Doctoral Thesis

Non-projective Parsing for Pre-ordering Statistical Machine Translation

Hwidong Na (나 휘동)
Department of Computer Science and Engineering
Pohang University of Science and Technology
2015

전처리 어순조정 통계기계번역을 위한 비투사 구문분석

Non-projective Parsing for Pre-ordering Statistical Machine Translation

Non-projective Parsing for Pre-ordering Statistical Machine Translation

by

Hwidong Na
Department of Computer Science and Engineering
Pohang University of Science and Technology

A dissertation submitted to the faculty of the Pohang University of Science and Technology in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Computer Science and Engineering

Pohang, Korea
10. 27. 2014
Approved by

Jong-Hyeok Lee
Academic Advisor

Non-projective Parsing for Pre-ordering Statistical Machine Translation

Hwidong Na

The undersigned have examined this dissertation and hereby certify that it is worthy of acceptance for a doctoral degree from POSTECH

10/27/2014

Committee Chair	Jong-Hyeok Lee	(Seal)
Member	Geunbae Lee	(Seal)
Member	Chee-Ha Kim	(Seal)
Member	Seung-Won Hwang	(Seal)
Member	Hyopil Shin	(Seal)

DCSE
20075087

나 휘동. Hwidong Na. Non-projective Parsing for Pre-ordering Statistical Machine Translation. 전처리 어순조정 통계기계번역을 위한 비투사 구문분석. Department of Computer Science and Engineering . 2015. 117P. Advisor: Jong-Hyeok Lee. Text in English.

Abstract

This dissertation thesis treats of the word reordering issue in machine translation Word reordering is an important issue in machine translation because the word orders in the source and target languages are generally different. The degree of the difference between linguistically divergent languages, such as Korean and English, becomes larger than that between similar ones. Although previous methods utilized syntactic trees for word reordering between divergent languages, their hierarchically imposed restrictions often prevents correct word reordering. Although the Inversion Transduction Grammar(ITG) constraints have been widely accepted, it often prevent to improve translation quality between linguistically divergent language pairs because of the restriction.

With three parallel corpora, we manually categorized the source of non-ITG word reordering. As a consequence, They appear in approximately 4% to 10% of all sentences. We also propose a novel reordering method via non-projective parsing. Instead of a syntactic tree from a traditional parser, a *reordering tree* is proposed, which is produced by a reordering parser that is trained with a parallel corpus. It aims to model reordering phenomena more suitably than a syntactic tree by relaxing projective constraints and explicitly encoding reordering orientation in the tree. A reordering parser inspired by a non-projective parser produces the reordering tree.

Under a pre-ordering SMT framework, we conducted experiments for three language pairs in both directions. Our proposed method showed promising improvement in both parsing accuracy and translation quality.

Contents

List of Tables

List of Figures

I. Introduction

1.1 Problem statement

The differences in linguistic typology between languages, such as word order or morphology, are major challenges in machine translation (MT). In terms of word order, human languages are categorized by the typical order of the essential components of a sentence such as subject (S), verb (V), and object (O). The flexibility of word order also varies for different languages, for example, Chinese has a fixed word order while Korean allows a relatively-free word order. Morphological typology is another view of human languages grouping them according to their morphological similarity. A spectrum of complexity of morphological structure ranges from isolating languages, a morpheme is nearly a word, to polysynthetic languages, multiple morphemes form a word.

Because of word order differences, word reordering is necessary for translation between linguistically divergent language pairs. In Figure 1.1, a Korean sentence is translated into an English sentence where a lines represents corresponding counterparts between two languages. Graphically, as the amount of the word order difference becomes greater, the number of line crossings in word alignment increases. In a matrix point of view, a roughly diagonal direction indicates that the word order is similar. On the other hand, linguistically divergent language pairs often have word alignment that are far from diagonal directions, as shown in Figure 1.2

Without loss of generality, MT is a black-box module that receives an *input* consisting of units in the source language and produce the *output* consisting of units in the target language. Different from MT, many natural language processing

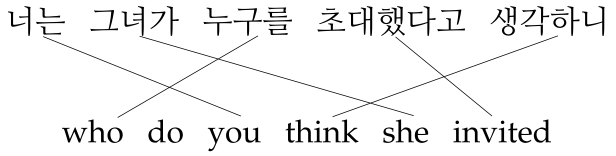


Figure 1.1: A parallel sentence with word alignment

	who	do	you	think	she	invited
너는			■			
그녀가					■	
누구를	■					
초대했다고						■
생각하니				■		

Figure 1.2: A matrix point of view of word alignment

(NLP) tasks such as word segmentation, part-of-speech/chunk tagging, and speech recognition do not suffer from the difference between input/output unit order. Some NLP tasks, e.g. sentence simplification, may requires the change of a sequence of output units, but the number of output units is small in general, and thus it does not cause serious problems as word reordering for MT. Therefore, word reordering is a distinguished issue for MT.

1.2 Scope of Thesis

Statistical machine translation (SMT) achieves relatively high scores by automatic evaluation metrics between similar language pairs [Birch 2008], where similarity of two languages measured using various criteria such as the amount of reordering or historical relatedness. One of the reasons is that statistical models widely used in SMT inherently fit to language pairs similar word order. The reordering models are puzzled by long-distance movement despite word reordering in a short distance can be captured by in-phrase manner. The heuristics of refining word alignment

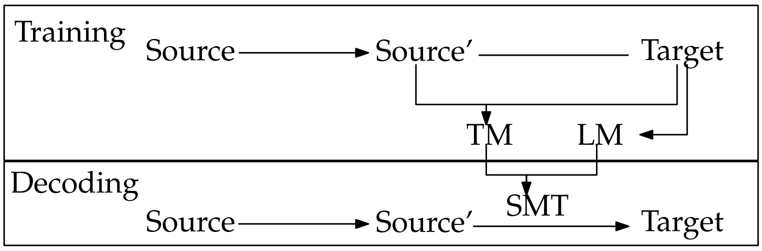


Figure 1.3: The general architecture of the pre-ordering SMT

presume a diagonal correspondence, which may introduce incorrect links or miss correct links. Word alignment models such as the IBM model 2 or HMM are suffer from different word order between languages, because they are based on the relative position of the aligned word. Thus, if we decrease the amount of reordering, it is expected that the translation quality increases.

For this reason, pre-processing approaches for word reordering have been widely studied in the literature of SMT, a.k.a pre-ordering SMT. At the training stage, a pre-processor rearranges source sentences as close to its corresponding target sentence as possible, and SMT models are trained between the reordered sentences and the target sentence. At the decoding stage, it does an identical job, and the reordered sentence are translated into the target sentence. In this manner, word order difference is reduced before translation, and a target-like source sentence ideally corresponds to a target sentence monotonically, i.e. without any word reordering at all. Figure 1.3 shows the general architecture of the pre-ordering SMT.

Compared with other approaches such as syntax-based SMT, pre-processing approaches are beneficial for word reordering. Formally speaking, word reordering is a problem of finding a correct permutation of the source sentence, where the number of all possible permutation grows factorial to the number of the words. If word reordering is dealt together with other problems in MT such as lexical choice, it aggravates the complexity of the problem and thus we often restrict the search space in practice. On the other hand, a separate module for word reordering eases the burden on SMT.

This thesis targets word reordering between Chinese/English and Korean/Japanese. Chinese/English and Korean/Japanese are linguistically divergent language pairs. Korean and Japanese have relatively flexible word order and both belong to SOV language family, while Chinese has a rigid word order and English a relatively fixed one and both are SVO languages. Specifically, our proposed methods utilize data-driven parsing techniques, originally developed for syntactic parsing. Word-aligned corpora provide reordering information for our proposed parser, namely, *reordering parser*s. Analogous to a syntactic parser, a reordering parser produces a hierarchical structure of an input sentence. The hierarchical structure produced by a reordering parser is simpler than syntactic structure, which embeds reordering information. Hereinafter, we call the hierarchical structure *reordering tree.*

1.3 Contribution

Our contributions are as follows:

- We suggest that the Inversion Transduction Grammar (ITG) constraints [Wu 1997] are insufficient to deal with word reordering in real situations. The ITG constraints have been widely used for word reordering in MT studies. They are, however, so restricted that some types of word reordering cannot be handled properly. We analyze three corpora between SVO and SOV languages: Chinese-Korean, English-Japanese, and English-Korean. In our analysis, sentences that require non-ITG word reordering are manually categorized. We also report the results for two quantitative measures that reveal the significance of non-ITG word reordering.

- To overcome such limitations of the ITG constraints, we propose a novel structure for word reordering, called a reordering tree. Monolingual parsing aims to analyze the (hierarchical) syntactic structure of a (flat) word sequence. The syntactic structure is usually represented in a tree form according to grammar. The tree transformation of the syntactic structure, however, may not be suitable for certain word reordering. Our proposed structure does not restrict a tree to be projective, in order to capture non-ITG word reordering as analyzed in the real data.

- Under ITG constrains, we propose three scalable methods to train the reordering parser based on a bottom-up algorithm, in order to utilize training instances as many as possible. Two of them introduce *parallelism* in the algorithm for fast processing. In addition, our proposed method extracts features for the first time, stores them on disk, and reuse them from the second epoch because the online learning process iterates over training instances.

- We propose a novel reordering method via non-projective parsing. Instead of a syntactic tree from a traditional parser, a reordering tree is produced by a reordering parser that is trained with a parallel corpus. It aims to model reordering phenomena more suitably than a syntactic tree by allowing non-projectivity and explicitly encoding reordering orientation in the tree. Our reordering parser mimics two non-projective parsers to obtain the reordering tree. We conducted experiments under a pre-ordering SMT framework and our proposed method showed promising improvement in both parsing and translation.

- We suggest an annotation guideline for word alignment between English and Korean. As two languages have different linguistic phenomena, annotation examples are provided for typical cases. Our guideline includes examples for main predicate, case markers, prepositions, some tenses (present progressive, passive, present perfect), idiom and idiomatic expression, to infinitive, quantitative, and shared component. Together with an automatic inspection of the annotation integrity, the annotation results show a high degree of agreement (κ 91.45).

1.4 Notation

Followings are the notation used throughout this thesis.

Table 1.1: Notations and their meaning used throughout this thesis

Notation	Meaning
f_j, e_i	the jth source, ith target word
$\bar{f}$, $\bar{e}$	a source, target translation unit
F, E	a source, target sentence
F'	a target-like sentence, result of pre-processing
$\mathcal{T}$	a parallel corpus, i.e. $\cup_t \langle F^{(t)}, E^{(t)} \rangle$
$\mathcal{A}$	word alignment $\{(j,i) \| f_j \sim e_i\}$
$\mathbf{w}$	weight vector
$\mathbf{v}$	weight vector for averaged perceptron
$\boldsymbol{\Phi}$	feature function
x	input of structured prediction
y	output (structure) of structured prediction
$\hat{y}$	predicted output of structured prediction
$\mathcal{L}(y, y')$	loss function
$\mathcal{Y}(x)$	a set of outputs for a given input
s_i^j	a span ranges $s_i \dots s_j$
$\mathcal{C}$	a parser configuration
σ	stack
β	buffer
E	a set of (head, modifier) relations $\{(j,i) \| w_i \leftarrow w_j \vee w_j \rightarrow w_i\}$

II. Background

This chapter delivers background knowledge for SMT and word reordering in SMT. Readers who are familiar with that may jump to the the next chapter.

2.1 Basic SMT

In rule-based MT (RBMT), word reordering as well as other problems such as lexical choices are encoded in the translation knowledge (a.k.a rules, dictionary) which is constructed by linguistic experts. It is intuitive and controllable that maintains translation knowledge by human, but the maintenance is also expansive and hard to scale up due to the conflict in legacy and new knowledge as the size of knowledge grows. SMT has evolved for around two decades, which the translation knowledge is automatically trained from parallel/comparable corpora. SMT is cost-effective and achieves comparable/better translation quality than RBMT as the size of data increases.

Conceptually, SMT is a statistical model for machine translation for a given source sentence. Bayes' rule decomposes the probability into two components: translation and language models. [1] Conceptually, translation model (TM) concerns the meaning preservation between the source and target sentence, while language model (LM) does the grammatically of the target sentence. However, sentence-level probabilities ($\Pr(F|E)$ and $\Pr(E)$) are infeasible to learn because of the infinite number of possible sentence constructions. Therefore, TM and LM are factorized into smaller units as follows:

[1] Assuming the probability of the source sentence $\Pr(F) = 1$.

$$\begin{aligned}\Pr(E|F) &= \Pr(F|E)\Pr(E) && (2.1)\\ \Pr(F|E) &= \prod_{\bar{e},\bar{f}} \Pr(\bar{e}|\bar{f}) && (2.2)\\ \Pr(E) &= \prod_{\bar{e}} \Pr(\bar{e}) && (2.3)\end{aligned}$$

According to the translation unit of TM and LM, there exits variety of SMT paradigms. For example, phrase-based TM utilizes a sequence of words as the translation units, which is called a phrase but not always linguistic one. n-gram LM is one of simple yet effective probabilistic model, which is based on the $n-1$ words (history) before the next word. In this thesis, we heavily rely on the phrase-based TM and n-gram LM for simplicity.

A log-linear model is a general framework widely used in SMT defined as follows:

$$Pr(E|F) \sim \sum_m \lambda_m h_m(E,F), \tag{2.4}$$

where λ_m is a weight and $h_m(E,F)$ is a feature function. In this framework, TM and LM are feature functions: $h_{TM}(E,F) = \log Pr(F|E) = \log \prod_k Pr(\bar{f}_k|\bar{e}_k)$ and $h_{LM}(E,F) = \log Pr(E) = \log \prod_i Pr(e_i|e_{i-n+1}^{i-1})$. As any feature function can be defined in this framework, it is assumed that a log-linear model is used for SMT in the later parts.

At decoding phase, phrase-based TM provides lexical choices for a given source sentence, and n-gram LM provides the information for ordering target phrases. Because the number of all possible sub-sequences (phrases) is quadratic to the sentence length, the time complexity of lexical choices is $O(n^2)$. On contrary, that of ordering target phrases is $O(n!)$ which is an NP-complete problem [Knight 1999]. Hence, word reordering has been generally approximated in previous studies.

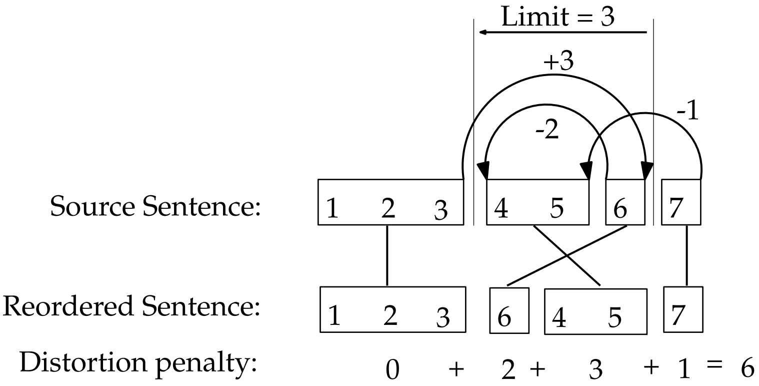


Figure 2.1: An example of distortion limit and penalty

2.2 Word reordering in SMT

Word order differences between the source and target languages require word reordering during translation. Unfortunately, word reordering makes translation more complicated than other natural language processing tasks such as speech recognition. Word reordering is required both in short distances (local reordering) and in long distances (global reordering). Although we cannot make clear distinction between local and global reordering, we refer global reordering as word order differences that go beyond constituency (linguistic phrase). For instance, movement from prepositions in an English constituent to postpositions in that of head-final languages is local reordering, while movement from SVO to SOV is global reordering. Local reordering has been successfully addressed in previous studies, e.g. phrase-based SMT memorizes word reordering in a phrase pair.

In addition to the intra-phrase word reordering, phrase-based SMT utilizes statistical reordering models. Distortion models are based on reordering distance, penalize the amount of phrase rearrangement [Koehn 2003, Green 2010]. For example, a distortion model $Pr(\text{start}_k, \text{end}_{k-1} - 1)$ captures the reordering distance between the current and previous phrases at *word* level, where start_k is the beginning of the current phrase and end_{k-1} is the ending of the previous phrase. Lexicalized reordering models complement distortion models using orientations of reordering, i.e. $Pr(o|\bar{f}_k, \bar{f}_{k-1})$. The orientations o used in [Tillmann 2004, Galley 2008] are monotone, swap, and discontinuous. The orientation is monotone if two phrases are adjacent and straight, swap if adjacent but inverted, and discontinuous if otherwise, respectively.

Although these reordering models are equipped in phrase-based SMT, it is not sufficient to handle long-distance reordering. Distortion models restrict reordering by prohibiting long-distance reordering beyond the distortion limit as shown in Figure 2.1. In lexicalized reordering models, the discontinuous orientation is dominant, for instance, approximately 40% to 80% [Galley 2008]. However, the discontinuous orientation gives little information to determine the word order. In addition, lexicalized reordering models are usually used together with distortion models, which leads to difficulty in global reordering.

For global reordering, previous studies have utilized hierarchical structures (tree). Any change in a (local) tree structure effects its corresponding span, i.e. a sequence of terminals. A synchronous grammar (SG) is one of the formalisms for this purpose. An SG denotes a pair of productions in both the source and target language, together with the one-to-one correspondence of non-terminals. There are variety of SGs such as synchronous context-free grammar (SCFG), synchronous tree substitution grammar (STSG), synchronous tree adjoining grammar (STAG), etc. However, the introduction to these SGs goes beyond the scope of this thesis, so readers are assumed to be familiar with SGs.

Pre-processing approaches that transform a tree using transfer rules have been well-studied [2]. Without loss of generality, transfer rules are described in a form of

[2] There are also a line of work that do not rely on tree structures for word reordering. Based on local information such as base chunks [Zhang 2007], a sequence of base chunks are permuted.

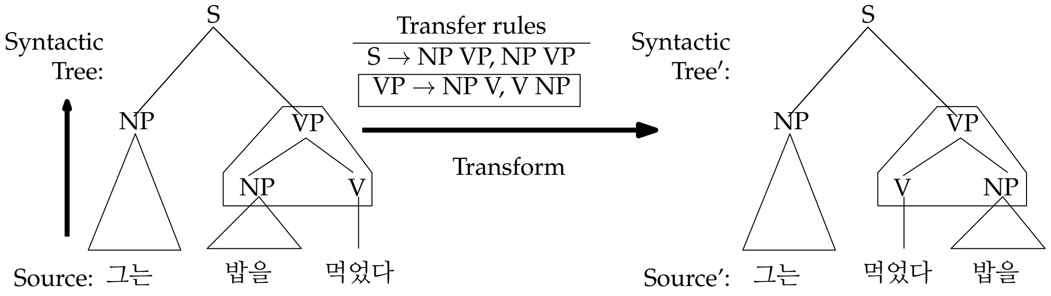


Figure 2.2: An example of pre-ordering using a constituency tree

SG. Previous studies first obtain a hierarchical structure using a syntactic parser. Then, they apply transfer rules that are hand-crafted [Collins 2005, Xu 2009] or automatically obtained from word-aligned corpora [Xia 2004, Genzel 2010]. Figure 2.2 shows an example of pre-ordering using a constituency tree.

In syntax-based SMT, its translation model is actually a SG with probability. A translation rule specifies reordering of substitution sites as well as translated words [Wu 1997]. Syntax-based SMT approaches incorporate tree structures of sentences to the translation rules in the source language [Huang 2006, Mi 2008a, Xiong 2007, Xie 2011] (tree-to-string), the target language [DeNero 2009, Galley 2006, Huang 2009, Shen 2008, Zhang 2006], (string-to-tree) or both [Ding 2005, Eisner 2003, Zhang 2010] (tree-to-tree) [3]. In tree-to-* translation, a translation rule for the source tree replaces the source part with the target part in a top-down manner. In string-to-* translation, on the other hand, it does in a bottom-up manner and eventually produce a target tree over a source sentence. Figure 2.3 shows an example of a tree-to-tree

Some cast word reordering problem to a well-known mathematical problem such as the Travelling Salesman Problem [Zaslavskiy 2009, Visweswariah 2011, Khapra 2013] or the Linear Ordering Problem [Tromble 2009]. Nevertheless, local information does not provide fruitful evidence for word reordering, and thus not suitable for global reordering.

[3] Hierarchical phrase-based SMT [Chiang 2007] is analogous to syntax-based SMT except it does not utilize any syntactic information but a single type of non-terminal X . Hence, it is a string-to-string translation in a perspective of syntax-based SMT.

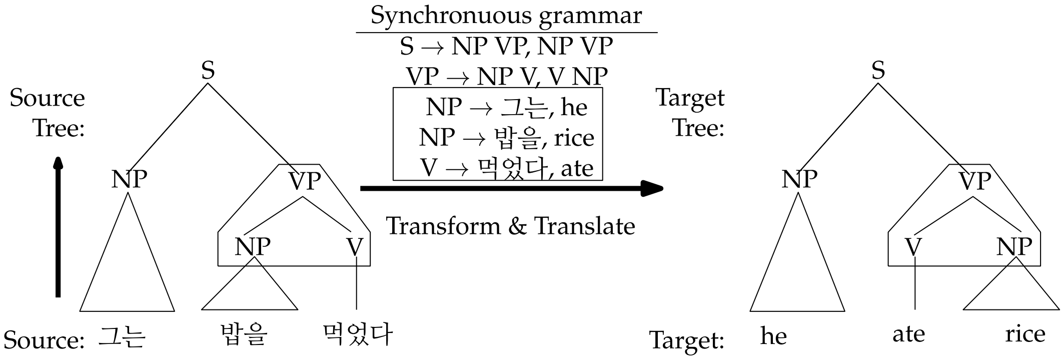


Figure 2.3: An example of a tree-to-tree translation

translation based on a SG over constituency trees.

In terms of reordering capacity, pre-ordering and syntax-based SMT are identical because word reordering rely on tree structure. For both approaches, therefore, transforming a hierarchical structure achieves more fluent translation than distortion and lexicalized reordering models. The accuracy and speed of the syntactic parser is crucial for the performance of both approaches. Although there have been successful efforts to alleviate the parsing error such as forest-based methods [Mi 2008a, Mi 2008b, Mi 2010, Tu 2010, Zhang 2009, Zhang 2010], syntax-based SMT deals with too complicated problem, both reordering and translation, to find the best result. The time complexity of either parsing (tree-to-*) or decoding (string-to-*) is the bottleneck of efficiency, especially for practical purpose. Therefore, we mainly focus on the pre-ordering SMT in the rest of this thesis.

III. Motivation of Thesis

3.1 Validation of the ITG constraints

Inversion Transduction Grammar (ITG) proposed by [Wu 1997] is a context-free grammar (CFG) with transduction, and produces a *pair* of symbols simultaneously. [1] Under the ITG constraint, only two orientations of non-terminals are allowed: straight (ST) and inverted (IV). Let γ be a production rule, consisting of a non-terminal symbol in the left-hand side, terminal/non-terminal symbols in the right-hand side α, terminal/non-terminal symbols in the transduction part β, and a one-to-one mapping between non-terminal symbols in the right-hand and transduction $\sim$. In short, γ is often described in a form analogous to a context-free grammar that denotes the orientation of non-terminal using a square/angle bracket for the straight/inverted orientation, respectively. For production rules only consisting of non-terminal symbols, for example, γ is either $X \rightarrow [YZ]$ (equivalent to $X \rightarrow YZ/YZ$) or $X \rightarrow \langle YZ \rangle$ (equivalent to $X \rightarrow YZ/ZY$).

The ITG constraints have been widely used for word reordering in machine translation studies, in order to overcome the limitation of the simple distortion model, for example, which restricts the maximum range of movements. Because every ITG can be expressed as an equivalent ITG in a two-normal form, there exist polynomial time algorithms for both monolingual $O(n^3)$ and bilingual $O(n^6)$ parsing. Thus, the ITG constraints provide an efficient approximation for restricting the search space of word reordering. Figure 3.1 depicts an example of ITG and its parse tree for the example sentence pair.

[1] As ITG describes a pair of language, it is also a SG.

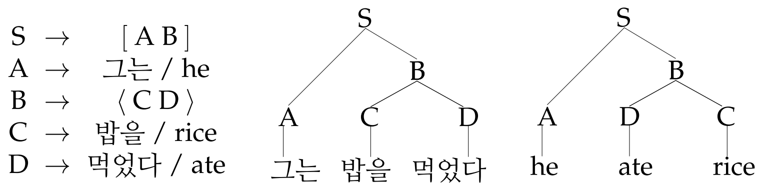


Figure 3.1: An example of ITG and its parse tree for the example sentence pair

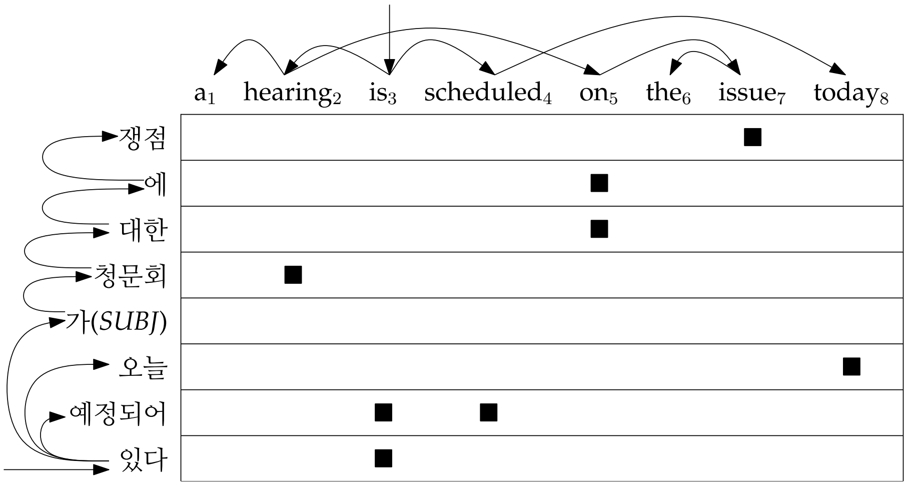


Figure 3.2: An example of Inside-Out word alignment with syntactic structure represented in a dependency grammar. *SUBJ* denotes a nominative case marker.

Although the ITG constraints have been widely used for word reordering in MT, it is impossible to cover all possible word reordering needs, especially for linguistically divergent language pairs. Under the ITG constraints, a non-terminal symbol always covers adjacent words in both the source and target languages. In other words, ITG cannot describe a pair of languages if a non-terminal needs to cover discontinuous words in either the source or target language. In real situations, non-ITG word reordering is often required when translating between linguistically divergent language pairs. A well-known case is the Inside-Out word alignment as shown in Figure 3.2.

Theoretically, ITG coverage decreases drastically as permutation length grows,

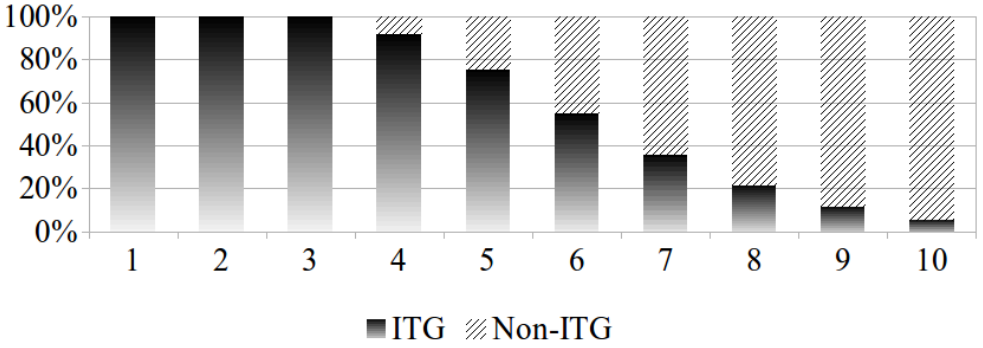


Figure 3.3: Theoretical coverage of ITG and non-ITG permutations with respect to sentence length

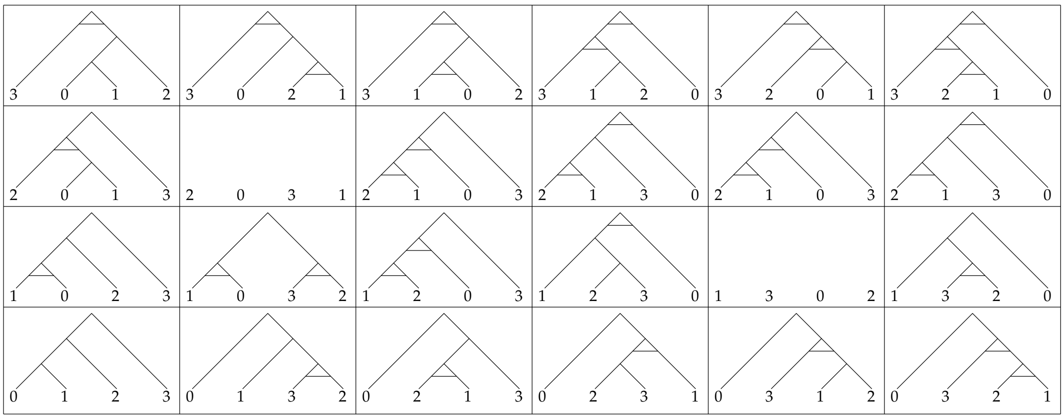


Figure 3.4: All possible permutations with a length-4 sequence. Two cases are unable to be described in the ITG constraints: 1302 and 2031

as shown in Figure 3.3. It is possible for two non-ITG cases to appear in a length-4 permutation as shown in Figure 3.4, which are known as Inside-Out. For a length-5 permutation, ITG cannot describe 25% of cases, though it is questionable this may happen in real situations. Hence, we analyze three word-aligned corpora, and manually identify four types of non-ITG word reordering in Chapter IV.

