A Tree Sequence Alignment-based Tree-to-Tree Translation Model

Min Zhang[1] Hongfei Jiang[2] Aiti Aw[1] Haizhou Li[1] Chew Lim Tan[3] and Sheng Li[2]
[1]Institute for Infocomm Research mzhang@i2r.a-star.edu.sg aaiti@i2r.a-star.edu.sg hli@i2r.a-star.edu.sg
[2]Harbin Institute of Technology hfjiang@mtlab.hit.edu.cn lisheng@hit.edu.cn
[3]National University of Singapore tancl@comp.nus.edu.sg

Abstract

This paper presents a translation model that is based on tree sequence alignment, where a tree sequence refers to a single sequence of sub-trees that covers a phrase. The model leverages on the strengths of both phrase-based and linguistically syntax-based method. It automatically learns aligned tree sequence pairs with mapping probabilities from word-aligned bi-parsed parallel texts. Compared with previous models, it not only captures non-syntactic phrases and discontinuous phrases with linguistically structured features, but also supports multi-level structure reordering of tree typology with larger span. This gives our model stronger expressive power than other reported models. Experimental results on the NIST MT-2005 Chinese-English translation task show that our method statistically significantly outperforms the baseline systems.

1 Introduction

Phrase-based modeling method (Koehn et al., 2003; Och and Ney, 2004a) is a simple, but powerful mechanism to machine translation since it can model local reorderings and translations of multi-word expressions well. However, it cannot handle long-distance reorderings properly and does not exploit discontinuous phrases and linguistically syntactic structure features (Quirk and Menezes, 2006). Recently, many syntax-based models have been proposed to address the above deficiencies (Wu, 1997; Chiang, 2005; Eisner, 2003; Ding and Palmer, 2005; Quirk et al, 2005; Cowan et al., 2006; Zhang et al., 2007; Bod, 2007; Yamada and Knight, 2001; Liu et al., 2006; Liu et al., 2007; Gildea, 2003; Poutsma, 2000; Hearne and Way, 2003). Although good progress has been reported, the fundamental issues in applying linguistic syntax to SMT, such as non-isomorphic tree alignment, structure reordering and non-syntactic phrase modeling, are still worth well studying.

In this paper, we propose a tree-to-tree translation model that is based on tree sequence alignment. It is designed to combine the strengths of phrase-based and syntax-based methods. The proposed model adopts tree sequence[1] as the basic translation unit and utilizes tree sequence alignments to model the translation process. Therefore, it not only describes non-syntactic phrases with syntactic structure information, but also supports multi-level tree structure reordering in larger span. These give our model much more expressive power and flexibility than those previous models. Experiment results on the NIST MT-2005 Chinese-English translation task show that our method significantly outperforms Moses (Koehn et al., 2007), a state-of-the-art phrase-based SMT system, and other linguistically syntax-based methods, such as SCFG-based and STSG-based methods (Zhang et al., 2007). In addition, our study further demonstrates that 1) structure reordering rules in our model are very useful for performance improvement while discontinuous phrase rules have less contribution and 2) tree sequence rules are able to model non-syntactic phrases with syntactic structure information, and thus contribute much to the performance improvement, but those rules consisting of more than three sub-trees have almost no contribution.

The rest of this paper is organized as follows: Section 2 reviews previous work. Section 3 elabo-

[1] A tree sequence refers to an ordered sub-tree sequence that covers a phrase or a consecutive tree fragment in a parse tree. It is the same as the concept "forest" used in Liu et al (2007).

Proceedings of ACL-08: HLT, pages 559–567,
Columbus, Ohio, USA, June 2008. ©2008 Association for Computational Linguistics

rates the modelling process while Sections 4 and 5 discuss the training and decoding algorithms. The experimental results are reported in Section 6. Finally, we conclude our work in Section 7.

2 Related Work

Many techniques on linguistically syntax-based SMT have been proposed in literature. Yamada and Knight (2001) use noisy-channel model to transfer a target parse tree into a source sentence. Eisner (2003) studies how to learn non-isomorphic tree-to-tree/string mappings using a STSG. Ding and Palmer (2005) propose a syntax-based translation model based on a probabilistic synchronous dependency insertion grammar. Quirk et al. (2005) propose a dependency treelet-based translation model. Cowan et al. (2006) propose a feature-based discriminative model for target language syntactic structures prediction, given a source parse tree. Huang et al. (2006) study a TSG-based tree-to-string alignment model. Liu et al. (2006) propose a tree-to-string model. Zhang et al. (2007b) present a STSG-based tree-to-tree translation model. Bod (2007) reports that the unsupervised STSG-based translation model performs much better than the supervised one. The motivation behind all these work is to exploit linguistically syntactic structure features to model the translation process. However, most of them fail to utilize non-syntactic phrases well that are proven useful in the phrase-based methods (Koehn et al., 2003).

