Linguistically Annotated Reordering: Evaluation and Analysis

Deyi Xiong*
Institute for Infocomm Research

Min Zhang**
Institute for Infocomm Research

Aiti Aw†
Institute for Infocomm Research

Haizhou Li‡
Institute for Infocomm Research

Linguistic knowledge plays an important role in phrase movement in statistical machine translation. To efficiently incorporate linguistic knowledge into phrase reordering, we propose a new approach: Linguistically Annotated Reordering (LAR). In LAR, we build hard hierarchical skeletons and inject soft linguistic knowledge from source parse trees to nodes of hard skeletons during translation. The experimental results on large-scale training data show that LAR is comparable to boundary word-based reordering (BWR) (Xiong, Liu, and Lin 2006), which is a very competitive lexicalized reordering approach. When combined with BWR, LAR provides complementary information for phrase reordering, which collectively improves the BLEU score significantly.

To further understand the contribution of linguistic knowledge in LAR to phrase reordering, we introduce a syntax-based analysis method to automatically detect constituent movement in both reference and system translations, and summarize syntactic reordering patterns that are captured by reordering models. With the proposed analysis method, we conduct a comparative analysis that not only provides the insight into how linguistic knowledge affects phrase movement but also reveals new challenges in phrase reordering.

1. Introduction

The phrase-based approach is a widely accepted formalism in statistical machine translation (SMT). It segments the source sentence into a sequence of phrases (not necessarily syntactic phrases), then translates and reorders these phrases in the target. The reason for the popularity of phrasal SMT is its capability of non-compositional translations and

* 1 Fusionopolis Way #21-01 Connexis Singapore 138632. E-mail: dyxiong@i2r.a-star.edu.sg.
** 1 Fusionopolis Way #21-01 Connexis Singapore 138632. E-mail: mzhang@i2r.a-star.edu.sg.
† 1 Fusionopolis Way #21-01 Connexis Singapore 138632. E-mail: aaiti@i2r.a-star.edu.sg.
‡ 1 Fusionopolis Way #21-01 Connexis Singapore 138632. E-mail: hli@i2r.a-star.edu.sg.

Submission received: 24 October 2008; revised submission received: 12 March 2010; accepted for publication: 21 April 2010.

local word reorderings within phrases. Unfortunately, reordering at the phrase level is still problematic for phrasal SMT. The default distortion-based reordering model simply penalizes phrase movement according to the jump distance, without considering any linguistic contexts (morphological, lexical, or syntactic) around phrases.

In order to utilize lexical information for phrase reordering, Tillman (2004) and Koehn et al. (2005) propose lexicalized reordering models which directly condition phrase movement on phrases themselves. One problem with such lexicalized reordering models is that they are restricted only to reorderings of phrases seen in training data. To eliminate this restriction, Xiong, Liu, and Lin (2006) suggest using boundary words of phrases (i.e., leftmost/rightmost words of phrases), instead of phrases, as reordering evidence. Although these lexicalized reordering models significantly outperform the distortion-based reordering model as reported, only using lexical information (e.g., boundary words) is not adequate to move phrases to appropriate positions.

Consider the following Chinese example with its English translation:

> [VP [PP 在(while) 制定(develop) 有关(related) 法律(legislation) 时] [VP [VV 考虑(consider)] [NP [DNP [NP 这次(this) 全民公决(referendum)] [DEG 的(of)]] [NP 结果(results)]]]][1]
> consider the results of this referendum while developing related legislation

In this example, boundary words 在 and 时 are able to decide that the translation of the PP phrase 在...时 should be postponed until some phrase that succeeds it is translated. But they cannot provide further information about exactly which succeeding phrase should be translated first. If high-level linguistic knowledge, such as the syntactic context VP→PP VP, is given, the position of the PP phrase can be easily determined since the pre-verbal modifier PP in Chinese is frequently translated into a post-verbal counterpart in English.

In this article, we focus on linguistically motivated phrase reordering, which integrates high-level linguistic knowledge in phrase reordering. We adopt a two-step strategy. In the first step, we establish a hierarchical skeleton in phrasal SMT by incorporating Bracketing Transduction Grammar (BTG) (Wu 1997) into phrasal SMT. In the second step, we inject soft linguistic information into nodes of the skeleton.

There are two significant advantages to using BTG in phrasal SMT. First, BTG is able to generate hierarchical structures.[2] This not only enhances phrasal SMT's capability for hierarchical and long-distance reordering but also establishes a platform for phrasal SMT to incorporate knowledge from linguistic structure. Second, phrase reordering is restricted by the ITG constraint (Wu 1997). Although it only allows two orders (**straight** or **inverted**) of nodes in any binary branching structure, it is broadly verified that the ITG constraint has good coverage of word reorderings on various language pairs (Wu, Carpuat, and Shen 2006). This makes phrase reordering in phrasal SMT a more tractable task.

After enhancing phrasal SMT with a hard hierarchical skeleton, we further inject soft linguistic information into the nodes of the skeleton. We annotate each BTG node

1 In this article, we use Penn Chinese Treebank phrase labels (Xue et al. 2000).

2 Chiang (2005) also generates hierarchical structures in phrasal SMT. One difference is that Chiang's hierarchical grammar is lexicon-sensitive because the model requires at least one pair of aligned words in each rule except for the "glue rule." The other difference is that his grammar allows multiple nonterminals. These two differences make Chiang's grammar more expressive than the BTG but at the cost of learning a larger model.

with syntactic and lexical elements by projecting the source parse tree onto the BTG binary tree. The challenge, of course, is that BTG hierarchical structures are not always aligned with the linguistic structures in the source language parse tree. To address this issue, we propose an annotation algorithm. The algorithm is able to label any BTG nodes during decoding with very little overhead, regardless of whether the BTG nodes are aligned with syntactic constituent nodes in the source parse tree. The annotated linguistic elements are then used to guide phrase reordering under the ITG constraint.

We call this two-step phrase reordering strategy **linguistically annotated reordering (LAR)** (Xiong et al. 2008a). Xiong, Liu, and Lin (2006) also adapt a two-step reordering strategy based on BTG. However, they use boundary words as reordering features at the second step. To distinguish this from our work, we call their approach **boundary word–based reordering (BWR)**. LAR and BWR can be considered as two reordering variants for BTG-based phrasal SMT, which have similar training procedures. Furthermore, they can be combined.

We evaluate LAR vs. BWR using the automatic metric BLEU (Papineni et al. 2002). The BLEU scores show that LAR is comparable to BWR and significantly improves phrase reordering when combined with BWR.

We want to further study what happens when we combine BWR with LAR. In particular, we want to investigate to what extent the integrated linguistic knowledge (from LAR) changes phrase movement in an actual SMT system, and in what direction the change takes place. The investigations will enable us to have a better understanding of the relationship between phrase movement and linguistic context, and therefore to explore linguistic knowledge more effectively in phrasal SMT.

Because syntactic constituents are often moved together across languages during translation (Fox 2002), we particularly study how linguistic knowledge affects syntactic constituent movement. To that end, we introduce a syntax-based analysis method. We parse source sentences, and align the parse trees with reference translations as well as system translations. We then summarize syntactic reordering patterns using context-free grammar (CFG) rules from the obtained tree-to-string alignments. The extracted reordering patterns clearly show the trace of syntactic constituent movement in both reference translations and system translations.

With the proposed analysis method, we analyze the combination of BWR and LAR vs. BWR alone. There are essentially three issues that are addressed in this syntax-based comparative analysis.

1. The first issue concerns syntactic constituent movement in human/machine translations. Fox (2002) investigates syntactic constituent movement in human translations. We study syntactic constituent movement in both human translations and machine translations that are generated by an actual SMT system and compare them.

2. The second issue concerns the change of phrase movement after rich linguistic knowledge is integrated into phrase reordering. To gain a better insight into this issue, we study phrase movement patterns for 13 specific syntactic constituents.

3. The last issue concerns which constituents remain difficult to reorder even though rich linguistic knowledge is employed.

The rest of the article is structured as follows. Section 2 introduces background information about BTG-based phrasal SMT and phrase reordering under the ITG

constraint. Section 3 describes algorithms which extract training instances for reordering models of BWR and LAR. Section 4 introduces the BWR model as our baseline reordering model. Section 5 describes LAR and the combination of LAR with BWR. Section 6 elaborates the syntax-based analysis method. Section 7 reports our evaluation results on large-scale data. Section 8 demonstrates our analysis results and addresses the various issues discussed above. Section 9 discusses related work. And finally, Section 10 summarizes our main conclusions.

2. Background

2.1 BTG-Based Phrasal SMT

We establish a unified framework for BTG-based phrasal SMT in this section. There are two kinds of rules in BTG, lexical rules (denoted as r^l) and merging rules (denoted as r^m):[3]

$$\begin{aligned} r^l &: A_p \rightarrow x/y \\ r^m &: A_p \rightarrow [A_l, A_r] | \langle A_l, A_r \rangle \end{aligned} \tag{1}$$

A lexical rule translates a source phrase x into a target phrase y and generates a leaf node A in the BTG tree. Merging rules combine left and right neighboring phrases A_l and A_r into a larger phrase A_p in an order $o \in \{straight, inverted\}$. In this article, we use "[]" to denote a straight order and "$\langle\rangle$" an inverted order.

We define a BTG derivation D as a sequence of independent applications of lexical and merging rules ($D = \langle r^l_{1..n_l}, r^m_{1..n_m} \rangle$). Given a source sentence, the decoding task of BTG-based SMT is to find a best derivation, which yields the best translation.[4]

We assign a probability to each rule using a log-linear model with different features and corresponding weights λ, then multiply them to obtain $P(D)$. To keep in line with the common understanding of standard phrasal SMT (Koehn, Och, and Marcu 2003), here we re-organize these features into a translation model (P_T), a reordering model (P_R), and a target language model (P_L) as follows:

$$P(D) = P_T(r^l_{1..n_l}) \cdot P_R(r^m_{1..n_m})^{\lambda_R} \cdot P_L(e)^{\lambda_L} \cdot exp(|e|)^{\lambda_w} \tag{2}$$

where $exp(|e|)$ is the word penalty.

The translation model is defined as

$$\begin{aligned} P_T(r^l_{1..n_l}) &= \prod_{i=1}^{n_l} P(r^l_i) \\ P(r^l) &= p(x|y)^{\lambda_1} \cdot p(y|x)^{\lambda_2} \cdot p_{lex}(x|y)^{\lambda_3} \cdot p_{lex}(y|x)^{\lambda_4} \cdot exp(1)^{\lambda_5} \end{aligned} \tag{3}$$

where $p(\cdot)$ represents the phrase translation probabilities in both directions, $p_{lex}(\cdot)$ denotes the lexical translation probabilities in both directions, and $exp(1)$ is the phrase penalty.

3 The subscripts l, r, p of A do not mean that we categorize A into three different nonterminals. We use them to represent the left node, right node, and parent node.