3.2 Word reordering beyond the ITG constraints

Recently, inducing parsers for pre-ordering SMT have been proposed [DeNero 2011, Neubig 2012, Na 2013]. Unlike syntactic parsers, these parsers are trained with a parallel corpus with word alignment. For a given sentence, these approaches either 1) first produce a parse tree and then permute the children of nodes in the tree [DeNero 2011], or 2) directly produce a parse tree that maximizes the reordering accuracy with latent derivations [Neubig 2012, Na 2013]. Both directions assumed that word reordering can be achieved under the ITG constraints.

Theoretically, however, ITG cannot handle non-ITG permutations, and thus suffers from the lack of coverage for all possible word reordering. To translate the Korean sentence to English in this example, the main predicate "*예정되어 있다 (is scheduled)*" needs to be located between the noun phrase "*청문회(hearing) 가 (SUBJ)*" and the postposition phrase "*쟁점(issue) 에(on)*" [2]. However, it is impossible to transform the syntactic structure regardless of the underlying grammar (constituency or dependency) used. Beside Inside-Out alignment, there also exist known difficulties in machine translation [Søgaard 2009a, Søgaard 2009b], which could be more severe with hierarchically imposed restrictions on the tree structure.

In general, tree-based reordering methods often fail to achieve correct word reordering because of the hierarchically imposed restrictions on the tree structure.

[2] We use glosses instead of Korean words for readability.

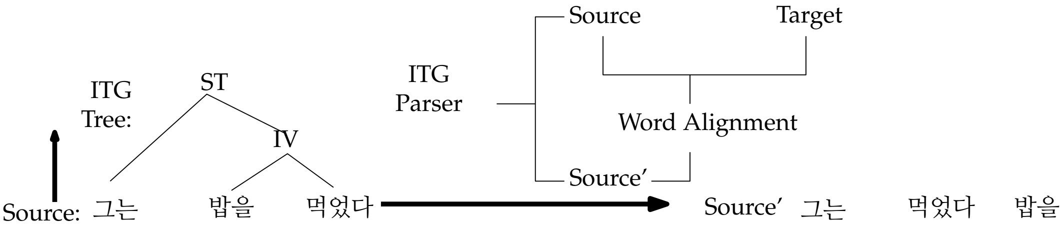


Figure 3.5: Pre-ordering using a reordering parser under the ITG constraints

Table 3.1: Comparison of previous approaches for word reordering

	Global reordering	Required resource	Efficiency
Distortion	Bad	X	Good
Lexicalized	Bad	X	Good
Word-level	Bad	X	Good
Chunk-level	Bad	Shallow Parser	Good
Syntactic tree	Limitedly Good	Syntactic Parser	Good
ITG tree	Limitedly Good	X	Good
Syntax-based	Limitedly Good	Syntactic Parser	Bad

This phenomenon is universal across the language pairs, such as English and Chinese, Romanian, Hindi, Spanish, and French [Wellington 2006]. The failure rates were from 61% (Chinese) to 15% (French) without allowing gaps in the tree structure in English. For relatively free-order languages like Korean and Japanese, certain restrictions become severe problem of word reordering. Table 3.1 summarizes previous approaches for word reordering.

In Chapter 5.2, we propose word reordering methods that go beyond the ITG constraints. First, we define a data structure which make non-ITG word reordering possible. Second, our proposed methods produce the tree in an efficient manner. They consists of training methods for parsing under the ITG constraints, and two parsing algorithms for parsing beyond the ITG constraints.

IV. Linguistic Evidence of Non-ITG Word Reordering

This chapter presents a linguistic analysis of word reordering beyond the ITG constraints between linguistically divergent language pairs. Most of contents come from the article [Na 2014].

4.1 Introduction

The ITG constraints restrict certain kinds of word reordering, for example, wh-questions which have non-ITG word reordering as shown in Table 4.1. We transcribe the gloss for each Korean word, where TOP is a topic marker, NOM a nominative marker, and ACC an accusative marker, respectively. The five words in English “who, you, think, she, invited” and their corresponding words in Korean “너/you, 그녀/she, 누구/who, 초대했다고/invited, 생각하니/think” [1] cannot be described with ITG constraints, because none of continuous words in one language has its corresponding continuous words in other language.

We raise the following research questions need to be answered: 1) Is it sufficient to cover word reordering phenomenon under the ITG constraints in real situation? 2) If not, what are the characteristics of non-ITG word reordering, and how significant are they? 3) If there are significant portion of non-ITG word reordering phenomena, how to deal with them in an efficient manner?

As the answers of the questions, mostly for the first two questions and partially for the third question, this chapter is organized as follows: First, we briefly illustrate

[1] (do, 생각하니) is a possible link, but excluded for brevity.

Table 4.1: An example of non-ITG word reordering with a gold-standard word alignment. A filled box denote a sure link, and an empty box a possible link, respectively.

Korean:	너	는	그녀	가	누구	를	초대했다고	생각하니	?
Gloss:	you	TOP	she	NOM	who	ACC	invite	think-do	?
who					■				
do								□	
you	■								
think								■	
she			■						
invited							■		
?									■

the characteristics of ITG for word reordering, and summarize related works (Section 4.2). Then, we investigate three corpora (Chinese-Korean, English-Japanese, and English-Korean) between subject-verb-object (SVO) and subject-object-verb (SOV) languages to analyze the characteristics of non-ITG word reordering (Section 4.3). Here, we identify four types of non-ITG word reordering, and find that they appear in approximately 4% to 10% of all sentences. It is possible to convert or rewrite sentences that require non-ITG word reordering, in order to satisfy ITG constraints (Section 4.4). Finally, we conclude that word reordering methods should deal with such non-ITG cases, especially between linguistically divergent language pairs (Section 4.5).

4.2 Related work

Previous studies have focused on investigating the capabilities and function of the ITG constraints between English and Chinese [Wu 1997], other European languages

[Zens 2003, Søgaard 2009a], and Arabic [Wu 2006]. Languages in theses studies belong to SOV (English, French, German, Spanish, Portuguese, Danish, Russian, and Chinese), or VSO (Arabic) languages. [2] A notable exception is [Wellington 2006], which summarized word reordering phenomena for an English-Hindi corpus, where Hindi is an SOV language, as well as for English-European languages and English-Chinese. While [Wu 1997] was “unable to find the real example” in their data, [Zens 2003, Wu 2006, Wellington 2006] reported the coverage of the ITG constraints at sentence-level. In addition to the sentence-level coverage, [Søgaard 2009a] estimated empirical lower bounds for the full class of ITGs at the level of translation units. [Xiong 2010b] analyzed reordering patterns in a phrase structure tree and investigated the three reasons for a non-reorderable case.

Although previous works reported the percentage of sentences or translation units violating the ITG constraints, we think that it is more novel to perform a linguistic analysis that identifies types of non-ITG word reordering with concrete examples. We also report two quantitative measures that reveal the significance of non-ITG word reordering, which are inspired by [Wellington 2006] and [Wu 1997]. French-English and Chinese-English pairs were already examined in [Wellington 2006] in a different way. The “failure rates for hierarchical alignment of bilingual bitexts under word alignment constraints only” (Table 2 in their paper) are 1% and 5% for French-English and Chinese-English, respectively, which are roughly comparable with the portion of non-ITG word reordering in this article (approximately 4% to 10%). In addition, we suggest a simple heuristic to convert sentences belonging two of four non-ITG types into ITG-constrained ones.

[2] Some languages have verb-second (V2) word order that is “a specific restriction on the placement of the finite verb in a sentence.” (from `http://en.wikipedia.org/wiki/V2_word_order`)

4.3 Analysis of non-ITG word reordering

In this section, three aspects of non-ITG word reordering are investigated. First, we categorize the types of non-ITG word reordering and collect their statistics. Second, we collect statistics of the size of the word sequence causing non-ITG word reordering. Third, we measure the distribution of minimum SCFG rank to handle non-ITG word reordering.

4.3.1 Experimental settings

We used three corpora consisting of different language pairs and genres. The Chinese-Korean corpus was collected from a newswire service, the English-Japanese corpus from Wikipedia articles [Neubig 2011a], and the English-Korean corpus from example sentences in an electronic dictionary, respectively. Each corpus consists of sentence pairs between SVO and SOV languages, but with different characteristics such as average sentence length. The statistics of the corpora are illustrated in Table 4.3.

For the Chinese-Korean and English-Japanese corpora, human-annotated word alignments are available, but not for the English-Korean corpus. As an alternative, an automatic word alignment was performed for the analysis of the English-Korean corpus. We used MGiza [Gao 2008] to obtain automatic word alignment of the development data together with the training data, consisting of approximately 600K sentences. For the high accuracy of word alignment, the intersection of bidirectional word alignment was analyzed. English sentences were segmented by `tokenizer.perl` included in Moses [Koehn 2007], Japanese sentences by KyTea [Neubig 2011b], and Chinese and Korean sentences by in-house morphological analyzers, respectively.

A simple way to identify the sentence that require non-ITG word reordering is

Table 4.2: An example of identifying non-ITG word reordering for the sentence in Table 4.1

Action	Stack σ	Buffer β
	[]	[who, do, you, think, she, invited]
Shift	[who]	[do, you, think, she, invited]
Shift	[who, do]	[you, think, she, invited]
Reduce	[who·do]	[you, think, she, invited]
Shift	[who·do, you]	[think, she, invited]
Shift	[who·do, you, think]	[she, invited]
Shift	[who·do, you, think, she]	[invited]
Shift	[who·do, you, think, she, invited]	[]

the use of a shift-reduce algorithm analogous to [Huang 2009]. The algorithm uses a stack σ and a buffer β, as shown in Table 4.2. Let $span(i)$ be a target span of a stack element i, which is defined by word alignment. If a stack element i does not have any word alignment, $span(i)$ is *don't-care.* In our experiments, two stack elements i and j are adjacent in the target sentence if $span(i)$ and $span(j)$ are consecutive, overlapped, or one of them are don't-care. Each step performs one of the actions below:

- Reduce: $([\sigma|i|j], \beta) \Rightarrow ([\sigma|i \cdot j], \beta)$ if i and j are adjacent in the target sentence

- Shift: $(\sigma, [k|\beta]) \Rightarrow ([\sigma|k], \beta)$ otherwise,

where i and j are top two elements in σ, $i \cdot j$ is a reduced element covering i and j, and k is the front element in β. After the termination (neither action is applicable), the ITG parsing is successful if the size of stack $|\sigma| = 1$: otherwise, the sentence requires non-ITG word reordering.

Table 4.3: Analysis of non-ITG word reordering between SVO and SOV languages

	Corpus	Chinese-Korean	English-Japanese	English-Korean
	# of sentences	367	1,235	1,895
	# source/target words	8,546/9,677	30,822/34,366	22,594/29,710
	average # words	23.28/26.36	24.95/27.82	11.92/15.67
	Aligned by	Human		MGiza
	Alignment error	3(0.8%)	10 (0.8%)	202 (10.65%)
	Total # of non-ITG w/o error	17 (4.67%)	123 (10.04%)	67 (3.95%)
Type	Wh-questions	0	0	16 (0.95%)
	Adverbials	3 (0.82%)	10 (0.82%)	14 (0.83%)
	Divergence	5 (1.37%)	61 (4.98%)	17 (1.00%)
	Others	9 (2.47%)	52 (4.24%)	20 (1.18%)
Size	∼ 5	2 (0.55%)	13 (1.06%)	17 (1.00%)
	6 ∼ 10	5 (1.37%)	31 (2.53%)	29 (1.71%)
	11 ∼ 15	7 (1.92%)	35 (2.86%)	13 (0.77%)
	16 ∼ 20	1 (0.27%)	15 (1.22%)	5 (0.30%)
	21 ∼	2 (0.55%)	29 (2.37%)	3 (0.18%)
SCFG Rank	4	14 (3.85%)	82 (6.69%)	30 (1.77%)
	5	0	13 (1.06%)	14 (0.83%)
	6	3 (0.82%)	14 (1.14%)	9 (0.53%)
	7	0	3 (0.24%)	6 (0.35%)
	8 ∼	0	11 (0.90%)	8 (0.47%)

Table 4.4: Examples of non-ITG word reordering. English glosses are given below non-English sentences with grammatical markers such as TOP (topic), NOM (nominative), ACC (accusative), LOC (locative), or GEN (genitive).

(a) Wh-questions

(1) English: *why don't the police raid the school-yard bullies?*									
Korean:	경찰	은	왜	학교	깡패들	을	단속하지	않는가	?
Gloss:	police	TOP	why	school-yard	bullies	ACC	raid	not-do	?
(2) English: *what did the Senator achieve on his trip to China?*									
Korean:	상원의원	은	중국	여행	에서	무엇	을	얻었는가	?
Gloss:	senator	TOP	china	trip	on	what	ACC	achieve-do	?

(b) Adverbials

(3) Chinese:	脂肪	主要	堆积	在	肠	与	肠	之间	。					
Gloss:	fat	mostly	accumulate	at	intestine	and	intestine	between	.					
Korean:	주로	창자	와	창자	사이	에	지방	이	쌓인다	.				
Gloss:	mostly	intestine	and	intestine	between	LOC	fat	NOM	accumulate	.				
(4) English: *in 1594, Hideyoshi began construction of Fushimi Castle for his residence*														
Japanese:	秀吉	は	居所	として	1594年	には	伏見	城	の	築城	を	始め	ている	。
Gloss:	Hideyoshi	TOP	residence	as	1594	in	Fushimi	castle	GEN	construct-castle	ACC	begin	did	.

Table 4.5: Examples of non-ITG word reordering. English glosses are given below non-English sentences with grammatical markers such as TOP (topic), NOM (nominative), ACC (accusative), LOC (locative), or GEN (genitive).

(a) Specific examples of divergence

(5) English: *he spent his life in Africa helping the poor.*											
Korean:	그	는	아프리카	에서	가난한	사람들	을	평생	도왔다	.	
Gloss:	he	NOM	Africa	LOC	poor	people	ACC	lifelong	helped	.	
(6) English: *a unit called Hokoshu or Bugyoshu was organized to handle practical matters.*											
Japanese:	奉公衆	や	奉行衆	と	呼ばれる	実務	官僚	を	整備	した	。
Gloss:	Hokoshu	or	Bugyoshu	as	called	handle-practical-matter	bureaucracy	ACC	organize	did	.
(7) English: *the construction of Jurakudai began in February 1586.*											
Japanese:	聚楽第	は	1586年	2月	に	着工	された	。			
Gloss:	Jurakudai	TOP	1586	February	in	construction-begin	was	.			

(b) Others

(8) English: *Fujiwara became one of the Emperor Uda's wives after his enthronement.*													
Japanese:	藤原	は	宇多	帝	即位	後に	入内	した	。				
Gloss:	Fujiwara	TOP	Uda	emperor	enthronement	after	become-queen	did	.				
(9) Chinese:	精制油	的	关税率	下调	至	10%	，	调和油	下调	至	8%	。	
Gloss:	refined-oil	GEN	tariff-rate	decrease	to	10%	,	mixed-oil	decrease	to	8%	.	
Korean:	정제유	관세율	은	10%	로	,	혼합유	는	8%	로	각각	인하된다	.
Gloss:	refined-oil	tariff-rate	TOP	10%	to	,	mixed-oil	TOP	8%	to	respectively	decrease	.

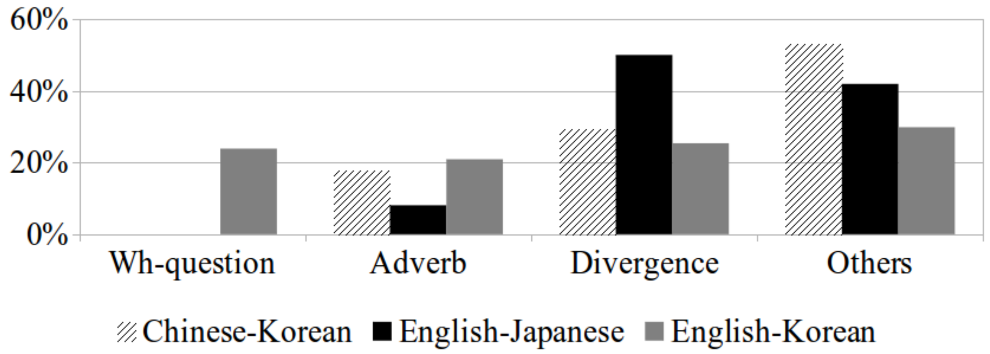


Figure 4.1: The proportions of each type of non-ITG word reordering across the three corpora

4.3.2 Types of non-ITG word reordering

We categorize four types of non-ITG word reordering: wh-questions, adverbials, specific examples of divergences, and others. For the English-Korean corpus, we identify non-ITG word reordering based on an automatic word alignment, which may contains many errors. Thus, we manually exclude 202 sentences which contain word alignment error from the analysis. Even in human annotated word alignment, there exist minor errors of word alignment. We also manually exclude these sentences (around 0.8% of sentences) in the Chinese-Korean and English-Japanese corpora.

Wh-questions

A language pair with different word-order typologies often requires non-ITG word reordering to translate wh-questions. For a rigid word-order language like English, the order of constituents in wh-questions is almost completely fixed. For a relatively free-order language like Korean, in contrast, it is flexible. Therefore, ITG word reordering does not guarantee correct results for wh-questions between rigid and relatively free word-order languages.

Some examples of wh-questions are shown in Table 4.4a. In (1), the word alignment is (why, 왜), (don't, 않는가), (police, 경찰), (raid, 단속하지), (school-yard, 학교), and (bullies, 깡패들). The word order in English is rearranged into "police, why, school-yard, bullies, raid, don't" in Korean, and none of adjacent words in one language is adjacent in the other language. The example (2) also requires non-ITG word reordering for the word order in Korean "Senator, China, trip, on, what, achieve".

Because newswire and Wikipedia rarely utilize wh-questions[3], we only found this type of non-ITG word reordering in the English-Korean corpus. We found a total of 16 wh-question sentences requiring non-ITG word reordering among 1,693 sentences with no errors of word alignment. They account for 39% of 41 wh-questions in this corpus.

We found that this type of non-ITG word reordering would be recognized as a generalized pattern. Let AUX be an auxiliary verb, S a subject, V a verb, and VP/PP a verb/preposition phrase, respectively. The example of Table 4.1 can be generalized as a pattern (Wh- AUX S_1 V_1 S_2 V_2 / S_1 S_2 Wh- V_2 V_1). The example (1) of Table 4.4a shows a pattern (Wh- AUX S_1 VP, S_1 Wh- VP AUX), and (2) a pattern (Wh- AUX S_1 V_1 PP , S_1 PP Wh- V_1), respectively.

Adverbials

Adverbials have a higher degree of freedom in word-order than other constituents. "In reality, adverbials are very free in their placement, appearing in different positions in the sentence, not just sentence final"[Brinton 2000] in English. Generally, the relatively free order of adverbials is language universal. Therefore, adverbials often result in non-ITG word reordering phenomena during translation.

[3] We cannot find any wh-questions in the Chinese-Korean and English-Japanese corpora.

Some examples of adverbials are shown in Table 4.4b. In (3), an adverb "主要/mostly" is located between the subject "脂肪/fat" and the main predicate "堆积/accumulate" in Chinese, while its corresponding word "주로/mostly" appears at the begging in Korean. The example (4) also shows that an adverb phrase "in 1594" at the beginning of the English sentence is located between "として/as" and "伏見/Fushimi" in Japanese.

Regardless of domains and language pairs, adverbials incur non-ITG word reordering. After excluding the erroneous word alignment, there remains around 0.8% of sentences that need non-ITG word reordering because of adverbials. This type takes the smallest portion among those we found for each corpus as shown in Figure 4.1. We also found that this type of non-ITG word reordering can be generalized. For (3), a pattern would be (S AdvP V PP / AdvP PP S V), and for (4), it is (AdvP S V PP / S PP AdvP V), where AdvP is an adverb phrase.

Specific examples of divergence

Translation divergences are a well-known problem in machine translation [Dorr 1994]. We found that certain specific examples of divergence, such as head-switching, structural, and categorial divergences, cause non-ITG word reordering. Head switching divergence refers to cases where the dependency relation between two constituents has switched after translation. For example, the example (5) of divergence in Table 4.5a shows that the dependency relation between a main verb "spent" and its argument "helping" in English is switched in Korean.

The structural or categorial divergences refer to the syntactic structure or category changes during translation, respectively. For example, (6) in Table 4.5a shows that the to-infinitive phrase "to handle practical matters" in English becomes the one-word noun phrase "実務/handle-practical-matters" in Japanese. The example

(7) also shows that the argument “Jurakudai” of the preposition “of” in English becomes the subject “聚楽第/Jurakudai” in Japanese[4].

In general, the percentage of non-ITG word reorderings caused by translation divergences is larger than by wh-questions and adverbial cases. Especially, in the English-Japanese corpus, translation divergences are the most significant reason for non-ITG word reordering. For the Chinese-Korean corpus, head-switching accounts for 40% and categorial 60%; For the English-Japanese corpus, head-switching 27.8%, structural 32.8%, and categorial 39.4%; For the English-Korean corpus, head-switching 23.5%, structural 35.3%, and categorial 41.2%, respectively. They show a wide variety of non-ITG word reorderings, which are hard to fully describe using simple patterns or rules.

Others

The fourth type of non-ITG reordering includes discontinuous translation units, natural translation, and so on. Here, discontinuous translation units mean that they have discontinuity in either the source or target language only. In such cases, discontinuous translation units in one language are translated into a word or adjacent words in the other language, so that correct word reordering of discontinuous translation units (DTUs) can improve machine translation. In Table 4.5b, the example (8) shows this type of non-ITG word reordering. A translation units “became one of the Emperor’s ... wives” in English is translated into two adjacent words “入内/become-queen した/did”.

As opposed to literal translation, natural translation often drops some constituents during translation. In (9), the first main verb “下调/decrease” in the Chinese sentence is omitted in the Korean sentence. Hence, two Chinese words

[4] There is also a conflational divergence, that a lexical gap “construction” and “began” is translated in a Japanese word “着工/begin-construction”.

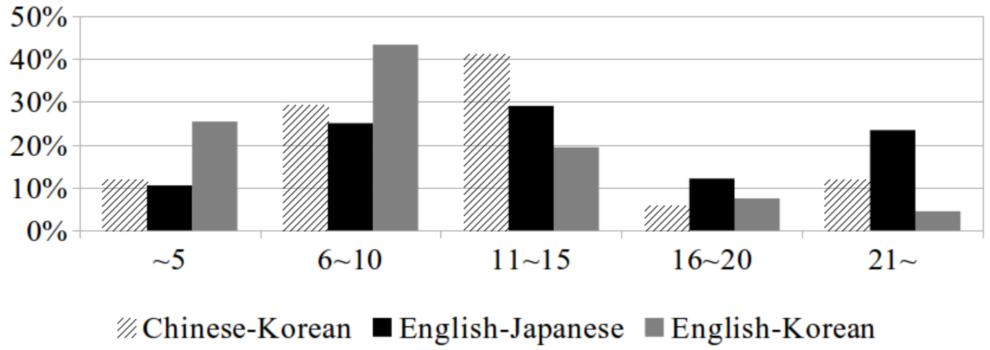


Figure 4.2: The distribution of the size of non-ITG word reordering across the three corpora

"下调/decrease" correspond to one Korean word "인하된다/decrease", which causes non-ITG word reordering. In addition, there are difference in writing style between languages. For example, a list of pairs in one language is written in two separated lists in the other language as follows: "A is a, B b, and C c, respectively" in English "A, B, C 는/TOP 각각/respectively a, b, c 이다/is" in other language.

This type of non-ITG word reordering accounts for the largest percentages of the Chinese-Korean and English-Korean corpora. As the corpora have different characteristics, the causes for this type are different. For the Chinese-Korean corpus, the omission of counterparts in one language accounts for 66.6% and the different style of writing 33.3%; For English-Japanese corpus, the omission 63.5%, DTUs 23%, and the style 13.5%; For English-Korean corpus, the omission 44%, DTUs 44%, and the style 12%, respectively. Similar to the third type, this type depends on so specific lexical pattern that we cannot find any generalized/structural patterns.

4.3.3 Size of non-ITG word sequence

We report the smallest size of the word sequence causing non-ITG word reordering, which is computed from failed ITG parsing results. After parsing, if a sequence of elements in the stack σ forms a continuous span in the target, we regard its corresponding word sequence in the source language as a non-ITG word reordering sequence. Many different sub-sequences in a sentence may require non-ITG word reordering. Among certain sequences, the minimum size is regarded to be equivalent to the size of non-ITG word reordering of the sentence. For example, the size of non-ITG word reordering of the example of Table 4.1 is 6, because the stack is [who, do, you, think, she, invited, ?] after the ITG parsing and “who, do, you, think, she, invited” is the minimum size of the word sequence forming a continuous span in the target sentence.

The histogram of the size of non-ITG word reordering is illustrated in Figure 4.2. The Chinese-Korean and English-Japanese corpora have longer sentences on average than English-Korean one, and the size of non-ITG word sequence from 11 to 15 accounts for the largest portion. For the English-Korean corpus, the largest portion ranges from 6 to 10 and the average is 9.25, which almost reaches the average sentence length of 11.92 for this corpus. This indicates that non-ITG word reordering is more likely to be global reordering than local one.

4.3.4 SCFG rank of non-ITG word reordering

It is also valuable to inspect the minimum SCFG rank to deal with non-ITG word reordering. The rank of an SCFG is the maximum number of the non-terminal symbols in the right hand side of the grammar. The expressive power of an SCFG with arbitrary rank, which generates all possible permutations for a given sequence, is generally much greater than that for ITG. The minimum rank of non-ITG word

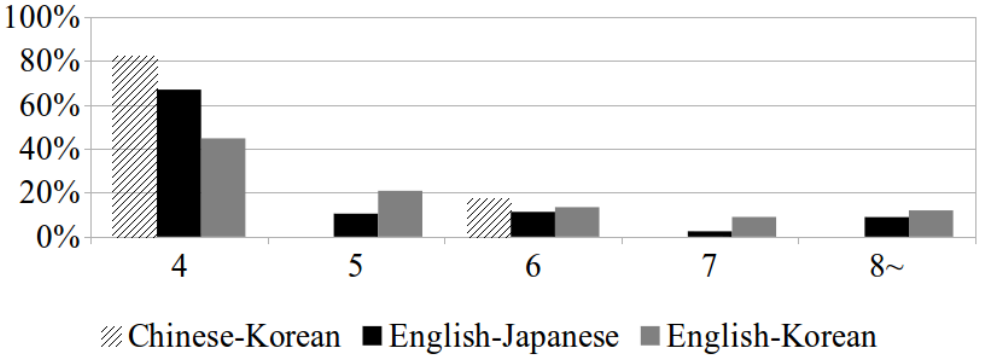


Figure 4.3: The distribution of the minimum SCFG rank for non-ITG word reorderings across the three corpora

reordering is also computed from the failed ITG parsing result. After parsing, if a sequence of elements in σ forms a continuous span in the target, the number of elements in the sequence is the rank. Among certain sequences consisting of at least 4 elements, the minimum rank is regarded as the minimum SCFG rank of non-ITG word reordering of the sentence.

The distribution of the minimum rank is illustrated in Figure 4.3. Most non-ITG word reordering requires a rank-4 SCFG, while there exist long tails of lager ranks. Obviously, ITG constraints, which has an expressive power of a rank-2 SCFG, is insufficient because it misses a certain amount of non-ITG word reordering. In next section, we explore the possibility of transforming non-ITG word reordering to satisfy ITG constraints.

4.4 Transformation of non-ITG word reordering

We found many non-ITG word reordering cases between SVO and SOV languages. One major reason is the degree of freedom in word order. The SOV languages have a general tendency of having flexible word order, and they are morphologically-rich languages to indicate the roles of constituents. Adverbials are another source of freedom of word order regardless of languages.

In a head-final language with a flexible word order such as Korean or Japanese, nearly all constituents except the main predicate can be located anywhere before its head. The examples (2) to (4) are such cases that constituents except the main predicate can be scrambled. A simple rule for handling the freedom of word order is moving the wh-words and adverbials in front of the clause they belong to. This is a possible solution converting the word order without harming the meaning of the source sentence. In Table 4.1, for example, the following two Korean sentences have the same meaning, but Korean$_1$ causes non-ITG word reordering, while Korean$_2$ does not. The brackets denote the clause boundaries.

Converting the wh-question type

English: who do you think she invited?
Korean$_1$: [너/you 는/TOP [그녀/she 가/NOM 누구/who 를/ACC 초대했다고/invite] 생각하니/think-do ?]
Korean$_2$: [너/you 는/TOP [누구/who 를/ACC 그녀/she 가/NOM 초대했다고/invite] 생각하니/think-do ?]

For the English-Korean corpus[5], we inspect the portion of acceptable conversions using the simple rule, and 30 sentences of 16 wh-questions and 14 adverbials types are indeed successful, accounting for 100%. This simple heuristic is also applicable to sentences including more than one adverbials as follows[6]:

[5] We focused on the English-Korean corpus because the numbers of the adverbial types in the Chinese-Korean and English-Japanese corpora are too small to inspect.