The formally syntax-based model for SMT was first advocated by Wu (1997). Xiong et al. (2006) propose a MaxEnt-based reordering model for BTG (Wu, 1997) while Setiawan et al. (2007) propose a function word-based reordering model for BTG. Chiang (2005)'s hierarchal phrase-based model achieves significant performance improvement. However, no further significant improvement is achieved when the model is made sensitive to syntactic structures by adding a constituent feature (Chiang, 2005).

In the last two years, many research efforts were devoted to integrating the strengths of phrase-based and syntax-based methods. In the following, we review four representatives of them.

1) Hassan et al. (2007) integrate supertags (a kind of lexicalized syntactic description) into the target side of translation model and language model under the phrase-based translation framework, resulting in good performance improvement. However, neither source side syntactic knowledge nor reordering model is further explored.

2) Galley et al. (2006) handle non-syntactic phrasal translations by traversing the tree upwards until a node that subsumes the phrase is reached. This solution requires larger applicability contexts (Marcu et al., 2006). However, phrases are utilized independently in the phrase-based method without depending on any contexts.

3) Addressing the issues in Galley et al. (2006), Marcu et al. (2006) create an xRS rule headed by a pseudo, non-syntactic non-terminal symbol that subsumes the phrase and its corresponding multi-headed syntactic structure; and one sibling xRS rule that explains how the pseudo symbol can be combined with other genuine non-terminals for acquiring the genuine parse trees. The name of the pseudo non-terminal is designed to reflect the full realization of the corresponding rule. The problem in this method is that it neglects alignment consistency in creating sibling rules and the naming mechanism faces challenges in describing more complicated phenomena (Liu et al., 2007).

4) Liu et al. (2006) treat all bilingual phrases as lexicalized tree-to-string rules, including those non-syntactic phrases in training corpus. Although the solution shows effective empirically, it only utilizes the source side syntactic phrases of the input parse tree during decoding. Furthermore, the translation probabilities of the bilingual phrases and other tree-to-string rules are not compatible since they are estimated independently, thus having different parameter spaces. To address the above problems, Liu et al. (2007) propose to use forest-to-string rules to enhance the expressive power of their tree-to-string model. As is inherent in a tree-to-string framework, Liu et al.'s method defines a kind of auxiliary rules to integrate forest-to-string rules into tree-to-string models. One problem of this method is that the auxiliary rules are not described by probabilities since they are constructed during decoding, rather than learned from the training corpus. So, to balance the usage of different kinds of rules, they use a very simple feature counting the number of auxiliary rules used in a derivation for penalizing the use of forest-to-string and auxiliary rules.

In this paper, an alternative solution is presented to combine the strengths of phrase-based and syn-

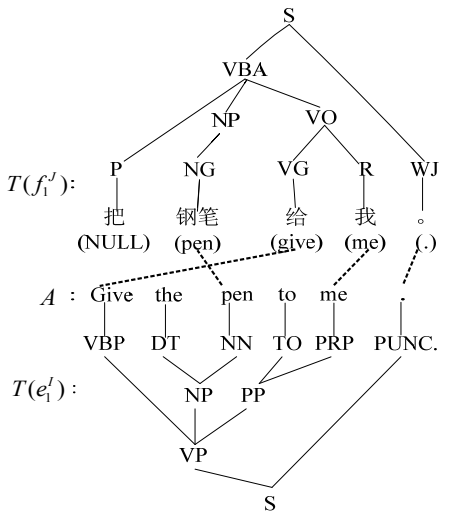

Figure 1: A word-aligned parse tree pairs of a Chinese sentence and its English translation

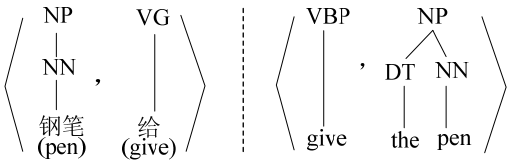

Figure 2: Two Examples of tree sequences

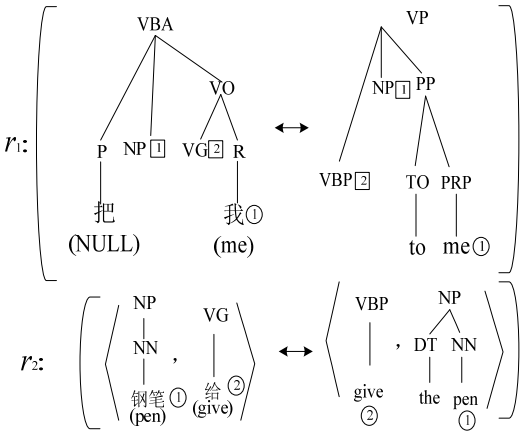

Figure 3: Two examples of translation rules

tax-based methods. Unlike previous work, our solution neither requires larger applicability contexts (Galley et al., 2006), nor depends on pseudo nodes (Marcu et al., 2006) or auxiliary rules (Liu et al., 2007). We go beyond the single sub-tree mapping model to propose a tree sequence alignment-based translation model. To the best of our knowledge, this is the first attempt to empirically explore the tree sequence alignment based model in SMT.