4 In this article, we use c to denote a source sentence and e a target sentence.

Similarly, the reordering model is defined on the merging rules as follows:

$$P_R(r^m_{1..n_m}) = \prod_{i=1}^{n_m} P(r^m_i) \tag{4}$$

One of the most important and challenging tasks in building a BTG-based phrasal SMT system is to define $P(r^m)$.

2.2 Reordering Under the ITG Constraint

Under the ITG constraint, three nodes $\{A_l, A_r, A_p\}$ are involved when we consider the order o between the two children $\{A_l, A_r\}$ in any binary subtrees. Therefore it is natural to define the ITG reordering $P(r^m)$ as a function as follows:

$$P(r^m) = f(A_l, A_r, A_p, o) \tag{5}$$

where $o \in \{straight, inverted\}$.

Based on this function, various reordering models are built according to different assumptions. For example, the flat reordering model in the original BTG (Wu 1996) assigns prior probabilities for the straight and inverted order assuming the order is highly related to the properties of language pairs. It is formulated as

$$P(r^m) = \begin{cases} p_s, & o = straight \\ 1 - p_s, & o = inverted \end{cases} \tag{6}$$

Supposing French and English are the source and target language, respectively, the value of p_s can be set as high as 0.8 to prefer monotone orientations because the two languages have similar word orders in most cases.

The main problem of the flat reordering model is also the problem of the standard distortion model (Koehn, Och, and Marcu 2003): Neither model considers linguistic contexts. To be context-dependent, the ITG reordering might directly model the conditional probability $P(o|A_l, A_r)$. This probability could be calculated using the maximum likelihood estimate (MLE) by taking counts from training data, in the manner of the lexicalized reordering model (Tillman 2004; Koehn et al. 2005):

$$P(o|A_l, A_r) = \frac{Count(o, A_l, A_r)}{Count(A_l, A_r)} \tag{7}$$

Unfortunately this lexicalized reordering method usually leads to a serious data sparseness problem under the ITG constraint because A_l and A_r become larger and larger due to the merging rules, and are finally unseen in the training data.

To avoid the data sparseness problem yet be contextually informative, attributes of A_l and A_r, instead of nodes themselves, are used as reordering evidence in a new perspective of the ITG reordering (Xiong, Liu, and Lin 2006). The new perspective treats the ITG reordering as a binary-classification problem where the possible order *straight* or *inverted* between two children nodes is the target class which the reordering model predicts given A_l, A_r, and A_p.

3. Reordering Example Extraction

Because we consider the ITG reordering as a classification problem, we need to obtain training instances to build a classifier. Here we refer to a training instance as a **reordering example**, which is formally defined as a triple of (o, b_l, b_r) where b_l and b_r are two neighboring blocks and $o \in \{straight, inverted\}$ is the order between them.

The **block** is a pair of aligned source phrase and target phrase

$$b = (c_{i_1}^{i_2}, e_{j_1}^{j_2}) \tag{8}$$

where b must be consistent with the word alignment M

$$\forall (i,j) \in M, i_1 \leq i \leq i_2 \leftrightarrow j_1 \leq j \leq j_2 \tag{9}$$

By this, we require that no words inside the source phrase $c_{i_1}^{i_2}$ are aligned to words outside the target phrase $e_{j_1}^{j_2}$ and that no words outside the source phrase are aligned to words inside the target phrase. This definition is similar to that of the bilingual phrase except that there is no length limitation over blocks. Figure 1 shows a word alignment matrix between a Chinese sentence and English sentence. In the matrix, each block can be represented as a rectangle, for example, blocks (c_4^4, e_4^4), (c_4^5, e_4^5), (c_4^7, e_4^9) in red rectangles, and (c_2^3, e_3^3), (c_1^3, e_1^3) in blue rectangles.

In this section, we discuss two algorithms for extracting reordering examples from word-aligned bilingual data. The first algorithm **AExtractor** (described in Section 3.1) extracts reordering examples directly from word alignments by extending the bilingual phrase extraction algorithm. The second algorithm **TExtractor** (described in Section 3.2) extracts reordering examples from BTG-style trees which are built from word alignments.

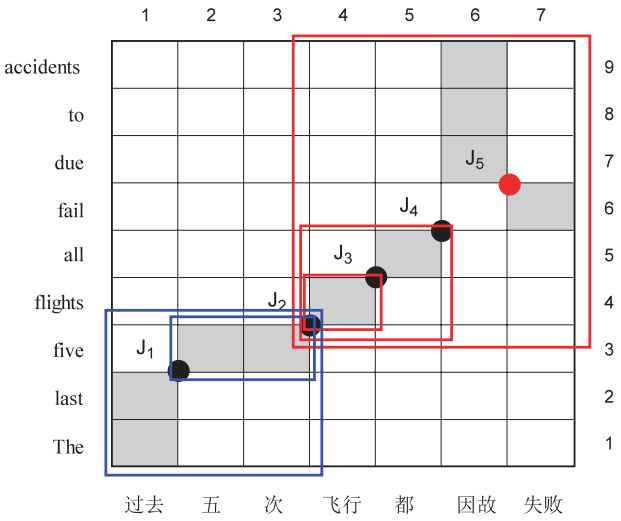

Figure 1
A word alignment matrix between a Chinese sentence and English sentence. Bold dots represent *junctions* which connect two neighboring blocks. Red and blue rectangles are blocks which are connected by junction J_2.

3.1 AExtractor: Extracting Reordering Examples from Word Alignments

Before we describe this algorithm, we introduce the concept of **junction** in the word alignment matrix. We define a junction as a vertex shared by two neighboring blocks. There are two types of junctions: a **straight junction**, which connects two neighboring blocks in a straight order (e.g., black dots J_1 – J_4 in Figure 1) and an **inverted junction**, which connects two neighboring blocks in an inverted order (e.g., the red dot J_5 in Figure 1).

The algorithm for AExtractor is shown in Figure 2. This completes three sub-tasks as follows.

1. Find blocks (lines 4 and 5). This is similar to the standard phrase extraction algorithm (Och 2002) except that we find blocks with arbitrary length.
2. Detect junctions and store blocks in the arrays of detected junctions (lines 7 and 8). Junctions that are included the current block can be easily detected by looking at the previous and next blocks. A junction can connect multiple blocks on its left and right sides. For example, the second junction J_2 in Figure 1 connects two blocks on the left side and three blocks on the right side. To store these blocks, we maintain two arrays (left and right) for each junction.
3. Select block pairs from each detected junction as reordering examples (lines 12–16). This is the most challenging task for AExtractor. Because a junction may have n blocks on its left side and m blocks on its right side, we will obtain nm reordering examples if we enumerate all block pairs. This will quickly increase the number of reordering examples, especially

1: **Input**: sentence pair (s, t), word alignment M and block selection rule r
2: $\Re := \emptyset$
3: **for** each span $(i_1, i_2) \in s$ **do**
4: Find block $b = (s_{i_1}^{i_2}, t_{j_1}^{j_2})$ that is consistent with M
5: Extend block b on the target boundary with any possible non-aligned target words to get blocks $E(b)$
6: **for** each block $b^* \in b \bigcup E(b)$ **do**
7: Detect possible junctions in b^*
8: Store b^* in the arrays of detected junctions
9: **end for**
10: **end for**
11: **for** each junction J in the matrix M **do**
12: Select b_l and b_r from the left and right array of J respectively according to the selection rule r
13: **if** J is a straight junction **then**
14: $\Re := \Re \bigcup \{(straight, b_l, b_r)\}$
15: **else if** J is an inverted junction **then**
16: $\Re := \Re \bigcup \{(inverted, b_l, b_r)\}$
17: **end if**
18: **end for**
19: **Output**: reordering examples $\Re$

Figure 2
AExtractor.

those with the straight order. To keep the number of reordering examples tractable, we define various selection rules r to heuristically select special block pairs as reordering examples.

We define four selection rules as follows.

1. **strINV**: We select the smallest blocks (in terms of the target length) for straight junctions, and the largest blocks for inverted junctions. Take the straight junction J_2 in Figure 1 as an example, the extracted reordering example is (*straight*, 五 次 | *five*, 飞行| *flights*).
2. **STRinv**: We select the largest blocks (in terms of the target length) for straight junctions, and the smallest blocks for inverted junctions. Still taking the straight junction J_2 as an example, this time the extracted reordering example is (*straight*, 过去 五 次 | *The last five*, 飞行 都 因故 失败| *flights all fail due to accidents*).
3. **RANDOM**: For any junction, we randomly select one block pair from its arrays.
4. **COMBO**: For each junction, we first select two block pairs using selection rule strINV and STRinv. If there are unselected blocks, we randomly select one block pair from the remaining blocks.

3.2 TExtractor: Extracting Reordering Examples from BTG-Style Trees

A potential problem for AExtractor is caused by the use of heuristic selection rules: keeping some block pairs as reordering examples while abandoning other block pairs. The kept block pairs are not necessarily the best training instances for tuning an ITG order predictor. To avoid this problem we can extract reordering examples from the BTG trees of sentence pairs. Reordering examples extracted in this way are naturally suitable for BTG order prediction.

There are various ways to build BTG trees over sentence pairs. One can use BTG to produce bilingual parses of sentence pairs, similar to the approaches proposed by Wu (1997) and Zhang and Gildea (2005) but using the more sophisticated reordering models BWR or LAR. After parsing, reordering examples can be extracted from bilingual parse trees and a better reordering model is therefore induced from the extracted reordering examples. Using the better reordering model, the bilingual sentences are parsed again. This procedure is run iteratively until no performance gain is obtained in terms of translation or parsing accuracy. Formally, we can use expectation-maximization (EM) training in this procedure. In the expectation step, we first estimate the likelihood of all BTG trees of sentence pairs with the current BTG model. Then we extract reordering examples and collect counts for them, weighted with the probability of the BTG tree where they occur. In the maximization step, we can train a more accurate reordering model with updated reordering examples. Unfortunately, this method is at high computational cost.

Instead, here we adopt a less expensive alternative method to produce BTG trees over sentence pairs. Supposing we have word alignments produced by GIZA++, we then use the shift-reduce algorithm (SRA) introduced by Zhang, Gildea, and Chiang (2008) to decompose word alignments into hierarchical trees. The SRA can guarantee that each node is a bilingual phrase in the decomposition tree. If the fan-out of a node is larger than two, we binarize it from left to right: for two neighboring child nodes, if

they are also neighboring on both the source and target sides, we combine them and create a new node to dominate them. In this way, we can transform the decomposition tree into a BTG-style tree. Note that not all multi-branching nodes can be binarized. We extract reordering examples only from binary nodes.

Figure 3 shows the BTG-style tree which is built from the word alignment in Figure 1 according to the method mentioned here. From this tree, we can easily extract four reordering examples in a straight order and one reordering example in an inverted order.

