the normel internal reordering. This gave an improvement of 0.3 on the training set, but performed worse on the unseen data. This looks like an overfitting effect.
So far I was not able to make use of the word based clustering and the reordering rules for the Japanese language.

English Source	Mr President , first of all , I would like to congratulate the Commission on the initiative to develop Article 13 of the Treaty further .
Baseline	Herr Präsident , zunächst beglückwünsche ich die Kommission zu der Initiative zu entwickeln Artikel 13 des Vertrags weiter .
POS	Herr Präsident ! zunächst möchte ich die Kommission zu ihrer Initiative beglückwünschen Artikel 13 des Vertrags entwickeln .
POS + reordered phrases	Herr Präsident , zunächst möchte ich die Kommission dazu beglückwünschen , die Initiative zu Artikel 13 des Vertrags weiterzuentwickeln .
KMeans5	Herr Präsident , zunächst möchte ich die Kommission zu ihrer Initiative beglückwünschen , Artikel 13 des Vertrags weiterzuentwickeln .
Chunking	Herr Präsident ! zunächst möchte ich die Kommission zu ihrer Initiative beglückwünschen Artikel 13 des Vertrags weiterzuentwickeln .

Table 5.14: Example Translations English to German

On the other hand this was in a way to expect because of the very different underlying sentence structure. The hope is that with the usage of a reliable chunk parser the reordering on the source side will give an improvement over the baseline.

5.5 Examples for translations

Here I present some example translations to illustrate the effect the different reordering strategies had. In Table 5.14 the translations for the system with the k Means based clusters and the chunk reordering are perfect, while the reordering using parts of speech tags makes some small mistakes. The translations created by the baseline system however are very bad compared to all the rule based reorderings.
The example in Table 5.15 gives an example sentence, where the chunk based reordering performed a long range reordering which made the translation better.
The third example in Table 5.16 is intended to show that not always every approach is helpful, even though the overall performance rose. Here it is the effect of the phrases extracted from the reordered corpus. Using these phrases decreased the translation quality for this sentence.
The final translation example in Table 5.17 provides an example, where there is an extreme difference between the translation using parts of speech tags and the translation using the k Means based clustering.

English Source	in the course of the last few years , it has been possible to bring about a distinct improvement in the situation , due in no small part to the influence of the European Institutions .
Baseline	in den letzten Jahren war es möglich , um eine deutliche Verbesserung der Situation , nicht zuletzt durch den Einfluss der Europäischen Institutionen .
POS + re-ordered	in den letzten Jahren war es möglich , eine deutliche Verbesserung der Lage herbeizuführen , nicht zuletzt durch den Einfluss der Europäischen Institutionen .
Chunking	in den letzten Jahren war es möglich , eine deutliche Verbesserung der Lage , nicht zuletzt durch den Einfluss der Europäischen Institutionen herbeizuführen .

Table 5.15: Example for a long range reordering in the chunk based translation

German Source	nun ist es an der Zeit , dass wir uns dieser Aufgabe so intensiv wie möglich annehmen .
Baseline	now is the time for us to this task as closely as possible to adopt .
POS	it is now time for us to take on this task as closely as possible .
POS + reordered phrases	it is now time that we this task accept as closely as possible .
KMeans5	it is now time for us to take on this task as closely as possible .

Table 5.16: Example translations where reordering with phrases from a reordered corpus performed worse

German Source	daher bin ich für jede Aussprache und Diskussion in dieser Angelegenheit offen .
Baseline	I am therefore in favour of any debate and discussion on this matter open .
POS	I am therefore in favour of any debate and open debate on this matter .
POS + re-ordered	I am therefore in favour of any debate and open debate on this matter .
KMeans5	that is why I am open to any debate and discussion on this matter .