3 Tree Sequence Alignment Model

3.1 Tree Sequence Translation Rule

The leaf nodes of a sub-tree in a tree sequence can be either non-terminal symbols (grammar tags) or terminal symbols (lexical words). Given a pair of source and target parse trees $T(f_1^J)$ and $T(e_1^I)$ in Fig. 1, Fig. 2 illustrates two examples of tree sequences derived from the two parse trees. A tree sequence translation rule r is a pair of aligned tree sequences $r = <TS(f_{j_1}^{j_2}), TS(e_{i_1}^{i_2}), \tilde{A}>$, where:

- $TS(f_{j_1}^{j_2})$ is a source tree sequence, covering the span $[j_1, j_2]$ in $T(f_1^J)$, and
- $TS(e_{i_1}^{i_2})$ is a target one, covering the span $[i_1, i_2]$ in $T(e_1^I)$, and
- $\tilde{A}$ are the alignments between leaf nodes of two tree sequences, satisfying the following condition: $\forall (i,j) \in \tilde{A} : i_1 \le i \le i_2 \leftrightarrow j_1 \le j \le j_2$.

Fig. 3 shows two rules extracted from the tree pair shown in Fig. 1, where r_1 is a tree-to-tree rule and r_2 is a tree sequence-to-tree sequence rule. Obviously, tree sequence rules are more powerful than phrases or tree rules as they can capture all phrases (including both syntactic and non-syntactic phrases) with syntactic structure information and allow any tree node operations in a longer span. We expect that these properties can well address the issues of non-isomorphic structure alignments, structure reordering, non-syntactic phrases and discontinuous phrases translations.

3.2 Tree Sequence Translation Model

Given the source and target sentences f_1^J and e_1^I and their parse trees $T(f_1^J)$ and $T(e_1^I)$, the tree sequence-to-tree sequence translation model is formulated as:

$$\begin{aligned} Pr(e_1^I \mid f_1^J) &= \sum_{T(f_1^J), T(e_1^I)} Pr(e_1^I, T(e_1^I), T(f_1^J) \mid f_1^J) \\ &= \sum_{T(f_1^J), T(e_1^I)} (Pr(T(f_1^J) \mid f_1^J) \\ &\quad \cdot Pr(T(e_1^I) \mid T(f_1^J), f_1^J) \\ &\quad \cdot Pr(e_1^I \mid T(e_1^I), T(f_1^J), f_1^J)) \end{aligned} \quad (1)$$

In our implementation, we have:

1) $Pr(T(f_1^J) \mid f_1^J) \equiv 1$ since we only use the best source and target parse tree pairs in training.
2) $Pr(e_1^I \mid T(e_1^I), T(f_1^J), f_1^J) \equiv 1$ since we just output the leaf nodes of $T(e_1^I)$ to generate e_1^I regardless of source side information.

Since $T(f_1^J)$ contains the information of f_1^J, now we have:

$$\begin{aligned} Pr(e_1^I \mid f_1^J) &= Pr(T(e_1^I) \mid T(f_1^J), f_1^J) \\ &= Pr(T(e_1^I) \mid T(f_1^J)) \end{aligned} \quad (2)$$

By Eq. (2), translation becomes a tree structure mapping issue. We model it using our tree sequence-based translation rules. Given the source parse tree $T(f_1^J)$, there are multiple derivations that could lead to the same target tree $T(e_1^I)$, the mapping probability $Pr(T(e_1^I) \mid T(f_1^J))$ is obtained by summing over the probabilities of all derivations. The probability of each derivation θ is given as the product of the probabilities of all the rules $p(r_i)$ used in the derivation (here we assume that a rule is applied *independently* in a derivation).

$$\begin{aligned} Pr(e_1^I \mid f_1^J) &= Pr(T(e_1^I) \mid T(f_1^J)) \\ &= \sum_{\theta} \prod_{r_i \in \theta} p(r_i :< TS(e_{i_1}^{i_2}), TS(f_{j_1}^{j_2}), \tilde{A} >) \end{aligned} \quad (3)$$

Eq. (3) formulates the tree sequence alignment-based translation model. Figs. 1 and 3 show how the proposed model works. First, the source sentence is parsed into a source parse tree. Next, the source parse tree is detached into two source tree sequences (the left hand side of rules in Fig. 3). Then the two rules in Fig. 3 are used to map the two source tree sequences to two target tree sequences, which are then combined to generate a target parse tree. Finally, a target translation is yielded from the target tree.