4. Boundary Word-Based Reordering

Following the binary-classification perspective of the ITG reordering, Xiong, Liu, and Lin (2006) propose a reordering model which exploits the maximum entropy (MaxEnt) classifier for BTG order prediction

$$P_{R_b}(r^m) = P_\theta(o|A_l, A_r, A_p) = \frac{exp(\sum_i \theta_i h_i(o, A_l, A_r, A_p))}{\sum_{o'} exp(\sum_i \theta_i h_i(o', A_l, A_r, A_p))} \tag{10}$$

where the functions $h_i \in \{0, 1\}$ are reordering features and the θ_i are the weights of these features.

Xiong, Liu, and Lin (2006) define reordering features using the boundary words of the source/target sides of both children $\{A_l, A_r\}$. Supposing that we have a reordering example (*inverted*, 于 *7*月 *15*日 | *on July 15*, 举行 总统 与 国会 选举 | *held its presidential and parliament elections*), leftmost/rightmost source words {于, *15*日, 举行, 选举} and target words {*on*, *15*, *held*, *elections*} will be extracted as boundary words. Each boundary word will form a reordering feature as follows

$$h_i(o, A_l, A_r, A_p) = \begin{cases} 1, & fn = bval, o = inverted \\ 0, & otherwise \end{cases}$$

where *fn* denotes the feature name, and *bval* is the corresponding boundary word.

There are two reasons why boundary words are used as important clues for reordering:

1. Phrases frequently cohere across languages (Fox 2002). In cohesive phrase movement, boundary words directly interact with the external contexts of

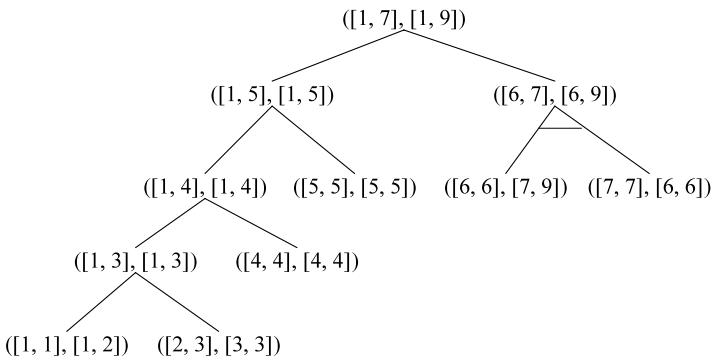

Figure 3
The BTG-style tree built from the word alignment in Figure 1. We use $([i, j], [p, q])$ to denote a tree node, where i, j and p, q are the beginning and ending indices in the source and target language, respectively.

phrases. This suggests that boundary words might contain information for phrase reordering.

2. The quantitative analysis in Xiong, Liu, and Lin (2006, page 525) further shows that boundary words indeed contain information for order prediction.

To train a BWR model, we follow three steps. First, we extract reordering examples from word-aligned bilingual data as described in the last section, then generate reordering features using boundary words from the reordering examples, and finally estimate feature weights.

5. Linguistically Annotated Reordering

In order to employ more linguistic knowledge in the ITG reordering, we annotate each BTG node involved in reordering using linguistic elements from the source-side parse trees. The linguistic elements include: (1) the head word *hw*, (2) the part-of-speech (POS) tag *ht* of the head word, and (3) its syntactic category *sc*. In this section, we describe the annotation algorithm and the LAR model, as well as the combination of LAR and BWR.

5.1 Annotation Algorithm

There are two steps to annotating a BTG node using source-side parse tree information: (1) determining the sequence on the source side which is exactly covered by the node, then (2) annotating the sequence according to the source-side parse tree. If the sequence is exactly covered by a single subtree in the source-side parse tree, it is called a **syntactic sequence**, otherwise it is a **non-syntactic sequence**. One of the challenges in this annotation is that phrases (BTG nodes) do not always cover syntactic sequences; in other words, they are not always aligned to constituent nodes in the source-side tree. To solve this problem, we generate a pseudo head word and **composite category** which consists of the syntactic categories of three relevant constituents for the non-syntactic sequences. In this way, our annotation is capable of labelling both syntactic and non-syntactic phrases and therefore providing linguistic information for any phrase reordering.

The annotation algorithm is shown in Figure 4. For a syntactic sequence, the annotation is trivial. Annotation elements directly come from the subtree that covers the sequence exactly. For a non-syntactic sequence, the process is more complicated. Firstly, we need to locate the smallest subtree c^* covering the sequence (line 6). Secondly, we try to identify the head word/tag of the sequence (lines 7–12) by using its head word directly if it is within the sequence. Otherwise, the word within the sequence which is nearest to *hw* will be assigned as the head word of the sequence. Finally, we determine the composite category of the sequence (lines 13–15), which is formulated as L-C-R. L/R refers to the syntactic category of the left/right **boundary node** of *s*, which is the highest leftmost/rightmost sub-node of c^* not overlapping the sequence. If there is no such boundary node (the sequence *s* is exactly aligned to the left/right boundary of c^*), L/R will be set to NULL. C is the syntactic category of c^*. L, R, and C together describe the external syntactic context of *s*. The composite category we define for non-syntactic phrases is similar to the CCG-style category in Zollmann, Venugopal, and Vogel (2008).

Figure 5 shows a syntactic parse tree for a Chinese sentence, with the head word annotated for each internal node. Some sample annotations are given in Table 1.

1: Annotator (sequence $s = \langle i, j \rangle$, source-side parse tree t)
2: **if** s is a syntactic sequence **then**
3: Find the subtree c in t which exactly covers s
4: $s.\{\ \} := \{c.hw, c.ht, c.sc\}$
5: **else**
6: Find the smallest subtree c^* covering s in t
7: **if** $c^*.hw \in s$ **then**
8: $s.hw := c^*.hw$ and $s.ht := c^*.ht$
9: **else**
10: Find the word $w \in s$ which is nearest to $c^*.hw$
11: $s.hw := w$ and $s.ht := w.t$ {$w.t$ is the POS tag of w}
12: **end if**
13: Find the left boundary node ln of s in c^*
14: Find the right boundary node rn of s in c^*
15: $s.sc := ln.sc\text{-}c^*.sc\text{-}rn.sc$
16: **end if**

Figure 4
The Annotation Algorithm.

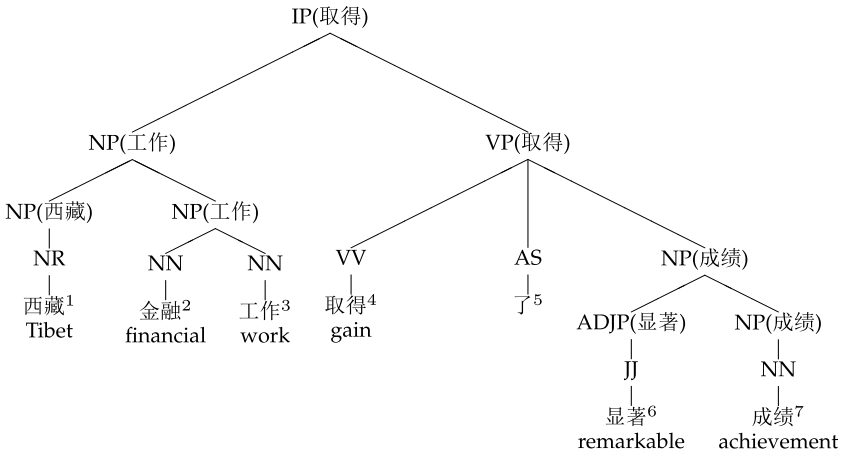

Figure 5
A syntactic parse tree with the head word annotated for each internal node. The superscripts on leaf nodes denote their surface positions from left to right.

5.2 Reordering Model

The linguistically annotated reordering model P_{R_a} is a MaxEnt-based classification model, which can be formulated as

$$P_{R_a}(r^m) = p_\theta(o|A_p^{a_p}, A_l^{a_l}, A_r^{a_r}) = \frac{exp(\sum_i \theta_i h_i(o, A_p^{a_p}, A_l^{a_l}, A_r^{a_r}))}{\sum_{o'} exp(\sum_i \theta_i h_i(o', A_p^{a_p}, A_l^{a_l}, A_r^{a_r}))} \quad (11)$$

where the feature functions $h_i \in \{0, 1\}$ are defined using annotated linguistic elements of each BTG node. Here we use the superscripts a_l, a_r, and a_p to stress that the BTG nodes are linguistically annotated.

Table 1
Annotation samples according to the tree shown in Figure 5.

sequence	*hw*	*ht*	*sc*
⟨1, 2⟩	金融	NN	NULL-NP-NN
⟨2, 3⟩	工作	NN	NP
⟨2, 4⟩	取得	VV	NP-IP-NP
⟨3, 4⟩	取得	VV	NP-IP-NP

hw/*ht* = the head word/tag, respectively; *sc* = syntactic category.

Each merging rule involves three nodes ($A_p^{a_p}, A_l^{a_l}, A_r^{a_r}$) and each node has three linguistic elements (hw, ht, sc). Therefore, the model has nine features in total. Taking the left node $A_l^{a_l}$ as an example, the model could use its head word w as a feature as follows:

$$h_i(o, A_p^{a_p}, A_l^{a_l}, A_r^{a_r}) = \begin{cases} 1, & A_l^{a_l}.hw = w, o = straight \\ 0, & otherwise \end{cases}$$

Training an LAR model also takes three steps. Firstly, we extract annotated reordering examples from source-side parsed, word-aligned bilingual data using the reordering example extraction algorithm and the annotation algorithm. We then generate features using the linguistic elements of these examples. Finally we tune feature weights to build the MaxEnt model.

5.3 Combining LAR and BWR

LAR and BWR can be combined at two different levels:

1. **Feature level.** Because both LAR and BWR are trained under the maximum entropy principle, we can combine linguistically annotated features from LAR and boundary word features from BWR together and train a single MaxEnt model. We call this method **All-in-One** combination.
2. **Model level.** We can also train two reordering models separately and integrate them into BTG-based SMT

$$\begin{aligned} P(D) = \quad & P_T(r_{1..n_l}^l) \quad \cdot P_{R_b}(r_{1..n_m}^m)^{\lambda_{R_b}} \cdot \\ & P_{R_a}(r_{1..n_m}^m)^{\lambda_{R_a}} \cdot P_L(e)^{\lambda_L} \cdot exp(|e|)^{\lambda_w} \end{aligned} \tag{12}$$

where P_{R_b} is the BWR reordering model and P_{R_a} is the LAR reordering model. We call this combination **BWR+LAR**.

We will empirically compare these two combination methods in Section 7.4.

6. A New Syntax-Based Reordering Analysis Method

In order to understand the influence of linguistic knowledge on phrase reordering, we propose a syntax-based method to analyze phrase reordering. In this analysis method,

we leverage the alignments between source-side parse trees and reference/system translations to summarize syntactic reordering patterns and calculate syntax-based measures of precision and recall for each syntactic constituent.