[6] It would be controversial whether the clause boundary include the shared subject "나/I 는/TOP" or not. Our analysis is based on the sentence structure as simple as possible, which is more like to be practical

Converting the adverbial type

English: I really want to$_1$ speak to$_2$ her directly.

Korean$_1$: [나/I 는/TOP [그녀/her 와/to$_2$ 반드시/really 직접/directly 통화-하/speak 고/to$_1$] 싶다/want].

Korean$_2$: [나/I 는/TOP [반드시/really 직접/directly 그녀/her 와/to$_2$ 통화-하/speak 고/to$_1$] 싶다/want].

The third and fourth types of non-ITG word reordering would be transformed by rewriting the sentence. For (7) of Table 4.5a, it is possible to make Japanese$_1$ as close to its literal translation as possible like Japanese$_2$.

Rewriting the divergence type

English: the construction of Jurakudai began in February 1586.

Japanese$_1$: 聚楽第/Jurakudai は/TOP 1586年/1586 2月/February に/in 着工/construction-begin された/was.

Japanese$_2$: 聚楽第/Jurakudai の/GEN 建設/construction は/TOP 1586年/1586 2月/February に/in 始ま/begin った/was.

For (9) of Table 4.5b, it is similarly possible to rewrite Korean$_1$ as Korean$_2$, as follows.

Rewriting the other type

Chinese: 精制油/refined-oil 的/GEN 关税率/tariff-rate 下调/decrease 至/to 10%, 调和油/mixed-oil 下调/decrease 至/to 8%.

Korean$_1$: 정제유/refined-oil 관세율/tariff-rate 은/TOP 10% 로/to, 혼합유/mixed-oil 는/TOP 8% 로/to 각각/respectively 인하된다/decrease.

Korean$_2$: 정제유/refined-oil 관세율/tariff-rate 은/TOP 10% 로/to 인하되고/decrease-

and, 혼합유/mixed-oil 는/TOP 8% 로/to 각각/respectively 인하된다/decrease.

Among 17 divergences and 20 other types in the English-Korean corpus, 20 sentences are successfully converted by moving words properly, and 17 sentences by re-wording, which account for 54% and 46%, respectively, Although we cannot find any non-ITG sentences that are genuinely impossible to rewrite in a way that allows the ITG constraint. In practice, it is hard to manually correct all sentences in parallel corpora, and an automatic method of this process would be necessary.

4.5 Concluding remarks

With three parallel corpora, we manually categorized the source of non-ITG word reordering between three SVO and SOV languages. We measured the size of non-ITG word reordering and the corresponding distribution of the minimum SCFG ranks. As a result, significant amount of non-ITG word reordering can be found. Although the portion of non-ITG word reordering in practice is much smaller than that in theory, it is nevertheless considerable. Therefore, ITG constraints is not adequate to the modelling of word reordering in machine translations in real situation. It remains difficult to correct sentences that violate ITG constraints automatically.

Dealing with non-ITG word reordering in an efficient manner is still a challenging problem. The search space beyond ITG constraints is much larger than that within ITG constraints, and "unconstrained reordering is only helpful if we are able to estimate the reordering probability reliably [Zens 2004]." One possible solution is to improve the distortion model used in statistical machine translation [Al-Onaizan 2006, Green 2010, Goto 2013]. With a large amount of distortion or unlimited distortion, these approaches achieved translation quality over a simple distance-based distortion model. Utilizing translation units with gaps (or discontinuous translation units) is also another approach [Simard 2005, Crego 2009, Galley 2010]. Some of pre-ordering approaches cast word reordering problem to a well-known mathematical problem such as the Travelling Salesman Problem [Zaslavskiy 2009, Visweswariah 2011, Khapra 2013] or the Linear Ordering Problem [Tromble 2009]. Their methods have potential to deal with non-ITG word reordering as well, though [Tromble 2009] restrict the search space with ITG constraints for efficiency. In the following chapter, we will deal with the problem of non-ITG word reordering to overcome the limitation of ITG constraints, to enhance the capabilities of word reordering, and in this way eventually to improve translation quality.

V. Pre-ordering via Parsing

Heretofore, we explore the limitation of the ITG constraints for word reordering. This chapter presents ways to deal with non-ITG word reordering under a pre-ordering SMT framework. First, we deliver a brief introduction to structured prediction and summarize parsing algorithms for syntactic trees as a relevant theory of our proposed methods. Then, it is illustrated that three methods for scalable training under the ITG constraints, and two parsers for word reordering beyond the ITG constraints.

5.1 Background knowledge

5.1.1 Structured prediction for NLP

Structured prediction is a machine learning (ML) technique trained over structured objects such as a sequence of POS tags or a parse tree. Equation 5.1 is a mathematical notation of structured prediction.

$$\hat{y} = \arg\max\nolimits_{y \in \mathcal{Y}(x)} \mathbf{w} \cdot \mathbf{\Phi}(x, y) \tag{5.1}$$

The major difference between structured prediction and multi-class (or simply binary) classification is the number of all possible labels, i.e. the size of the target domain. For multi-class classification, the number of all possible labels (or classes) is fixed. If we consider all possible labels, which are usually exponential to the input size, structured prediction is infeasible. Hence, an approximated (inexact) search substitutes the exhaustive (exact) search at the training stage.

The goal of the training is to find an indicator (weight **w**) to discriminate good

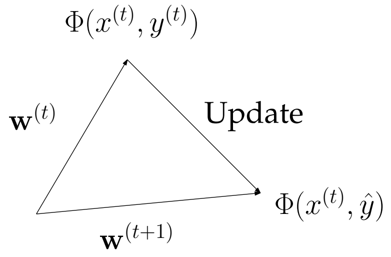


Figure 5.1: A graphical view of online learning for each training instance

structured objects from bad ones. For scalable training, structured perceptron tries to update $\mathbf{w}$ for each training instance, a.k.a online learning. For each training epoch, the current weight vector is updated toward the difference of feature vectors between the gold-standard (derivation) y and the prediction $\hat{y}$, i.e. $\Delta\mathbf{\Phi}(x, y, \hat{y}) = \mathbf{\Phi}(x, y) - \mathbf{\Phi}(x, \hat{y})$. In a graphical view point, current weight vector $\mathbf{w}^{(t)}$ changes toward the vector representation of the gold-standard y, and against the prediction $\hat{y}$, as shown in Figure 5.1.

One of the well-known techniques is the structured perceptron with averaging weights suggested by [Collins 2002]. For each training instance, the difference is $\Delta\mathbf{\Phi}(x, y, \hat{y})$ accumulated to $\mathbf{v}$ in addition to the immediate update of $\mathbf{w}$ At the end of the training, $\mathbf{v}$ are divided by the number of updates, i.e. $N \times |T|$, where N is the number of iterations. It is known to avoid over-fitting problem in online learning.

Standard update with inexact search, however, does not hold the theoretical property of the convergence as pointed out in [Huang 2012]. The early update is one of solutions for online learning with inexact search [Collins 2004]. Let y_1^i be a i-prefix of the structured output y, e.g, y_1^3 is “D N V” for a sequence of POS tags “D N V D N”. Whenever the gold-standard derivation is pruned out during inexact search, the early update reflects the difference of feature vector between y_1^i and $\hat{y}_1^i$, where i is the first difference of the gold and best in beam. In general, any violation-fixing update with inexact search still hold the theoretical property of the convergence. A violation is defined as follows:

$$y_1^i \neq \hat{y}_1^i \wedge \mathbf{w} \cdot \Delta\boldsymbol{\Phi}(x, y_1^i, \hat{y}_1^i) \leq 0 \tag{5.2}$$

By definition, it is possible to exist more than one violation. Let $\mathcal{V}$ be a set of triples which are violations, i.e. $\mathcal{V} = \{(x, y_1^i, \hat{y}_1^i) | y_1^i \neq \hat{y}_1^i \wedge \mathbf{w} \cdot \Delta\boldsymbol{\Phi}(x, y_1^i, \hat{y}_1^i) \leq 0\}$. The update against the triple that maximize the amount of violation fastens the convergence as well as improves the search quality [Huang 2012, Yu 2013, Zhang 2013, Zhao 2014], i.e.

$$\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} + \mathbf{w}^{(t)} \cdot \Delta\boldsymbol{\Phi}(x, y_1^{i^*}, \hat{y}_1^{i^*}), \tag{5.3}$$

where $i^* = \arg\min_i \mathbf{w} \cdot \Delta\boldsymbol{\Phi}(x, y_1^i, \hat{y}_1^i)$.

For learning with latent variables, e.g. translation lattice, the gold-standard derivation is not unique. In [Yu 2013], for instance, the authors defined y-good derivations as "the set of partial derivations whose English side is a prefix of the reference translation y, and whose Chinese projection covers exactly i words on the input sentence x", and did y-bad derivations conversely. The max-violation of Equation 5.3 changes as follows:

$$\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} + \mathbf{w}^{(t)} \cdot \Delta\boldsymbol{\Phi}(x, d_i^+(x, y), d_i^-(x, y)), \tag{5.4}$$

where $d_i^+(x, y)$ and $d_i^-(x, y)$ are the best derivations among y-good and y-bad derivations during inexact search, respectively.

Large-margin methods have been utilized for structured prediction for many NLP tasks [McDonald 2005a, Daumé 2005, McDonald 2006a, Watanabe 2007, Chiang 2008]. In binary classification case, for example, "a margin of an example is proportional to the distance between the instance and the hyperplane" [Crammer 2006] which separates positive and negative instances. Because "the perceptron algorithm ... does not optimize any notion of classification margin, which is widely accepted to reduce generalization error" [McDonald 2006a], a large-margin method tries to maximize the margin. For example, the Margin Infused Relaxed Algorithm (MIRA) is a large-margin method that the update is as follows:

$$\mathbf{w}^{(t+1)} \leftarrow \arg\min \|\mathbf{w} - \mathbf{w}^{(t)}\| \tag{5.5}$$

such that $\forall y', \mathbf{w} \cdot \Delta\mathbf{\Phi}(x, y, y') \geq \mathcal{L}(y, y')$. While this is an constrained optimization, a.k.a quadratic programming, it would be approximated to use a small number of k-best, even 1-best [McDonald 2005b, McDonald 2006b]. It is also possible to average weights from the MIRA iterations to prevent the over-fitting problem.

5.1.2 Data-driven parsing algorithms

Because our proposed methods adopt existing parsing algorithms, it helps readers understand this thesis in detail. Specifically, dependency parsing algorithms are illustrated although constituency or other formalisms are also able to be adopted for the same purpose. For whom familiar with data-driven dependency parsing, this section is negligible.

For a given sentence, a dependency parser produces a dependency tree consisting of dependency relations between words. A dependency relation is a binary relation between a head and modifier. A head is a word that *dominates* one or more modifiers, which are *dependent* upon the head. While a dependency tree is assumed to be *well-formed*, i.e. rooted, single-headed, and acyclic [Nivre 2008], projectivity is often not assumed for languages such as Czech and Danish. In a structural viewpoint, a dependency tree is a directed graph of a set of words.

A tree-bank is a set of syntactic trees constructed by human annotators. Existing dependency tree-banks are available for many languages such as Brazilian-Portuguese, English, Finnish, French, German, Italian, Indonesian, Japanese, Ko-

Table 5.1: An example of transition-based projective dependency parsing. SH denotes Shift, LA Left-Arc, and RA Right-Arc, respectively.

Action	Stack σ	Buffer β	Arc
SH	[he]	[ate, rice, with, chicken]	
SH	[he, ate]	[rice, with, chicken]	
LA	[ate]	[rice, with, chicken]	(he, ate)
SH	[ate, rice]	[with, chicken]	
SH	[ate, rice, with]	[chicken]	
RA	[ate, rice]	[with]	(chicken, with)
RA	[ate]	[rice]	(with, rice)
RA	[]	[are]	(rice, ate)

rean, Spanish and Swedish [1]. The number of sentences in a tree-bank ranges approximately from 5K to 20K, which make us to decide to annotate at least 10K sentences for our parser. Details will be explained in Chapter VI.

With a discriminative model as described in above, decoding plays the central role of dependency parsing. We categorize dependency parsing algorithms according to the time complexity. Linear-time algorithms have been spotlighted mostly for efficient parsing. Cubic-time algorithms, on the other hand, have more opportunities to produce the correct parse tree because of the larger search space than linear-time algorithms. Quadratic-time algorithms that cast the problem to finding the maximum spanning tree are originally proposed for non-projective dependency parsing, but they are not used in this thesis.

A linear-time algorithm runs in linear time to the sentence length. In a graphical view, it processes a word sequence in a left-to-right manner. Transition-based dependency parsing has provided the foundation of linear-time algorithms [Nivre 2003, Nivre 2008], though there were quadratic-time algorithms in early period [Kudo 2000,

[1] https://code.google.com/p/uni-dep-tb/

ALGORITHM 1: Eisner's algorithm for projective dependency tree

```
input  : A sentence $w_1 \dots w_N$
output: A well-formed dependency tree
Initialize Chart C
for $L \in [1, N]$ do
    for $i \in [0, n - L)$ do
        $k \leftarrow i + L$ unless $k > N$
        $C(i, k, \leftarrow, 0) = \max_{i \leq j < k} C(i, j, \rightarrow, 1) + C(j + 1, k, \leftarrow, 1) + score(k, i)$
        $C(i, k, \rightarrow, 0) = \max_{i \leq j < k} C(i, j, \rightarrow, 1) + C(j + 1, k, \leftarrow, 1) + score(i, k)$
        $C(i, k, \leftarrow, 1) = \max_{i \leq j < k} C(i, j, \leftarrow, 1) + C(j, k, \leftarrow, 0)$
        $C(i, k, \rightarrow, 1) = \max_{i < j \leq k} C(i, j, \rightarrow, 1) + C(j, k, \rightarrow, 0)$
    end
end
```

Kudo 2002, Yamada 2003], It "tries to perform parsing deterministically, using a classifier trained on gold standard derivations from a tree-bank to guide the parser" [Hall 2008]. The classifier determines one of actions that incur a transition. A transition refers a change of a parser configuration $\mathcal{C}$ consisting of $\langle \sigma, \beta, E \rangle$. For example, a transition-based system for projective dependency parsing incorporates the following actions:

- *Shift*: $\langle \sigma, w_j | \beta, E \rangle \Rightarrow \langle \sigma | w_j, \beta, E \rangle$
- *Right-Arc*: $\langle \sigma | w_i, w_j | \beta, E \rangle \Rightarrow \langle \sigma, w_i | \beta, E \cup \{(w_i, w_j)\} \rangle$
- *Left-Arc*: $\langle \sigma | w_i, w_j | \beta, E \rangle \Rightarrow \langle \sigma, w_j | \beta, E \cup \{(w_j, w_i)\} \rangle$

This is called the "arc-standard" version of the Malt parser [Nivre 2006]. For each word, exactly one *Shift* and either *Right-Arc* or *Left-Arc* action is taken. Hence, the time complexity is linear to the sentence length. Table 5.1 shows an example of projective dependency parsing using a transition system.

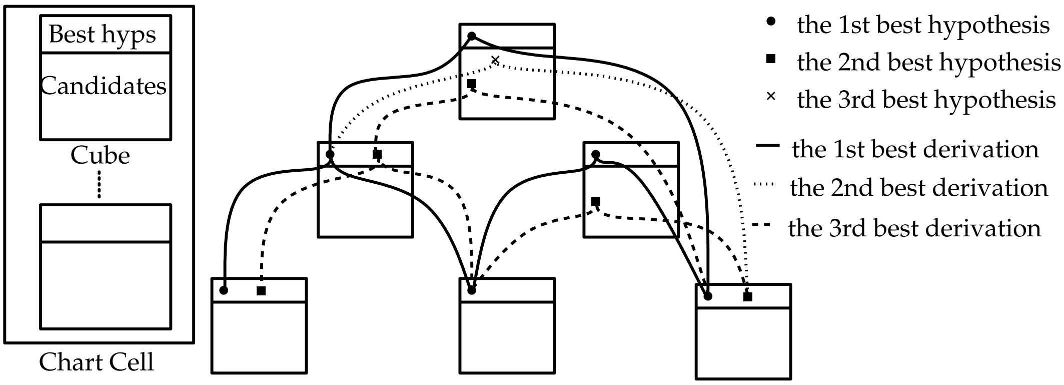


Figure 5.2: An intermediate stage of Cube Growing. When the best hypothesis is requested, it produces the second and third best hypotheses, requesting the second best hypotheses in the cell below, and recursively so on.

A cubic-time algorithm runs slower than linear-time because of larger search space. In a bottom-up manner, it produces a larger structure by composing smaller structures. The Cocke-Younger-Kasami (CYK) algorithm is a cubic-time algorithm originally developed for context-free grammars. [Eisner 1996] suggested an efficient algorithm producing a projective dependency tree in cubic time. As a deductive system, the Eisner's algorithm is defined as follows:

- Incomplete cases: $\dfrac{\langle i,j,\rightarrow,1\rangle \quad \langle j+1,k,\leftarrow,1\rangle}{\langle i,k,\leftarrow,0\rangle} \quad \dfrac{\langle i,j,\rightarrow,1\rangle \quad \langle j+1,k,\rightarrow,1\rangle}{\langle i,k,\rightarrow,0\rangle}$
- Complete cases: $\dfrac{\langle i,j,\leftarrow,1\rangle \quad \langle j,k,\leftarrow,0\rangle}{\langle i,k,\leftarrow,1\rangle} \quad \dfrac{\langle i,j,\rightarrow,0\rangle \quad \langle j,k,\rightarrow,1\rangle}{\langle i,k,\rightarrow,1\rangle}$

where $\langle i,j,d,c\rangle$ denotes the span $[i,j]$, the direction $d \in \{\leftarrow,\rightarrow\}$, and the completeness $c \in \{0,1\}$. Because the Eisner's algorithms considers the dependency relations between words in boarders, it runs in cubic time. By maintaining back pointers, $\langle 0,n,\rightarrow,1\rangle$ yields a complete dependency tree for a length-n sentence, where w_0 is reserved for the dummy root. Algorithm 1 shows a pseudo code of the Eisner's algorithm, which runs in a cubic time to the sentence length N.

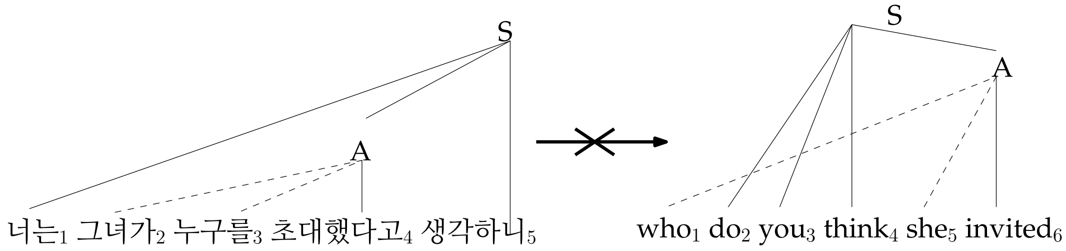

S
A
너는1 그녀가2 누구를3 초대했다고4 생각하니5
S
A
who1 do2 you3 think4 she5 invited6

Figure 5.3: An example of impossible tree transformation using dependency grammar

With beam search, both parsers produce k-best parse trees. k-best parse trees are not only useful for training a discriminative model, but also for achieving higher accuracy, e.g. by re-ranking. The Cube Pruning/Growing algorithms are an efficient dynamic programming technique to produce k-best parse trees [Huang 2005]. The basic idea is to minimize unnecessary enumerations over the search space. Specifically, both algorithms maintain (hyper) cubes during search. A cube consists of a list of best hypotheses and a priority queue for candidates. If the k-th hypothesis is not explored yet, candidates in the priority queue supply hypotheses to the list and additional candidates are generated from the supplied hypotheses. The k-th best hypothesis is produced on-the fly in the Cube Growing algorithm, while every cube produces k-best hypothesis in the Cube Pruning algorithm. Therefore, the Cube Growing algorithm is more efficient than the Cube Pruning algorithm. Figure 5.2 illustrates a intermediate stage of Cube Growing.

5.2 Reordering Tree

We propose a hierarchical structure specialized for word reordering, called a **reordering tree**. Monolingual parsing aims to produce the (hierarchical) syntactic structure from a (flat) word sequence. The syntactic structure is usually represented in a tree form according to a linguistically-motivated grammar. The tree transformation of the syntactic structure, however, is imperfect to deal with word reordering phenomena. Figure 5.3 shows an example of *impossible* tree transformation using dependency grammar from Korean to English, where English has non-projective dependency (dashed lines).

Let v be a node in a tree, and $v.\mathsf{ll}$ and $v.\mathsf{rr}$ be the leftmost/rightmost words covered by v, respectively. Let $\mathit{span}(v)$ be the range of the word sequence $[v.\mathsf{ll}, v.\mathsf{rr}]$. Let v and v' be *disjoint* if v is neither an ancestor nor descendant of v', and $disjoint(v)$ be the set of nodes that are disjoint with v. Without loss of generality, a tree is *projective* if the span of a node does not have any overlapping span with disjoint nodes, i.e. $\forall v' \in disjoint(v), \mathit{span}(v) \cap \mathit{span}(v') = \emptyset$. According to this definition, word reordering under the ITG constraints is suitable for transforming a projective tree. However, non-ITG word reordering is generally impossible by transforming a projective tree.

To overcome such limitations of tree transformation, we do not constrain a reordering tree to be projective. For simplicity, we assume that a source sentence can be reordered as a target-like order. A reordering tree represents a reordered sentence, i.e., the permutation of a source sentence. For an original sentence $w_1 \ldots w_N$, let π be the permutation $w'_1 \ldots w'_N$.

Formal definition of a reordering tree is as follows. A reordering tree G consists of nodes V and hyper-edges E. A node $v \in V$ consists of ω that covers a set of words in the original sentence, i.e. $\{w_i | 1 \leq i \leq N\}$, and its label χ. For a non-terminal node, the label χ indicates the reordering orientation $\chi \in \{S, I\}$ of the children, where S and I are straight and inverted analogous to ITG constraints, respectively. For a terminal node, the label χ is denoted as $-$. An hyper-edge e consists of a parent node $\mathit{head}(e)$ and its ordered children $\mathit{tails}(e)$. Let v_0 be the root node of the reordering tree, and $\mathcal{Y}(v)$ be the (partial) permutation under v. $\mathcal{Y}(v_0)$, or in

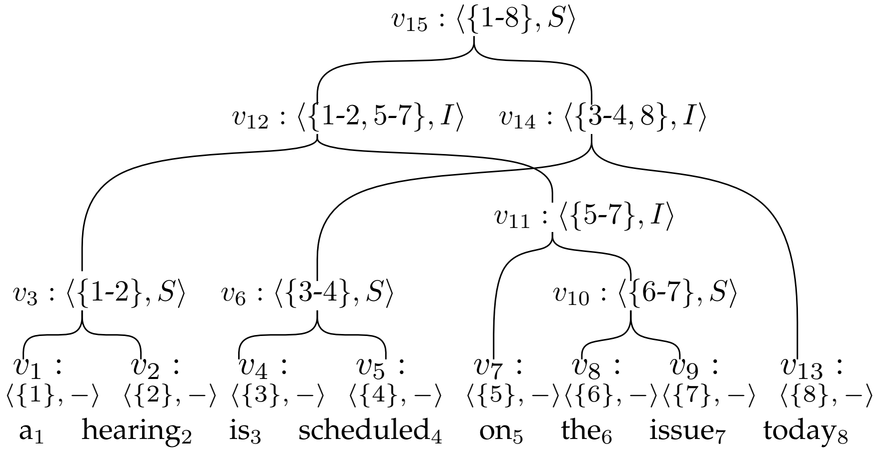

$v_{15} : \langle\{1\text{-}8\}, S\rangle$
$v_{12} : \langle\{1\text{-}2, 5\text{-}7\}, I\rangle$
$v_{14} : \langle\{3\text{-}4, 8\}, I\rangle$
$v_{11} : \langle\{5\text{-}7\}, I\rangle$
$v_3 : \langle\{1\text{-}2\}, S\rangle$
$v_6 : \langle\{3\text{-}4\}, S\rangle$
$v_{10} : \langle\{6\text{-}7\}, S\rangle$
$v_1 : \langle\{1\}, -\rangle$
$v_2 : \langle\{2\}, -\rangle$
$v_4 : \langle\{3\}, -\rangle$
$v_5 : \langle\{4\}, -\rangle$
$v_7 : \langle\{5\}, -\rangle$
$v_8 : \langle\{6\}, -\rangle$
$v_9 : \langle\{7\}, -\rangle$
$v_{13} : \langle\{8\}, -\rangle$
a$_1$ hearing$_2$ is$_3$ scheduled$_4$ on$_5$ the$_6$ issue$_7$ today$_8$

Figure 5.4: A reordering tree for the example sentence. *i-j* abbreviates a continuous index from *i* to *j* (both inclusive). A horizontal line under a node denotes the *inverted* orientation (otherwise *straight*).

short $\mathcal{Y}$, is therefore a permutation π of the original sentence.

A reordering tree is different from a constituency tree or dependency tree in two aspects. First, the tree represents not the syntactic information, but the reordering information. Second, a node v does not necessarily covers a linguistic constituent. Figure 5.4 shows a reordering tree for the example sentence of Figure 3.2. The permutation under the root node corresponds to the reordered sentence "*the issue on a hearing today is scheduled*". A reordering tree may not be projective. For example, v_{12} in Figure 5.4 covers the span $[1, 7]$, which overlaps the span $[3, 8]$ of v_{14}. As non-projectivity is allowed, a reordering tree breaks down Inside-Out word alignment. Hence, it covers broader range of word reordering phenomena during translation.

In this paper, we utilize a reordering tree under a pre-processing framework for SMT. Similar to a monolingual parser, a reordering parser first produces a reordering tree for a given source sentence. Then, the permutation given by the reordering tree,

ALGORITHM 2: Online learning for a training instance

```
procedure UpdateWeight(F, A, w)
    $\mathcal{D} \leftarrow$ Parse(F, w)
    $\dot{D} \leftarrow \arg\max_{D \in \mathcal{D}} Score(D|F, \mathbf{w}) + L(D|F, A)$
    $\hat{D} \leftarrow \arg\min_{D \in \mathcal{D}} L(D|F, A) - \alpha Score(D|F, \mathbf{w})$
    if $L(\hat{D}|F, A) \neq L(\dot{D}|F, A)$ then
        $\mathbf{w} \leftarrow \beta(\mathbf{w} + \gamma(\phi(\hat{D}, F) - \phi(\dot{D}, F)))$
    end
```

i.e. a reordered sentence, is translated into the target language using phrase-based SMT. Then, two research questions remain: how to obtain the reordering (parse) tree and how to train the parser for the reordering tree.

5.3 Parsing under the ITG constraints

We briefly summarize the discriminative reordering parser under the ITG constrains in this section. The most relevant work of this paper was proposed to induce a tree for word reordering produced by a discriminative parser [Neubig 2012]. The goal of their method is to find the best permutation $\hat{\pi}$ for a given source sentence F, according to the following discriminative model.

$$\begin{aligned} \hat{\pi} &= \arg\max_{\pi} Score(\pi|F) \\ Score(\pi|F) &= Score(D|F, \mathbf{w}) \\ &= \sum_i w_i \phi_i(D, F) \end{aligned} \tag{5.6}$$

where D is a reordering tree for word reordering which yields π, and w_i and ϕ_i are the ith feature weight and function, respectively.

To learn the weight vector $\mathbf{w}$, the loss-driven large-margin training is used[Crammer 2006] by finding $\dot{D}$ with the highest model score (Eq. 5.6) and $\hat{D}$ with the smallest loss. A loss function $L(D|F, A)$ is defined by word alignment A, where [Neubig 2012] suggested two kinds of loss functions. Finally, the weight is updated using the difference between the model parse $\dot{D}$ and the oracle parse $\hat{D}$. It is computationally intractable to search over all possible permutations for π. Hence, $\dot{D}$ and $\hat{D}$ are selected among K-best parses encoded in a hyper graph $\mathcal{D}$. To break the tie, $Score(D|F, \mathbf{w})$ and $L(D|F, A)$ are mutually augmented when selecting $\dot{D}$ and $\hat{D}$. The online learning for a training instance $\langle F, A \rangle$ with the current weight vector $\mathbf{w}$ is shown in Algorithm 2 (taken from [Neubig 2012]).

5.3.1 Scalable training method

We illustrate three methods to scale up the online learning method which iterates several epochs over the training instances. First, we adopted a faster search algorithm known as Cube Growing, and integrated it into a parallel CYK parsing method. Second, the feature generation process run in parallel because it is a major bottleneck of parsing efficiency. Third, the features generated at the first iteration are stored on disk and used in the remaining epochs. In a consequence, our proposed methods enable us to utilize tens of thousands training instances in our experiments.