Our model is implemented under log-linear framework (Och and Ney, 2002). We use seven basic features that are analogous to the commonly used features in phrase-based systems (Koehn, 2004): 1) bidirectional rule mapping probabilities; 2) bidirectional lexical rule translation probabilities; 3) the target language model; 4) the number of rules used and 5) the number of target words. In addition, we define two new features: 1) the number of lexical words in a rule to control the model's preference for lexicalized rules over un-lexicalized rules and 2) the average tree depth in a rule to balance the usage of hierarchical rules and flat rules. Note that we do not distinguish between larger (taller) and shorter source side tree sequences, i.e. we let these rules compete directly with each other.

4 Rule Extraction

Rules are extracted from word-aligned, bi-parsed sentence pairs $< T(f_1^J), T(e_1^I), A >$, which are classified into two categories:

- ***initial rule***, if all leaf nodes of the rule are terminals (i.e. lexical word), and
- ***abstract rule***, otherwise, i.e. at least one leaf node is a non-terminal (POS or phrase tag).

Given an *initial rule* $< TS(f_{j_1}^{j_2}), TS(e_{i_1}^{i_2}), \tilde{A} >$, its ***sub initial rule*** is defined as a triple $< TS(f_{j_3}^{j_4}), TS(e_{i_3}^{i_4}), \hat{A} >$ if and only if:

- $< TS(f_{j_3}^{j_4}), TS(e_{i_3}^{i_4}), \hat{A} >$ is an *initial rule*.
- $\forall (i, j) \in \tilde{A} : i_3 \le i \le i_4 \leftrightarrow j_3 \le j \le j_4$, i.e. $\hat{A} \subseteq \tilde{A}$
- $TS(f_{j_3}^{j_4})$ is a sub-graph of $TS(f_{j_1}^{j_2})$ while $TS(e_{i_3}^{i_4})$ is a sub-graph of $TS(e_{i_1}^{i_2})$.

Rules are extracted in two steps:

1) Extracting *initial rules* first.
2) Extracting *abstract rules* from extracted *initial rules* with the help of *sub initial rules*.

It is straightforward to extract *initial rules*. We first generate all fully lexicalized source and target tree sequences using a dynamic programming algorithm and then iterate over all generated source and target tree sequence pairs $< TS(f_{j_1}^{j_2}), TS(e_{i_1}^{i_2}) >$. If the condition "$\forall (i, j) \in A : i_1 \le i \le i_2 \leftrightarrow j_1 \le j \le j_2$" is satisfied, the triple $< TS(f_{j_1}^{j_2}), TS(e_{i_1}^{i_2}), \tilde{A} >$ is an *initial rule*, where $\tilde{A}$ are alignments between leaf nodes of $TS(f_{j_1}^{j_2})$ and $TS(e_{i_1}^{i_2})$. We then derive *abstract rules* from *initial rules* by removing one or more of its *sub initial rule*s. The *abstract rule* extraction algorithm presented next is implemented using dynamic programming. Due to space limitation, we skip the details here. In order to control the number of rules, we set three constraints for both finally extracted initial and abstract rules:

1) The depth of a tree in a rule is not greater

than h.

2) The number of non-terminals as leaf nodes is not greater than c.
3) The tree number in a rule is not greater than d.

In addition, we limit *initial rules* to have at most seven lexical words as leaf nodes on either side. However, in order to extract long-distance reordering rules, we also generate those *initial rules* with more than seven lexical words for *abstract rules* extraction only (not used in decoding). This makes our *abstract rules* more powerful in handling global structure reordering. Moreover, by configuring these parameters we can implement other translation models easily: 1) STSG-based model when $d = 1$; 2) SCFG-based model when $d = 1$ and $h = 2$; 3) phrase-based translation model only (no reordering model) when $c = 0$ and $h = 1$.

Algorithm 1: *abstract rules* extraction

Input: *initial rule* set r_{ini}

Output: *abstract rule* set r_{abs}

1: **for each** $r_i \in r_{ini}$, **do**
2: put all ***sub initial rules*** of r_i into a set r_i^{subini}
3: **for each** subset $\Theta \subseteq r_i^{subini}$ **do**
4: **if** there are spans overlapping between any two rules in the subset Θ **then**
5: **continue** //**go to** line 3
6: **end if**
7: generate an abstract rule by removing the portions covered by Θ from r_i and co-indexing the pairs of non-terminals that rooting the removed source and target parts
8: add them into the *abstract rule* set r_{abs}
9: **end do**
10: **end do**

5 Decoding

Given $T(f_1^J)$, the decoder is to find the best derivation θ that generates $<T(f_1^J), T(e_1^I)>$.