6.1 Overview

The alignment between a source parse tree and a target string is a collection of relationships between parse tree nodes and their corresponding target spans.[5] A **syntactic reordering pattern (SRP)** is defined as

$$\langle \alpha \rightarrow \beta_1 ... \beta_n \propto [i_1] ... [i_n] \rangle$$

The first part of an SRP is a CFG structure on the source side and the second part $[i_1]...[i_n]$ indicates the order of target spans $\beta_1^T ... \beta_n^T$ of nonterminals $\beta_1 ... \beta_n$ on the target side.[6]

Let's take the VP structure $\text{VP} \rightarrow \text{PP}_1\text{VP}_2$ as an example to explain how the precision and recall can be obtained. On the target side, the order of PP_1^T and VP_2^T might be [1][2] or [2][1]. Therefore we have two syntactic reordering patterns for this structure:

$$\langle \text{VP} \rightarrow \text{PP}_1\text{VP}_2 \propto [1][2] \rangle \quad \textit{and} \quad \langle \text{VP} \rightarrow \text{PP}_1\text{VP}_2 \propto [2][1] \rangle$$

Suppose that the two reordering patterns occur a times in the alignments between source parse trees and reference translations, b times in the alignments between source parse trees and system translations, and c times in both alignments. Then the reordering precision/recall for this structure is c/b and c/a, respectively. We can further calculate the F_1-score as $2 \times c/(a+b)$. These syntax-based metrics intuitively show how well the reordering model can reorder this structure. By summarizing all reordering patterns of all constituents, we can obtain an overall precision, recall, and F_1-score for the tested reordering model.

This new syntax-based analysis for reordering is motivated in part by recent work which transforms the order of nodes in the source-side parse tree before translation (Xia and McCord 2004; Collins, Koehn, and Kucerova 2005; Li et al. 2007; Wang, Collins, and Koehn 2007). Here we focus on the order transformation of syntactic constituents performed by reordering models during translation. In addition to aligning parse trees with reference translations, we also align parse trees with system translations so that we can learn the movement of syntactic constituents carried out by the reordering models and investigate the performance of the reordering models by comparing both alignments.

For notational convenience, we denote syntactic reordering patterns that are extracted from the alignments between source parse trees and reference translations as **REF-SRP** and those from the alignments between source parse trees and system translations as **SYS-SRP**. We refer to those present in both alignments under some conditions

5 We adopt the definition of **span** from Fox (2002): Given a node n that covers a word sequence $s_p ... s_i ... s_q$ and a word alignment matrix M, the target words aligned to n are $\{t_i : t_i \in M(s_i)\}$. We define the target span of node n as $n^T = (min(\{t_i\}), max(\{t_i\}))$. Note that n^T may contain words that are not in $\{t_i\}$.

6 Please note that the order of structures may not be defined in some cases (see Section 6.3).

that will be described in Section 6.4 as **MATCH-SRP**. To conduct a thorough analysis on the reorderings, we carry out the following steps on the test corpus (source sentences + reference translations):

1. Parse source sentences.
2. Generate word alignments between source sentences and reference translations as well as word alignments between source sentences and system translations.
3. According to the word alignments of Step 2, for each multi-branching node $\alpha \rightarrow \beta_1 ... \beta_n$ in the source parse tree generated in Step 1, find the target spans $\beta_1^T ... \beta_n^T$ and their order $[i_1]...[i_n]$ in the reference and system translations, respectively.
4. Generate REF-SRPs, SYS-SRPs, and MATCH-SRPs according to the target orders generated in Step 3 for each multi-branching node.
5. Summarize all SRPs and calculate the precision and recall as described above.

We further elaborate Steps 2–4 in the Sections 6.2–6.4.

6.2 Generating Word Alignments

To obtain word alignments between source sentences and multiple reference translations, we pair the source sentences with each of the reference translations and include the created sentence pairs in our bilingual training corpus. Then we run GIZA++ on the new corpus in both directions, and apply the "grow-diag-final" refinement rule (Koehn et al. 2005) to produce the final word alignments.

To obtain word alignments between source sentences and system translations, we store the word alignments within each phrase pair in our phrase table. When we output the system translation for a source sentence, we trace back the original source phrase for each target phrase in the system translation. This will generate a phrase alignment between the source sentence and system translation. Given the phrase alignment and word alignments within the phrase stored in the phrase table, we can easily obtain word alignments between the whole source sentence and system translation.

6.3 Generating Target Spans and Orders

Given the source parse tree and the word alignment between a source sentence and a reference/system translation, for each multi-branching node $\alpha \rightarrow \beta_1 ... \beta_n$, we firstly determine the target span β_i^T for each child node β_i following Fox (2002). If one child node is aligned to NULL, we define a special target span for it. The order for this special target span will remain the same as the child node occurring in $\beta_1 ... \beta_n$.

Two target spans may overlap with each other because of inherent divergences between two languages or noise in the word alignment. When this happens on two neighboring nodes β_i and β_{i+1}, we combine these two nodes together and redefine a target span $\beta_{i\&i+1}^T$ for the combined node. This process will be repeated until no more neighboring nodes can be combined. For example, the target span of nodes *a* and *b* in

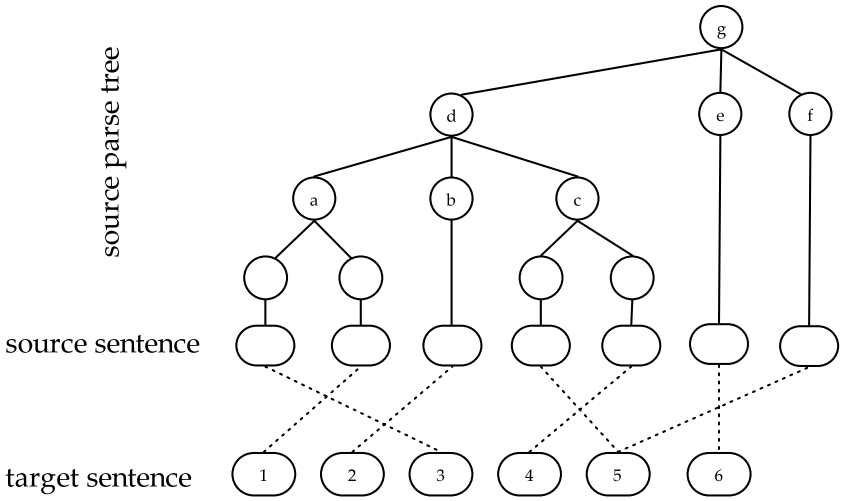

Figure 6
An example source parse tree with the word alignment between the source sentence and the target translation. Dotted lines show the word alignment.

Figure 6 overlap ((1, 3) vs. (2, 2)). Therefore these two nodes are to be combined into a new node, whose target span is (1, 3).

After performing all necessary node combinations, if there are no more overlaps, we call the multi-branching node **reorderable**, otherwise **non-reorderable**. To get a clearer picture of the reorderable nodes, we divided them into two categories:

- **fully reorderable** if all target spans of child nodes don't overlap;
- **partially reorderable** if some child nodes are combined due to overlapping.

In Figure 6, both nodes a and c are fully reorderable nodes.[7] Node d is a partially reorderable node. Node g is a non-reorderable node because (1) the target spans of its child nodes d and f overlap, and (2) child nodes d and f cannot be combined because they are not neighbors.

Because we have multiple reference translations for each source sentence, we can define multiple orders for $\{\beta_i^T\}_1^n$. If one node is non-reorderable in all reference translations, we call it **REF-non-reorderable**, otherwise **REF-reorderable**. To specify the reorderable attribute of a node in the system translation, we prefix "SYS-" to {non-reorderable, reorderable, fully-reorderable, partially-reorderable}.

6.4 Generating SRPs

After we obtain the orders of the child nodes for each multi-branching node, we generate REF-SRPs and SYS-SRPs from the fully/partially reorderable nodes. We obtain the

7 Their target translations are interrupted by the other node's translation. We will discuss this situation in Section 8.5.

MATCH-SRP for each multi-branching node by comparing the obtained SYS-SRP with the REF-SRPs for this node under the following conditions:

1. Because we have multiple reference translations, we may have different REF-SRPs. We compare the SYS-SRP with the REF-SRP where the reference translation for this node (the sequence within the target span of the node defined by the REF-SRP) has the shortest Levenshtein distance (Navarro 2001) to that of the system translation.
2. If there are combined nodes in SYS/REF-SRPs, they are treated as a unit when comparing, without considering the order within each combined node. If the order of the SYS-SRP and the selected REF-SRP matches, we have one MATCH-SRP for the node.

Let's give an example to explain these conditions. Suppose that we are processing the structure VP $\rightarrow$ $PP_1ADVP_2VP_3$. We obtain four REF-SRPs from four different reference translations and one SYS-SRP from the system output. Here we only show the orders:

Ref.a : [3][1][2]

Ref.b : [3][2][1]

Ref.c&d : [2][3][1]

SYS : [3][1&2]

References *c* and *d* have the same order. Therefore we have three different REF-SRPs for this structure. In the SYS-SRP, PP_1 and $ADVP_2$ are combined and moved to the right side of VP_3. Supposing that the system translation for this structure has the shortest edit distance to that of Reference *b*, we use the order of Reference *b* to compare the system order. In the Reference *b* order, both PP_1 and $ADVP_2$ are also moved to the right side of VP_3. Therefore the two orders of Reference *b* and SYS match. We have one matched SRP for this structure.

7. Evaluation

Our system is a BTG-based phrasal SMT system, developed following Section 2. We integrate the boundary word–based reordering model and the linguistically annotated reordering model into our system according to our reordering configuration. We carried out various experiments to evaluate the reordering example extraction algorithms of Section 3, the linguistically annotated reordering model vs. boundary word–based reordering model, and the effects of linguistically annotated features on the Chinese-to-English translation task of the NIST MT-05 using large scale training data.

7.1 Experimental Setup

We ran GIZA++ (Och and Ney 2000) on the parallel corpora (consisting of 101.93M Chinese words and 112.78M English words) listed in Table 2 in both directions and then applied the "grow-diag-final" refinement rule (Koehn, Och, and Marcu 2003) to

Table 2
Corpora used.

Corpus	LDC catalog	Chinese words	English words
United Nations	LDC2004E12	68.63M	76.99M
Hong Kong News	LDC2004T08	15.07M	15.89M
Sinorama Magazine	LDC2005T10	10.26M	9.64M
FBIS	LDC2003E14	7.09M	9.28M
Xinhua	LDC2002E18	0.40M	0.43M
Chinese News Translation	LDC2005T06	0.28M	0.31M
Chinese Treebank	LDC2003E07	0.10M	0.13M
Multiple Translation Chinese	LDC2004T07	0.10M	0.11M
Total	——	101.93M	112.78M

obtain many-to-many word alignments. From the word-aligned corpora, we extracted bilingual phrases.