Cube Growing in parallel CYK parsing

Cube Growing is a dynamic programming algorithm for searching over a hyper graph, proposed by [Huang 2005]. It produces the kth-best parse on-the-fly, and thus does not enumerate unnecessary hypotheses during the search process. More specifically, two data structures manage the hypotheses: a list of best hypotheses and a priority queue of candidates for the next best hypothesis. If the kth-best parse is already produced, it is the kth hypothesis in the best list. Otherwise, Cube Growing enumerates hypotheses by taking the best candidate from the priority queue

ALGORITHM 3: Cube Growing in parallel CYK parsing

```
procedure Parse
input  : A sentence $w_1 \dots w_N$
output: A hyper graph with $K$-best parses
  for $L \in [1, N]$ do
      for $l \in [0, N - L]$                                   // in parallel
      do
          $r \leftarrow l + L$
          ModifiedCubeGrowing($l$, $r$)
      end
      wait for terminating the cell-level parallelization
  end
  root ← The root cell
  for $k \in [1, K]$ do
      LazyKthBest(root.$q$, $k$)
  end

procedure ModifiedCubeGrowing
input  : A cell covering $[l, r]$
output: A priority queue $q$ with candidates
  for $m \in (l, r]$ do
      left ← cell [l,m]
      right ← cell [m,r]
      L ← peek(left.q)
      R ← peek(right.q)
      push( q, Hyp(L, R))                                        // straight
      push( q, Hyp(R, L))                                        // inverted
  end
```

ALGORITHM 4: Lazy production in parallel CYK parsing

```
procedure LazyKthBest
input  : A priority queue q and the demanded $k$
output: The $k$th hypothesis in b
  b ← the list of best hypothesis
  while size(b) $< k+1$ and size(q) $> 0$ do
     best ← pop(q)
     LazyNext(q, best)
     push(b, best)
  end

procedure LazyNext
input  : A priority queue q and the hypothesis best
output: An extended priority queue q'
  /* best.L and best.R are the left and right children of best,
     respectively                                              */
  L ← LazyKthBest(left, rank(best.L)+1)
  if L exists then
     push(q, Hyp(L, best.R))                          // straight
     push(q, Hyp(best.R, L))                          // inverted
  end
  R ← LazyKthBest(right, rank(best.R)+1)
  if R exists then
     push(q, Hyp(best.L, R))                          // straight
     push(q, Hyp(R, best.L))                          // inverted
  end
```

until the kth hypothesis can be found. Whenever the best candidate is taken from the priority queue, successors of the candidate are pushed on the priority queue, if possible. To obtain the successors, Cube Growing is recursively performed.

[Dunlop 2011] proposed that the original CYK parsing algorithm can be parallelized in three levels: sentence-level, cell-level, and grammar-level. Although they reported the grammar-level parallelization achieved the fastest result using thousands of GPUs, we adopted the cell-level parallelization. It is possible to parallelize the original CYK parsing at cell-level because the hypotheses in different chart cells covering same number of words in the sentence do not affect each other. Unfortunately, this property does not hold anymore in Cube Growing because k-best hypotheses are enumerated on demand. Therefore, a race condition arises if we directly apply Cube Growing in the cell-level parallelization.

We modified Cube Growing to fit in the cell-level parallelization. To avoid the race condition, the modified Cube Growing directly accesses to the priority queue for the first best hypothesis. It is postponed to move the first best hypothesis to the best list until the second best hypothesis is requested. From the second best parses, the modified Cube Growing does not run in parallel, which is identical to the original one. Algorithm 3 and 4 show the entire procedures for the cell-level parallelization with the modified Cube Growing.

Parallel feature generation

The feature function ϕ in the discriminative model (Eq. 5.6) is further decomposed into the edge level in a reordering tree.

$$\begin{aligned} \phi_i(D, F) &= \sum_{d \in D} \phi_i(d, F) && (5.7) \\ Score(D|F, \mathbf{w}) &= \sum_{d \in D} \sum_i w_i \phi_i(d, F) && (5.8) \end{aligned}$$

where d is a hyper edge in the hyper graph. Because most of feature functions $\phi_i(d, F)$ involve string operations, the feature generation becomes a major bottleneck of parsing efficiency. In a pilot study of our experiments, the feature generation is the most time-consuming process, which takes over 80% of the total parsing time.

Our proposed method parallelizes the feature generation in advance to produce a reordering tree. For a length-N sentence, there are $N(N-1)/2$ hyper edges in the hyper graph $\mathcal{D}$. For each hyper edge, there are two possible orientations *straight* and *inverted*. Hence, the feature generation is performed $N(N-1)$ times in total.

With careful design of the feature function, the feature generation can be parallelized: If the feature function is defined only in a single level of the tree, in other words, a feature set generated from the feature function for a hyper edge is independent from that for the other edge. Therefore, two feature sets for two orientations are stored for each hyper edge, and thus $N(N-1)$ feature sets in the hyper graph in total.

On disk feature

For each iteration, the feature sets generated by the feature function are identical, and the feature weights are only updated. To avoid redundant feature generation processes, reusing the generated features help the later iteration speed up. As the number of generated features is usually huge, however, it might be impossible store them in memory.

Our proposed method writes the features on disk after the generation instead of

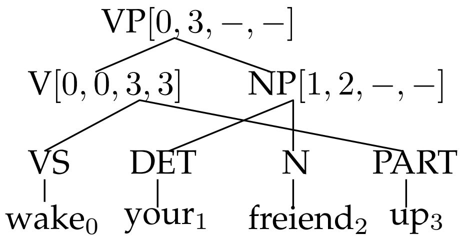


Figure 5.5: An example of discontinuous constituency tree

keeping them in memory. We simply create a file for each sentence with a identification of the sentence in the file name. At the actual parsing time, the generated features are recovered from the file for each sentence. Then the parser begins to search the best permutation π using the features according to the discriminative model. From the second iteration, in other words, we skip the feature generation process and reuse the generated features at the first time.

5.4 Parsing beyond the ITG constraints

In this section, we propose two parsers to produce a reordering tree beyond the ITG constraints by adopting parsing schemes for syntactic trees. Out of previous parsing works, discontinuous constituency and non-projective dependency parsing are selected because they do not restrict a tree to be projective. Specifically, a bottom-up parsing algorithm and a shift-reduce algorithm are adopted. According to the parsing algorithm, efficiency and search quality can be trade-off. A bottom-up parsing we used is a CYK-style algorithm, which runs slower than a shift-reduce algorithm but has more potential to find better solution. Nevertheless, our shift-reduce algorithm is more useful in practice, and shows promising results together with beam-search and dynamic programming techniques. For both parsers, a discriminative model is trained using feature templates, which will be explained.

5.4.1 Bottom-up reordering parser

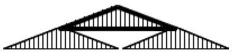
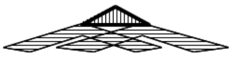
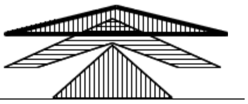
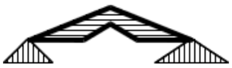
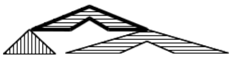
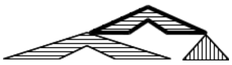
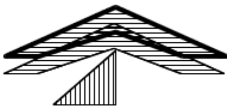
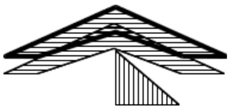
Decoding algorithm

A bottom-up for discontinuous constituency as shown in Figure 5.5 was proposed by [Waxmonsky 2006]. It also allows for non-projective in a constituency tree using $2C$-dimensional representation of items in the agenda, where C is the maximum number of continuous spans for each item and there are at most $C-1$ gaps. An item is a $2C$-dimensional vector $[b_1, e_1, \ldots, b_C, e_C]$, where b_i and e_i are the beginning and end of a continuous span, respectively. For some $1 < k \leq C$, $\forall i$ s.t. $k \leq i$, b_i and e_i can be *don't care*, denoted as $-$. In this manner, the $2C$-dimensional vector can represent continuous constituency as well as discontinuous one. Because $C = 1$ leads to a constituency parser without non-projectivity, their method is a general case of projective constituency parsing. Unfortunately, the time and space complexities are impractically high, respectively, $O(N^{3C})$ and $O(N^{2C})$, where N is the number of words.

Our proposed method adopts the bottom-up parser to allow non-projectivity in a reordering tree. For practical purposes, we restrict the maximum number of gaps to one, i.e., $C = 2$, and the maximum number of words in a gap to D. We also consider eight cases of combining two antecedents into a single consequence, as shown in Table 5.2. The restriction reduces the time complexity to $O(DN^4)$ via a CYK-style algorithm (Algorithm 5). Note that the eight cases of combination can be grouped by the type of the consequence. The results of (1), (2), and (3) are continuous spans, whereas those of (4), (5), (6), (7), and (8) are discontinuous ones.
[2]

[2] Each case is denoted as a comment in Algorithm 5.

Table 5.2: Compositions for a bottom-up parser

Type	ID	Composition	Graphically
Projective	(1)	$\dfrac{[l,i,-,-] \quad [i+1,r,-,-]}{[l,r,-,-]}$	
	(2)	$\dfrac{[l,m,n,i] \quad [m+1,n-1,i+1,r]}{[l,r,-,-]}$	
	(3)	$\dfrac{[l,m,n,r] \quad [m+1,n-1,-,-]}{[l,r,-,-]}$	
Non-projective	(4)	$\dfrac{[l,m,-,-] \quad [n,r,-,-]}{[l,m,n,r]}$	
	(5)	$\dfrac{[l,i,-,-] \quad [i+1,m,n,r]}{[l,m,n,r]}$	
	(6)	$\dfrac{[l,m,n,i] \quad [i+1,r,-,-]}{[l,m,n,r]}$	
	(7)	$\dfrac{[l,m,i,r] \quad [n,i-1,-,-]}{[l,m,n,r]}$	
	(8)	$\dfrac{[l,i,n,r] \quad [i+1,m,-,-]}{[l,m,n,r]}$	

ALGORITHM 5: CYK-style parsing algorithm for reordering tree

```
input  : A sentence $w_1 \ldots w_N$
output: A reordering tree $\mathcal{G}$
for $L \in [1, N]$ do
    for $l \in [0, n - L)$ do
        $r \leftarrow l + L$
        for $i \in [l, r)$ do
            $[l, r, -, -] = [l, i, -, -] \otimes [i + 1, r, -, -]$                      // (1)
            for $d \in [1, D]$ do
                for $n \in [m + 1 + d, r)$ do
                    $[l, r, -, -] = [l, m, n, i] \otimes [m + 1, n - 1, i + 1, r]$      // (2)
                end
                $[l, r, -, -] = [l, m, m + 1 + d, r] \otimes [m + 1, m + d, -, -]$      // (3)
                $[l, m, m + 1 + d, r] = [l, m, -, -] \otimes [m + 1 + d, r, -, -]$      // (4)
                for $n \in [m + 1 + d, r)$ do
                    $[l, m, n, r] = [l, i, -, -] \otimes [i + 1, m, n, r]$              // (5)
                    $[l, m, n, r] = [l, m, n, i] \otimes [i + 1, r, -, -]$              // (6)
                    $[l, m, n, r] = [l, m, i, r] \otimes [n, i - 1, -, -]$              // (7)
                    $[l, m, n, r] = [l, i, n, r] \otimes [i + 1, m, -, -]$              // (8)
                end
            end
        end
    end
end
```

Training discriminative model

In training phase, a classifier is trained to discriminate one composition from another. The oracle labels for the classifier are inferred from the pseudo tree bank. In the bottom-up reordering parser, the orientation of two children is the oracle label. When two antecedents are i and j, the oracle label is defined as follows:

- Straight if i and j are adjacent and straight
- Inverted if i and j are adjacent and inverted
- Undefined otherwise

Therefore, if the oracle label is undefined, it is excluded from the set of the training instances. In addition, we utilize alignment templates [Och 2004] found in a word-alignment sentence pair. A source part of alignment templates is regarded as a terminal symbol. A method to extract an alignment template in linear time [Zhang 2008] minimizes the time consumption on this step. We also utilize k-best reordering trees to generate the training data if possible.

Feature template

For each label, the classifier is trained with a set of features. In the CYK-style algorithm, we know the left and right children v_l and v_r of each binary edge, and simply extract features from them. For the boundary/context features, we denote b and e as the beginning and end indexes, respectively. The feature template for a classifier is summarized in Table 5.3. A notable feature is the reordered n-gram, which is an n-gram language model built from reordered source sentences. The reordered source sentences are obtained using a parallel corpus after identifying the word alignment. We also use the translation boundary feature proposed by [Xiong 2010a]. $IsBegin(w_i)/IsEnd(w_i)$ are binary features indicating that the word

Table 5.3: Features used for classifier. b and e is the beginning and end indexes of a node v, respectively. v_l and v_r are the left and right children, respectively.

Type	Template
Bag-of-words	$\{w_i \| i \in v\}$
Bag-of-POSs	$\{t_i \| i \in v\}$
Boundary words	w_b, w_{b+1}, w_{e-1}, w_e
Boundary POSs	t_b, t_{b+1}, t_{e-1}, t_e
Context words	w_{b-2}, w_{b-1}, w_{e+1}, w_{e+2}
Context POSs	t_{b-2}, t_{b-1}, t_{e+1}, t_{e+2}
Punctuation	$IsPunct(w_b)$, $IsPunct(w_e)$
Reordered n-gram	$LM'(v_l \| v_r)$, $LM'(v_r \| v_l)$
Translation boundary [Xiong 2010a]	$IsBegin(w_b)$, $IsEnd(w_e)$

w_i is the begin/end of a translation boundary. In our experiments, we use various combinations of features.

5.4.2 Shift-reduce reordering parser

Decoding algorithm

A shift-reduce parser for non-projective dependency proposed by [Nivre 2009] is adopted for our purpose. It is a transition system analogous to the parser for projective dependency, but incorporates an additional action *Swap*.

- *Shift*: $\langle \sigma, w_j|\beta, E\rangle \Rightarrow \langle \sigma|w_j, \beta, E\rangle$
- *Right-Arc*: $\langle \sigma|w_i, w_j|\beta, E\rangle \Rightarrow \langle \sigma, w_i|\beta, E \cup \{(w_i, w_j)\}\rangle$
- *Left-Arc*: $\langle \sigma|w_i, w_j|\beta, E\rangle \Rightarrow \langle \sigma, w_j|\beta, E \cup \{(w_j, w_i)\}\rangle$
- *Swap*: $\langle \sigma|w_i, w_j, \beta, E\rangle \Rightarrow \langle \sigma|w_j, w_i|\beta\rangle$ if $i < j$

Table 5.4: An example of transition-based non-projective dependency parsing [Nivre 2009]. SH denotes Shift, LA Left-Arc, RA Right-Arc, and SW Swap action, respectively

Action	Stack σ	Buffer β	Arc
SH	[a]	[hearing, . . .]	
SH	[a, hearing]	[is, . . .]	
LA	[hearing]	[is, . . .]	(a, hearing)
SH	[hearing, is]	[scheduled, . . .]	
SH	[. . . , is, scheduled]	[on, . . .]	
SH	[. . . , scheduled, on]	[the, . . .]	
SH	[. . . , on, the]	[issue, today]	
SH	[. . . , the, issue]	[today]	
LA	[. . . , on, issue]	[today]	(the, issue)
RA	[. . . , scheduled, on]	[today]	(issue, on)
SW	[. . . , is, on]	[scheduled, . . .]	
SW	[hearing, on]	[is, . . .]	
RA	[hearing]	[is, . . .]	(on, hearing)
SH	[hearing, is]	[scheduled, . . .]	
LA	[is]	[scheduled, . . .]	(hearing, is)
SH	[is, scheduled]	[today]	
SH	[is, scheduled, today]	[]	
RA	[is, scheduled]	[]	(today, scheduled)
RA	[is]	[]	(scheduled, is)

Table 5.5: An example of the transition-based reordering parser proposed in this paper. SH denotes *Shift*, ST *Straight*, IV *Inverted*, and SW *Swap* action, respectively.

Action	Stack σ	Buffer β	Node	Edge
	$[]$	$[w_1, \dots]$		
SH	$[v_1]$	$[w_2, \dots]$	$v_1 = \langle\{\text{a}\}, -\rangle$	
SH	$[v_1, v_2]$	$[w_3, \dots]$	$v_2 = \langle\{\text{hearing}\}, -\rangle$	
ST	$[v_3]$	$[w_3, \dots]$	$v_3 = \langle\{\text{a, hearing}\}, S\rangle$	$\langle v_3, \{v_1, v_2\}\rangle$
SH	$[v_3, v_4]$	$[w_4, \dots]$	$v_4 = \langle\{\text{is}\}, -\rangle$	
SH	$[\dots, v_4, v_5]$	$[w_5, \dots]$	$v_5 = \langle\{\text{scheduled}\}, -\rangle$	
ST	$[v_3, v_6]$	$[w_5, \dots]$	$v_6 = \langle\{\text{is, scheduled}\}, S\rangle$	$\langle v_6, \{v_3, v_4\}\rangle$
SH	$[\dots, v_6, v_7]$	$[w_6, \dots]$	$v_7 = \langle\{\text{on}\}, -\rangle$	
SH	$[\dots, v_7, v_8]$	$[w_7, \dots]$	$v_8 = \langle\{\text{the}\}, -\rangle$	
SH	$[\dots, v_8, v_9]$	$[w_8, \dots]$	$v_9 = \langle\{\text{issue}\}, -\rangle$	
ST	$[\dots, v_7, v_{10}]$	$[w_8, \dots]$	$v_{10} = \langle\{\text{the, issue}\}, S\rangle$	$\langle v_{10}, \{v_8, v_9\}\rangle$
IV	$[\dots, v_6, v_{11}]$	$[w_8, \dots]$	$v_{11} = \langle\{\text{on}, \dots, \text{issue}\}, I\rangle$	$\langle v_{11}, \{v_7, v_{10}\}\rangle$
SW	$[v_3, v_{11}]$	$[v_6, \dots]$		
IV	$[v_{12}]$	$[v_6, \dots]$	$v_{12} = \langle\{\text{on}, \dots, \text{hearing}\}, I\rangle$	$\langle v_{12}, \{v_3, v_{11}\}\rangle$
SH	$[v_{12}, v_6]$	$[w_8, \dots]$		
SH	$[\dots, v_6, v_{13}]$	$[]$	$v_{13} = \langle\{\text{today}\}, -\rangle$	
IV	$[v_{12}, v_{14}]$	$[]$	$v_{14} = \langle\{\text{today}, \dots, \text{scheduled}\}, I\rangle$	$\langle v_{14}, \{v_6, v_{13}\}\rangle$
ST	$[v_{15}]$	$[]$	$v_{15} = \langle\{\text{the}, \dots, \text{scheduled}\}, S\rangle$	$\langle v_{15}, \{v_{12}, v_{14}\}\rangle$

The swap action makes it possible to produce the non-projective dependency tree. It dynamically reorders the source sentence during parsing. Table 5.4 illustrates an example of transition-based non-projective parsing. Two swap actions pull out *is* and *scheduled* from σ and push *on*, resulting in dynamic reordering during parsing. Although the worst time complexity is quadratic to the number of words in a sentence, it is expected linear because of the few number of swap actions as non-projective dependencies are rare.

Our proposed method is also a transition system using four actions. Instead of the dependency relations, however, it manages the reordering information. Let ω_i be the word set covered by a node v_i. For a parsing configuration $\mathcal{C} = \langle \sigma, \beta, V, E \rangle$, four actions are defined as follows:

- *Shift*: $\langle \sigma, w_i|\beta, V, E \rangle \Rightarrow \langle \sigma|v_i, \beta, V \cup \{v_i\}, E \rangle$, where $v_i = \langle \{w_i\}, - \rangle$ [3]
- *Straight*: $\langle \sigma|v_i|v_j, \beta, V, E \rangle \Rightarrow \langle \sigma|v_k, \beta, V \cup \{\langle \omega_i \cup \omega_j, S \rangle\}\rangle, E \cup \{\langle w_k, \{v_i, v_j\}\rangle\}\}\rangle$
- *Inverted*: $\langle \sigma|v_i|v_j, \beta, V, E \rangle \Rightarrow \langle \sigma|v_k, \beta, V \cup \{\langle \omega_i \cup \omega_j, I \rangle\}\rangle, E \cup \{\langle w_k, \{v_i, v_j\}\rangle\}\}\rangle$
- *Swap*: $\langle \sigma|v_i|v_j, \beta, V, E \rangle \Rightarrow \langle \sigma|v_j, v_i|\beta, V, E \rangle$ if $i < j$

Analogous to the non-projective dependency parsing, the swap action relax the projective constraint in the structure. Table 5.5 illustrates an example of our proposed method. A reduce action (straight or inverted) merges two elements with an orientation. Therefore, we obtain a reordering tree after the transition system ends. Similar to non-projective parsing, the worst time complexity is quadratic, but it is expected linear to the length of the sentence.

For k-best parsing, our proposed method utilizes beam search with dynamic programming proposed by [Huang 2010]. A parsing configuration is a dynamic programming state, which merges with *equivalent* states. Intuitively, two states are equivalent if they make no difference in further search results. For dependency parsing, [Huang 2010] defined the equivalence of two states using *kernel features* which are "the smallest set of atomic templates". By merging equivalent states, dynamic programming explores exponentially large space in a polynomial time and space. Together with beam search, k-best parsing still runs in a linear time.

As a deductive system, Table 5.6 illustrates our proposed method. Note that it manages a stack for swapping states γ, and the *Shift* action pushes the state which is

[3] If a node is swapped out and then shifted again, *Shift*: $\langle \sigma, v_i|\beta, V, E \rangle \Rightarrow \langle \sigma|v_i, \beta, V, E \rangle$

Table 5.6: A deductive system for reordering parser beyond the ITG constraints. A state consists of the step i, covered source span l, r, partial parsing results σ, swapped states γ, prefix and inside costs c and v, and predictor states φ. n is the number of words in a sentence, and m the maximum number of swap actions taken, respectively. The costs of the *Shift* action is ξ, *Straight* δ, *Inverted* λ, and *Swap* ρ, respectively, where $\xi_p = \mathbf{w} \cdot \Phi_{SH}(p), \delta_p^q = \mathbf{w} \cdot \Phi_{SH}(q) + \mathbf{w} \cdot \Phi_{ST}(p), \lambda_p^q = \mathbf{w} \cdot \Phi_{SH}(q) + \mathbf{w} \cdot \Phi_{IV}(p)$, and $\rho_p = \mathbf{w} \cdot \Phi_{SW}(p)$. The *Idle* action (*ID*) completes the deductions by increasing the step count.

State form $\quad i : \langle l,\ r,\ j,\ \sigma,\ \gamma \rangle : (c,\ v,\ \varphi)$

Equivalence $\quad i : \langle l,\ r,\ j,\ \sigma,\ \gamma \rangle \quad i : \langle l,\ r,\ j,\ \sigma',\ \gamma' \rangle$ if $\tilde{f}(j, \sigma) = \tilde{f}(j, \sigma')$

Ordering $\quad i : \_ : (c,\ v,\ \_) \prec i : \_ : (c',\ v',\ \_)$ if $c < c' \vee (c = c \wedge v < v')$

p_0

$$0 : \langle 0\ , 0,\ 0,\ \epsilon,\ \epsilon \rangle : (0,\ 0,\ \emptyset)$$

SH

$$\frac{\overbrace{i : \langle \_,\ \_,\ j,\ \sigma,\ \epsilon \rangle : (c,\ \_,\ \_)}^{\text{state } p}}{i+1 : \langle j,\ j+1,\ j+1,\ \sigma | \langle \{w_j\}, - \rangle,\ \epsilon \rangle : (c + \xi_p,\ 0,\ \{p\})}\ j < n$$

SH

$$\frac{\overbrace{\_ : \langle l,\ r,\ \_,\ \sigma' | v_0,\ \_ \rangle : (\_,\ v',\ \_)}^{\text{state } q} \quad \overbrace{i : \langle \_,\ \_,\ j,\ \sigma,\ \gamma | v_0 \rangle : (c,\ \_,\ \_)}^{\text{state } p}}{i+1 : \langle l,\ r,\ j,\ \sigma | v_0, \gamma \rangle : (c + v' + \xi_q,\ v',\ \{p\})} \gamma \neq \epsilon$$

ST

$$\frac{\overbrace{\_ : \langle l,\ \_,\ \_,\ \sigma' | v'_0,\ \_ \rangle : (c',\ v',\ \varphi')}^{\text{state } q} \quad \overbrace{i : \langle \_,\ r,\ j,\ \sigma | v_0,\ \gamma \rangle : (\_,\ v,\ \varphi)}^{\text{state } p}}{i+1 : \langle l,\ r,\ j,\ \sigma' | \langle \omega_0 \cup \omega'_0,\ S \rangle,\ \gamma \rangle : (c' + v + \delta_p^q,\ v' + v + \delta_p^q,\ \varphi')}\ q \in \varphi$$

IV

$$\frac{\overbrace{\_ : \langle l,\ \_,\ \_,\ \sigma' | v'_0,\ \_ \rangle : (c',\ v',\ \varphi')}^{\text{state } q} \quad \overbrace{i : \langle \_,\ r,\ k,\ \sigma | v_0,\ \gamma \rangle : (\_,\ v,\ \varphi)}^{\text{state } p}}{i+1 : \langle l,\ r,\ k,\ \sigma' | \langle \omega_0 \cup \omega'_0,\ I \rangle,\ \gamma \rangle : (c' + v + \lambda_p^q,\ v' + v + \lambda_p^q,\ \varphi')}\ q \in \varphi$$

SW

$$\frac{\overbrace{\_ : \langle \_,\ \_,\ \_,\ \_,\ \_ \rangle : (\_,\ \_,\ \varphi')}^{\text{state } q} \quad \overbrace{i : \langle l,\ r,\ j,\ \sigma | v_1 | v_0,\ \gamma \rangle : (c,\ v,\ \varphi)}^{\text{state } p}}{i+1 : \langle l,\ r,\ j,\ \sigma | v_0,\ \gamma | v_1 \rangle : (c + \rho_p,\ v + \rho_p,\ \varphi')}\ q \in \varphi$$

ID

$$\frac{i : \langle 0,\ n,\ n,\ \sigma,\ \gamma \rangle : (c,\ v,\ \varphi)}{i+1 : \langle 0,\ n,\ n,\ \sigma,\ \gamma \rangle : (c,\ v,\ \varphi)}$$

goal

$$2(n+m) - 1 : \langle 0\ , n,\ n,\ \sigma,\ \epsilon \rangle : (c,\ \_,\ \{p_0\})$$

Table 5.7: An example derivation using the deduction system. The nodes are same as Table 5.5. The scores are as follows: $\xi_i = \mathbf{w} \cdot \Phi_{SH}(p_i); \delta_i^j = \mathbf{w} \cdot \Phi_{SH}(q_j) + \mathbf{w} \cdot \Phi_{ST}(p_i); \lambda_i^j = \mathbf{w} \cdot \Phi_{SH}(q_j) + \mathbf{w} \cdot \Phi_{IV}(p_i);$ and $\rho_i = \mathbf{w} \cdot \Phi_{SW}(p_i)$.

p_0	$0 : \langle 0, 0, 0, \epsilon, \epsilon \rangle : (0, 0, \emptyset)$
p_1	$1 : \langle 0, 1, 1, v_1, \epsilon \rangle : (\xi_0, 0, \{p_0\})$
p_2	$2 : \langle 1, 2, 2, v_1 \vert v_2, \epsilon \rangle : (\xi_0 + \xi_1, 0, \{p_1\})$
p_3	$3 : \langle 0, 2, 2, v_3, \epsilon \rangle : (\xi_0 + \delta_2^1, \delta_2^1, \{p_0\})$
p_4	$4 : \langle 2, 3, 3, v_3 \vert v_4, \epsilon \rangle : (\xi_0 + \delta_2^1 + \xi_3, 0, \{p_3\})$
p_5	$5 : \langle 3, 4, 4, v_3 \vert v_4 \vert v_5, \epsilon \rangle : (\xi_0 + \delta_2^1 + \xi_3 + \xi_4, 0, \{p_4\})$
p_6	$6 : \langle 2, 4, 4, v_3 \vert v_6, \epsilon \rangle : (\xi_0 + \delta_2^1 + \xi_3 + \delta_5^4, \delta_5^4, \{p_3\})$
p_7	$7 : \langle 4, 5, 5, v_3 \vert v_6 \vert v_7, \epsilon \rangle : (\xi_0 + \delta_2^1 + \xi_3 + \delta_5^4 + \xi_6, 0, \{p_6\})$
p_8	$8 : \langle 5, 6, 6, v_3 \vert v_6 \vert v_7 \vert v_8, \epsilon \rangle : (\xi_0 + \delta_2^1 + \xi_3 + \delta_5^4 + \xi_6 + \xi_7, 0, \{p_7\})$
p_9	$9 : \langle 6, 7, 7, v_3 \vert v_6 \vert v_7 \vert v_8 \vert v_9, \epsilon \rangle : (\xi_0 + \delta_2^1 + \xi_3 + \delta_5^4 + \xi_6 + \xi_7 + \xi_8, 0, \{p_8\})$
p_{10}	$10 : \langle 5, 7, 7, v_3 \vert v_6 \vert v_7 \vert v_{10}, \epsilon \rangle : (\xi_0 + \delta_2^1 + \xi_3 + \delta_5^4 + \xi_6 + \xi_7 + \delta_9^8, \delta_9^8, \{p_7\})$
p_{11}	$11 : \langle 4, 7, 7, v_3 \vert v_6 \vert v_{11}, \epsilon \rangle : (\xi_0 + \delta_2^1 + \xi_3 + \delta_5^4 + \xi_6 + \delta_9^8 + \lambda_{10}^7, \delta_9^8 + \lambda_{10}^7, \{p_6\})$
p_{12}	$12 : \langle 4, 7, 7, v_3 \vert v_{11}, v_6 \rangle : (\xi_0 + \delta_2^1 + \xi_3 + \delta_5^4 + \xi_6 + \delta_9^8 + \lambda_{10}^7 + \rho_{11}, \delta_9^8 + \lambda_{10}^7 + \rho_{11}, \{p_3\})$
p_{13}	$13 : \langle 0, 7, 7, v_{12}, v_6 \rangle : (\xi_0 + \delta_2^1 + \delta_9^8 + \lambda_{10}^7 + \rho_{11} + \lambda_{12}^3, \delta_2^1 + \delta_9^8 + \lambda_{10}^7 + \rho_{11} + \lambda_{12}^3, \{p_0\})$
p_{14}	$14 : \langle 2, 4, 7, v_{12} \vert v_6, \epsilon \rangle : (\xi_0 + \delta_2^1 + \delta_9^8 + \lambda_{10}^7 + \rho_{11} + \lambda_{12}^3 + \xi_2 + \delta_5^4, \delta_5^4, \{p_{13}\})$
p_{15}	$15 : \langle 7, 8, 8, v_{12} \vert v_6 \vert v_{13}, \epsilon \rangle : (\xi_0 + \delta_2^1 + \delta_9^8 + \lambda_{10}^7 + \rho_{11} + \lambda_{12}^3 + \xi_2 + \delta_5^4 + \xi_{14}, 0, \{p_{14}\})$
p_{16}	$16 : \langle 2, 8, 8, v_{12} \vert v_{14}, \epsilon \rangle : (\xi_0 + \delta_2^1 + \delta_5^4 + \delta_9^8 + \lambda_{10}^7 + \rho_{11} + \lambda_{12}^3 + \xi_2 + \delta_5^4 + \lambda_{15}^{14}, \delta_5^4 + \lambda_{15}^{14}, \{p_{13}\})$
p_{17}	$17 : \langle 0, 8, 8, v_{15}, \epsilon \rangle : (\xi_0 + \delta_2^1 + \delta_9^8 + \lambda_{10}^7 + \rho_{11} + \lambda_{12}^3 + \delta_5^4 + \lambda_{15}^{14} + \delta_{16}^{13}, \_, \{p_0\})$

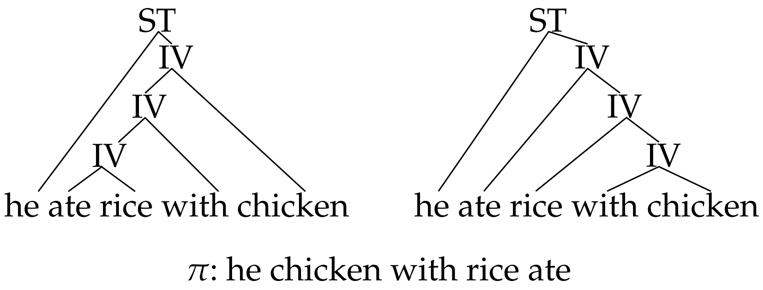


Figure 5.6: Two reordering trees in y-good derivations

swapped out recently. In practice, we restrict the maximum number of swap actions taken during parsing upto m. Table 5.7 shows the deduction for the reordering tree shown in Figure 5.4.