$$\begin{aligned} \hat{e} &= \arg\max_{e_1^I} Pr(T(e_1^I) \mid T(f_1^J)) \\ &\approx \arg\max_{e_1^I, \theta} \prod_{r_i \in \theta} p(r_i) \end{aligned} \quad (4)$$

Algorithm 2: Tree Sequence-based Decoder

Input: $T(f_1^J)$	**Output:** $T(e_1^I)$

Data structures:

$h[j_1, j_2]$ To store translations to a span $[j_1, j_2]$

1: **for** $s = 0$ to J -1 **do** // s: span length
2: **for** $j_1 = 1$ to $J - s$, $j_2 = j_1 + s$ **do**
3: **for each** rule r spanning $[j_1, j_2]$ **do**
4: **if** r is an *initial rule* **then**
5: insert r into $h[j_1, j_2]$
6: **else**
7: generate new translations from r by replacing non-terminal leaf nodes of r with their corresponding spans' translations that are already translated in previous steps
8: insert them into $h[j_1, j_2]$
9: **end if**
10: **end for**
11: **end for**
12: **end for**
13: output the hypothesis with the highest score in $h[1, J]$ as the final best translation

The decoder is a span-based beam search together with a function for mapping the source derivations to the target ones. Algorithm 2 illustrates the decoding algorithm. It translates each span iteratively from small one to large one (lines 1-2). This strategy can guarantee that when translating the current span, all spans smaller than the current one have already been translated before if they are translatable (line 7). When translating a span, if the usable rule is an *initial rule*, then the tree sequence on the target side of the rule is a candidate translation (lines 4-5). Otherwise, we replace the non-terminal leaf nodes of the current *abstract rule* with their corresponding spans' translations that are already translated in previous steps (line 7). To speed up the decoder, we use several thresholds to limit search beams for each span:

1) α, the maximal number of rules used
2) β, the minimal log probability of rules
3) γ, the maximal number of translations yield

It is worth noting that the decoder does not force a complete target parse tree to be generated. If no rules can be used to generate a complete target parse tree, the decoder just outputs whatever have

been translated so far monotonically as one hypothesis.

6 Experiments

6.1 Experimental Settings

We conducted Chinese-to-English translation experiments. We trained the translation model on the FBIS corpus (7.2M+9.2M words) and trained a 4-gram language model on the Xinhua portion of the English Gigaword corpus (181M words) using the SRILM Toolkits (Stolcke, 2002) with modified Kneser-Ney smoothing. We used sentences with less than 50 characters from the NIST MT-2002 test set as our development set and the NIST MT-2005 test set as our test set. We used the Stanford parser (Klein and Manning, 2003) to parse bilingual sentences on the training set and Chinese sentences on the development and test sets. The evaluation metric is case-sensitive BLEU-4 (Papineni et al., 2002). We used GIZA++ (Och and Ney, 2004) and the heuristics "grow-diag-final" to generate *m-to-n* word alignments. For the MER training (Och, 2003), we modified Koehn's MER trainer (Koehn, 2004) for our tree sequence-based system. For significance test, we used Zhang et al's implementation (Zhang et al, 2004).

We set three baseline systems: Moses (Koehn et al., 2007), and SCFG-based and STSG-based tree-to-tree translation models (Zhang et al., 2007). For Moses, we used its default settings. For the SCFG/STSG and our proposed model, we used the same settings except for the parameters d and h ($d=1$ and $h=2$ for the SCFG; $d=1$ and $h=6$ for the STSG; $d=4$ and $h=6$ for our model). We optimized these parameters on the training and development sets: c=3, α=20, β=-100 and γ=100.

6.2 Experimental Results

We carried out a number of experiments to examine the proposed tree sequence alignment-based translation model. In this subsection, we first report the rule distributions and compare our model with the three baseline systems. Then we study the model's expressive ability by comparing the contributions made by different kinds of rules, including *strict* tree sequence rules, non-syntactic phrase rules, structure reordering rules and discontinuous phrase rules[2]. Finally, we investigate the impact of maximal sub-tree number and sub-tree depth in our model. All of the following discussions are held on the training and test data.

Rule	*Initial Rules*	*Abstract Rules*		
	L	**P**	**U**	**Total**
BP	322,965	0	0	322,965
TR	443,010	144,459	24,871	612,340
TSR	225,570	103,932	714	330,216

Table 1: # of rules used in the testing ($d=4$, $h=6$) (**BP**: bilingual phrase (used in Moses), **TR**: tree rule (only 1 tree), **TSR**: tree sequence rule (> 1 tree), **L**: fully lexicalized, **P**: partially lexicalized, **U**: unlexicalized)

Table 1 reports the statistics of rules used in the experiments. It shows that:

1) We verify that the **BP**s are fully covered by the *initial rules* (i.e. lexicalized rules), in which the lexicalized **TSR**s model all non-syntactic phrase pairs with rich syntactic information. In addition, we find that the number of *initial rules* is greater than that of bilingual phrases. This is because one bilingual phrase can be covered by more than one *initial rule* which having different sub-tree structures.