We used all corpora listed in Table 2 except for the United Nations corpus to train our reordering models, which consist of 33.3M Chinese words and 35.79M English words. We ran the reordering example extractor AExtractor and TExtractor of Section 3 on the chosen word-aligned corpora. We then extracted boundary word features from the reordering examples. To extract linguistically annotated features, we parsed the Chinese side of the chosen parallel text using a Chinese parser (Xiong, Liu, and Lin 2005) which was trained on the Penn Chinese Treebank with an F_1-score of 79.4%. We ran the off-the-shelf MaxEnt toolkit[8] to tune the reordering feature weights with the iteration number set to 100 and Gaussian prior to 1 to avoid overfitting.

We built our 4-gram language model using the SRILM toolkit (Stolcke 2002), which was trained on the Xinhua section of the English Gigaword corpus (181.1M words). We selected 580 short sentences (not exceeding 50 characters per sentence) from the NIST MT-02 evaluation test data as our development set (18 words/31 characters per sentence). The NIST MT-05 test set includes 1,082 sentences with an average of 27.4 words/47.6 characters per sentence. The reference corpus for the NIST MT-05 test set contains four translations per source sentence. Both the development and test sets were also parsed using the parser mentioned above.

Our evaluation metric is the case-insensitive BLEU-4 (Papineni et al. 2002) using the shortest reference sentence length for the brevity penalty. The model feature weights are tuned on the development set to maximize BLEU using MERT (Och 2003). Statistical significance in BLEU score differences is tested by paired bootstrap re-sampling (Koehn 2004).

7.2 Bias in AExtractor

As described in Section 3, AExtractor selectively extracts reordering examples. This selective extraction raises three questions:

1. Is it necessary to extract all reordering examples?

8 Available at: `http://homepages.inf.ed.ac.uk/s0450736/maxent_toolkit.html`.

2. If it is not necessary, do the heuristic selection rules impose any bias on the reordering model? For example, if we use the strINV selection rule, meaning that we always extract the largest block pairs for inverted reordering examples, does the reordering model prefer swappings on larger blocks to those on smaller blocks?
3. Does the bias have a strong impact on the performance in terms of BLEU score?

The answer to the first question is no. Firstly, it is practically undesirable to extract all reordering examples because even a very small training set will produce millions of reordering examples if we enumerate all block pair combinations. Secondly, extracting all reordering examples introduces a great amount of noise into training and therefore undermines the final reordering model. In Table 3, we show the number of reordering examples extracted using different extraction algorithms and selection rules. The AExtractor with the COMBO selection rule extracts the largest number of reordering examples. However, it does not obtain the highest BLEU score compared with other selection rules which extract a smaller number of reordering examples. This empirically suggests that there is no need to extract all reordering examples.

To answer the second question, we trace the best BTG trees produced on the test set by our system. The BWR reordering model is trained on reordering examples which are extracted using different selection rules. Then we calculate the average number of words on the target side which are covered by binary nodes in a straight order. We refer to this number as **straight average length**. Similarly, **inverted average length** is calculated on all binary nodes in an inverted order. The third and fourth columns of Table 3 show the two average variables. Comparing these average numbers, we clearly observe that two selection rules indeed impose noticeable bias on the reordering model.

- The strINV selection rule, which always extracts the largest block pairs for inverted reordering examples, has the largest inverted average length. This indicates that the strINV rule biases the reordering model towards larger swappings.
- On the contrary, the STRinv selection rule, which extracts the largest block pairs for straight reordering examples and smallest pairs for inverted reordering examples, has the largest straight average length and a

Table 3
Comparison of reordering example extraction algorithms and selection rules. We only use BWR as the reordering model for this comparison.

Ext. Alg. (Sel. rule)	Reordering Examples	Straight Avg. Len.	Inverted Avg. Len.	BLEU
AExtractor (strINV)	10.06M	15.8	14.5	32.37
AExtractor (STRinv)	10.06M	17.3	12.8	32.47
AExtractor (RANDOM)	10.06M	14.7	11.8	32.24
AExtractor (COMBO)	23.27M	13.8	13.5	32.10
TExtractor	14.30M	15.0	14.1	29.95

relatively much smaller inverted average length. This suggests that the STRinv rule makes the reordering model prefer smaller swappings.

Note that the selection rules RANDOM and COMBO do not impose bias on the length of extracted reordering examples compared with strINV and STRinv. The latter two selection rules have special preferences on the length of reordering examples and transfer these preferences to the reordering models as shown in Table 3.

Because the preference for reordering larger/smaller blocks is imposed by the reordering example extraction algorithm with special selection rules, one might wonder whether we can allow the decoder to decide its own reordering preference. We add two new features to our translation model: reordering count penalty (*rc*) and reordering length penalty (*rl*). We accumulate *rc* whenever two neighboring BTG nodes are reordered. And at the same time we add the number of words which are covered by these two neighboring nodes to *rl*. Their weights are tuned using MERT to maximize BLEU score on the development set with other model feature weights. These two features are similar to the widely used word/phrase penalty features. Tuning the weights of the word/phrase penalty features, we can allow the decoder to favor shorter or longer phrases. Similarly, with the two new features *rc* and *rl*, we can allow the decoder to favor shorter or longer reorderings.

We conducted experiments using reordering examples which are extracted with the RANDOM and COMBO selection rules because these two rules do not impose bias on the length of reordering examples. Observing the optimized weights of *rc* and *rl* on the development set, we find that the decoder rewards larger *rc* but smaller *rl*. This means that the decoder prefers shorter reorderings to longer reorderings. However, the BLEU scores on the test set are 31.89 and 32.0 for RANDOM and COMBO, respectively, which are worse than the BLEU scores of RANDOM and COMBO without using *rc* and *rl* in Table 3, and also worse than the performance of strINV and STRinv which impose preferences on reordering examples. This seems to suggest that the preference for shorter/longer reorderings imposed by the reordering example extraction algorithm is better than that decided by the decoder itself.

Finally, for the last question, we observe from Table 3 that BLEU scores are not that much different although we have quite the opposite bias imposed by different selection rules. The changes in BLEU score, which happen when we shift from one selection rule to the other, are limited to a maximum of 1.2%. Among the four selection rules, the STRinv rule achieves the highest BLEU score. The reason might be that the bias towards smaller swappings imposed by this rule helps the decoder to reduce incorrect long-distance swappings (Xiong et al. 2008b).

7.3 AExtractor vs. TExtractor

We further compared the two algorithms for reordering example extraction. In Table 3, we find that TExtractor significantly underperforms in comparison to AExtractor. This is because the transformation from decomposition trees to BTG trees is not complete. Many crossing links due to errors and noise in word alignments generated by GIZA++ make it impossible to build BTG nodes over the corresponding words. It would be better to use alignments induced by the ITG and EM procedure described in Section 3.2 but this has a very high cost.

Given the comparison in Table 3, we use AExtractor with the STRinv selection rule to extract reordering examples for both BWR and LAR in all experiments described below.

7.4 LAR vs. BWR

Table 4 shows the results of the different integration of BWR and LAR into our systems. Only using LAR achieves a BLEU score of 32.17, which is comparable to that of BWR. This suggests that LAR is promising given that:

- LAR uses many fewer features than BWR does. According to our statistics, LAR contains only 166.1k linguistically annotated features whereas BWR has 451.4k boundary word features.
- Syntactic divergences between the source and target languages as well as parse errors prevent the effective use of syntactic knowledge for phrase reordering (see the in-depth analysis in Section 8.2.2).

Although BWR marginally outperforms LAR (32.47 vs. 32.17), simple boundary word features are not adequate to move phrases to appropriate positions because they cannot recognize syntactic contexts which are very relevant to phrase reordering. Therefore the best way to reorder a phrase is to combine BWR and LAR so that we can use syntactic information on the one hand and not worry too much about syntactic divergences on the other hand.

As described in Section 5.3, we can combine BWR and LAR at two levels: the feature level and the model level. When we combine them at the model level, we achieve an absolute improvement of 0.83 and 1.13 BLEU points over BWR and LAR, respectively, which are both statistically significant ($p < 0.01$). This shows that LAR and BWR are complementary to each other and in particular that using linguistic knowledge can significantly improve a very competitive lexicalized reordering model (BWR).

The other combination method All-in-One (at the feature level) also obtains significant improvements over BWR and LAR but marginally underperforms compared to BWR+LAR. In our later experiments we use the combination method BWR+LAR.

7.5 Varying Training Data Size

To investigate how LAR improves BWR when we vary our training data size, we carried out experiments on three different training data sets: **FBIS** (7.09M Chinese words, 9.28M English words); **Large1**, which includes all corpora listed in Table 2 except for the United Nations corpus (33.3M Chinese words, 35.79M English words); and **Large2**, which

Table 4
BLEU scores for LAR, BWR, and their combinations.

Reordering Configuration	BLEU
BWR	32.47
LAR	32.17
All-in-One	33.03**++
BWR+LAR	33.30**++

** = Significantly better than BWR ($p < 0.01$); ++ = significantly better than LAR ($p < 0.01$).

Table 5
BLEU scores on different training data sets. Large1 refers to the corpora listed in Table 2 except for the United Nations corpus. Large2 includes all corpora listed in Table 2.

Training Data	BWR	BWR+LAR	Improvement
FBIS	24.97	26.52	1.55
Large1	29.96	30.78	0.82
Large2	32.47	33.30	0.83

consists of Large1 and the United Nations corpus (101.93M Chinese words, 112.78 English words). The language model remains the same for these three data sets because it is trained on a much larger data set (181.1M words).

Table 5 shows the results. We observe that BWR+LAR is able to achieve a larger improvement of 1.55 BLEU points over BWR on smaller training data. When we enlarge the training data set from FBIS to Large1, both BWR and BWR+LAR improve quite a bit. The difference between them is narrower, 0.82 BLEU points, but still significant. When we continue to use more training data (Large2), the improvement obtained by integrating LAR becomes stable at the 0.8 level.

7.6 Effects of Linguistically Annotated Features

We conducted further experiments to evaluate the effects of individual linguistically annotated features. Using the reordering configuration of BWR+LAR, we augment LAR's feature pool incrementally: firstly using only syntactic categories[9] (*sc*) as features (170 features in total), then constructing composite categories (*cc*) for non-syntactic phrases ($sc + cc$) (8.6K features), and finally introducing head words and their POS tags into the feature pool ($sc + cc + hw + ht$) (166.1K features). This series of experiments demonstrates the impact and degree of contribution made by each feature for reordering. The experimental results are presented in Table 6, from which we have the following observations:

1. Syntactic category alone improves the performance statistically significantly. The baseline feature set *sc* with only 170 features improves the BLEU score from 32.47 to 32.87.
2. Other linguistic information, provided by the categories of boundary nodes (*cc*) and head word/tag pairs ($hw + ht$), also improves phrase reordering. Producing composite categories for non-syntactic BTG nodes and integrating head word/tag pairs into LAR as reordering features are both effective, indicating that context information complements syntactic category for capturing reordering patterns.

9 For a non-syntactic node, we only use the single category C, without constructing the composite category L-C-R.

Table 6
The effect of the linguistically annotated reordering model. (*sc*) is the baseline feature set, (*sc* + *cc*) and (*sc* + *cc* + *hw* + *ht*) are extended feature sets for LAR.