Training discriminative model

Our proposed parser utilizes a discriminative model to classify sub-steps during parsing. Unlike monolingual parsing, however, we do not rely on human-annotated reordering trees. Instead, we utilize a *pseudo* tree bank from a parallel corpus analogous to [DeNero 2011, Neubig 2012]. The word alignment can shed light on the orientation of any two sets of words, left and right children, in a sentence pair. For instance, v_{12} and v_{14} establish a *inverted* orientation, whereas v_3 and v_{11} *inverted* orientation, according to the word alignment in Figure 3.2.

For $v = \langle \omega, \chi \rangle$ and $\pi = \mathcal{Y}(v)$, let $v.\mathsf{tl}$ and $v.\mathsf{tr}$ be the leftmost/rightmost words of the reordered sentence under v, i.e. $v.\mathsf{tl} = \pi_1$ and $v.\mathsf{tr} = \pi_{|\omega|}$. Also, let $\mathbf{r} = r_1 \ldots r_n$ be the ranks of the source words, which are relative positions in the target sentence, defined by word alignment. [4] For example, the ranks of the example sentence in Figure 5.6 is $r_1 = 0, r_2 = 4, r_3 = 3, r_4 = 2$ and $r_5 = 1$. An oracle action for a parsing configuration $\mathcal{C} = \langle \sigma | v_i | v_j, \beta, V, E \rangle$ is defined as follows:

[4] There are various options of defining the ranks, e.g., attaching null aligned word to the nearest left/right aligned word.

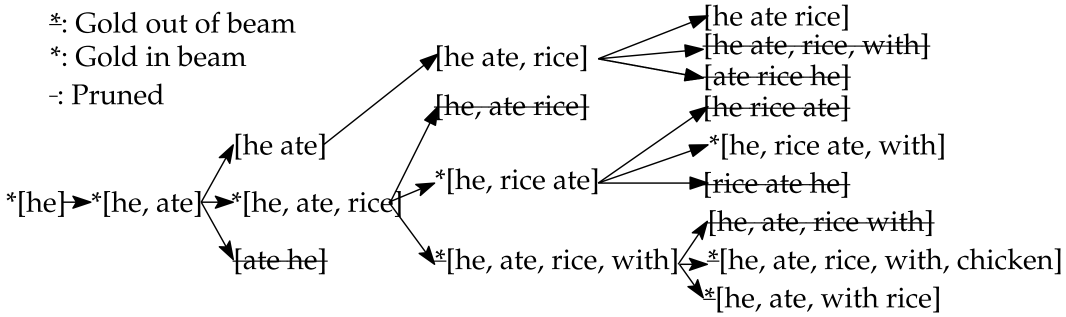


Figure 5.7: Beam search with y-good derivations. Each state is depicted by its stack for brevity.

- *Straight* if $r_{v_i.\mathsf{tr}}$ and $r_{v_j.\mathsf{tl}}$ are adjacent and straight
- *Inverted* if $r_{v_i.\mathsf{tr}}$ and $r_{v_j.\mathsf{tl}}$ are adjacent and inverted
- Swap if $\exists v_{i'} \in \sigma, r_{v_{i'}.\mathsf{tr}}$, and $r_{v_j.\mathsf{tl}}$ are adjacent and $i' < j$
- Shift otherwise

Two learning schemes MaxForce and Pegasos were compared in our experiments. First, MaxForce is analogous to the scheme in [Yu 2013]. As the gold-standard derivation is not unique, we utilize the online learning scheme with latent variables [Yu 2013]. For example, two reordering trees in Figure 5.6 correspond to the correct permutation π. Let y-good derivations be the set of partial derivations whose the action sequence produces a reordering tree corresponding to the correct permutation. Conversely, y-bad derivations do not produce the correct permutations. During beam search, the parser manages not only top k states, but also y-good derivations. Whenever a y-good derivation is produced, it is stored in a separate place. If a y-good derivation falls off at step i, the parser continues to produce the next y-good derivation at step $i + 1$ from the separate place. Figure 5.7 shows an example. Although the y-good derivation "[he, ate, rice, with]" falls off at step

4, y-good derivations at step 5 ("[he, ate, rice, with, chicken]" and "[he, ate, with rice]") are produced.

Second, we adopted a loss-driven large margin training Pegasos utilized in [Neubig 2012]. To learn the weight vector, they used the loss-driven large-margin training [Crammer 2006] by finding a derivation with the highest model score (model parse) and a derivation with the smallest loss (oracle parse). A loss function is defined by word alignment, where [Neubig 2012] suggested two kinds of loss functions. Finally, the weight is updated using the difference between the model parse and the oracle parse. It is computationally intractable to search over all possible derivations. Hence, model and oracle parses are selected among k-best parses during beam search. To break the tie, the model score and loss are mutually augmented when selecting the oracle and model parse, respectively, as shown in Algorithm 2.

Feature template

We develop the feature template for the shift-reduce reordering parser inspired by [Huang 2010, DeNero 2011, Neubig 2012]. Different from dependency parsing, boundary words play key roles in our proposed method: The leftmost/rightmost boundary words (ll/rr, respectively), the rightmost/leftmost boundary words of the left/right child (lr/rl, respectively), and the before/after words of the leftmost/rightmost boundary words (b/a, respectively). In addition to the boundary words, the action type (t) to be taken is utilized. The phrase features used in [Neubig 2012] are also adopted. Table 5.8 illustrates the feature template and features.

Table 5.8: The feature template and features. s_i is the i-th element from the stack top, and q_j is the j-th word/word class from the buffer front.

Template	Description
ll/rr	the leftmost/rightmost boundaries
rl/lr	the rightmost boundary of the left chlid and vice versa
b/a	before/after boundaries
t	action taken t $\in \{shift, straight, inverted, swap\}$
$\phi_{table}(x)$	phrase feature used in [DeNero 2011]
Type	**Feature**
word	s_0.ll $\circ$ t s_0.rr $\circ$ t s_0.lr $\circ$ t s_0.rl $\circ$ t
	s_0.ll $\circ$ s_0.rr $\circ$ t s_0.lr $\circ$ s_0.rl $\circ$ t
	s_1.ll $\circ$ t s_1.rr $\circ$ t s_1.lr $\circ$ t s_1.rl $\circ$ t
	s_1.ll $\circ$ s_1.rr $\circ$ t s_1.lr $\circ$ s_1.rl $\circ$ t
	s_0.b $\circ$ t s_0.a $\circ$ t s_1.b $\circ$ t s_1.a $\circ$ t q_0 $\circ$ t $\phi_{table}(s_0)$
word class	s_0.ll $\circ$ t s_0.rr $\circ$ t s_0.lr $\circ$ t s_0.rl $\circ$ t
	s_0.ll $\circ$ s_0.rr $\circ$ t s_0.lr $\circ$ s_0.rl $\circ$ t
	s_1.ll $\circ$ t s_1.rr $\circ$ t s_1.lr $\circ$ t s_1.rl $\circ$ t
	s_1.ll $\circ$ s_1.rr $\circ$ t s_1.lr $\circ$ s_1.rl $\circ$ t
	s_2.ll $\circ$ t s_2.rr $\circ$ t s_2.lr $\circ$ t s_2.rl $\circ$ t
	s_2.ll $\circ$ s_2.rr $\circ$ t s_2.lr $\circ$ s_2.rl $\circ$ t
	s_0.b $\circ$ t s_0.a $\circ$ t s_1.b $\circ$ t s_1.a $\circ$ t
	s_2.b $\circ$ t s_2.a $\circ$ t q_0 $\circ$ t q_1 $\circ$ t q_2 $\circ$ t
Kernel Features	
s_0.ll s_0.rr s_0.lr s_0.rl s_1.ll s_1.rr s_1.lr s_1.rl s_2.ll s_2.rr	
q_0 q_1 q_2 t	

VI. Experiments

In this chapter, we report experimental results for both bottom-up and shift-reduce reordering parsers. For the bottom-up parser, the experiment was focused on combinations of feature templates, and compared with strong baselines. For the shift-reduce parser, on the other hand, the experiment was focused on effectiveness of our proposed method on various language pairs.

6.1 Experiments for bottom-up parser

6.1.1 Experimental Environment

We used a Korean-English parallel corpus obtained from the web, which consists of 601,896 sentence pairs. We randomly split the corpus into the training (80M/103M)[1], development (10K/14K), and test corpora (120K/155K). The training corpus was used not only to align words and extract the phrase pairs, but also to provide training instances for the classifier of reordering parsers. We utilized only 10% of whole training instances to train the classifier, though, in order to reduce the training time. For comparison, we evaluated a reordering parser using same classifier, which is a greedy version transition-based parser illustrated in 5.4. Giza++ [Och 2003a] and Moses [Koehn 2007] supplied the baseline phrase-based SMT system.

We evaluated our proposed methods in two aspects. First, we measured the accuracy of the reordering parsers. Unlike monolingual parsing, however, a gold-standard tree bank is absent. Even if there exists manually constructed reordering trees, evaluation metrics in monolingual parsing cannot be directly utilized. For

[1] The number of the source and target words, respectively

Table 6.1: The accuracy of the classifier and the reordering parsers

	System	Classifier	Hamming	Kendal's Tau
Greedy	(1)	71.55	8.86	45.45
	(1)+(2)	73.83	9.70	**48.40**
	(1)+(3)	71.62	9.60	47.50
	(1)+(4)	71.49	9.73	47.72
	(1)+(2)+(3)	74.10	**10.93**	41.61
	(1)+(2)+(4)	73.90	9.86	48.19
	(1)+(3)+(4)	71.80	9.26	43.75
	All	74.12	9.12	45.14
Bottom-up	(1)	87.46	27.59	46.19
	(1)+(2)	87.52	28.29	46.76
	(1)+(2)+(4)	87.67	**28.45**	**49.79**

example, there exists spurious ambiguity caused by the fact that many different reordering trees may yield the identical permutation. Therefore, we adopted evaluation metrics originally proposed for translation. After building reordering trees for sentences, the accuracy of word reordering is estimated using the oracle permutation given by word alignment. Second, we inspected the translation quality of the SMT system that adopted the parser-based word reordering as a preprocessing step. An end-to-end machine translation utilized a reordering parser and a phrase-base SMT system. Distortion and lexicalized reordering methods in a phrase-based SMT system are also enabled to capture local reordering that cannot be found in the reordering parser, or to fix incorrect reordering.

The Hamming and Kendal's tau distances are a reordering distance metrics used in evaluating translation quality [Birch 2011]. Both distances measure the similarity between two permutations in terms of the reordering distance. The higher either of the two distance values is, the more similar the two permutations would be. Thus the

the distance value between the yield of a reordering tree and its corresponding oracle permutation is regarded as the accuracy of the reordering parser. The underlying assumption is that "the ordering of the reference sentence must be acceptable" [Birch 2011]. Please refer to the definition and details in their paper.

The translation quality of the machine translation was measured with Bleu [Papineni 2002], Nist [Doddington 2002], Ter [Zaidan 2010] and Ribes [Isozaki 2010]. Pairwise human evaluation was also conducted between the best result according to the automatic evaluation and the result from Moses baseline. Bidirectional word alignments are symmetrized using a GrowDiagFinalAnd heuristic. The parameters for the phrase-based SMT system were tuned using the minimum error rate training [Och 2003b] to maximize Bleu score on the development corpus. Table 6.1 and 6.2 show the accuracy of the reordering parsers and the translation quality when used for pre-ordering, respectively. The pairwise human evaluation between the result using the our proposed method and the baseline is shown in Figure 6.3.

We also assume that there exist no word reordering in a base chunk, e.g., NP chunk. This assumption aims to prevent excessive word reordering between Korean and English. For example, the articles in English seldom match anything in Korean, and the word alignment of the articles tends to be incorrect. If we regard the word alignment as correct, word reordering would be unnecessarily excessive. Therefore, we let a base chunk be minimum unit of reordering instead of a word. We used CrfChunker [Phan 2006] to identify base chunks in English. We also restrict the word reordering with the clause/sentence boundary, while approximating the boundary with punctuations.

We used LibLinear (version 1.92) as a classifier [Fan 2008]. Table 6.1 shows the accuracy of the combination of features. Among all the combinations of feature templates, we investigated only some combinations of the following features:

Table 6.2: The translation quality. † denotes the score is statistically significant at 99% confidence level than Moses baseline.

	System	DevBLEU	Bleu	Nist	Ter	Ribes
Greedy	(1)	19.95	18.46	5.56	72.96	65.47
	(1)+(2)	20.68	19.64	5.56	71.88	66.54
	(1)+(3)	19.30	18.24	5.66	73.57	65.14
	(1)+(4)	20.97	**19.90**	5.69	71.90	**66.62**
	(1)+(2)+(3)	20.27	18.81	5.67	72.66	64.83
	(1)+(2)+(4)	20.55	19.51	5.70	**71.72**	66.59
	(1)+(3)+(4)	21.10	19.74	**5.76**	71.76	65.60
	All	20.99	19.49	5.57	73.34	64.03
Bottom-up	(1)	21.10	20.08†	5.75	70.93	67.91
	(1)+(2)	21.34	20.28†	5.79†	70.82	**68.18**
	(1)+(2)+(4)	21.58	**20.45†**	**5.82†**	**70.61**	67.95
	Moses	21.12	19.58	5.76	71.63	65.69
	Hiero	23.47	19.13	5.42	71.26	67.26
	Lader	21.01	20.10†	5.80†	70.72	67.76
	Lader with gaps	22.86	**21.10†**	**5.91†**	**69.71**	**68.61**

(1) word-based features: bag-of-words, boundary word, context words, and punctuation, (2) POS-based features: bag-of-POSs, boundary POSs, and context POSs, (3) reordered n-gram feature, and (4) translation boundary feature. Note that the reordered n-gram feature is not used for the classifier in the bottom-up reordering parser since it scores the items during beam search. We set the size of gap D to 7 for all experiments.

6.1.2 Discussion

The accuracy of the classifier generally increases as more features are combined. For the greedy reordering parser, (2) improves the accuracy more than (3) and (4). The

Table 6.3: Pairwise human evaluation result

	Bottom-up	Equal	Moses	Total
Bottom-up	13	5	3	21
Equal	5	50	7	62
Moses	1	10	6	17
Total	19	65	16	100

combination of all features achieved a 2.57% improvement over (1). The difference in the accuracy between the two parsers is notable. Even with the minimum set of features (1), the classifier for our parser achieved much higher accuracy. Because two parsers work differently (greedy vs. dynamic programming), the characteristics of training instances are also different. In addition, the bottom-up reordering parser utilized k-best trees if possible. As a result, we believe that the training instances for the bottom-up reordering parser provide more statistically meaningful data.

The accuracy of the reordering parsers also shows that the bottom-up reordering parser is better than the transition-based one. Both distances indicate that the reordered sentence of the bottom-up parser is much more similar to the oracle permutation given by word alignment. The Hamming distance is more sensitive to the position of the reordering, showing greater difference between two parsers. In terms of the Kendal's tau distance, the bottom-up parsers tend to be more accurate than the transition-based parsers. Hence, we expect that the translation quality of the bottom-up parser would be better than that of the transition-based parser.

The translation quality of the bottom-up parser is indeed better than that of the transition-based parser. The difference between the result of the bottom-up parsers (20.45 Bleu) and that of the greedy parser (19.90 BLEU) is statistically significant at a 99% confidence. It is also better than that of Moses baseline (19.58 BLEU). As Ter score reflects more about post-editing effort which is more suitable for checking the reordering quality, Table 6.2 shows that the improvement is indeed achieved from the pre-ordering using the bottom-up reordering parser. Ribes scores, a reordering-oriented evaluation measure, also indicates that our proposed method achieves improvement over the baseline. Pairwise human evaluation also shows that the difference is meaningful.

We conducted further comparison beyond the Moses baseline. Allowing larger distortion in PBSMT achieved higher scores than the baseline and our proposed method. Surprisingly, a hierarchical PBSMT system (Hiero) degraded the translation quality, which suggested global reordering is not handled properly. A discriminative reordering parser under the ITG constraint (Lader), which involves more sophisticated training procedure than ours, improved translation quality as well. Note that it is unfair to directly compare Lader and our proposed method because of the difference of features and the training method. Therefore, we allow discontinuity in Lader as Algorithm 5. The last two rows in Table 6.2 shows that this modification further improves the translation quality of Lader. It almost reaches to the PBSMT with larger distortion, which takes decoding longer time.

6.2 Experiments for transition-based parser

6.2.1 Experimental Environment

We further tested our proposed method using parallel corpora between language pairs which have drastically different word orders: Korean-English (K-E), Korean-Chinese (K-C), and Japanese-English (J-E). Except a K-E corpus which was manually constructed by human translators (Systran), a K-E corpus was collected online dictionary (Naver), a K-C corpus from newswire services (DongA), a J-E corpus from Wikipedia articles (Kftt) [Neubig 2011a]. Table 6.4 shows the statistics of corpora we used.

Table 6.4: The statistics of corpora. Figures are the number of sentence pairs.

		Parser	SMT		
Language Pair	ID	Train	Train	Dev	Test
Korean-English	Systran	9,989	97,085	1,483	483
	Naver		599,673		
Korean-Chinese	DongA	367	85,211	1,832	1,342
Japanese-English	Kftt	1,235	329,981	1,235	1,160

As the training data of our reordering parser, a word alignment corpus is required. We manually built word alignment corpora for K-E and K-C, and utilized the provided word alignment for J-E. Especially for the Systran corpus, the word alignment corpus was massively constructed, in order to provide enough training data for reordering parsing. It took 6 man-months to annotate the word alignment for approximately 9,000 sentence pairs, after first trained annotators using 1,000 sentences.

The accuracy of our reordering parser was measured by the similarity between the permutation obtained from parsing and the correct permutation under the word alignment. Unlike monolingual parsing, the gold-standard tree bank is absent. Even if there exists manually constructed reordering trees, evaluation metrics in monolingual parsing cannot be directly utilized. For example, there exists spurious ambiguity caused by the fact that many different reordering trees may yield the identical permutation as shown in Figure 5.6.

The Kendal's tau (τ) and chunk fragmentation (*chunk*) distances indicate the similarity defined as follows:

- τ: pairwise ordering

$$\tau(\pi, \mathbf{r}) = \frac{2}{n(n+1)} \sum_{i}^{n-1} \sum_{j=i+1}^{n} \delta(r_{\pi_i} > r_{\pi_j})$$

- Chunk fragment: word adjacency

$$chunk(\pi, \mathbf{r}) = \frac{1}{n} \sum_{i}^{n-1} \delta(r_{\pi_i} \nsim r_{\pi_{i+1}}),$$

where the ranks of words in the original sentence $\mathbf{r} = r_1 \dots r_n$ is inferred from the word alignment, $\delta(\cdot)$ is the Kronecker' delta and $\delta(i \nsim j) = 1$ if $i \neq j$ and $i + 1 \neq j$. For comparison, Lader was used for the baseline reordering parser under ITG constraints, which supports both Standard perceptron and Pegasos for online learning. Our proposed method Srdp is implemented on the top of Lader, but MaxForce is supported in addition to Pegasos. For reordering parsing beyond ITG constraints, the maximum number of swap actions (m) was set to one. Note that our proposed method obeys ITG constraints if $m = 0$, and we also compared Srdp under ITG constraints. At training stage, the maximum number of the iteration was set to 100 and the beam size 10, respectively. Whenever an epoch increased the accuracy of reordering parsing, evaluated on a held-out data, the discriminative model was stored. A reordering parser then utilized the last stored discriminative model.

The translation quality of the machine translation was measured with Bleu [Papineni 2002], Ter [Snover 2006] and Ribes [Isozaki 2010]. Bleu provides an n-gram precision with brevity penalty, which has been criticized for neglecting word order difference. Ter reflects more about post-editing effort which is more suitable for checking the reordering quality. Ribes is also a reordering-oriented evaluation measure. For machine translation, Moses was used for the baseline phrase-based SMT system with MGiza [Gao 2008] to obtain automatic word alignment. Moses also provided a syntax-based SMT system (Syntax) for better comparison. Bidirectional word alignments were symmetrized using the GrowDiagFinalAnd heuristic. The parameters for the SMT systems were tuned using the minimum error rate

Table 6.5: The accuracy and speed of the reordering parsers

	English			Korean		
Parser$_m^{\text{Learner}}$	τ	chunk	speed	τ	chunk	speed
$\textsc{Lader}^{\textsc{Standard}}$	79.45	73.71	9.09	79.25	68.44	21.23
$\textsc{Lader}^{\textsc{Pegasos}}$	81.25	77.20	8.76	80.71	72.01	26.16
$\textsc{Srdp}_0^{\textsc{MaxForce}}$	81.03	78.92	0.76	81.73	73.65	1.39
$\textsc{Srdp}_1^{\textsc{MaxForce}}$	82.06	**80.00**	1.04	82.54	74.82	1.37
$\textsc{Srdp}_0^{\textsc{Pegasos}}$	81.81	77.77	0.88	82.38	74.10	1.12
$\textsc{Srdp}_1^{\textsc{Pegasos}}$	**82.20**	78.01	1.11	**83.57**	**75.37**	1.45

training [Och 2003b] to maximize Bleu score on the development corpus.

For each corpus, we conducted experiments on the phrase-based SMT with/without pre-ordering. The evaluation of reordering parsing was two-folds: the accuracy and speed. The result of machine translation was evaluated by automatic metrics described above. For the reasonable comparisons, we ran experiments five times for each setting. Table 6.5 shows the accuracy and speed of the reordering parsers, and Table 6.6 shows the translation quality.

During the construction of the training data for the reordering tree, i.e. word-aligned corpora, we had periodically measured the accuracy of a reordering parser and its performance applied to a pre-ordering SMT. There were five points of measurements consisting of 890, 1690, 4190, 8389, and 9989 sentences, respectively. [2] Figure 6.1 shows the effect of the size of training data.

[2] After the first 890 sentences were annotated by one annotator, rest of sentences were annotated by 6 annotators for a month. At the end of each week, the effect of the size of training data was measured using the annotated corpus until that moment.

Table 6.6: The translation quality

	English→Korean			Korean→English		
	Bleu	Ter	Ribes	Bleu	Ter	Ribes
Moses	21.11	67.98	67.15	17.41	68.62	65.72
Syntax	17.60	69.41	67.59	14.83	72.23	64.65
Lader$^{\text{Standard}}$	22.83	64.51	72.08	18.81	65.32	70.94
Lader$^{\text{Pegasos}}$	23.65	63.51	73.44	20.16	62.23	72.19
Srdp$_0^{\text{MaxForce}}$	22.70	64.91	72.63	18.28	65.31	69.47
Srdp$_1^{\text{MaxForce}}$	22.42	64.10	**73.92**	19.19	63.52	70.74
Srdp$_0^{\text{Pegasos}}$	23.17	63.80	73.81	**20.84**	62.84	71.99
Srdp$_1^{\text{Pegasos}}$	**24.14**	**63.37**	73.76	20.57	**62.08**	**72.62**
	Chinese→Korean			Korean→Chinese		
	Bleu	Ter	Ribes	Bleu	Ter	Ribes
Moses	21.87	67.52	72.95	20.34	66.93	74.64
Syntax	14.26	81.12	70.28	14.32	70.54	71.95
Lader$^{\text{Standard}}$	22.04	66.23	74.32	19.95	66.04	74.01
Lader$^{\text{Pegasos}}$	23.01	**64.54**	**75.22**	20.77	65.48	74.28
Srdp$_0^{\text{MaxForce}}$	**23.59**	64.78	74.31	**20.95**	65.16	74.74
Srdp$_1^{\text{MaxForce}}$	22.10	67.56	70.95	20.92	65.27	**74.92**
Srdp$_0^{\text{Pegasos}}$	21.63	67.19	72.47	20.68	**64.72**	74.75
Srdp$_1^{\text{Pegasos}}$	21.48	67.58	72.51	20.46	65.48	74.62
	English→Japanese			Japanese→English		
	Bleu	Ter	Ribes	Bleu	Ter	Ribes
Moses	21.33	73.28	67.92	18.13	71.94	65.76
Syntax	18.40	75.91	65.78	**19.10**	72.39	65.21
Lader$^{\text{Standard}}$	19.78	75.30	68.08	18.80	71.32	**67.81**
Lader$^{\text{Pegasos}}$	**22.32**	**72.18**	**69.85**	18.72	**71.04**	67.77
Srdp$_0^{\text{MaxForce}}$	20.38	74.92	67.34	17.76	72.94	65.09
Srdp$_1^{\text{MaxForce}}$	21.16	73.60	68.17	17.65	72.90	64.37
Srdp$_0^{\text{Pegasos}}$	20.85	74.31	68.01	17.44	73.34	63.57
Srdp$_1^{\text{Pegasos}}$	20.69	74.32	67.44	17.91	72.76	64.78

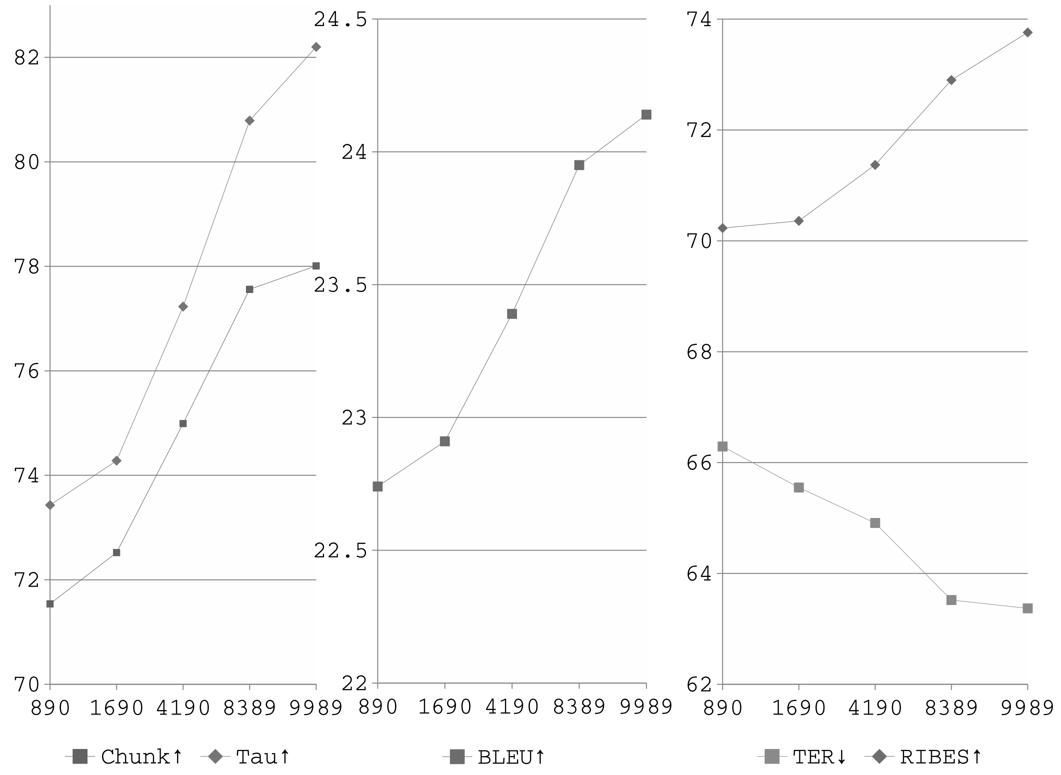


Figure 6.1: The effect of the size of training data

6.2.2 Discussion

Because a large size of training data, i.e. word-aligned corpus, is available, we evaluated reordering parsers between English and Korean using 600 sentences separated from the training data. The accuracy of Srdp is comparable or even higher, while the speed is much faster than Lader. MaxForce outperformed Standard update in a framework of structured prediction. The amounts of the improvement have similar tendency for both languages (+1.8 vs. +1.5 and +3.49 vs. +3.57 for τ and chunk, respectively). Pegasos outperformed both learning algorithms in terms of reordering accuracy . Compare with $\text{Srdp}_0^{\text{MaxForce}}$, $\text{Srdp}_0^{\text{Pegasos}}$ achieved gains of 0.78 (English) and 0.65 (Korean) τ and 0.45 (English) chunk accuracy. Differences between$\text{S}{\small\text{RDP}}_1^{\text{MaxForce}}$ and $\text{S}{\small\text{RDP}}_1^{\text{Pegasos}}$ were smaller than $m = 0$ cases, except τ for Korean. Unlike learning methods of structured prediction, PEGASOS directly concerns the loss of reordering during training. It also incorporates regularization technique and thus leads relatively stable result.