2) *Abstract rules* generalize *initial rules* to unseen data and with structure reordering ability. The number of the *abstract rule* is far less than that of the *initial rules*. This is because leaf nodes of an *abstract rule* can be non-terminals that can represent any sub-trees using the non-terminals as roots.

Fig. 4 compares the performance of different models. It illustrates that:

1) Our tree sequence-based model significantly outperforms ($p < 0.01$) previous phrase-based and linguistically syntax-based methods. This empirically verifies the effect of the proposed method.

2) Both our method and STSG outperform Moses significantly. Our method also clearly outperforms STSG. These results suggest that:

- The linguistically motivated structure features are very useful for SMT, which can be cap-

[2] To be precise, we examine the contributions of *strict* tree sequence rules and single tree rules separately in this section. Therefore, unless specified, the term "tree sequence rules" used in this section only refers to the *strict* tree sequence rules, which must contain at least two sub-trees on the source side.

tured by the two syntax-based models through tree node operations.

- Our model is much more effective in utilizing linguistic structures than STSG since it uses tree sequence as basic translation unit. This allows our model not only to handle structure reordering by tree node operations in a larger span, but also to capture non-syntactic phrases, which circumvents previous syntactic constraints, thus giving our model more expressive power.

3) The linguistically motivated SCFG shows much lower performance. This is largely because SCFG only allows sibling nodes reordering and fails to utilize both non-syntactic phrases and those syntactic phrases that cannot be covered by a single CFG rule. It thereby suggests that SCFG is less effective in modelling parse tree structure transfer between Chinese and English when using Penn Treebank style linguistic grammar and under word-alignment constraints. However, formal SCFG show much better performance in the formally syntax-based translation framework (Chiang, 2005). This is because the formal syntax is learned from phrases directly without relying on any linguistic theory (Chiang, 2005). As a result, it is more robust to the issue of non-syntactic phrase usage and non-isomorphic structure alignment.

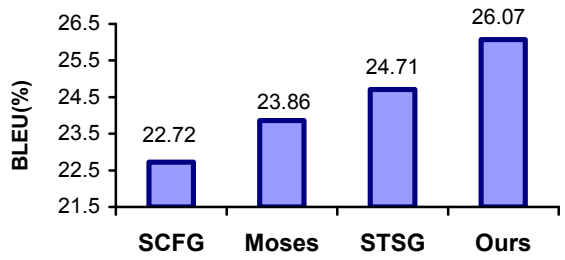

Figure 4: Performance comparison of different methods

Rule Type	TR (STSG)	TR +TSR_L	TR+TSR_L +TSR_P	TR +TSR
BLEU(%)	24.71	25.72	25.93	26.07

Table 2: Contributions of **TSR**s (see Table 1 for the definitions of the abbreviations used in this table)

Table 2 measures the contributions of different kinds of tree sequence rules. It suggests that:

1) All the three kinds of **TSR**s contribute to the performance improvement and their combination further improves the performance. It suggests that they are complementary to each other since the lexicalized **TSR**s are used to model non-syntactic phrases while the other two kinds of **TSR**s can generalize the lexicalized rules to unseen phrases.

2) The lexicalized **TSR**s make the major contribution since they can capture non-syntactic phrases with syntactic structure features.

Rule Type	BLEU (%)
TR+**TSR**	26.07
(**TR**+**TSR**) w/o **SRR**	24.62
(**TR**+**TSR**) w/o **DPR**	25.78

Table 3: Effect of **S**tructure **R**eordering **R**ules (**SRR**: refers to the structure reordering rules that have at least two non-terminal leaf nodes with inverted order in the source and target sides, which are usually not captured by phrase-based models. Note that the reordering between lexical words and non-terminal leaf nodes is not considered here) and **D**iscontinuous **P**hrase **R**ules (**DPR**: refers to these rules having at least one non-terminal leaf node between two lexicalized leaf nodes) in our tree sequence-based model ($d = 4$ and $h = 6$)

Rule Type	# of rules	# of rules overlapped (Intersection)
SRR	68,217	18,379 (26.9%)
DPR	57,244	18,379 (32.1%)

Table 4: numbers of SRR and DPR rules

Table 3 shows the contributions of SRR and DPR. It clearly indicates that SRRs are very effective in reordering structures, which improve performance by 1.45 (26.07-24.62) BLEU score. However, DPRs have less impact on performance in our tree sequence-based model. This seems in contradiction to the previous observations[3] in literature. However, it is not surprising simply because we use tree sequences as the basic translation units. Thereby, our model can capture all phrases. In this sense, our model behaves like a phrase-based model, less sensitive to discontinuous phras-