Reordering Configuration	BLEU
BWR	32.47
BWR + LAR (*sc*)	32.87*
BWR + LAR (*sc* + *cc*)	33.06**
BWR + LAR (*sc* + *cc* + *hw* + *ht*)	33.30**++

* = almost significantly better than BWR ($p < 0.075$); ** = significantly better than BWR ($p < 0.01$); ++ = significantly better than BWR + LAR (*sc*) ($p < 0.01$).

8. Analysis

We first obtain system translations of the test corpus. We generate word alignments between source sentences and system/reference translations as described in Section 6.2. Then we follow the analysis steps of Section 6.1 to investigate syntactic constituent movement in the reference translations and system translations which are generated using two different reordering configurations: BWR+LAR vs. BWR. In LAR, we use the best reordering feature set (*sc* + *cc* + *hw* + *ht*).

8.1 Syntactic Constituent Movement: Overview

If a syntactic constituent is fully reorderable or partially reorderable, it is considered to be movable as a unit. To denote the proportion of syntactic constituents to be moved as a unit, we introduce two variables **REF-R-rate** and **SYS-R-rate**, which are defined as

$$\text{SYS-R-rate} = \frac{count(\text{SYS-reorderable nodes})}{count(\text{multi-branching nodes})} \tag{13}$$

$$\text{REF-R-rate} = \frac{count(\text{REF-reorderable nodes})}{count(\text{multi-branching nodes})} \tag{14}$$

Table 7 shows the statistics of REF/SYS-reorderable nodes on the test corpus. From this table, we have the following observations:

1. A large number of nodes are REF-reorderable, accounting for 79.82% of all the multi-branching nodes. This number shows that, in reference translations, a majority of syntactic constituent movement across Chinese–English can be performed by directly permuting constituents in a sub-tree.
2. The R-rates of BWR and BWR+LAR are 77.46% and 81.79%, respectively. The R-rate of BWR+LAR is obviously higher than that of BWR, which suggests that BWR+LAR tends towards moving more syntactic constituents together than BWR does. We will discuss this further later.

Table 7
Statistics of multi-branching and REF/SYS-reorderable nodes per sentence.

	BWR	BWR+LAR
multi-branching node	18.68	
REF-reorderable node	14.91	
REF-R-rate	79.82%	
SYS-fully-reorderable node	13.16	14.01
SYS-partially-reorderable node	1.31	1.26
SYS-R-rate	77.46%	81.79%

8.2 Syntactic Constituent Movement among Multiple Reference Translations

8.2.1 Differences in Movement Orientation. Because each source sentence is translated by four different human experts, we would like to analyze the differences among reference translations, especially on the orders of constituents being translated. Table 8 shows the overall distribution over the number of different orders for each multi-branching constituent among the reference translations.

In most cases (75.4%), four reference translations have completely the same order for syntactic constituents. This makes it easier for our analysis to compare the system order with the reference order. However, there are 22% cases where two different orders are provided, which shows the flexibility of translation. According to our study, noun phrases taking DNP or CP modifiers, as well as DNPs and CPs themselves, are more likely to be translated in two different orders. Table 9 shows the percentages in which two different orders for these constituents are observed in the reference corpus.

DNP and CP are always used as pre-modifiers of noun phrases in Chinese. They often include the particle word 的 (*of*) at the ending position. The difference is that DNP constructs a phrasal modifier whereas CP constructs a relative-clause modifier. There is no fixed reordering pattern for DNP and CP and therefore for NP which takes DNP/CP as a pre-modifier. In the DNP → NP DEG structure, the DEG (的) can be

Table 8
Distribution of number of different orders by which syntactic constituents are translated in references.

Number of different orders	1	2	3	4
Percentage	75.40	22	2.33	0.33

Table 9
Two-order translation distribution of 4 NP-related constituents.

Constituent	**2-order translation percentage**
NP → DNP NP	16.93
NP → CP NP	9.43
CP → IP DEC	24.79
DNP → NP DEG	34.58

translated into *'s* or *of*, which are both appropriate in most cases, depending on the translator's preference. If the former is chosen, the order of DNP and therefore the order for NP → DNP NP will both be straight: [1][2]. Otherwise, the two orders will be inverted: [2][1]. Similarly, there are also different translation patterns for CP → IP DEC and NP → CP NP. CP can be translated into "*that* + clause" or adjective-like phrases in English. Figure 7 shows an example where the CP constituent is translated into an adjective-like phrase. Although the "*that* + clause" must be placed behind the noun phrase which it modifies, the order for adjective-like phrases is flexible (see Figure 7).

For those constituents with different reference orders, we compare the order of the system translation to that of the reference translation which has the shortest edit distance to the system translation as described herein so that we can take into account the potential influence of different translations on the order of syntactic constituents.

8.2.2 REF-Non-Reorderable Constituents. We also study REF-R-rates for the 13 most frequent constituents listed in Table 10. We find that two constituents, VP_1 → PP VP_2 and NP_1 → CP NP_2, have the lowest REF-R-rates, 58.20% and 61.77%, respectively. This means that about 40% of them are REF-non-reorderable. In order to understand the reasons why they are non-reorderable in reference translations, we further investigate REF-non-reorderable cases for the constituent type VP_1 → PP VP_2 and roughly classify the reasons into three categories as follows.

1. **Outside interruption.** The reordering of PP and VP_2 is interrupted by other constituents outside VP_1. For example, the Chinese sentence *[NP* 某人*/somebody] [*VP_1 *[PP* 在...时*/when...] [*VP_2 *[*说*/say NP[...]]]]* is translated into *when..., somebody said ....* Here the translation of the first NP which is outside VP_1 is inserted between the translations of PP and VP_2 and therefore interrupts their reordering. Outside interruption accounts for 21.65% of REF-non-reorderable cases.
2. **Inside interruption.** The reordering of PP and VP_2 is interrupted by the combination of PP's subnodes with VP_2's subnodes. Inside interruption accounts for 48.45% of REF-non-reorderable cases, suggesting that it is the major factor which decreases the reorderability of VP → PP VP. Because both PP and VP have their own complex sub-structures, the inside

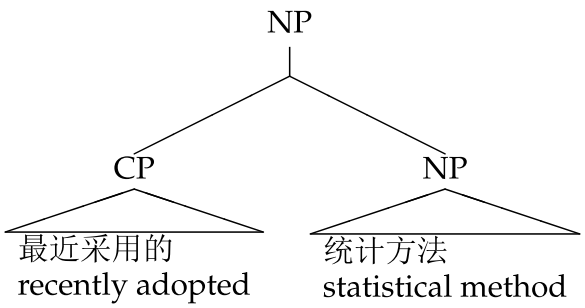

Figure 7
An example of the translation of NP → CP NP. This constituent can be translated in two different orders: 1) the recently adopted statistical method (straight order); 2) the statistical method recently adopted (inverted order).

Table 10
F_1-scores (BWR+LAR vs. BWR) for the 13 most frequent constituents in the test corpus. Constituents indicated in **bold** have relatively lower F_1 score for reordering.

Type	Constituent	Percent. (%)	SYS-R-rate (%)		F_1-score (%)	
			BWR	BWR+LAR	BWR	BWR+LAR
VP	VP → VV NP	8.12	79.22	84.10	76.97	80.53
	VP → ADVP VP	4.30	63.45	65.86	70.83	73.67
	VP → PP VP	1.87	**60.32**	**70.37**	**39.29**	**40.33**
	VP → VV IP	1.82	79.35	86.14	77.16	82.26
NP	NP → NN NN	6.88	84.68	85.18	76.17	79.10
	NP → NP NP	5.12	82.13	84.93	69.25	72.17
	NP → DNP NP	2.14	**69.75**	**74.83**	**56.68**	**56.61**
	NP → CP NP	2.12	**59.67**	**73.43**	**48.75**	**54.48**
Misc.	IP → NP VP	6.78	71.99	79.80	63.22	65.79
	PP → P NP	3.63	80.63	85.95	82.75	84.93
	CP → IP DEC	3.51	83.94	87.89	69.91	72.24
	QP → CD CLP	2.74	66	65	67.52	68.47
	DNP → NP DEG	2.43	85.98	89.84	67.5	68.75

interruption is very complicated and includes a variety of cases, some of which are quite unexpected. Here we show two frequent examples of inside interruption:

a. The preposition in the PP and the verb word/phrase of VP_2 are aligned to only one target word or one continuous phrase. For example, 向...施压/*pressure*, 对...有信心/*be confident of*, 因...而受苦/*suffer from*, and so on. This is caused by the lexical divergence problem.

b. The PP is first combined with the verb word of VP_2 in an inverted order, then combined with the remainder of VP_2 in a straight order. For example, *[PP [P 从] [omission$_1$]] [VP [VV 了解到] [omission$_2$]]* might be translated into *learned from omission$_1$ that omission$_2$*.

3. **Parse error.** This accounts for 29.90% of REF-non-reorderable cases.

Although these reasons are summarized from our analysis on the constituent type VP → PP VP, they can be used to explain other REF-non-reorderable constituents, such as NP → CP NP.

8.3 Syntactic Constituent Movement in System Translations

8.3.1 Overall Reordering Precision and Recall of Syntactic Constituents. By summarizing all syntactic reordering patterns (REF-SRP, SYS-SRP, and Match-SRP) for all constituents, we can calculate the overall reordering precision and recall of syntactic constituents. Table 11 shows the results for both BWR+LAR and BWR, where BWR+LAR clearly outperforms BWR.

Table 11
Syntactic reordering precision and recall of BWR+LAR vs. BWR on the test corpus.

	Precision	Recall	F_1
BWR	70.89	68.79	69.83
BWR+LAR	71.32	73.08	72.19

8.3.2 The Effect of Linguistic Knowledge on Phrase Movement. To understand the change in phrase movement caused by linguistic knowledge, we further investigate how well BWR and BWR+LAR reorder certain constituents, especially those with high distribution probability. Table 10 lists the 13 most frequent constituents, which jointly account for 51.46% of all multi-branching constituents. Except for NP → DNP NP, the reordering F_1 score of all these constituents in BWR+LAR is better than that in BWR.

Our hypothesis for the phrase movement change in BWR+LAR is that the integrated linguistic knowledge makes phrase movement in BWR+LAR pay more respect to syntactic constituent boundaries. The overall R-rates of BWR+LAR vs. BWR described in Section 8.1 indicate that BWR+LAR tends towards moving more syntactic constituents together than BWR does. We want to know whether this is also true for a specific constituent type. The fourth and fifth columns in Table 10 present the R-rate for each individual constituent type that we have analyzed. It is obvious that the R-rate of BWR+LAR is much higher than that of BWR for almost all constituents. This indicates that higher R-rate is one of the reasons for the higher performance of BWR+LAR.