On the other aspect, allowing non-projectivity in a tree structure ($m = 1$) achieves a gain over ITG reordering parser ($m = 0$) without increasing running time significantly. $\text{S}{\small\text{RDP}}_1^{\text{Pegasos}}$ achieved the best scores for English τ and Korean τ and chunk accuracy. Its running time was increased a little (+0.23 and +0.33 seconds for English and Korean, respectively), which was much faster than LADER counterparts (-7.65 and -24.71 seconds for English and Korean, respectively). Note that Korean LADER run three times slower than English LADER, while Korean SRDP and English SCDP run in almost similar time. Therefore, our proposed method is much appropriate for the practical purpose.

Different from reordering parsing, SMT systems including MOSES and SYNTAX baseline were compared using three corpora in both directions. The translation quality measured by automatic evaluation metrics shows that SRDP significantly outperformed MOSES baseline between English/Chinese and Korean. Interestingly, the results of SYNTAX baselines between English/Chinese and Korean and English→Japanese were much poorer than that of MOSES baselines. For Japanese→English, however, SYNTAX baseline outperformed all competitors.

For English and Korean, SRDP achieved improvements over MOSES more than 3 BLUE, 4 TER, and 6 RIBES scores, respectively. Non-projective reordering parsers further enhanced the translation quality, and achieved better scores than LADER. For Chinese and Korean, SRDP also outperformed both MOSES and LADER, while non-projectivity did not improve the translation quality. For English and Japanese, our proposed method underperformed the competitors. Allowing non-projectivity improved Srdp with MaxForce for English→Japanese and with Pegasos for Japanese→English.

We further investigated the reasons of the differences in the translation quality. The sentence length greatly influences Srdp because search errors made in earlier stages cannot be recovered in later stages. In other words, early decisions require information of remaining sentence, which is unknown at that time. The average sentence length of English/Chinese and Korean corpus is shorter than that of English Japanese. Therefore, Srdp suffer from more ambiguity of longer sentences than Lader, which produce a bigger structure using smaller structures in a bottom-up manner, and thus incorporate more information.

The size of training data greatly influences the reordering and translation quality. At the first time-stamp (890), it starts with much lower τ and chunk accuracy than the final stage (9989). Nevertheless, the translation quality already was higher than Moses baseline (21.11 vs. 22.74, 67.98 vs. 66.29, and 67.15 vs. 70.23 for Bleu, Ter, and Ribes, respectively) As the data size increased, the reordering and translation quality increased proportional to the amount until the fourth time-stamp (8389).

Table 6.7 to 6.9 are examples of the translation results from the five timestamps for English→Korean. Given an English sentence, a Korean reference sentence and the translation results are shown below. For each result, underlined words denote ungrammatical translations, and strike-through words incorrect lexical choices, respectively.

Example 6.7a has a prepositional phrase at the beginning of the sentence, followed by the main clause. As the reference sentence, word reordering is required within the prepositional phrase and the main clause independently. Among the results, the fourth time-stamp (8389) produced the best translation result, reordering words correctly yet select an incorrect functional word (“가”). On the other hand, Moses produced incorrect translation for the main clause because it did not perform word reordering and translated the main clause monotonically (“조치 를 취하 는 것 을 방해”). Example 6.7b also shows a similar result. The translation of the main predicate (“called”) needs to be placed at the end of Korean sentence. While the translation results of the five timestamps succeeded word reordering, that Moses failed and produce incorrect translation for the main predicate (“하 기 에 다”).

For more complex sentences, the difference between our proposed method and Moses were amplified. Example 6.8a consists of an adverb, adverbial phrase, and the main clause containing a preposition phrase with a negation pattern (“not ... any more”). The failure of word reordering caused a translation result with totally different meaning as Moses (“더 이상 말 하 지 않 는다 고 약속 한다 면 나쁜 농담”). Contrarily, our proposed method produced relatively good translations after 1690, basically relocated the main predicate at the end of the sentence (“promised”). Example 6.8b has a relative clause as the object of the main predicate. Such pattern often brings difficulty in word reordering for Moses because long distance movement are not preferred. As a consequence, Moses produced incorrect translation results for the main predicate (“cautioned”), rather regarded the predicate in the relative clause (“would be contingent”) as the main predicate. Our proposed method successfully placed the main predicate and the predicate in the relative clause, and produced correct translation.

Table 6.9 shows translation results of relatively short sentences. Even in a short sentence, word reordering plays a crucial role for translation. Example 6.9a has a negation pattern (“not ... any”), which makes Moses difficult to translate the sentence correctly. Example 6.9b is a simple sentence but Moses failed to achieve correct translation of the main predicate (“wanted”). For both examples, our proposed method generated the correct word reordering and eventually produced more

Table 6.7: Examples of the translation results from the five timestamps and MOSES

(a)

English:	In cases where threats appear imminent , steps might be taken to disrupt publication .
Korean:	위협 이 심각 하 게 보이 는 상황 에서 는 , 출판 을 방해 하 는 절차 가 개입 된다 .
890:	~~아~~ 위협 이 임박 경우 에 , ~~조치 를 발표 하 는 것 을 방해~~ 할 수 있 다 .
1690:	~~, 사건 들 에서 임박 한 위협 으로 보이 는 곳 을 출판~~ 하 기 위해 조치 를 취할 수 있 다 .
4190:	경우 에 위협 을 임박 한 것 처럼 보이 는 곳 인 , ~~방해 를 발표 하 기 위해~~ 수도 있 다 .
8398:	위협 가 임박 한 것 으로 보이 는 경우 , 출판 을 방해 하 는 조치 를 취해 야 할 것 이 다 .
9899:	~~사건 에서 위협 을 즉각 적 으로 보이 지만~~ , ~~출판 절차~~ 들 을 방해 하 게 될 수도 있 다 .
MOSES:	위협 을 임박 한 것 으로 보이 는 경우 들 에 는 , ~~조치 를 취하 는 것 을 방해~~ 할 수 있 게 되 었 다

(b)

English:	He called on Russia to expel Mr. Snowden .
Korean:	그 는 러시아 가 스노우 덴 씨 를 내쫓 을 것 을 요구 했 습니다 .
890:	그 는 그 가 러시아 에 스노 든 을 추방 할 것 을 요구 했 다 .
1690:	그 는 러시아 에 게 스노우 든 을 추방 에 요구 했 다 .
4190:	그 는 러시아 가 스노우 든 을 추방 할 것 을 요구 했 다 .
8389:	그 는 러시아 가 스노 든 을 추방 할 것 을 요구 했 다 .
9899:	그 는 러시아 에 게 스노우 든 을 추방 할 것 을 요청 했 다 .
MOSES:	그 는 러시아 가 스노 든 을 추방 ~~하 가 에 다~~ .

Table 6.8: Examples of the translation results from the five timestamps and MOSES

(a)

English:	Later , after a break , West promised not to tell any more bad jokes .
Korean:	잠시 휴식 을 가진 후 , 웨스트 는 더이상 의 좋 지 않 은 농담 을 하 지 않 을 것 을 약속 했 다 .
890:	나중 에 , ~~한 후~~ , 웨스트 는 ~~말 하 지 않 을 것 을 약속 하 는 어떠한 더 나쁜 농담~~ 했 다 .
1690:	이후 , 나중 에 , 웨스트 는 어떠한 더 많 은 나쁜 농담 을 하 지 않 기 로 약속 했 다 .
4190:	~~, 한 후 에~~ , 웨스트 는 어떠한 더 나쁜 농담 을 하 지 않 기 로 약속 하 였 다 .
8389:	~~후 에 ,~~ 나중 에 , 웨스트 는 더 나쁜 농담 도 말 하 지 않 기 로 약속 했 다 .
9989:	웨스트 는 어떠한 더 나쁜 농담 을 나중 에 ~~,~~ 하 지 않 기 로 약속 했 다 .
MOSES:	나중 에 , ~~후 ,~~ 웨스트 는 ~~더 이상 말 하 지 않 는다 고 약속 한다 면 나쁜 농담~~ 했 다 .

(b)

English:	However , Fed officials cautioned that the timeline would be contingent on the health of the economy .
Korean:	그러나 연준 의 관계자들 은 이 시간표 는 경제 의 건전성 의 여하 에 달렸 다 고 경고 했 다 .
890:	그러나 , 연준 관계자들 은 타임 라인 ~~의 건강 이 경제 에 파견 될 것 이라고~~ 경고 했 다 .
1690:	그러나 , 연준 관계자들 은 타임 라인 은 경제 의 건강 에 달려 있 을 것 이라고 경고 했 다 .
4190:	그러나 , 연준 관계자들 은 타임 라인 은 ~~건강 에 대한 여부 는 것 은 경제 의~~ 경고 했 다 .
8398:	그러나 , 연준 관계자들 은 타임 라인 은 경제 의 건강 에 달려 있 을 것 이라고 경고 했 다 .
9989:	그러나 , 연준 관계자들 은 타임 라인 은 경제 의 건강 에 ~~파견 될 것 이~~ 라고 경고 했 다 .
MOSES:	그러나 , 연준 관계자들 은 타임 라인 을 ~~하 기 위해 서 는 것 의 건강 에 달려 있 다~~

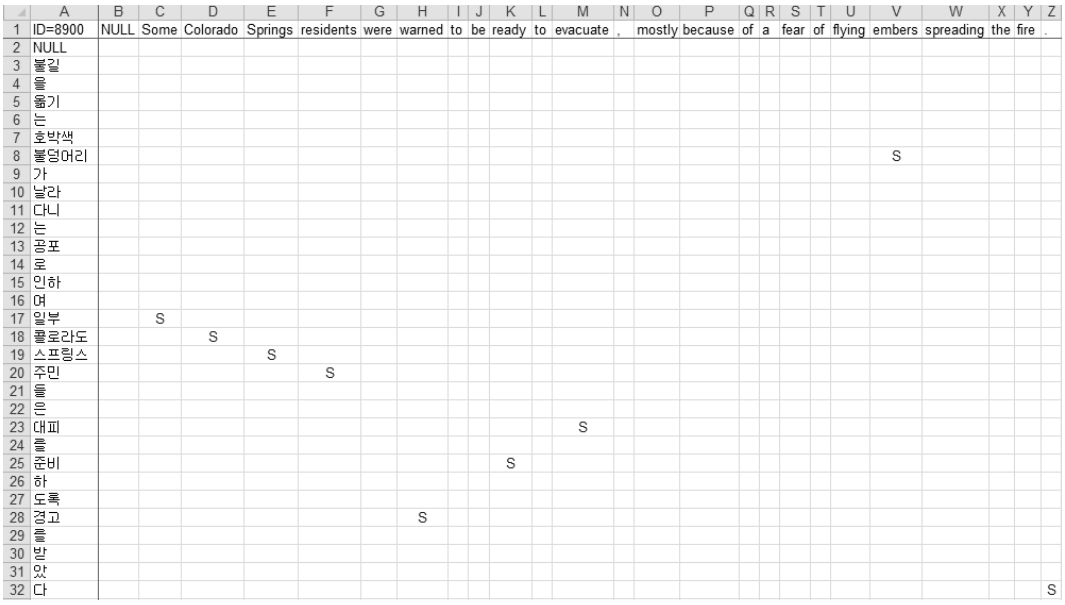
Table 6.9: Examples of the translation results from the five timestamps and Moses

(a)

English:	I can 't say how it will end . ”
Korean:	나 는 이것 이 어떻 게 끝 나 게 될 것 인지 는 언급 하 지 않 겠 다 . ”
890:	~~그것 이 어떻 게 할 것 이라고~~ 나 는 말 할 수 없 다 ” ~~고 말 했 다~~ .
1690:	어떻 게 그것 이 끝 나 는 말 할 수 없 다 . ”
4190:	어떻 게 그것 이 끝 나 는 말 할 수 없 다 . ”
8389:	나 는 그것 이 어떻 게 끝 날 것 이라고 말 할 수 없 다 . ”
9989:	내 가 어떻 게 끝 날 것 이라고 말 할 수 없 다 . ”
Moses:	내 가 ~~말 할 수 없 는 그것 이 어떻 게 끝 날 것 이 다~~ .

(b)

English:	“ They wanted to be masters of their own country . ”
Korean:	“ 그 들 은 자신 의 나라 에 주인 이 되 고 싶 었 다 . ”
890:	“ 그 들 이 원했 던 것 은 그 들 의 자국 의 주인 이 다 . ”
1690:	“ 그 들 은 그들 자신 의 나라 의 주인 하 고자 하 는 것 이 다 . ”
4190:	“ 그 들 은 그들 자신 의 나라 의 주인 이 고 싶 었 다 . ”
8389:	“ 그 들 은 그들 자신 의 나라 의 주인 하 길 원했 다 . ”
9989:	“ 그 들 은 그들 자신 의 나라 의 ~~고 싶 었 을 것~~ 이 다 . ”
Moses:	이 되 기 를 원했 던 그 들 은 그들 자신 의 나라 의 ~~석사~~ 다 .



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	ID=8900	NULL	Some	Colorado	Springs	residents	were	warned	to	be	ready	to	evacuate	,	mostly	because	of	a	fear	of	flying	embers	spreading	the	fire	.
2	NULL																									
3	불길																									
4	을																									
5	옮기																									
6	는																									
7	호박색																									
8	불덩어리																					S				
9	가																									
10	날라																									
11	다니																									
12	는																									
13	공포																									
14	로																									
15	인하																									
16	여																									
17	일부		S																							
18	콜로라도			S																						
19	스프링스				S																					
20	주민					S																				
21	들																									
22	은																									
23	대피												S													
24	를																									
25	준비										S															
26	하																									
27	도록																									
28	경고							S																		
29	를																									
30	받																									
31	았																									
32	다																									S

Figure 6.2: A graphical user interface for annotation. S and P denote sure and possible annotation, respectively

reference-like output.

6.3 Annotation Guideline for English-Korean

In this section, the annotation guideline for English-Korean is described. An automatic word alignment often has incorrect or missing links, but provide a lot of correct links. Therefore, we decided to provide an automatic word alignment as precise as possible to the annotators as a seed. For the seed word alignment, bidirectional word alignments and their intersection was utilized. As the intersection allows word-to-word correspondence appearing in both directions, it tends to provide word alignment with high precision and low recall. Figure 6.2 illustrates a graphical user interface for the annotation, providing the seed word alignment.

The annotators then manipulated the seed word alignment using their insight.

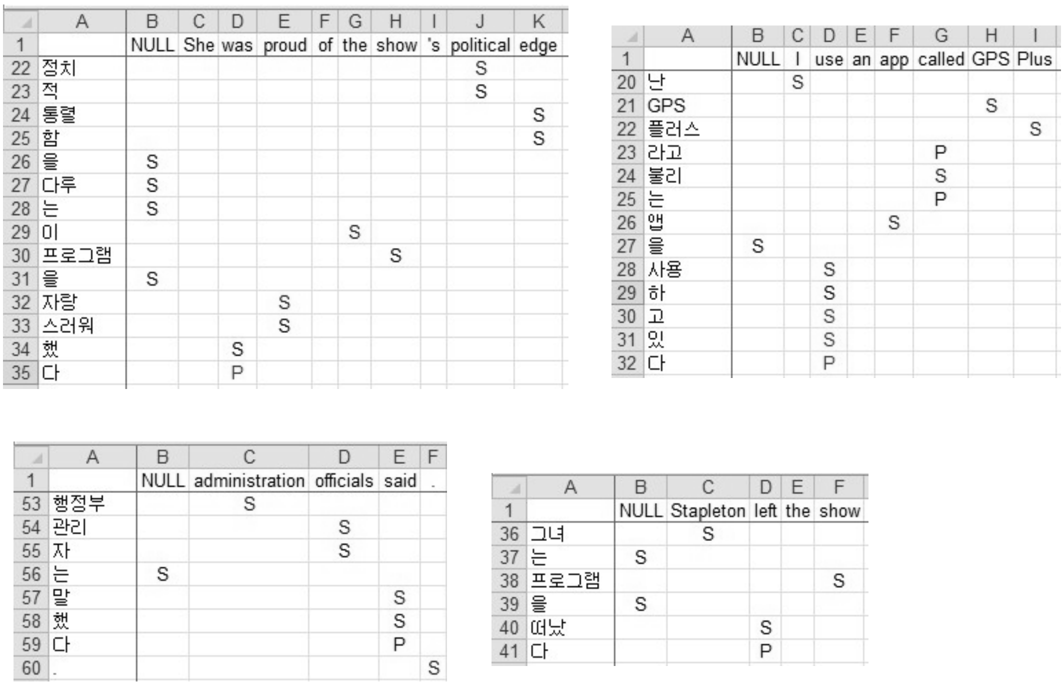
More specifically, they deleted incorrect links and inserted missing links. There are two types of links: *sure* and *possible*. We borrowed the distinction of sure and possible links as follows: "each alignment point may be marked as sure, meaning that both words are a translation of each other in any context, or as possible, meaning that the words are a translation of each other in some contexts. [Graca 2008]" Unaligned words were allowed as few as possible so that "words should only remain unaligned when you can answer 'Yes' to the following question: If the seemingly extraneous words were simply deleted from their verse, would the two verses become more similar in meaning? [Kruijff-Korbayová 2006]" Note that unaligned words were also annotated using a special word ("Null") in the other language.

Because the above distinction is often ambiguous in practice, however, the guideline described the following examples as reference annotations.

Main predicate A Korean main predicate consists of various morphemes representing modality such as voice, tense, and aspect, in addition to the content morpheme. In our guideline, we let the annotators mark a morpheme to sure as many as possible, except the verbal ending. Figure 6.3 shows annotation examples for main predicates.

Case markers In Korean, there exist explicit markers for nominative and accusative cases, where English has no corresponding words. In our guideline, therefore, nominative and accusative case markers are Null-aligned words in general.

Prepositions An English preposition often correspond to multiple Korean morphemes. Sometimes, an English preposition is translated in a sequence of Korean words including a case marker as shown in Figure 6.4. If Korean words do not change the meaning of the sentence even they are absent, the words correspond to



	A	B	C	D	E	F	G	H	I	J	K
1		NULL	She	was	proud	of	the	show	's	political	edge
22	정치									S	
23	적									S	
24	통렬										S
25	함										S
26	을	S									
27	다루	S									
28	는	S									
29	이						S				
30	프로그램							S			
31	을	S									
32	자랑				S						
33	스러워				S						
34	했			S							
35	다			P							

	A	B	C	D	E	F	G	H	I
1		NULL	I	use	an	app	called	GPS	Plus
20	난		S						
21	GPS							S	
22	플러스								S
23	라고						P		
24	불리						S		
25	는						P		
26	앱					S			
27	을	S							
28	사용			S					
29	하			S					
30	고			S					
31	있			S					
32	다			P					

	A	B	C	D	E	F
1		NULL	administration	officials	said	.
53	행정부		S			
54	관리			S		
55	자			S		
56	는	S				
57	말				S	
58	했				S	
59	다				P	
60	.					S

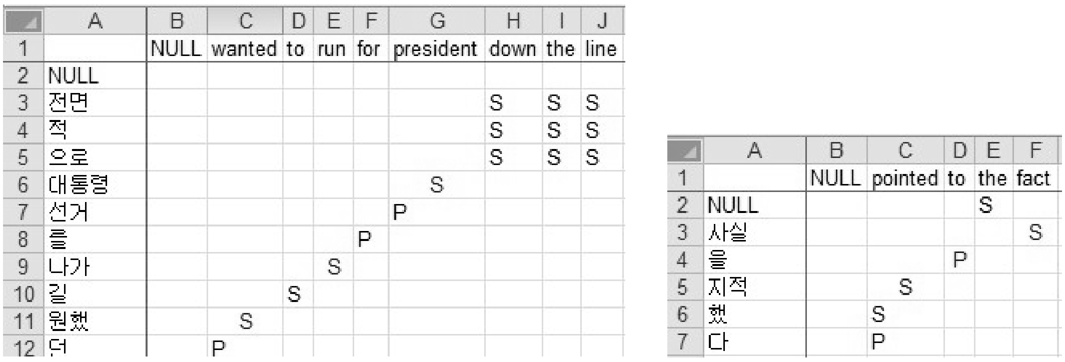
	A	B	C	D	E	F
1		NULL	Stapleton	left	the	show
36	그녀		S			
37	는	S				
38	프로그램					S
39	을	S				
40	떠났			S		
41	다			P		

Figure 6.3: Annotation examples for main predicates

Null as shown in Figure 6.5. Table 6.10 collects the following examples as reference annotations, yet not complete.

Present progressive Present progressive in English basically consists of a copula and a present particle. Our guideline is to annotate a present progressive to the corresponding Korean words with sure links except the verbal ending mapping to the copula with a possible link. Figure 6.6 shows annotation examples for present progressive.

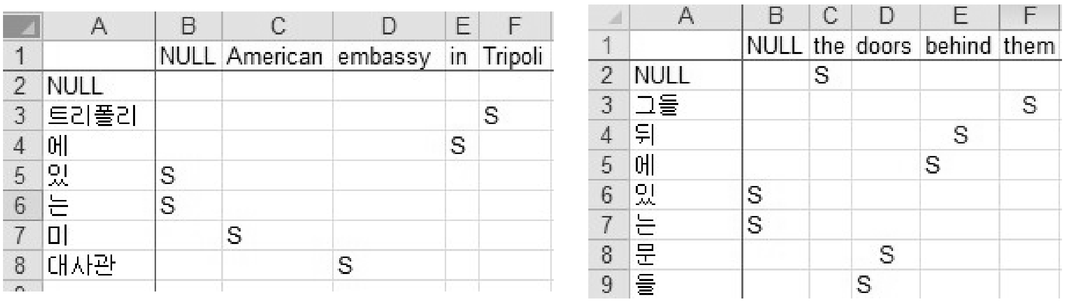
Passive Passive in English basically consists of a copula and a past particle. Analogous to present progressive, the verbal ending maps to the copula with a possible link. Figure 6.7 shows annotation examples for present progressive.



	A	B	C	D	E	F	G	H	I	J
1		NULL	wanted	to	run	for	president	down	the	line
2	NULL									
3	전면							S	S	S
4	적							S	S	S
5	으로							S	S	S
6	대통령						S			
7	선거						P			
8	를					P				
9	나가				S					
10	길			S						
11	원했		S							
12	던		P							

	A	B	C	D	E	F
1		NULL	pointed	to	the	fact
2	NULL				S	
3	사실					S
4	을			P		
5	지적		S			
6	했		S			
7	다		P			

Figure 6.4: Annotation examples for prepositions



	A	B	C	D	E	F
1		NULL	American	embassy	in	Tripoli
2	NULL					
3	트리폴리					S
4	에				S	
5	있	S				
6	는	S				
7	미		S			
8	대사관			S		

	A	B	C	D	E	F
1		NULL	the	doors	behind	them
2	NULL		S			
3	그들					S
4	뒤				S	
5	에				S	
6	있	S				
7	는	S				
8	문			S		
9	들			S		

Figure 6.5: Annotation examples for null alignment for the preposition

Present perfect Present perfect in English basically consists of an auxiliary verb and a past particle. A present perfect corresponding Korean words with sure links except the verbal ending mapping to the auxiliary verb with a possible link. Figure 6.8 shows annotation examples for present prefect.

Idiom, idiomatic expression Idiom and idiomatic expression cannot be decomposed into the word level. Hence, they were regarded as a unit and marked as a block of annotations as shown in Figure 6.9. Note that above mentioned guideline were applied for unaligned words.

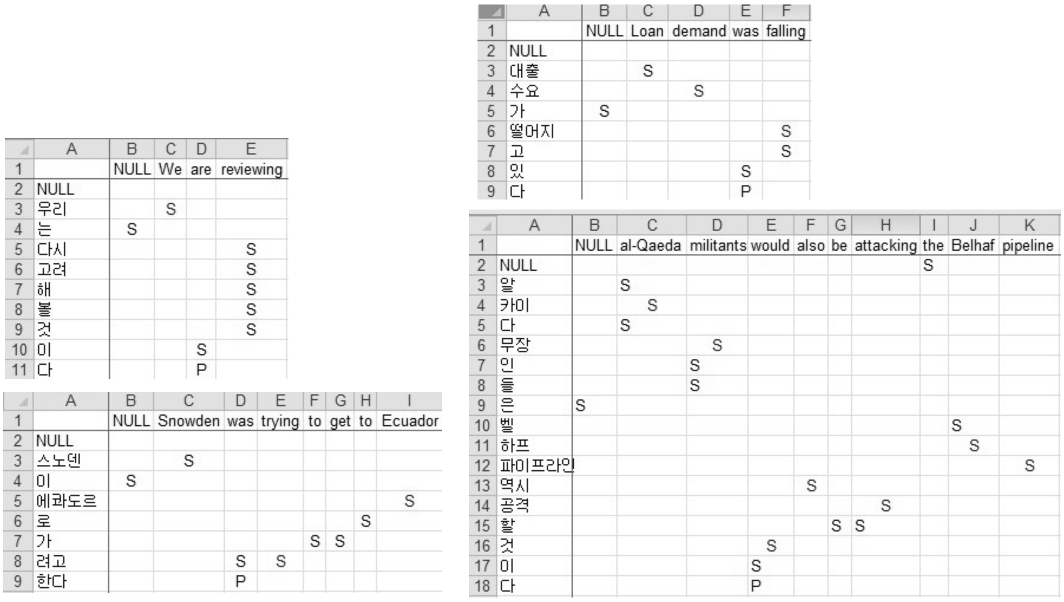
Table 6.10: Annotation example for prepositions

English	Korean
over	에 (P) 거쳐(S)
about	에 (P) 대해 (S) / 에 (P) 관한 (S)
on	에 (P) 대해 (S) / 에 (P) 관해 (S)
against	에 (P) 대한 (S) / 에 (P) 반대하여 (S)
for	에 (P) 대한 (S) / 을(P) 위한 (S)
by	에 (P) 의해 (S)
to	에 (P) 대한 (S)
through	을 (P) 통해 (S)
from	로 (P) 부터 (S)
despite	에도 (P) 불구하고 (S)
of	중 (P) 에 (S)
in	을 (P) 통해 (S)

To infinitive A "to infinitive" pattern in English is translated in various styles in Korean. Our guideline simply regarded the "to" word as an independent word, and annotated the corresponding Korean words such as "도록" or "것" with sure link if possible. Figure 6.10 shows annotation examples for to infinitive.

Quantitative Korean has dependent nouns to represent quantitative such as "년, 월, 일, 명, 가지, 개", which often do not appear in English explicitly. Our guideline annotated them with possible links unless the quantitative word appears explicitly. Figure 6.11 shows annotation examples.

Shared component Sometimes a natural translation do not repeat shared component in a sentence. Our guideline aims to recovered the omitted word at the later stage, and marked the last corresponding word among the candidates. Figure 6.12 shows annotation examples.