[3] Wellington et al. (2006) reports that discontinuities are very useful for translational equivalence analysis using binary-branching structures under word alignment and parse tree constraints while they are almost of no use if under word alignment constraints only. Bod (2007) finds that discontinues phrase rules make significant performance improvement in linguistically STSG-based SMT models.

es (Wellington et al., 2006). Our additional experiments also verify that discontinuous phrase rules are complementary to syntactic phrase rules (Bod, 2007) while non-syntactic phrase rules may compromise the contribution of discontinuous phrase rules. Table 4 reports the numbers of these two kinds of rules. It shows that around 30% rules are shared by the two kinds of rule sets. These overlapped rules contain at least two non-terminal leaf nodes plus two terminal leaf nodes, which implies that longer rules do not affect performance too much.

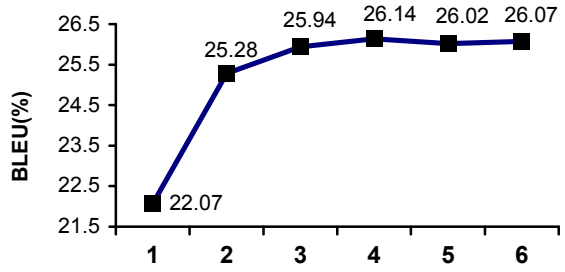

Figure 5: Accuracy changing with different maximal tree depths ($h = 1$ to 6 when $d = 4$)

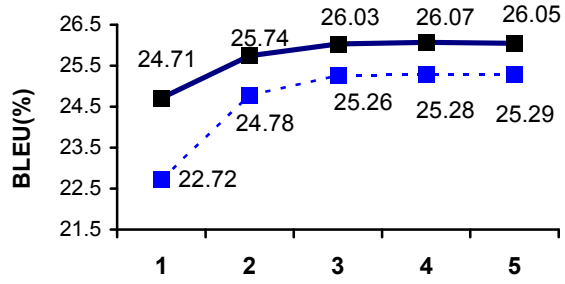

Figure 6: Accuracy changing with the different maximal number of trees in a tree sequence (d =1 to 5), the upper line is for $h = 6$ while the lower line is for $h = 2$.

Fig. 5 studies the impact when setting different maximal tree depth (h) in a rule on the performance. It demonstrates that:

1) Significant performance improvement is achieved when the value of h is increased from 1 to 2. This can be easily explained by the fact that when $h = 1$, only monotonic search is conducted, while $h = 2$ allows non-terminals to be leaf nodes, thus introducing preliminary structure features to the search and allowing non-monotonic search.

2) Internal structures and large span (due to h increasing) are also useful as attested by the gain of 0.86 (26.14-25.28) Blue score when the value of h increases from 2 to 4.

Fig. 6 studies the impact on performance by setting different maximal tree number (d) in a rule. It further indicates that:

1) Tree sequence rules (d >1) are useful and even more helpful if we limit the tree depth to no more than two (lower line, h=2). However, tree sequence rules consisting of more than three subtrees have almost no contribution to the performance improvement. This is mainly due to data sparseness issue when d >3.

2) Even if only two-layer sub-trees (lower line) are allowed, our method still outperforms STSG and Moses when d>1. This further validates the effectiveness of our design philosophy of using multi-sub-trees as basic translation unit in SMT.

7 Conclusions and Future Work

In this paper, we present a tree sequence alignment-based translation model to combine the strengths of phrase-based and syntax-based methods. The experimental results on the NIST MT-2005 Chinese-English translation task demonstrate the effectiveness of the proposed model. Our study also finds that in our model the tree sequence rules are very useful since they can model non-syntactic phrases and reorderings with rich linguistic structure features while discontinuous phrases and tree sequence rules with more than three sub-trees have less impact on performance.

There are many interesting research topics on the tree sequence-based translation model worth exploring in the future. The current method extracts large amount of rules. Many of them are redundant, which make decoding very slow. Thus, effective rule optimization and pruning algorithms are highly desirable. Ideally, a linguistically and empirically motivated theory can be worked out, suggesting what kinds of rules should be extracted given an input phrase pair. For example, most function words and headwords can be kept in *abstract rules* as features. In addition, word alignment is a hard constraint in our rule extraction. We will study direct structure alignments to reduce the impact of word alignment errors. We are also interested in comparing our method with the forest-to-string model (Liu et al., 2007). Finally, we would also like to study unsupervised learning-based bilingual parsing for SMT.

References

Rens Bod. 2007. *Unsupervised Syntax-Based Machine Translation: The Contribution of Discontinuous Phrases*. MT-Summmit-07. 51-56.

David Chiang. 2005. *A hierarchical phrase-based model for SMT*. ACL-05. 263-270.

Brooke Cowan, Ivona Kucerova and Michael Collins. 2006. *A discriminative model for tree-to-tree translation*. EMNLP-06. 232-241.