To gain a more concrete understanding of this change, we show two examples for the reordering of VP → PP VP in Figure 8. In both examples, BWR fails to move the PP constituent to the right of the VP constituent, whereas BWR+LAR does it successfully. By tracing the binary BTG trees generated by the decoder, we find that BWR generated a very different BTG tree from the source parse tree whereas the BTG tree in BWR+LAR almost matches the source parse tree. In the first example, BWR combines the VP phrase

Input:	[**NP** NHK] [**VP** [**PP** 向 政治 压力] [**VP** 低头]]
Ref:	[**NP** NHK] [**VP** [**VP** bowed] [**PP** to political pressure]]
BWR:	[[⟨the/向 NHK/NHK⟩ political/政治] pressure/压力低头]
BWR+LAR:	[NHK/NHK ⟨bow/低头 [to political/向政治 pressure/压力]⟩]
Input:	[**NP** 五 角 大厦] [**VP** [**PP** 为 亚洲 海啸 灾民] [**VP** 展开 空前 救援 行动]]
Ref:	[**NP** Pentagon] [**VP** [**VP** launches unprecedented relief operations] [**PP** for Asian tsunami victims]]
BWR:	[Pentagon/五 角 大 厦 [is/为 ⟨[⟨[an/展 开 unprecedented/空 前] [tsunami/海 啸 disaster/灾 民]⟩ relief operations/救 援 行 动] in Asia/亚洲⟩
BWR+LAR:	[Pentagon/五 角 大 厦 ⟨[[launched/展 开 an unprecedented/空 前] rescue operations/救 援 行 动] [[for Asian/为 亚 洲 tsunami/海 啸] victims/灾民]⟩]

Figure 8
Two examples for the translation of VP → PP VP. Square brackets indicate combinations in a straight order and angular brackets represent combinations in an inverted order.

低头 with 压力 and then combines 政治. The preposition word 向 is combined with the NP phrase *NHK*, which makes the translation of *NHK* interrupt the reordering of VP → PP VP in this example. The BWR tree in the second example is even worse. The non-syntactic phrase 展开 空前 in the VP phrase is first combined with 海啸 灾民, which is a sub-phrase of PP preceding VP in an inverted order. The remaining part of the VP phrase is then merged. This merging process continues regardless of the source parse tree. The comparison of BTG trees of BWR+LAR and BWR in the two examples suggests that reordering models should respect syntactic structures in order to capture reorderings under these structures.

Our observation on phrase movement change resonates with the recent efforts in phrasal SMT that allow the decoder to prefer translations which show more respect for syntactic constituent boundaries (Cherry 2008; Marton and Resnik 2008; Yamamoto, Okuma, and Sumita 2008). Mapping to syntactic constituent boundaries, or in other words, syntactic cohesion (Fox 2002; Cherry 2008), has been studied and used in early syntax-based SMT models (Wu 1997; Yamada and Knight 2001). But its value has receded in more powerful syntax-based models (Galley et al. 2004; Chiang 2005) and non-syntactic phrasal models (Koehn, Och, and Marcu 2003). Marton and Resnik (2008) and Cherry (2008) use syntactic cohesion as a soft constraint by penalizing hypotheses which violate constituent boundaries. Yamamoto, Okuma, and Sumita (2008) impose this as a hard constraint on the ITG constraint to allow reorderings which respect the source parse tree. They all report significant improvements on different language pairs, which indicates that syntactic cohesion is very useful for phrasal SMT. Our analysis demonstrates that linguistically annotated reordering provides an alternative way to incorporate syntactic cohesion into phrasal SMT.

8.4 Challenges in Phrase Reordering and Suggestions

We highlight three constituent types in Table 10 (indicated in bold) which are much more difficult to reorder, as indicated by their relatively lower F_1 scores. The lower F_1 scores indicate that BWR+LAR is not fully sufficient for reordering these constituents although it performs much better than BWR. We find two main reasons for the lower F_1 scores and provide suggestions accordingly as follows.

1. **Constrained decoding.** We observe that in reorderable constituents which involve long-distance reorderings, their boundaries are easily violated by phrases outside them. To prohibit boundary violations, we propose constrained decoding. In constrained decoding, we define special **zones** in source sentences. Reorderings and translations within the zones cannot be interrupted by fragments outside the zones. We can also define other constrained operations on the zones. For example, we can prohibit swappings in any zones which contain punctuation (Xiong et al. 2008b). The beginning and ending positions of a zone are automatically learned. To be more flexible, they are not necessarily constituent boundaries. Constrained decoding is different from both soft constraints (Cherry 2008; Marton and Resnik 2008) and hard constraints (Yamamoto, Okuma, and Sumita 2008). It can be considered as in between both of these because it is harder than the former but softer than the latter.

2. **Integrating special reordering rules.** Some constituents are indeed non-reorderable as we discussed in Section 8.2.2. Inside or outside

interruptions have to be allowed to obtain fluent translations for these constituents. However, the allowance of interruptions is sometimes beyond the representability of BTG rules. For example, to solve the lexical divergence problem, bilingual rules with aligned lexicons have to be introduced. To capture reorderings of these constituents, we propose to integrate special reordering rules with richer contextual information into BTG to extend BTG's ability to deal with interruptions. Completely replacing BTG with richer formalisms, such as hierarchical phrase (Chiang 2005) and tree-to-string (Liu, Liu, and Lin 2006) or string-to-tree (Marcu et al. 2006), introduces a huge extra cost. Instead, integrating a small number of reordering rules into BTG to model reorderings of non-reorderable constituents would be more desirable.

8.5 Discussion

In the definition of syntactic reordering patterns, we only consider the relative order of individual constituents on the target side. We do not consider whether or not they remain contiguous on the target side. It is possible that other words are inserted between spans of two contiguous constituents. We use the term **gap** to refer to when this happens. The absence of a gap in the definition of syntactic reordering patterns may produce more matched SRPs and therefore lead to higher precision and recall. Table 12 shows the revised overall precision and recall of syntactic reordering patterns when we also compare gaps. The revised results show that BWR+LAR still significantly outperforms BWR. This also applies to the 13 constituents identified in Table 10. The analysis results obtained before are still valid when we consider gaps.

9. Related Work

9.1 Linguistically Motivated Phrase Reordering

There are various approaches which are devoted to incorporating linguistic knowledge into phrase reordering. Generally, these approaches can be roughly divided into three categories: (1) reordering the source language in a preprocessing step before decoding begins; (2) estimating phrase movement with reordering models; and (3) capturing reorderings by synchronous grammars. The preprocessing approach applies manual or automatically extracted reordering knowledge from linguistic structures to transform the source language sentence into a word order that is closer to the target sentence. The second reordering approach moves phrases under certain reordering constraints and estimates the probabilities of movement with linguistic information. In the third

Table 12
Revised overall precision and recall of BWR+LAR vs. BWR on the test corpus when we consider the gap in syntactic reordering patterns.

	Precision	Recall	F_1
BWR (gap)	46.28	44.91	45.58
BWR+LAR (gap)	48.80	50	49.39

approach, reordering knowledge is included in synchronous rules. The last two categories reorder the source sentence during decoding, which distinguishes them from the first approach. Note that some researchers integrate multiple reordering approaches in one decoder (Lin 2004; Quirk, Menezes, and Cherry 2005; Ge, Ittycheriah, and Papineni 2008).

9.1.1 The Preprocessing Approach. In early work, Brown et al. (1992) describe an approach to reordering French phrases in a preprocessing step. Xia and McCord (2004) present a preprocessing approach which automatically learns reordering patterns based on CFG productions. Since then, the preprocessing approach seems to have been more popular. Collins, Koehn, and Kucerova (2005) propose reordering German clauses with six types of manual rules. Similarly, Wang, Collins, and Koehn (2007) reorder Chinese parse trees using fine-grained human-written rules, mostly concentrating on VP and NP structures. Li et al. (2007) improve the preprocessing approach by generating *n*-best reordered source sentences with reordering knowledge automatically learned from the alignments between source parse trees and target translations. The approach proposed in Li et al. also enhances the connection between the preprocessing and decoding by adding a source reordering probability feature. Other approaches introduced in Nießn and Ney (2001), Popović and Ney (2006), and Zhang, Zens, and Ney (2007) use morphological, POS, and chunk knowledge in the preprocessing approach, respectively.

9.1.2 Estimating Phrase Movement with Embedded Reordering Models. Under the IBM constraint (Zens and Ney 2003), the early work uses a distortion-based reordering model to penalize word movements (Koehn, Och, and Marcu 2003). Similarly, under the ITG constraint, the corresponding model is the flat model which assigns a prior probability to the straight or inverted order (Wu 1996). These two models don't respect the content of phrases which are moved. To address this issue, lexicalized reordering models which are sensitive to lexical information about phrases are introduced (Tillman 2004; Koehn et al. 2005; Kumar and Byrne 2005; Al-Onaizan and Papineni 2006). Xiong, Liu, and Lin (2006) introduce a more flexible reordering model under the ITG constraint using discriminative features which are automatically learned from a training corpus. Zhang et al. (2007) propose a model for syntactic phrase reordering which uses syntactic knowledge from source parse trees. Our reordering approach is most similar to those in Xiong, Liu, and Lin (2006) and Zhang et al. but extends them further by using syntactic knowledge and allowing non-syntactic phrase reordering.

9.1.3 Capturing Reorderings by Synchronous Grammars. Wu (1997) and Eisner (2003) use synchronous grammars to capture reorderings between two languages. Chiang (2005) introduces formal synchronous grammars for phrase-based translation. In his work, hierarchical reordering knowledge is included in synchronous rules which are automatically learned from word-aligned corpus. In linguistically syntax-based models, string-to-tree (Marcu et al. 2006), tree-to-string (Huang, Knight, and Joshi 2006; Liu, Liu, and Lin 2006), and tree-to-tree (Zhang et al. 2008) translation rules, just to name a few, are explored. Linguistical reordering knowledge is naturally included in these syntax-based translation rules.

9.2 Automatic Analysis of Reordering

Although there is a variety of work on phrase reordering, automatic analysis of phrase reordering is not widely explored in the SMT literature. Chiang et al. (2005) propose

an automatic method to compare different system outputs in a fine-grained manner with regard to reordering. In their method, common word *n*-grams occurring in both reference translations and system translations are extracted and generalized to part-of-speech tag sequences. A recall is calculated for each certain tag sequence to indicate the ability of reordering models to capture this tag sequence in system translations. Popovic et al. (2006) use the relative difference between WER (word error rate) and PER (position independent word error rate) to indicate reordering errors. The larger the difference, the more reordering errors there are.

Callison-Burch et al. (2007) propose a constituent-based evaluation that is very similar to our method in Steps (1)–(3). They also parse the source sentence and automatically align the parse tree with the reference/system translations. The difference is that they highlight constituents from the parse tree to enable human evaluation of the translations of these constituents, rather than automatically analyzing constituent movement. They use this method for human evaluation in the shared translation task of the 2007 and 2008 ACL Workshop on Statistical Machine Translation.