Table 5.17: Example of very different word order in k Means based translation

6. Conclusions and future work

In this thesis I presented my work on the word reordering task. My work focused on the reordering of the source side, to generate a word order that has a word order more similar to the target side. To make no hard decisions in this preprocessing step that result in a loss of information I created for every sentence a reordering lattice. Such a lattice contains all the possible reorderings for one sentence.
The paths that are part of the lattice in addition to the original source sentence are created by reordering rules. Those rules are learned in a prior step from the word aligned bilingual training corpus. Since the use of only words results in a data sparseness problem for the reordering rules, I used additional annotation information on the source side. For this additional information I tested the use of parts of speech tags, chunk parse information and automatically learned word clusterings.
In the reordering rules I used additional context information that describes the typical surrounding words or tags that occur with the rule. By doing that I was able to learn rules that better distinguished between reorderings for the same word sequence and I was able to learn additional rules that only had a very low relative frequency when no context information was used. Experiments showed that the usage of context for translation quality improves the system, but depends on the language pairs.
For the annotation of the source corpus I showed that the usage of the linguistically motivated annotations, like parts of speech tags and chunk parse information, resulted in an improvement in translation quality.
Results that were comparable to the scores obtained with the linguistically motivated annotations were achieved by the usage of automatically learned word clusters. However the experiments showed that clustering methods that are based on reordering information obtained from the alignment outperformed other clustering methods that were not modeled for the reordering task. Also the results showed that fewer classes are preferable.

Furthermore I evaluated the effect of a reordering of the source training corpus. The results showed that an improvement was possible when the reordered training corpus is used for the phrase extraction, however a language model based on this corpus,

used as a scoring of the reorderings in the lattice, did not turn out to improve the system.
For the phrase extraction based on the reordered training corpus I showed that those reorderings have to be preferred that are obtained the same way as the reorderings of the sources during translation. This effect also was language dependent, but for the tested languages the reordering using the same strategy as for creating the translation lattices turned out to be never worse than the reordering using alignment information. As a reason for that I showed the problem that a reordering motivated by alignment information suffers from alignment errors, while reorderings based on the rules are more robust.
Another result of my experiments is that the reordering approaches are language dependent and not every language pair is able to profit from those approaches. So the strategies are not for blind application, but have to be verified for the specific task. But for four of the five tested language pairs the proposed approaches provided a significant improvement.

All together the results imply that the most improvements are achievable by using parts of speech tags for the annotation of the source side. The rules that are used for the reordering task should make use of the context, the reorderings were seen in. The source corpus should be reordered using the learned rules and a phrase table based on the new source side should be extracted. Using the rules to create translation lattices and using the phrases extracted from the reordered source corpus should result in a system that covers the reorderings better than internal reorderings.
When no parts of speech tags are available, the usage of automatically learned clusters instead of the parts of speech tags makes sense.
But in any case it is still important to verify the usage of the strategies for the specific task.

6.1 Future work

For the future it makes sense to look into the problems that caused the problems with the Japanese to English translation. The hypothesis that chunk parse based reorderings do improve the system has to be verified.
Directly related to the problems seen for Japanese to English translation is the task of finding a method for deciding which reordering approach is the best for a specific language pair. In Chapter five I gave a first approach for analysing languages with regard to their reordering behavior. This should be further investigated to provide a way to automatically decide for the best approach.

Also the techniques for the automatically built word clusters for the annotation of the source side still leave room for further improvements. The so far used clusters have a problem with unknown words. This can be improved by integrating a language model for the clusters into the clustering of a source sentence. That would allow to decide for a cluster even when the word is unknown.

A very interesting future task is the integration of the proposed techniques into applications like simultaneous speech translation systems. In those systems parts of speech tagger or chunk parser introduce the problem of being usually very slow or they are very hard to integrate into another system. There the integration of the automatical clustering of the source words according to their reordering behavior would provide the needed functionality and the needed speed.

Furthermore it still makes sense to look into other approaches for modeling longer ranging reorderings. Chunk based reorderings are able to move words over a longer spanning subsequence, however they are not able to move single words that are part of one chunk over a longer subsequence into another chunk. Solutions have to be found for that.
As I mentioned in my introduction, there is the problem of dealing with disjoint phrases. The strategies proposed in this thesis are not able to cover those reorderings. As a solution for the problem, either a reordering on the target side after translation or the usage of disjoint phrases would be possible.

And of course the evaluation of the strategies for other language pairs is always very important.

Literature

[BaLa05] S. Banerjee und A. Lavie. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments, June 2005.