Yuan Ding and Martha Palmer. 2005. *Machine translation using probabilistic synchronous dependency insertion grammars*. ACL-05. 541-548.

Jason Eisner. 2003. *Learning non-isomorphic tree mappings for MT*. ACL-03 (companion volume).

Michel Galley, Mark Hopkins, Kevin Knight and Daniel Marcu. 2004. *What's in a translation rule?* HLT-NAACL-04.

Michel Galley, J. Graehl, K. Knight, D. Marcu, S. DeNeefe, W. Wang and I. Thayer. 2006. *Scalable Inference and Training of Context-Rich Syntactic Translation Models.* COLING-ACL-06. 961-968

Daniel Gildea. 2003. *Loosely Tree-Based Alignment for Machine Translation.* ACL-03. 80-87.

Jonathan Graehl and Kevin Knight. 2004. *Training tree transducers*. HLT-NAACL-2004. 105-112.

Mary Hearne and Andy Way. 2003. *Seeing the wood for the trees: data-oriented translation*. MT Summit IX, 165-172.

Liang Huang, Kevin Knight and Aravind Joshi. 2006. *Statistical Syntax-Directed Translation with Extended Domain of Locality*. AMTA-06 (poster).

Dan Klein and Christopher D. Manning. 2003. *Accurate Unlexicalized Parsing*. ACL-03. 423-430.

Philipp Koehn, F. J. Och and D. Marcu. 2003. *Statistical phrase-based translation*. HLT-NAACL-03. 127-133.

Philipp Koehn. 2004. *Pharaoh: a beam search decoder for phrase-based statistical machine translation models*. AMTA-04, 115-124

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin and Evan Herbst. 2007. *Moses: Open Source Toolkit for Statistical Machine Translation*. ACL-07 (poster) 77-180.

Yang Liu, Qun Liu and Shouxun Lin. 2006. *Tree-to-String Alignment Template for Statistical Machine Translation*. COLING-ACL-06. 609-616.

Yang Liu, Yun Huang, Qun Liu and Shouxun Lin. 2007. *Forest-to-String Statistical Translation Rules*. ACL-07. 704-711.

Daniel Marcu, W. Wang, A. Echihabi and K. Knight. 2006. *SPMT: Statistical Machine Translation with Syntactified Target Language Phrases*. EMNLP-06. 44-52.

I. Dan Melamed. 2004. *Statistical machine translation by parsing*. ACL-04. 653-660.

Franz J. Och and Hermann Ney. 2002. *Discriminative training and maximum entropy models for statistical machine translation*. ACL-02. 295-302.

Franz J. Och. 2003. *Minimum error rate training in statistical machine translation*. ACL-03. 160-167.

Franz J. Och and Hermann Ney. 2004a. *The alignment template approach to statistical machine translation*. Computational Linguistics, 30(4):417-449.

Kishore Papineni, Salim Roukos, ToddWard and Wei-Jing Zhu. 2002. *BLEU: a method for automatic evaluation of machine translation*. ACL-02. 311-318.

Arjen Poutsma. 2000. *Data-oriented translation. COLING-2000*. 635-641

Chris Quirk and Arul Menezes. 2006. *Do we need phrases? Challenging the conventional wisdom in SMT*. COLING-ACL-06. 9-16.

Chris Quirk, Arul Menezes and Colin Cherry. 2005. *Dependency treelet translation: Syntactically informed phrasal SMT*. ACL-05. 271-279.

Stefan Riezler and John T. Maxwell III. 2006. *Grammatical Machine Translation*. HLT-NAACL-06. 248-255.

Hendra Setiawan, Min-Yen Kan and Haizhou Li. 2007. *Ordering Phrases with Function Words*. ACL-7. 712-719.

Andreas Stolcke. 2002. *SRILM - an extensible language modeling toolkit*. ICSLP-02. 901-904.

Benjamin Wellington, Sonjia Waxmonsky and I. Dan Melamed. 2006. *Empirical Lower Bounds on the Complexity of Translational Equivalence*. COLING-ACL-06. 977-984.

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

Deyi Xiong, Qun Liu and Shouxun Lin. 2006. *Maximum Entropy Based Phrase Reordering Model for SMT*. COLING-ACL-06. 521–528.

Kenji Yamada and Kevin Knight. 2001. *A syntax-based statistical translation model*. ACL-01. 523-530.

Min Zhang, Hongfei Jiang, Ai Ti Aw, Jun Sun, Sheng Li and Chew Lim Tan. 2007. *A Tree-to-Tree Alignment-based Model for Statistical Machine Translation*. MT-Summit-07. 535-542.

Ying Zhang. Stephan Vogel. Alex Waibel. 2004. *Interpreting BLEU/NIST scores: How much improvement do we need to have a better system?* LREC-04. 2051-2054.