Fox (2002) systematically studies syntactic cohesion between French and English using human translations and alignments. Compared with her work, our analysis here includes, but is not limited to, an investigation of syntactic cohesion *in an actual MT system*.

10. Conclusion

We have presented a novel linguistically motivated phrase reordering approach: Linguistically Annotated Reordering. The LAR approach incorporates soft linguistic knowledge from the source parse tree into hard hierarchical skeletons generated by BTG in phrasal SMT. To automatically learn reordering features, we have introduced algorithms for reordering example extraction and linguistic annotation. We have also proposed a new syntax-based analysis method to detect syntactic constituent movement in human/machine translations.

We have conducted experiments on large-scale training data to evaluate LAR and BWR as well as the reordering example extraction algorithms. Our evaluation results show that:

1. Extracting reordering examples directly from word alignments is much better than from BTG-style trees which are built from word alignments.
2. Selection rules which bias the reordering model towards smaller swappings improve translation quality.
3. BWR+LAR significantly outperforms BWR, which suggests that the integration of linguistic knowledge improves reordering; and tuning two separate reordering models is better than the All-in-One combination method.

We have further analyzed the outputs of BWR+LAR vs. BWR using the proposed syntax-based analysis method. Our analysis results show that:

1. BWR+LAR achieves a significantly higher reordering precision and recall than BWR does with regard to syntactic constituent movement.

2. For most reorderable constituents, integrating source-side linguistic knowledge into the reordering model can significantly improve reorderings by guiding reordering models to prefer hypotheses that pay more respect to constituent boundaries.

3. For non-reorderable constituents or constituents involving long-distance reorderings, integrating source-side linguistic knowledge into the reordering model is not sufficient to avoid illegal boundary violations or to capture reordering patterns.

To avoid illegal boundary violations in long-span constituents, we suggest constrained decoding, which protects special zones in the source sentence from being interrupted by phrases outside the zones. Beginning and ending positions of the zones are automatically learned using lexical and syntactic knowledge. To capture complex reorderings which cross constituent boundaries, phrasal SMT should integrate reordering rules with richer contextual information.

Acknowledgments

We would like to thank the three anonymous reviewers for their helpful comments and suggestions.

References

Al-Onaizan, Yaser and Kishore Papineni. 2006. Distortion models for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 529–536, Sydney.

Brown, Peter F., Stephen A. Della Pietra, Vincent J. Della Pietra, John D. Lafferty, and Robert L. Mercer. 1992. Analysis, statistical transfer, and synthesis in machine translation. In *Proceedings of the Fourth International Conference on Theoretical and Methodological Issues in Machine Translation*, pages 83–100, Montreal.

Callison-Burch, Chris, Cameron Fordyce, Philipp Koehn, Christof Monz, and Josh Schroeder. 2007. (Meta-) evaluation of machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 136–158, Prague.

Cherry, Colin. 2008. Cohesive phrase-based decoding for statistical machine translation. In *Proceedings of ACL-08: HLT*, pages 72–80, Columbus, OH.

Chiang, David. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 263–270, Ann Arbor, MI.

Chiang, David, Adam Lopez, Nitin Madnani, Christof Monz, Philip Resnik, and Michael Subotin. 2005. The hiero machine translation system: Extensions, evaluation, and analysis. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 779–786, Vancouver.

Collins, Michael, Philipp Koehn, and Ivona Kucerova. 2005. Clause restructuring for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 531–540, Ann Arbor, MI.

Eisner, Jason. 2003. 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.

Fox, Heidi. 2002. Phrasal cohesion and statistical machine translation. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 304–311, Philadelphia, PA.

Galley, Michel, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What's in a translation rule? In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 273–280, Boston, MA.

Ge, Niyu, Abe Ittycheriah, and Kishore Papineni. 2008. Multiple reorderings in phrase-based machine translation. In *Proceedings of the ACL-08: HLT Second Workshop on Syntax and Structure in*

Statistical Translation (SSST-2), pages 61–68, Columbus, OH.

Huang, Liang, Kevi Knight, and Aravind Joshi. 2006. Statistical syntax-directed translation with extended domain of locality. In *Proceedings of the 7th Conference of the Association for Machine Translation of the Americas*, pages 66–73, Cambridge, MA.

Koehn, Philipp. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP 2004*, pages 388–395, Barcelona.

Koehn, Philipp, Amittai Axelrod, Alexandra Birch Mayne, Chris Callison-Burch, Miles Osborne, and David Talbot. 2005. Edinburgh system description for the 2005 IWSLT speech translation evaluation. In *Proceedings of the International Workshop on Spoken Language Translation 2005*, pages 78–85, Pittsburgh, PA.

Koehn, Philipp, Franz Joseph Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 58–54, Edmonton.

Kumar, Shankar and William Byrne. 2005. Local phrase reordering models for statistical machine translation. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 161–168, Vancouver.

Li, Chi-Ho, Minghui Li, Dongdong Zhang, Mu Li, Ming Zhou, and Yi Guan. 2007. A probabilistic approach to syntax-based reordering for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 720–727, Prague.

Lin, Dekang. 2004. A path-based transfer model for machine translation. In *Proceedings of the 20th International Conference on Computational Linguistics (Coling 2004)*, pages 625–630, Geneva.

Liu, Yang, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 609–616, Sydney.

Marcu, Daniel, Wei Wang, Abdessamad Echihabi, and Kevin Knight. 2006. SPMT: Statistical machine translation with syntactified target language phrases. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 44–52, Sydney.

Marton, Yuval and Philip Resnik. 2008. Soft syntactic constraints for hierarchical phrased-based translation. In *Proceedings of ACL-08: HLT*, pages 1003–1011, Columbus, OH.

Navarro, Gonzalo. 2001. A guided tour to approximate string matching. *ACM Computing Surveys*, 33(1):31–88.

Nießn, Sonja and Hermann Ney. 2001. Morpho-syntactic analysis for reordering in statistical machine translation. In *Proceedings of MT Summit VIII*, pages 247–252, Santiago de Compostela.

Och, Franz Josef. 2002. *Statistical Machine Translation: From Single-Word Models to Alignment Templates*. Ph.D. thesis, RWTH Aachen University, Germany.

Och, Franz Josef. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167, Sapporo.

Och, Franz Josef and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 440–447, Hong Kong.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, PA.

Popovic, Maja, Adrià de Gispert, Deepa Gupta, Patrik Lambert, Hermann Ney, José B. Mariño, Marcello Federico, and Rafael Banchs. 2006. Morpho-syntactic information for automatic error analysis of statistical machine translation output. In *Proceedings on the Workshop on Statistical Machine Translation*, pages 1–6, New York, NY.

Popović, Maja and Hermann Ney. 2006. Pos-based word reorderings for statistical machine translation. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC 2006)*, pages 1278–1283, Genoa.

Quirk, Chris, Arul Menezes, and Colin Cherry. 2005. Dependency treelet translations: Syntactically informed phrasal smt. In *Proceedings of the 43rd*

Annual Meeting of the ACL, pages 271–279, Ann Arbor, MI.

Stolcke, Andreas. 2002. SRILM—an extensible language modeling toolkit. In *Proceedings of the 7th International Conference on Spoken Language Processing (ICSLP 2002)*, pages 901–904, Denver, CO.

Tillman, Christoph. 2004. A unigram orientation model for statistical machine translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2004): Short Papers*, pages 101–104, Boston, MA.

Wang, Chao, Michael Collins, and Philipp Koehn. 2007. Chinese syntactic reordering for statistical machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 737–745, Prague.

Wu, Dekai. 1996. A polynomial-time algorithm for statistical machine translation. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pages 152–158, Santa Cruz, CA.

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

Wu, Dekai, Marine Carpuat, and Yihai Shen. 2006. Inversion transduction grammar coverage of Arabic-English word alignment for tree-structured statistical machine translation. In *Proceeding of the IEEE/ACL 2006 Workshop on Spoken Language Technology (SLT 2006)*, pages 234–237, Aruba.

Xia, Fei and Michael McCord. 2004. Improving a statistical MT system with automatically learned rewrite patterns. In *Proceedings of the 20th International Conference on Computational Linguistics (Coling 2004)*, pages 508–514, Geneva.

Xiong, Deyi, Qun Liu, and Shouxun Lin. 2005. Parsing the penn chinese treebank with semantic knowledge. In *Proceedings of The 2nd International Joint Conference on Natural Language Processing (IJCNLP-05)*, pages 70–81, Jeju Island.

Xiong, Deyi, Qun Liu, and Shouxun Lin. 2006. Maximum entropy based phrase reordering model for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 521–528, Sydney.

Xiong, Deyi, Min Zhang, Aiti Aw, and Haizhou Li. 2008a. A linguistically annotated reordering model for BTG-based statistical machine translation. In *Proceedings of ACL-08: HLT, Short Papers*, pages 149–152, Columbus, OH.

Xiong, Deyi, Min Zhang, Aiti Aw, Haitao Mi, Qun Liu, and Shouxun Lin. 2008b. Refinements in BTG-based statistical machine translation. In *Proceedings of the Third International Joint Conference on Natural Language Processing*, pages 505–512, Hyderabad.

Xue, Nianwen, Fei Xia, Shizhe Huang, and Anthony Kroch. 2000. The bracketing guidelines for the Penn Chinese treebank (3.0). Technical report IRCS 00-07, University of Pennsylvania Institute for Research in Cognitive Science, Philadelphia.

Yamada, Kenji and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proceedings of 39th Annual Meeting of the Association for Computational Linguistics*, pages 523–530, Toulouse.

Yamamoto, Hirofumi, Hideo Okuma, and Eiichiro Sumita. 2008. Imposing constraints from the source tree on ITG constraints for SMT. In *Proceedings of the ACL-08: HLT Second Workshop on Syntax and Structure in Statistical Translation (SSST-2)*, pages 1–9, Columbus, OH.

Zens, Richard, and Hermann Ney. 2003. A comparative study on reordering constraints in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics* (ACL), pages 144–151, Sapporo, Japan.

Zhang, Dongdong, Mu Li, Chi-Ho Li, and Ming Zhou. 2007. Phrase reordering model integrating syntactic knowledge for SMT. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 533–540, Prague.

Zhang, Hao and Daniel Gildea. 2005. Stochastic lexicalized inversion transduction grammar for alignment. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 475–482, Ann Arbor, MI.

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

Zhang, Min, Hongfei Jiang, Aiti Aw, Haizhou Li, Chew Lim Tan, and Sheng Li. 2008. A tree sequence alignment-based tree-to-tree translation model. In *Proceedings of ACL-08: HLT*, pages 559–567, Columbus, OH.

Zhang, Yuqi, Richard Zens, and Hermann Ney. 2007. 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, Rochester, NY.

Zollmann, Andreas, Ashish Venugopal, and Stephan Vogel. 2008. The CMU syntax-augmented machine translation system: SAMT on hadoop with N-best alignments. In *Proceedings of International Workshop on Spoken Language Translation (IWSLT)*, pages 18–25, Honolulu, HI.