[BePP96] A. L. Berger, S. A. Della Pietra und V. J. Della Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1), 1996, S. 39.

[BPPM93] Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra und Robert L. Mercer. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, 19(2), 1993, S. 263.

[Bril95] E. Brill. Transformation-based error-driven learning and natural language processing: A case study in part of speech tagging, 1995.

[Cent06] TALP Research Center. FreeLing 1.5 An Open Source Suite of Language Analyzers, http://garraf.epsevg.upc.es/freeling/, 2006.

[ChCF06] B. Chen, M. Cettolo und M. Federico. Reordering rules for phrase-based statistical machine translation. In *Int. Workshop on Spoken Language Translation Evaluation Campaign on Spoken Language Translation*, 2006, S. 1 – 15.

[CrMa06] Josep M. Crego und Jose B. Marino. Reordering Experiments for N-Gram-Based SMT. In *Spoken Language Technology Workshop*, Palm Beach, Aruba, 2006. S. 242 – 245.

[DiHe04] Chris Ding und Xiaofeng He. K-means Clustering via Principal Component Analysis. In *Proc. of Int'l Conf. Machine Learning (ICML 2004)*, July 2004, S. 225 – 232.

[Dijk59] E. W. Dijkstra. A note on two problems in connexion with graphs, 1959.

[KAMCB+05] P. Koehn, A. Axelrod, A. B. Mayne, C. Callison-Burch, M. Osborne und D. Talbot. Edinburgh system description for the 2005 IWSLT speech translation evaluation. In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT)*, Pittsburgh, PA, 2005.

[Knig99] Kevin Knight. Decoding Complexity in Word-Replacement Translation Models. *Computational Linguistics*, 25(4), 1999, S. 607 – 615.

[KnNe95] Reinhard Kneser und Hermann Ney. Improved backing-off for m-gram language modeling. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Detroit, Michigan, 1995. S. 181 –184.

[KoMo05] P. Koehn und C. Montz. Shared task: statistical machine translation between European languages. In *ACL Workshop on Building and Using Parallel Texts*, Ann Arbor, Michigan, 2005. S. 119 – 124.

[KoOM03] P. Koehn, F. Och und D. Marcu. Statistical Phrase-Based Translation, 2003.

[Lloy82] S. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, Band 28, 1982, S. 129–137.

[LZNB+07] Ian Lane, Andreas Zollmann, Thuy Linh Nguyen, Nguyen Bach, Ashish Venugopal, Stephan Vogel, Kay Rottmann, Ying Zhang und Alex Waibel. The CMU-UKA Statistical Machine Translation Systems for IWSLT 2007, 2007.

[NiNe01] S. Nieen und H. Ney. Morphosyntactic analysis for reordering in statistical machine translation. In *Proc. MT Summit VIII*, Santiago de Compostela, Galicia, Spain, September 2001. S. 247 – 252.

[Och99] Franz Josef Och. An Efficient Method for Determining Bilingual Word Classes. In *Ninth Conf. of the Europ. Chapter of the Association for Computational Linguistics (EACL '99)*, Bergen, Norway, 1999. S. 71 – 76.

[Och03] F. J. Och. Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, Sapporo, Japan, 2003. S. 160 – 167.

[OcNe00] Franz Josef Och und Hermann Ney. Improved Statistical Alignment Models. In *ACL 2000*, 2000, S. 440 – 447.

[OcNe03] Franz Josef Och und Hermann Ney. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1), 2003, S. 19 – 51.

[OGKS+04] F. J. Och, D. Gildea, S. Khudanpur, A. Sarkar, K. Yamada, A. Fraser, S. Kumar, L. Shen, D. Smith, K. Eng, V. Jain, Z. Jin und D. Radev. A smorgasbord of features for statistical machine translation. *Proc. 2004 HLT-NAACL*, 2004, S. 161.

[PiJBC66] John R. Pierce und et al. John B. Carroll. Language and Machines - Computers in Translation and Linguistics. *ALPAC report, National Academy of Sciences, National Research Council*, 1966.