	A	B	C	D	E
1		NULL	We	are	reviewing
2	NULL				
3	우리		S		
4	는	S			
5	다시				S
6	고려				S
7	해				S
8	볼				S
9	것				S
10	이			S	
11	다			P	

	A	B	C	D	E	F
1		NULL	Loan	demand	was	falling
2	NULL					
3	대출		S			
4	수요			S		
5	가	S				
6	떨어지					S
7	고					S
8	있				S	
9	다				P	

	A	B	C	D	E	F	G	H	I
1		NULL	Snowden	was	trying	to	get	to	Ecuador
2	NULL								
3	스노덴		S						
4	이	S							
5	에콰도르								S
6	로							S	
7	가					S	S		
8	려고			S	S				
9	한다			P					

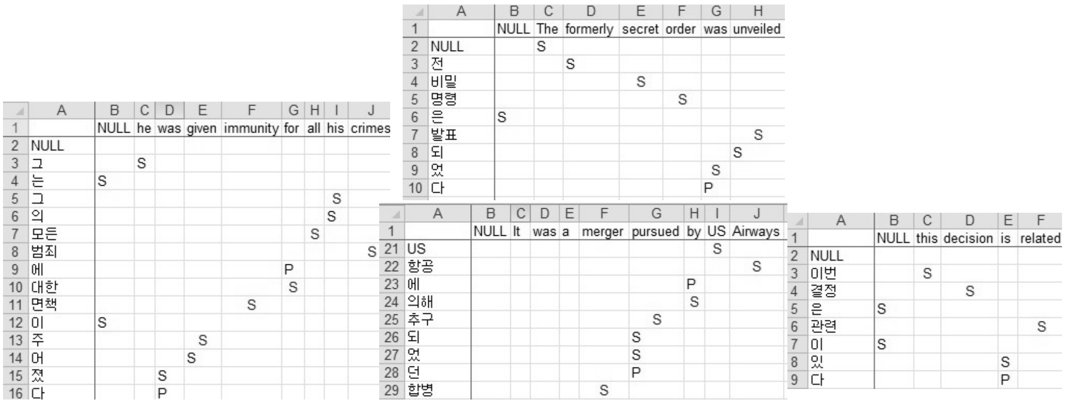
	A	B	C	D	E	F	G	H	I	J	K
1		NULL	al-Qaeda	militants	would	also	be	attacking	the	Belhaf	pipeline
2	NULL								S		
3	알		S								
4	카이		S								
5	다		S								
6	무장			S							
7	인			S							
8	들			S							
9	은	S									
10	벨									S	
11	하프									S	
12	파이프라인										S
13	역시					S					
14	공격							S			
15	할						S	S			
16	것				S						
17	이				S						
18	다				P						

Figure 6.6: Annotation examples for present progressive

For integrity of the annotation, an automatic procedure checked the annotation result. Although the procedure could not check all possible cases, it greatly reduced the time for manual inspection of the annotation result. There were four cases that the procedure concerned as follows:

- A word had annotation with both Null and other words.
- A word did not have any annotation .
- An annotation was neither sure nor possible.
- A word was an empty string (the annotators unexpectedly modifies English/Korean sentences)

Figure 6.13 shows an example of erroneous annotations. The 21th Korean word ("들") maps to both Null and the English word in the column F ("residents"). The English word in the column E (”Springs”) also maps to both Null and the



	A	B	C	D	E	F	G	H	I	J
1		NULL	he	was	given	immunity	for	all	his	crimes
2	NULL									
3	그		S							
4	는	S								
5	그								S	
6	의								S	
7	모든							S		
8	범죄									S
9	에						P			
10	대한						S			
11	면책					S				
12	이	S								
13	주				S					
14	어				S					
15	졌			S						
16	다			P						

	A	B	C	D	E	F	G	H
1		NULL	The	formerly	secret	order	was	unveiled
2	NULL		S					
3	전			S				
4	비밀				S			
5	명령					S		
6	은	S						
7	발표							S
8	되							S
9	었						S	
10	다						P	

	A	B	C	D	E	F	G	H	I	J
1		NULL	It	was	a	merger	pursued	by	US	Airways
21	US								S	
22	항공									S
23	에							P		
24	의해							S		
25	추구						S			
26	되						S			
27	었						S			
28	던						P			
29	합병					S				

	A	B	C	D	E	F
1		NULL	this	decision	is	related
2	NULL					
3	이번		S			
4	결정			S		
5	은	S				
6	관련					S
7	이	S				
8	있				S	
9	다				P	

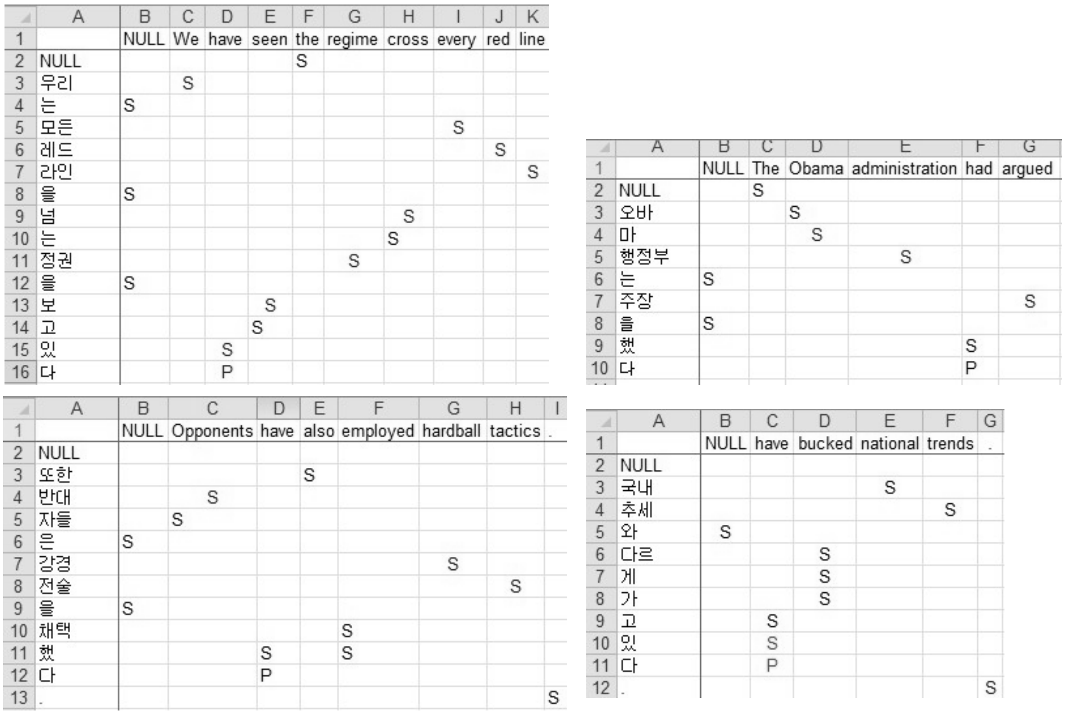
Figure 6.7: Annotation examples for passive

19th Korean word ("스프링스"). Many Korean words (in the row 5,6,8, and 10 to 16) does not have any annotation. The annotation at the row 17 and column C is neither sure nor possible.

Cohen's kappa coefficient is one of the measure the agreement between annotators. For the test sentences used to measure the accuracy of reordering parsers, we measured the inter-annotator agreement using the kappa statistics. Although there are modified kappa coefficient for word alignment with sure and possible annotations scheme, a simple kappa coefficient was utilize in our experiment as follows:

- true-true (A): if two annotators mark same word as word alignment
- true-false (B): if a word have corresponding counterpart, but not the other
- false-true (C): if a word maps to Null but not the other
- false-false (D): if two annotators marks same word as Null

The statistics computed as the original Cohen's kappa coefficient κ was 91.45 for our annotation results. Although the corpus was constructed by human effort, the annotators are beginners not experts unlike tree-bank annotation. After the first



	A	B	C	D	E	F	G	H	I	J	K
1		NULL	We	have	seen	the	regime	cross	every	red	line
2	NULL					S					
3	우리		S								
4	는	S									
5	모든								S		
6	레드									S	
7	라인										S
8	을	S									
9	넘							S			
10	는							S			
11	정권						S				
12	을	S									
13	보				S						
14	고				S						
15	있			S							
16	다			P							

	A	B	C	D	E	F	G
1		NULL	The	Obama	administration	had	argued
2	NULL		S				
3	오바			S			
4	마			S			
5	행정부				S		
6	는	S					
7	주장						S
8	을	S					
9	했					S	
10	다					P	

	A	B	C	D	E	F	G	H	I
1		NULL	Opponents	have	also	employed	hardball	tactics	.
2	NULL								
3	또한				S				
4	반대		S						
5	자들		S						
6	은	S							
7	강경						S		
8	전술							S	
9	을	S							
10	채택					S			
11	했			S		S			
12	다			P					
13	.								S

	A	B	C	D	E	F	G
1		NULL	have	bucked	national	trends	.
2	NULL						
3	국내				S		
4	추세					S	
5	와	S					
6	다르			S			
7	게			S			
8	가			S			
9	고		S				
10	있		S				
11	다		P				
12	.						S

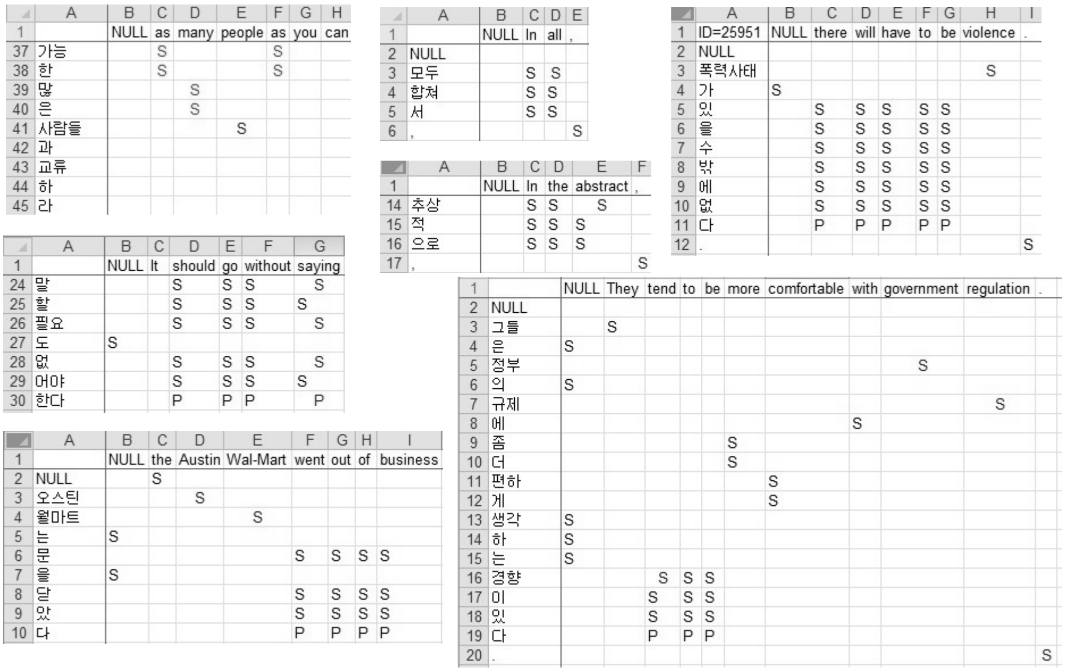
Figure 6.8: Annotation examples for present perfect

week, they were familiar with the annotation guideline, and produce the results very quickly. Hence, it is expected that our proposed method can be applicable other language pairs without heavy investment. For example, crowd-sourcing is a general approach which fits for this purpose. We hope that many researchers utilizes our findings to handle word reordering in an efficient manner for both engineering and economical viewpoints.

$$Pr(a) = \frac{A+B}{A+B+C+D} \qquad (6.1)$$

$$Pr(e) = \frac{(A+B)(A+C)+(C+D)(B+D)}{(A+B+C+D)^2} \qquad (6.2)$$

$$\kappa = \frac{Pr(a)-Pr(e)}{1-Pr(e)} \qquad (6.3)$$



	A	B	C	D	E	F	G	H
1		NULL	as	many	people	as	you	can
37	가능		S			S		
38	한		S			S		
39	많			S				
40	은			S				
41	사람들				S			
42	과							
43	교류							
44	하							
45	라							

	A	B	C	D	E
1		NULL	In	all	,
2	NULL				
3	모두		S	S	
4	합쳐		S	S	
5	서		S	S	
6	,				S

	A	B	C	D	E	F	G	H	I
1	ID=25951	NULL	there	will	have	to	be	violence	.
2	NULL								
3	폭력사태							S	
4	가	S							
5	있		S	S	S	S	S		
6	을		S	S	S	S	S		
7	수		S	S	S	S	S		
8	밖		S	S	S	S	S		
9	에		S	S	S	S	S		
10	없		S	S	S	S	S		
11	다		P	P	P	P	P		
12	.								S

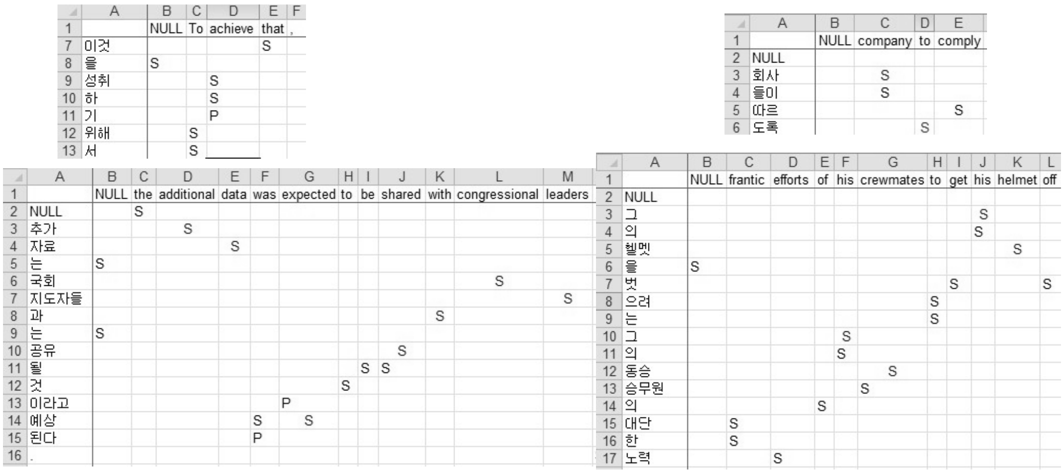
	A	B	C	D	E	F
1		NULL	In	the	abstract	,
14	추상		S	S	S	
15	적		S	S	S	
16	으로		S	S	S	
17	,					S

	A	B	C	D	E	F	G
1		NULL	It	should	go	without	saying
24	말			S	S	S	S
25	할			S	S	S	S
26	필요			S	S	S	S
27	도	S					
28	없			S	S	S	S
29	어야			S	S	S	S
30	한다			P	P	P	P

	A	B	C	D	E	F	G	H	I
1		NULL	the	Austin	Wal-Mart	went	out	of	business
2	NULL		S						
3	오스틴			S					
4	월마트				S				
5	는	S							
6	문					S	S	S	S
7	을	S							
8	닫					S	S	S	S
9	았					S	S	S	S
10	다					P	P	P	P

1		NULL	They	tend	to	be	more	comfortable	with	government	regulation	.
2	NULL											
3	그들		S									
4	은	S										
5	정부									S		
6	의	S										
7	규제										S	
8	에								S			
9	좀						S					
10	더						S					
11	편하							S				
12	게							S				
13	생각	S										
14	하	S										
15	는	S										
16	경향			S	S	S						
17	이			S	S	S						
18	있			S	S	S						
19	다			P	P	P						
20	.											S

Figure 6.9: Annotation examples for idiom and idiomatic expression



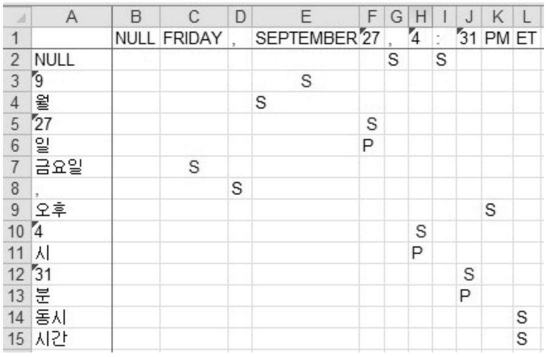
	A	B	C	D	E	F
1		NULL	To	achieve	that	,
7	이것				S	
8	을	S				
9	성취			S		
10	하			S		
11	기			P		
12	위해		S			
13	서		S			

	A	B	C	D	E
1		NULL	company	to	comply
2	NULL				
3	회사		S		
4	들이		S		
5	따르				S
6	도록			S	

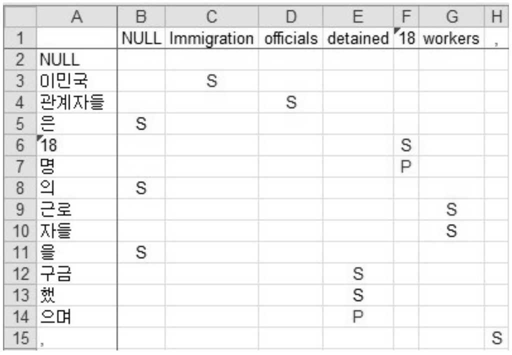
	A	B	C	D	E	F	G	H	I	J	K	L	M
1		NULL	the	additional	data	was	expected	to	be	shared	with	congressional	leaders
2	NULL		S										
3	추가			S									
4	자료				S								
5	는	S											
6	국회											S	
7	지도자들												S
8	과										S		
9	는	S											
10	공유									S			
11	될								S	S			
12	것							S					
13	이라고						P						
14	예상					S	S						
15	된다					P							
16	.												

	A	B	C	D	E	F	G	H	I	J	K	L
1		NULL	frantic	efforts	of	his	crewmates	to	get	his	helmet	off
2	NULL											
3	그									S		
4	의									S		
5	헬멧										S	
6	을	S										
7	벗								S			S
8	으려							S				
9	는							S				
10	그					S						
11	의					S						
12	동승						S					
13	승무원						S					
14	의				S							
15	대단		S									
16	한		S									
17	노력			S								

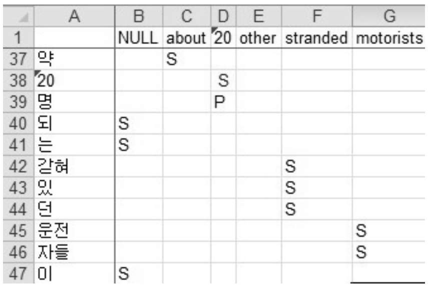
Figure 6.10: Annotation examples for to infinitive



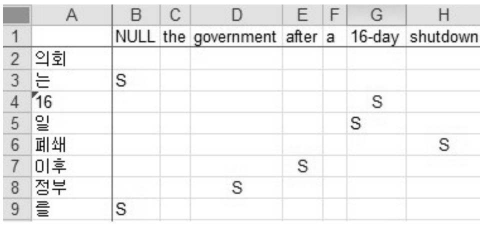
	A	B	C	D	E	F	G	H	I	J	K	L
1		NULL	FRIDAY	,	SEPTEMBER	27	,	4	:	31	PM	ET
2	NULL						S		S			
3	9				S							
4	월				S							
5	27					S						
6	일					P						
7	금요일		S									
8	,			S								
9	오후										S	
10	4							S				
11	시							P				
12	31									S		
13	분									P		
14	동시											S
15	시간											S



	A	B	C	D	E	F	G	H
1		NULL	Immigration	officials	detained	18	workers	,
2	NULL							
3	이민국		S					
4	관계자들			S				
5	은	S						
6	18					S		
7	명					P		
8	의	S						
9	근로						S	
10	자들						S	
11	을	S						
12	구금				S			
13	했				S			
14	으며				P			
15	,							S

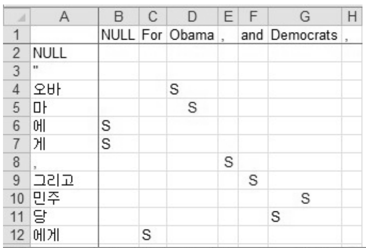


	A	B	C	D	E	F	G
1		NULL	about	20	other	stranded	motorists
37	약		S				
38	20			S			
39	명			P			
40	되	S					
41	는	S					
42	갇혀					S	
43	있					S	
44	던					S	
45	운전						S
46	자들						S
47	이	S					

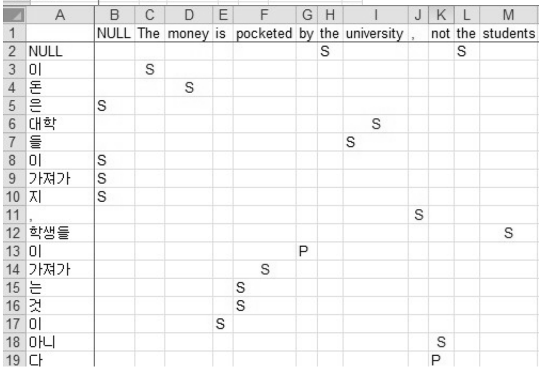


	A	B	C	D	E	F	G	H
1		NULL	the	government	after	a	16-day	shutdown
2	의회							
3	는	S						
4	16						S	
5	일						S	
6	폐쇄							S
7	이후				S			
8	정부			S				
9	를	S						

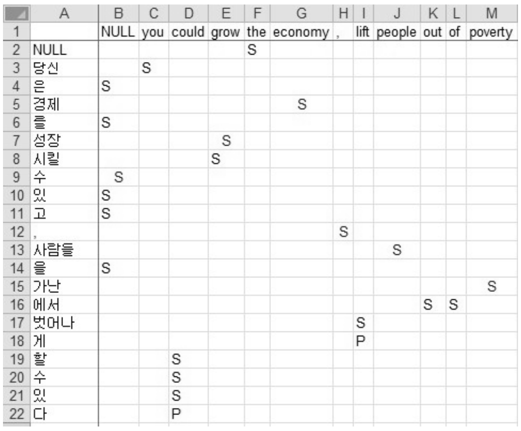
Figure 6.11: Annotation examples for quantitative



	A	B	C	D	E	F	G	H
1		NULL	For	Obama	,	and	Democrats	,
2	NULL							
3	"							
4	오바			S				
5	마			S				
6	에	S						
7	게	S						
8	,				S			
9	그리고					S		
10	민주						S	
11	당						S	
12	에게		S					

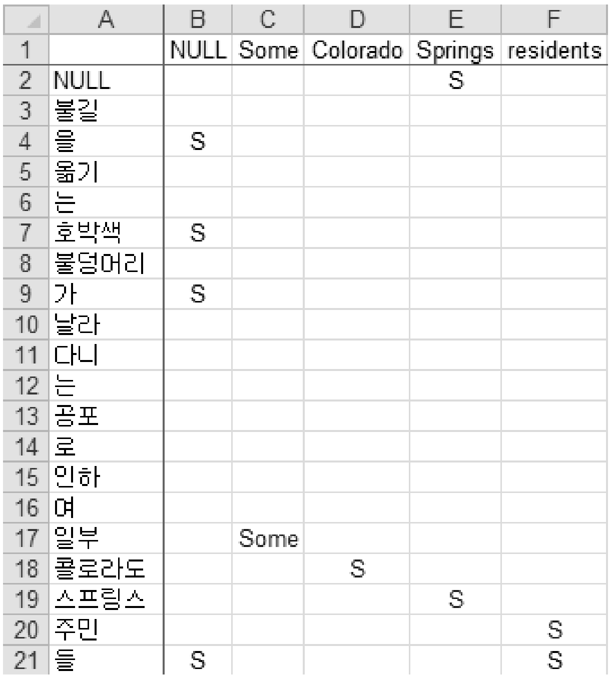


	A	B	C	D	E	F	G	H	I	J	K	L	M
1		NULL	The	money	is	pocketed	by	the	university	,	not	the	students
2	NULL							S				S	
3	이		S										
4	돈			S									
5	은	S											
6	대학								S				
7	들								S				
8	이	S											
9	가져가	S											
10	지	S											
11	,									S			
12	학생들												S
13	이						P						
14	가져가					S							
15	는					S							
16	것					S							
17	이				S								
18	아니										S		
19	다										P		



	A	B	C	D	E	F	G	H	I	J	K	L	M
1		NULL	you	could	grow	the	economy	,	lift	people	out	of	poverty
2	NULL					S							
3	당신		S										
4	은	S											
5	경제						S						
6	를	S											
7	성장				S								
8	시킬				S								
9	수	S											
10	있	S											
11	고	S											
12	,							S					
13	사람들									S			
14	을	S											
15	가난												S
16	에서										S	S	
17	벗어나								S				
18	게								P				
19	할			S									
20	수			S									
21	있			S									
22	다			P									

Figure 6.12: Annotation examples for sharing components



	A	B	C	D	E	F
1		NULL	Some	Colorado	Springs	residents
2	NULL				S	
3	불길					
4	을	S				
5	옮기					
6	는					
7	호박색	S				
8	불덩어리					
9	가	S				
10	날라					
11	다니					
12	는					
13	공포					
14	로					
15	인하					
16	여					
17	일부		Some			
18	콜로라도			S		
19	스프링스				S	
20	주민					S
21	들	S				S

Figure 6.13: An example of erroneous annotations

VII. Conclusion and Future Work

In conclusion, we have thoroughly explored research issues for word reordering. First, we insist that word reordering under the ITG constraints is insufficient to cover certain phenomena appearing in real situation. The analysis of non-ITG word reordering phenomena between Chinese/English and Korean/Japanese showed the evidence of our claim. Second, it is possible to mitigate the limitations of syntactic tree-based reordering methods by allowing non-projectivity and explicitly encoding reordering orientations in the tree. In a novel tree structure for word reordering we proposed, non-ITG permutations are achieved. Third, the non-ITG word reordering is achieved by parsing, in which monolingual parsing techniques are adopted. Analogous to a monolingual parser, a reordering parser produces a reordering tree for a given sentence. Since we train the reordering parsers using a parallel corpus, any syntactic parser is unnecessary. Thus, this technique may be useful for resource-poor languages. Fourth, our proposed method showed promising improvement in both parsing accuracy and translation quality. We investigated the effectiveness of the reordering parsers under the pre-ordering framework of SMT. Fifth, annotation for reordering parsers is much lighter than tree-bank annotation, which is more appropriate for practice. We suggested an annotation guideline for word alignment between English and Korean, and the guideline was easy to understand even for beginners.

We emphasize the following points among our findings.

- It is more general to express non-ITG word reordering in a non-projective tree than in a projective tree. In our thesis, a bottom-up and shift-reduce algorithms adopted from monolingual parsing produce the non-projective tree.

- In a greedy setting, a bottom-up algorithm outperforms a shift-reduce algorithm. In a beam-search setting, however, a shift-reduce algorithm is better than a bottom-up algorithm for non-projective reordering tree.

- To train a reordering parser, we constructed approximately 10K sentences with word alignment, which eventually bring an improvement of translation quality. The resource are relatively cheaper to construct than more linguistically-motivated ones such as tree-bank.

The corpora consisting of shorter sentences contains fewer number of non-ITG word reordering. Nevertheless, we did not construct annotated data for the longer sentences and not prove the effectiveness of our proposed method for longer sentences yet. It is an interesting work to investigate that our proposed method also works for that circumstance in future.

Note that our proposed method does not utilize any linguistic resources such as POS taggers. This scheme is thus useful for resource-poor languages, which at least have a tokenizer/segmentor for raw text. In future, we will further investigate a joint model of tokenization and parsing, which has two beneficial points. First, it produces a reordering tree from a raw sentence directly. Second, error propagation through the pipeline of tokenization and parsing is reduced.

Summary in Korean
요 약 문

본 학위논문은 기계 번역에서 중요한 사안인 어순 조정에 대한 연구 결과이다. 일반적으로 원문과 번역문의 어순이 다르고 한국어와 영어와 같이 언어학적으로 상이한 언어 사이에는 이러한 차이가 더 심각해진다. 비록 기존 연구에서 구문 구조 분석 결과를 어순 조정을 위해 활용해 왔지만, 이러한 방법은 계층 구조에 제약을 받기 때문에 올바른 어순 조정을 제한하는 경우가 발생한다. Inversion Transduction Grammar(ITG)의 이러한 제약 사항은 기존에 널리 통용되어 왔으나 언어학적 차이가 큰 어순 사이에서는 적지 않은 경우에 올바른 어순 조정을 제한하여 번역 성능의 향상을 어렵게 한다.

본 논문에서는 먼저 한-영, 한-중, 일-영 말뭉치를 대상으로 ITG 제약 하에서 어순 조정에 실패하는 경우를 분석하였다. 그 결과 4Ĩ0%가량의 문장이 ITG 제약 하에서 어순 조정에 실패하는 것으로 나타났다. 이렇게 무시할 수 없는 비율의 어순 조정을 올바르게 수행하기 위하여 본 논문에서는 비투사 구문분석 기법을 통한 어순 조정을 하는 방법을 제안한다. 주어진 문장에 대해 기존의 방법은 전통적인 구문분석을 통해 문법적인 계층구조(syntactic tree)를 나타내지만, 제안하는 방법은 어순 조정을 명시한 계층구조(reordering tree)를 활용한다. 이러한 계층 구조를 얻는 분석기는 병렬 말뭉치에서 학습되어 투사성에 따른 제약을 해소한다. 구체적으로는 비투사 의존구문 분석을 위해 제안된 어순 조정 분석기(reordering)를 제안한다. 제안된 분석기를 전 처리 어순 조정에 기반한 통계기계번역에 적용하여 언어학적으로 상이한 언어 사이에서 번역을 수행한 결과, 제안된 방법은 분석기의 성능과 번역 품질에서 향상을 보였다.

References

[Al-Onaizan 2006] Yaser Al-Onaizan and Kishore Papineni. *Distortion Models for Statistical Machine Translation.* In Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics, ACL-44, pages 529–536, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics.

[Birch 2008] Alexandra Birch, Miles Osborne and Philipp Koehn. *Predicting Success in Machine Translation.* In Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '08, pages 745–754, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[Birch 2011] Alexandra Birch and Miles Osborne. *Reordering metrics for MT.* In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1, pages 1027–1035. Association for Computational Linguistics, 2011.

[Brinton 2000] Laurel J Brinton. The structure of modern english. 2000.

[Chiang 2007] David Chiang. *Hierarchical phrase-based translation.* computational linguistics, vol. 33, no. 2, pages 201–228, 2007.

[Chiang 2008] David Chiang, Yuval Marton and Philip Resnik. *Online Large-margin Training of Syntactic and Structural Translation Features.* In Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '08, pages 224–233, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[Collins 2002] Michael Collins. *Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms.* In Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10, EMNLP '02, pages 1–8, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[Collins 2004] Michael Collins and Brian Roark. *Incremental Parsing with the Perceptron Algorithm.* In Proceedings of the 42Nd Annual Meeting on Association for Computational Linguistics, ACL '04, Stroudsburg, PA, USA, 2004. Association for Computational Linguistics.