[PoNe06] M. Popovic und H. Ney. POS-based word reorderings for statistical machine translation. In *Proc. of the 5th Int. Conf. on Language Resources and Evaluation (LREC)*, Genoa, Italy, 2006. S. 1278.

[PRWZ01] K. Papineni, S. Roukos, T. Ward und W. Zhu. Bleu: a method for automatic evaluation of machine translation, 2001.

[SaCa07] Germán Sanchis und Francisco Casacuberta. Reordering via N-Best Lists for Spanish-Basque Translation. In *Proceedings of the 11th International Conference on Theoretical and Methodological Issues in Machine Translation*, Skövde, Sweden, September 2007. S. 191 – 198.

[Schm94] H. Schmid. Probabilistic Part-of-Speech Tagging Using Decision Trees. In *International Conference on New Methods in Language Processing*, Manchester, UK, 1994.

[ShSO04] L. Shen, A. Sarkar und F. J. Och. Discriminative reranking for machine translation. *In HLT-NAACL 2004: Main Proc.*, 2004, S. 177.

[Till04] Christoph Tillmann. A Unigram Orientation Model for Statistical Machine Translation. In *Companian Vol. of the Joint HLT and NAACL Conference (HLT 04)*, Boston, MA, May 2004. S. 101 – 104.

[TiZh05] C. Tillmann und T. Zhang. A localized prediction model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, Ann Arbor, Michigan, 2005. S. 557 – 564.

[TsTs05] Yoshimasa Tsuruoka und Jun'ichi Tsujii. Chunk Parsing Revisited. In *Proceedings of the 9th International Workshop on Parsing Technologies (IWPT 2005)*, 2005, S. 133 – 140.

[VoNT96] Stephan Vogel, Hermann Ney und Christoph Tillmann. HMM-based WordAlignment in Statistical Translation. In *Proc. of Coling '96: The 16th International Conference on Computational Linguistics*, Copenhagen, Denmark, 1996. S. 836 – 841.

[VXDN06] David Vilar, Jia Xu, Luis Fernando D'Haro und Hermann Ney. Error Analysis of Statistical Machine Translation Output, 2006.

[WaWa98] Yeyi Wang und Alex Waibel. Fast Decoding for Statistical Machine Translation. In *Proc. ICSLP 98*, 98, S. 2775 – 2778.

[Wu96] D. Wu. A polynomial-time algorithm for statistical machine translation. *Proc. 34th Annual Meeting of the Assoc. for Computational Linguistics*, 1996, S. 152.

[Wu97] D. Wu. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3), 1997, S. 377.

[YaKn00] Kenji Yamada und Kevin Knight. A Syntax-based Statistical Translation Model. *ACL 2000*, 2000.

[YaNg01] David Yarowsky und Grace Ngai. Inducing multilingual POS taggers and NP bracketers via robust projection across aligned corpora. In *Second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies*, Pittsburgh, PA, June 2001. S. 1 – 8.

[ZeNe03] R. Zens und H. Ney. A comparative study on reordering constraints in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, Band 1, Sapporo, Japan, 2003. S. 144 – 151.

[ZeNe06] R. Zens und H. Ney. Pos-based word reorderings for statistical machine translation. In *Proc. of the Fifth Int. Conf. on Language Resources and Evaluation (LREC)*, 2006.

[ZhVo06] Ying Zhang und Stephan Vogel. Suffix array and its applications in empirical natural language processing. Technical Report CMU-LTI-06-010, Language Technologies Institute, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, Dec. 2006.

[ZhXW05] Bing Zhao, Eric P. Xing und Alex Waibel. Bilingual Word Spectral Clustering for Statistical Machine Translation. In *ACL 2005 Workshop on Building and using Parallel Texts: Data Driven Machine Translation and Beyond (ACL WPT-05)*, June 2005.

[ZhZN07] Yuqi Zhang, Richard Zens und Hermann Ney. Chunk-Level Reordering of Source Language Sentences with Automatically Learned Rules for Statistical Machine Translation. In *Proceedings of SSST, NAACL-HLT 2007 / AMTA Workshop on Syntax and Structure in Statistical Translation*, Rochester, New York, April 2007. Association for Computational Linguistics, S. 1 – 8.