[Collins 2005] Michael Collins, Philipp Koehn and Ivona Kučerová. *Clause restructuring for statistical machine translation.* In Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, pages 531–540. Association for Computational Linguistics, 2005.

[Crammer 2006] Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz and Yoram Singer. *Online Passive-Aggressive Algorithms.* J. Mach. Learn. Res., vol. 7, pages 551–585, December 2006.

[Crego 2009] Josep M Crego and François Yvon. *Gappy Translation Units under Left-to-Right SMT Decoding.* In 13th Annual Conference of the European Association for Machine Translation, page 66, Barcelona, Spain, 2009.

[Daumé 2005] Hal Daumé III and Daniel Marcu. *Learning As Search Optimization: Approximate Large Margin Methods for Structured Prediction.* In Proceedings of the 22Nd International Conference on Machine Learning, ICML '05, pages 169–176, New York, NY, USA, 2005. ACM.

[DeNero 2009] John DeNero, Mohit Bansal, Adam Pauls and Dan Klein. *Efficient Parsing for Transducer Grammars.* In Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pages 227–235, Boulder, Colorado, June 2009. Association for Computational Linguistics.

[DeNero 2011] John DeNero and Jakob Uszkoreit. *Inducing sentence structure from parallel corpora for reordering.* In Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '11, pages 193–203, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[Ding 2005] Yuan Ding and Martha Palmer. *Machine Translation Using Probabilistic Synchronous Dependency Insertion Grammars.* In Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05), pages 541–548, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics.

[Doddington 2002] George Doddington. *Automatic evaluation of machine translation quality using n-gramco-occurrence statistics.* In Proceedings of the second international conference on Human Language Technology Research, pages 138–145. Morgan Kaufmann Publishers Inc., 2002.

[Dorr 1994] Bonnie J. Dorr. *Machine translation divergences: a formal description and proposed solution.* Comput. Linguist., vol. 20, no. 4, pages 597–633, December 1994.

[Dunlop 2011] Aaron Dunlop, Nathan Bodenstab and Brian Roark. *Efficient matrix-encoded grammars and low latency parallelization strategies for CYK.* In Proceedings of the 12th International Conference on Parsing Technologies, IWPT '11, pages 163–174, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[Eisner 1996] Jason M. Eisner. *Three New Probabilistic Models for Dependency Parsing: An Exploration.* In Proceedings of the 16th Conference on Computational Linguistics - Volume 1, COLING '96, pages 340–345, Stroudsburg, PA, USA, 1996. Association for Computational Linguistics.

[Eisner 2003] Jason Eisner. *Learning Non-Isomorphic Tree Mappings for Machine Translation.* In The Companion Volume to the Proceedings of 41st Annual Meeting of the Association for Computational Linguistics, pages 205–208, Sapporo, Japan, July 2003. Association for Computational Linguistics.

[Fan 2008] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang and Chih-Jen Lin. *LIBLINEAR: A library for large linear classification.* The Journal of Machine Learning Research, vol. 9, pages 1871–1874, 2008.

[Galley 2006] Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang and Ignacio Thayer. *Scalable Inference and Training of Context-Rich Syntactic Translation Models.* In Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, pages 961–968, Sydney, Australia, July 2006. Association for Computational Linguistics.

[Galley 2008] Michel Galley and Christopher D Manning. *A simple and effective hierarchical phrase reordering model.* In Proceedings of the Conference on Empirical Methods in Natural Language Processing, pages 848–856. Association for Computational Linguistics, 2008.

[Galley 2010] Michel Galley and Christopher D. Manning. *Accurate Non-hierarchical Phrase-based Translation.* In Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, HLT '10, pages 966–974, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[Gao 2008] Qin Gao and Stephan Vogel. *Parallel implementations of word alignment tool.* In Software Engineering, Testing, and Quality Assurance for Natural Language Processing, SETQA-NLP '08, pages 49–57, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[Genzel 2010] Dmitriy Genzel. *Automatically learning source-side reordering rules for large scale machine translation.* In Proceedings of the 23rd International Conference on Computational Linguistics, pages 376–384. Association for Computational Linguistics, 2010.

[Goto 2013] Isao Goto, Masao Utiyama, Eiichiro Sumita, Akihiro Tamura and Sadao Kurohashi. *Distortion Model Considering Rich Context for Statistical Machine Translation.* In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 155–165, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

[Graca 2008] Joao Graca, Joana Paulo Pardal, Luísa Coheur and Diamantino Caseiro. *Building a Golden Collection of Parallel Multi-Language Word Alignment.* In LREC, 2008.

[Green 2010] Spence Green, Michel Galley and Christopher D. Manning. *Improved Models of Distortion Cost for Statistical Machine Translation.* In Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pages 867–875, Los Angeles, California, June 2010. Association for Computational Linguistics.

[Hall 2008] Johan Hall. *Transition-based natural language parsing with dependency and constituency representations.* PhD thesis, Växjö University, 2008.

[Huang 2005] Liang Huang and David Chiang. *Better k-best parsing.* In Proceedings of the Ninth International Workshop on Parsing Technology, Parsing '05, pages 53–64, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

[Huang 2006] Liang Huang. *Statistical syntax-directed translation with extended domain of locality.* In In Proc. AMTA 2006, pages 66–73, 2006.

[Huang 2009] Liang Huang, Hao Zhang, Daniel Gildea and Kevin Knight. *Binarization of synchronous context-free grammars.* Comput. Linguist., vol. 35, no. 4, pages 559–595, December 2009.

[Huang 2010] Liang Huang and Haitao Mi. *Efficient incremental decoding for tree-to-string translation.* In Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, EMNLP '10, pages 273–283, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[Huang 2012] Liang Huang, Suphan Fayong and Yang Guo. *Structured Perceptron with Inexact Search.* In Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL HLT '12, pages 142–151, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

[Isozaki 2010] Hideki Isozaki, Katsuhito Sudoh, Hajime Tsukada and Kevin Duh. *Head finalization: A simple reordering rule for sov languages.* In Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR, pages 244–251. Association for Computational Linguistics, 2010.

[Khapra 2013] Mitesh M. Khapra, Ananthakrishnan Ramanathan and Karthik Visweswariah. *Improving reordering performance using higher order and structural features.* In Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 315–324, Atlanta, Georgia, June 2013. Association for Computational Linguistics.

[Knight 1999] Kevin Knight. *Decoding complexity in word-replacement translation models.* Computational Linguistics, vol. 25, no. 4, pages 607–615, 1999.

[Koehn 2003] Philipp Koehn, Franz Josef Och and Daniel Marcu. *Statistical phrase-based translation.* In Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1, pages 48–54. Association for Computational Linguistics, 2003.

[Koehn 2007] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens*et al. Moses: Open source toolkit for statistical machine translation.* In Annual meeting-association for computational linguistics, volume 45, page 2, 2007.

[Kruijff-Korbayová 2006] Ivana Kruijff-Korbayová, Klára Chvátalová and Oana Postolache. *Annotation guidelines for Czech-English word alignment.* In Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006), pages 1256–1261. Citeseer, 2006.

[Kudo 2000] Taku Kudo and Yuji Matsumoto. *Japanese Dependency Structure Analysis Based on Support Vector Machines.* In Proceedings of the 2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora: Held in Conjunction with the 38th Annual Meeting of the Association for Computational Linguistics - Volume 13, EMNLP '00, pages 18–25, Stroudsburg, PA, USA, 2000. Association for Computational Linguistics.

[Kudo 2002] Taku Kudo and Yuji Matsumoto. *Japanese Dependency Analysis Using Cascaded Chunking.* In Proceedings of the 6th Conference on Natural Language Learning - Volume 20, COLING-02, pages 1–7, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[McDonald 2005a] Ryan McDonald, Koby Crammer and Fernando Pereira. *Online Large-margin Training of Dependency Parsers.* In Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, ACL '05, pages 91–98, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

[McDonald 2005b] Ryan McDonald, Fernando Pereira, Kiril Ribarov and Jan Hajič. *Non-projective Dependency Parsing Using Spanning Tree Algorithms.* In Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing, HLT '05, pages 523–530, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

[McDonald 2006a] Ryan McDonald. *Discriminative learning and spanning tree algorithms for dependency parsing.* PhD thesis, University of Pennsylvania, 2006.

[McDonald 2006b] Ryan McDonald and Fernando Pereira. *Online learning of approximate dependency parsing algorithms.* In Proceedings of 11th Conference of the European Chapter of the Associationfor Computational Linguistics (EACL-2006)), volume 6, pages 81–88, 2006.

[Mi 2008a] Haitao Mi and Liang Huang. *Forest-based Translation Rule Extraction.* In Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing, pages 206–214, Honolulu, Hawaii, October 2008. Association for Computational Linguistics.

[Mi 2008b] Haitao Mi, Liang Huang and Qun Liu. *Forest-Based Translation.* In Proceedings of ACL-08: HLT, pages 192–199, Columbus, Ohio, June 2008. Association for Computational Linguistics.

[Mi 2010] Haitao Mi, Liang Huang and Qun Liu. *Machine Translation with Lattices and Forests.* In Coling 2010: Posters, pages 837–845, Beijing, China, August 2010. Coling 2010 Organizing Committee.

[Na 2013] Hwidong Na and Jong-Hyeok Lee. *A Discriminative Reordering Parser for IWSLT 2013.* In Proceedings of the tenth International Workshop on Spoken Language Trans lation (IWSLT), 2013.

[Na 2014] Hwidong Na and Jong-Hyeok Lee. *Linguistic Analysis of Non-ITG Word Reordering between Language Pairs with Different Word Order Typologies.* ACM Transactions on Asian Language Information Processing, vol. 3, no. 13, pages 11:1–11:12, 2014.

[Neubig 2011a] Graham Neubig. *The Kyoto Free Translation Task.* http://www.phontron.com/kftt, 2011.

[Neubig 2011b] Graham Neubig, Yosuke Nakata and Shinsuke Mori. *Pointwise Prediction for Robust, Adaptable Japanese Morphological Analysis.* In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, pages 529–533, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

[Neubig 2012] Graham Neubig, Taro Watanabe and Shinsuke Mori. *Inducing a discriminative parser to optimize machine translation reordering.* In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL '12, pages 843–853, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

[Nivre 2003] Joakim Nivre. *An Efficient Algorithm for Projective Dependency Parsing.* In Proceedings of the 8th International Workshop on Parsing Technologies (IWPT). Citeseer, 2003.

[Nivre 2006] Joakim Nivre, Johan Hall and Jens Nilsson. *MaltParser: A data-driven parser-generator for dependency parsing.* In In Proc. of LREC-2006, pages 2216–2219, 2006.

[Nivre 2008] Joakim Nivre. *Algorithms for Deterministic Incremental Dependency Parsing.* Comput. Linguist., vol. 34, no. 4, pages 513–553, December 2008.

[Nivre 2009] Joakim Nivre. *Non-Projective Dependency Parsing in Expected Linear Time.* In Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP, pages 351–359, Suntec, Singapore, August 2009. Association for Computational Linguistics.

[Och 2003a] Franz Josef Och. *Minimum error rate training in statistical machine translation.* In Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1, ACL '03, pages 160–167, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

[Och 2003b] Franz Josef Och and Hermann Ney. *A systematic comparison of various statistical alignment models.* Computational linguistics, vol. 29, no. 1, pages 19–51, 2003.

[Och 2004] Franz Josef Och and Hermann Ney. *The alignment template approach to statistical machine translation.* Computational Linguistics, vol. 30, no. 4, pages 417–449, 2004.

[Papineni 2002] Kishore Papineni, Salim Roukos, Todd Ward and Wei-Jing Zhu. *Bleu: a Method for Automatic Evaluation of Machine Translation.* In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL), pages pp. 311–318., Philadelphia, July 2002.

[Phan 2006] Xuan-Hieu Phan. *CRFChunker: CRF English Phrase Chunker*, 2006.

[Shen 2008] Libin Shen, Jinxi Xu and Ralph Weischedel. *A New String-to-Dependency Machine Translation Algorithm with a Target Dependency Language Model.* In Proceedings of ACL-08: HLT, pages 577–585, Columbus, Ohio, June 2008. Association for Computational Linguistics.

[Simard 2005] Michel Simard, Nicola Cancedda, Bruno Cavestro, Marc Dymetman, Eric Gaussier, Cyril Goutte, Kenji Yamada, Philippe Langlais and Arne Mauser. *Translating with Non-contiguous Phrases.* In Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing, HLT '05, pages 755–762, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

[Snover 2006] Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla and John Makhoul. *A study of translation edit rate with targeted human annotation.* In In Proceedings of Association for Machine Translation in the Americas, pages 223–231, 2006.

[Søgaard 2009a] Anders Søgaard and Jonas Kuhn. *Empirical Lower Bounds on Alignment Error Rates in Syntax-Based Machine Translation.* In Proceedings of the Third Workshop on Syntax and Structure in Statistical Translation (SSST-3) at NAACL HLT 2009, pages 19–27, Boulder, Colorado, June 2009. Association for Computational Linguistics.

[Søgaard 2009b] Anders Søgaard and Dekai Wu. *Empirical lower bounds on translation unit error rate for the full class of inversion transduction grammars.* In Proceedings of the 11th International Conference on Parsing Technologies, IWPT '09, pages 33–36, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[Tillmann 2004] Christoph Tillmann. *A unigram orientation model for statistical machine translation.* In Proceedings of HLT-NAACL 2004: Short Papers, pages 101–104. Association for Computational Linguistics, 2004.

[Tromble 2009] Roy Tromble and Jason Eisner. *Learning linear ordering problems for better translation.* In Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2, pages 1007–1016. Association for Computational Linguistics, 2009.

[Tu 2010] Zhaopeng Tu, Yang Liu, Young-Sook Hwang, Qun Liu and Shouxun Lin. *Dependency Forest for Statistical Machine Translation.* In Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 1092–1100, Beijing, China, August 2010. Coling 2010 Organizing Committee.

[Visweswariah 2011] Karthik Visweswariah, Rajakrishnan Rajkumar, Ankur Gandhe, Ananthakrishnan Ramanathan and Jiri Navratil. *A Word Reordering Model for Improved Machine Translation.* In Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '11, pages 486–496, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[Watanabe 2007] Taro Watanabe, Jun Suzuki, Hajime Tsukada and Hideki Isozaki. *Online Large-Margin Training for Statistical Machine Translation.* In Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), pages 764–773, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[Waxmonsky 2006] Sonjia Waxmonsky and I Dan Melamed. *A dynamic data structure for parsing with discontinuous constituents.* NYU Proteus Project Technical Report, pages 06–001, 2006.

[Wellington 2006] Benjamin Wellington, Sonjia Waxmonsky and I. Dan Melamed. *Empirical lower bounds on the complexity of translational equivalence.* In Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics, ACL-44, pages 977–984, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics.

[Wu 1997] Dekai Wu. *Stochastic inversion transduction grammars and bilingual parsing of parallel corpora.* Computational linguistics, vol. 23, no. 3, pages 377–403, 1997.

[Wu 2006] Dekai Wu, Marine Carpuat and Yihai Shen. *Inversion transduction grammar coverage of arabic-english word alignment for tree-structured statistical machine translation.* In In Proceedings of the IEEE/ACL Workshop on Spoken Language Technology, 2006.

[Xia 2004] Fei Xia and Michael McCord. *Improving a statistical MT system with automatically learned rewrite patterns.* In Proceedings of the 20th international conference on Computational Linguistics, page 508. Association for Computational Linguistics, 2004.

[Xie 2011] Jun Xie, Haitao Mi and Qun Liu. *A novel dependency-to-string model for statistical machine translation.* In Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, pages 216–226, Edinburgh, Scotland, UK., July 2011. Association for Computational Linguistics.

[Xiong 2007] Deyi Xiong, Qun Liu and Shouxun Lin. *A dependency treelet string correspondence model for statistical machine translation.* In Proceedings of the Second Workshop on Statistical Machine Translation, StatMT '07, pages 40–47, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics.

[Xiong 2010a] Deyi Xiong, Min Zhang, Aiti Aw and Haizhou Li. *Linguistically Annotated Reordering: Evaluation and Analysis.* Comput. Linguist., vol. 36, no. 3, pages 535–568, September 2010.

[Xiong 2010b] Deyi Xiong, Min Zhang and Haizhou Li. *Learning translation boundaries for phrase-based decoding.* In Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pages 136–144. Association for Computational Linguistics, 2010.

[Xu 2009] Peng Xu, Jaeho Kang, Michael Ringgaard and Franz Och. *Using a dependency parser to improve SMT for subject-object-verb languages.* In Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pages 245–253. Association for Computational Linguistics, 2009.

[Yamada 2003] Hiroyasu Yamada and Yuji Matsumoto. *Statistical dependency analysis with support vector machines.* In Proceedings of IWPT, volume 3, 2003.

[Yu 2013] Heng Yu, Liang Huang, Haitao Mi and Kai Zhao. *Max-Violation Perceptron and Forced Decoding for Scalable MT Training.* In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pages 1112–1123, Seattle, Washington, USA, October 2013. Association for Computational Linguistics.

[Zaidan 2010] Omar F. Zaidan and Chris Callison-Burch. *Predicting human-targeted translation edit rate via untrained human annotators.* In Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, HLT '10, pages 369–372, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[Zaslavskiy 2009] Mikhail Zaslavskiy, Marc Dymetman and Nicola Cancedda. *Phrase-based Statistical Machine Translation As a Traveling Salesman Problem.* In Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1 - Volume 1, ACL '09, pages 333–341, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[Zens 2003] Richard Zens and Hermann Ney. *A Comparative Study on Reordering Constraints in Statistical Machine Translation.* In Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1, ACL '03, pages 144–151, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

[Zens 2004] Richard Zens, Hermann Ney, Taro Watanabe and Eiichiro Sumita. *Reordering Constraints for Phrase-based Statistical Machine Translation.* In Proceedings of the 20th International Conference on Computational Linguistics, COLING '04, Stroudsburg, PA, USA, 2004. Association for Computational Linguistics.

[Zhang 2006] Hao Zhang, Liang Huang, Daniel Gildea and Kevin Knight. *Synchronous Binarization for Machine Translation.* In Proceedings of the Human Language Technology Conference of the NAACL, Main Conference, pages 256–263, New York City, USA, June 2006. Association for Computational Linguistics.

[Zhang 2007] Yuqi Zhang, Richard Zens and 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, pages 1–8, 2007.

[Zhang 2008] Hao Zhang, Daniel Gildea and David Chiang. *Extracting synchronous grammar rules from word-level alignments in linear time.* In 22nd International Conference on Computational Linguistics (Coling), pages 1081–1088, 2008.

[Zhang 2009] Hui Zhang, Min Zhang, Haizhou Li, Aiti Aw and Chew Lim Tan. *Forest-based Tree Sequence to String Translation Model.* In Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP, pages 172–180, Suntec, Singapore, August 2009. Association for Computational Linguistics.

[Zhang 2010] Hui Zhang, Min Zhang, Haizhou Li and Eng Siong Chng. *Non-Isomorphic Forest Pair Translation.* In Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, pages 440–450, Cambridge, MA, October 2010. Association for Computational Linguistics.

[Zhang 2013] Hao Zhang, Liang Huang, Kai Zhao and Ryan McDonald. *Online Learning for Inexact Hypergraph Search.* In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pages 908–913, Seattle, Washington, USA, October 2013. Association for Computational Linguistics.

[Zhao 2014] Kai Zhao, Liang Huang, Haitao Mi and Abe Ittycheriah. *Hierarchical MT Training using Max-Violation Perceptron.* In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2:

Short Papers), pages 785–790, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

Acknowledgements
감사의 글

감사의 글을 적고 있으니 정든 포항을 떠나게 됨을 또 한 번 실감하게 됩니다. 우선 여기까지 저를 올 수 있도록 지도해주신 교수님께 깊은 감사의 마음을 드립니다. 짧지 않은 시간 동안 때로는 엄격한 모습으로 때로는 다정한 모습으로 가르쳐주시고 다독여주신 교수님의 지도 덕분에 제가 원만히 학업을 마칠 수 있었습니다. 더불어 귀한 시간 내 주셔서 논문 심사를 맡아주신 김치하 교수님, 이근배 교수님, 신효필 교수님, 황승원 교수님께도 감사드립니다.

포항이란 낯선 도시에 적응하는데 큰 힘이 된 대학원 동기, 후배들에게도 많은 빚을 졌습니다. 즐거울 때도 힘들 때도 곁에서 함께한 이들이 없었다면 저 혼자 버텨내기는 어려웠을 것입니다. 비록 연구실은 달랐지만 잘 챙겨주신 제영이형, 가끔씩 함께 술잔을 기울일 수 있었던 현목이형, 지금까지도 모임을 이끌어 오는 구심점인 기영이형, 위트있는데다 훈남이기까지한 정욱이형, 학부 때부터 대학원까지 함께한 상협이형, 끊임없이 도전하는 모습이 아름다운 유석이형, 주변 사람을 촘촘히 엮어주는 연결고리 희진이, 부끄러워하지만 활화산 같은 끼를 품고 있는 건혁이, 카리스마 넘치는 대장님 지혜, 노래방으로 함께하던 순기와 더불어 오랜 시간 룸메이트였던 상섭이형, 맛과 멋을 즐길 줄 아는 태훈이, 알게 모르게 동문회의 시초가 된 아영이, 조금 친해질 때쯤 되니 헤어지게 된 진하, 항상 웃는 모습으로 기억에 남는 룸메이트 국보에게 모두 감사를 전합니다.

연구실에서 매일 마주치며 함께한 KLE 구성원들에게도 감사합니다. 선배들에게는 철 없는 후배가 아니었을지, 후배들에게는 좀 더 잘 챙겨주지 못한 선배가 아니었을지 괜시리 돌아보게 됩니다. 신입생일 때부터 항상 신세만 지게 되는 미훈누나, 연구방향을 이끌어주고 많은 도움을 주었던 금희누나, 무슨 일이든 해내고 마는 연구실 일꾼

용훈이형, 졸업 후에 오히려 더 많은 도움을 받는 승훈이형, 정말 많은 시간을 함께해 든든한 힘이 된 예하형, 꼼꼼하고 성실한 자세로 연구실의 정신적 지주가 되어준 세종이형, 운동도 좋아하고 밝은 성격에서 많이 배운 성국이형, 독특하면서도 동질감을 느낄 수 있었던 한경이형, 실력과 노력을 겸비한 능력자 준기형, 언젠가 크게 성공할 거라 믿어 의심치 않는 우상이형, 함께 축구도 많이하고 이야기도 많이 나누던 장호형, 한결같이 긍정적인 성격이 인상깊은 헌영이, 도전적이고 창의적인 승부사 기영이, 낯가림만 지나면 엄청난 친화력을 보여주는 CK, 어느새 연구실 중견으로 성장한 건일이, 한국사람보다 더 한국사람 같은 싹싹한 영영, 언제나 기대를 충족시켜주는 만물박사 재훈이, 뛰어난 실력과 겸손한 자세를 갖춘 종구, Vinh, Zhen, Chow, and Girma, thank you for being my friend and colleague. 연구실 생활이라는 테두리를 벗어나 이제 사회초년생으로 제가 모자랐던 부분을 반면교사하는 마음을 다져봅니다.

초등학교 때부터 이어져온 친구인 동호, 재인, 강희에게는 고마움을 어떻게 표현해야 할 지 모르겠습니다. 항상 그 자리에 있어주는 친구들이 있었기에 저 역시 제자리에서 제가 할 몫을 감당할 수 있었습니다. 먼길을 마다하지 않고 함께 해주기도 했었고 또다른 시각에서 중요한 조언을 얻을 수 있었습니다. 그동안 함께한 시간이 오랜만큼 앞으로 더 많은 시간을 함께하고 제가 받은 것을 돌려주고 싶습니다. 마지막으로 사랑하는 가족들에게 이 모든 영광을 돌리고 싶습니다. 타지에 홀로 보낸 아들 걱정에 많이 애태우셨을 부모님과 하나뿐인 형인데도 많이 아껴주지 못해 동생에게 고맙고 미안합니다. 조금 더 많이 챙기고 사랑하며 추억을 만들어나가는 모습을 오래도록 보여드리고 싶습니다. 사랑합니다.

Curriculum Vitae

Education

2003. 3. - 2007. 2. Computer Science and Engineering, Chung-Ang University, (B.S.)

2007. 3. - 2015. 2. Computer Science and Engineering, Pohang University of Science and Technology (Ph.D.)

Experience

Spring 2007 Teaching Assistant, Compiler, Computer Science and Engineering, Pohang University of Science and Technology

Spring 2008 Teaching Assistant, Programming Language, Computer Science and Engineering, Pohang University of Science and Technology

Spring 2011 Teaching Assistant, Introduction to Programming, Computer Science and Engineering, Pohang University of Science and Technology

Publication

International Journal

Hwidong Na and Jong-Hyeok Lee, “Linguistic Analysis of Non-ITG Word Reordering between Language Pairs with Different Word Order Typologies”, TALIP, Vol.3 (13), Article 11, 2014

Domestic Journal

나휘동, 이종혁, “통계기반 기계번역을 위한 어순 조정: 비투사 의존구문분석 기법의 활용”, 정보과학회 논문지: 소프트웨어 및 응용, Vol. 40 (11), pp.657-662, 2013

김한경, 나휘동, 이금희, 이종혁, “문장구조 유사도와 단어 유사도를 이용한 클러스터링 기반의 통계기계번역”, 정보과학회논문지: 소프트웨어 및 응용, Vol. 37 (4), pp.297-304, 2010

김한경, 나휘동, 이금희, 이종혁, “일영 통계기계번역에서 의존문법 문장 구조와 품사 정보를 사용한 클러스터링 기법”, 정보과학회논문지 : 컴퓨팅의 실제 및 레터 제15권 제12호, Vol. 15 (12), pp. 993-997, 2009

International Conference

Jian-Ri Li, Se-Jong Kim, Hwidong Na and Jong-Hyeok Lee, “Postech’s System Description for Medical Text Translation Task”, WMT 2014, pp. 229–232, 2014

Hwidong Na, Jong-Hyeok Lee, “A Discriminative Reordering Parser for IWSLT 2013”, IWSLT 2013, pp. 83-86, 2013

Hwidong Na, and Jong-Hyeok Lee, “Forest-to-String Translation using Binarized Dependency Forest for IWSLT 2012 OLYMPICS Task”, IWSLT 2012, pp. 130-135, 2012

Mei-Xun Jin, Hwidong Na and Jong-Hyeok Lee, “Beyond Chart Parsing: An Analytic Comparison of Dependency Chart Parsing Algorithms”, IWPT 2011, pp.220-224, 2011

Hwidong Na, Jin-Ji Li, Se-Jong Kim and Jong-Hyeok Lee, “POSTECH’s Statistical Machine Translation Systems for NTCIR-9 PatentMT Task (English-to-Japanese)”, NTCIR-9, pp. 652-656, 2011

Se-Jong Kim, Hwidong Na, and Jong-Hyeok Lee, “The KLE’s Subtopic Mining System for the NTCIR-9 INTENT Task”, NTCIR-9, pp. 135-137, 2011

Hwidong Na, and Jong-Hyeok Lee, “Multi-word unit dependency forest-based translation rule extraction”, ACL 2011 workshop SSST-5, pp.41-51, 2011

Hwidong Na, Jin-Ji Li, Yeha Lee and Jong-Hyeok Lee “A Synchronous Context Free Grammar using Dependency Sequence for Syntax-base Statistical Machine Translation”, AMTA Studuent Workshop, 2010

Hwidong Na, and Jong-Hyeok Lee, “The POSTECH’s Statistical Machine Translation Systems for the IWSLT 2010”, IWSLT 2010, pp.153-156, 2010

Hwidong Na, Hankyoung Kim and Jong-Hyeok Lee, “The POSTECH’s Statistical Machine Translation Systems for the NTCIR-8 Patent Translation Task”, NTCIR-8, pp.397-402, 2010

Hwidong Na, Jin-Ji Li, Jungi Kim and Jong-Hyeok Lee “Improving Fluency by Reordering Target Constituents using MST Parser in English-to-Japanese phrase-based SMT” ,MT Summit XII, 2009

Jin-Ji Li, Hwi-Dong Na, Hankyong Kim, Chang-Hu Jin, Jong-Hyeok Lee, “The POSTECH Statistical Machine Translation Systems for NTCIR-7 Patent Translation Task”, NTCIR-7, pp.445-449, 2008

Domestic Conference

오진, 김미훈, 나휘동, 이종혁, “Chinese Dependency Parsing With Heterogeneous Part-Of-Speech Annotations”, 한국정보과학회, pp. 625-627, 2013

김장호, 이금희, 나휘동, 김동일, 이종혁, "통계적 수정규칙을 이용한 한국어-중국어 단어정렬 개선방법", 한글 및 한국어 정보처리 학술대회(HCLT), 논문집 Vol. 36(1-A), pp. 88-89 , 2009

김장호, 이금희, 나휘동, 이종혁, "한국어 형태소 유형에 따른 한국어-중국어 단어정렬 결과분석", 한국정보과학회, 학술발표논문집, Vol. 36(1-C), pp. 325-329, 2009

나휘동, 이금희, 이종혁, "일영 통계기계번역에서 먼 거리 의존 관계를 이용한 일본어 어순 조정", 한국정보과학회, 학술발표논문집 Vol. 35(1-A), pp. 76-77, 2008