A Tree-to-String Phrase-based Model for Statistical Machine Translation

Thai Phuong Nguyen
College of Technology
Vietnam National University, Hanoi
thainp@vnu.edu.vn
Akira Shimazu[1], Tu-Bao Ho[2], Minh Le Nguyen[1], and Vinh Van Nguyen[1]
[1]School of Information Science
[2]School of Knowledge Science
Japan Advanced Institute of Science and Technology
{shimazu,bao,nguyenml,vinhnv}@jaist.ac.jp

Abstract

Though phrase-based SMT has achieved high translation quality, it still lacks of generalization ability to capture word order differences between languages. In this paper we describe a general method for tree-to-string phrase-based SMT. We study how syntactic transformation is incorporated into phrase-based SMT and its effectiveness. We design syntactic transformation models using unlexicalized form of synchronous context-free grammars. These models can be learned from source-parsed bitext. Our system can naturally make use of both constituent and non-constituent phrasal translations in the decoding phase. We considered various levels of syntactic analysis ranging from chunking to full parsing. Our experimental results of English-Japanese and English-Vietnamese translation showed a significant improvement over two baseline phrase-based SMT systems.

1 Introduction

Based on the kind of linguistic information which is made use of, syntactic SMT can be divided into four types: tree-to-string, string-to-tree, tree-to-tree, and hierarchical phrase-based. The tree-to-string approach (Collins et al., 2005; Nguyen and Shimazu, 2006; Liu et al., 2006 and 2007) supposes that syntax of the source language is known. This approach can be applied when a source language parser is available. The string-to-tree approach (Yamada and Knight, 2001; Galley et al., 2006) focuses on syntactic modelling of the target language in cases it has syntactic resources such as treebanks and parsers. The tree-to-tree approach models the syntax of both languages, therefore extra cost is required. The fourth approach (Chiang, 2005) constraints phrases under context-free grammar structure without any requirement of linguistic annotation.

In this paper, we present a tree-to-string phrase-based method which is based on synchronous CFGs. This method has two important properties: syntactic transformation is used in the decoding phase including a word-to-phrase tree transformation model and a phrase reordering model; phrases are the basic unit of translation. Since we design syntactic transformation models using un-lexicalized synchronous CFGs, the number of rules is small[1]. Previous studies on tree-to-string SMT are different from ours. Collins et al. Collins et al. (2005) used hand crafted rules to carry out word reordering in the preprocessing phase but not decoding phase. Nguyen and Shimazu (2006) presented a more general method in which lexicalized syntactic reordering models based on PCFGs can be learned from source-parsed bitext and then applied in the preprocessing phase. Liu et al. (2006) changed the translation unit from phrases to tree-to-string alignment templates (TATs) while we do not. TATs was represented as xRs rules while we use synchronous CFG rules. In order to overcome the limitation that TATs can not capture non-constituent phrasal translations, Liu et al. (2007) proposed forest-to-string rules while our system can naturally make use of such kind of phrasal translation by word-to-phrase tree transformation.

We carried out experiments with two language pairs English-Japanese and English-Vietnamese. Our system achieved significant improvements over Pharaoh, a state-of-the-art phrase-based SMT system. We also analyzed the dependence of translation quality on the level of syntactic analysis (shallow or deep).

Figure 1 shows the architecture of our system. The input of this system is a source-language tree and the output is a target-language string. This system uses all features of conventional phrase-based SMT as in (Koehn et al., 2003). There are two new features including a word-to-phrase tree transformation model and a phrase reordering model. The decoding algo-

[1] See Section 6.2.

CoNLL 2008: Proceedings of the 12th Conference on Computational Natural Language Learning, pages 143–150
Manchester, August 2008

rithm is a tree-based search algorithm.

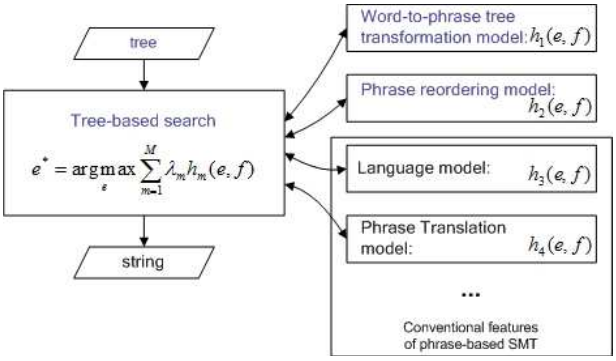
tree
Tree-based search
$e^* = \underset{e}{\operatorname{argmax}} \sum_{m=1}^{M} \lambda_m h_m(e, f)$
string
Word-to-phrase tree transformation model: $h_1(e, f)$
Phrase reordering model: $h_2(e, f)$
Language model: $h_3(e, f)$
Phrase Translation model: $h_4(e, f)$
...
Conventional features of phrase-based SMT

Figure 1: A syntax-directed phrase-based SMT architecture.

2 Translation Model

We use an example of English-Vietnamese translation to demonstrate the translation process as in Figure 2. Now we describe a tree-to-string SMT model based on synchronous CFGs. The translation process is:

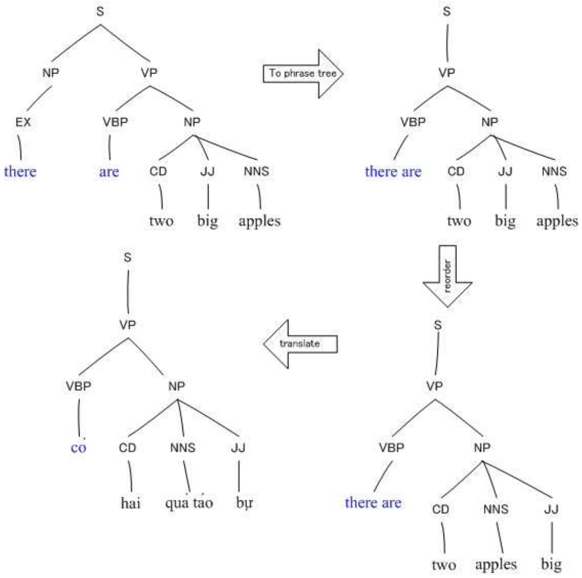

Figure 2: The translation process.

$$T_1 \rightarrow T_2 \rightarrow T_3 \rightarrow T_4 \quad (1)$$

where T_1 is a source tree, T_2 is a source phrase tree, T_3 is a reordered source phrase tree, and T_4 is a target phrase tree.

Using the first order chain rule, the join probability over variables (trees) in graphical representation 1 is approximately calculated by:

$$P(T_1, T_2, T_3, T_4) = P(T_1) \times P(T_2|T_1) \times P(T_3|T_2) \times P(T_4|T_3) \quad (2)$$

$P(T_1)$ can be omitted since only one syntactic tree is used. $P(T_2|T_1)$ is a word-to-phrase tree transformation model we describe later. $P(T_3|T_2)$ is a reordering model. $P(T_4|T_3)$ can be calculated using a phrase translation model and a language model. This is the fundamental equation of our study represented in this paper. In the next section, we will describe how to transform a word-based CFG tree into a phrase-based CFG tree.

3 Word-to-Phrase Tree Transformation

3.1 Penn Treebank's Tree Structure

According to this formalism, a tree is represented by phrase structure. If we extract a CFG from a tree or set of trees, there will be two possible rule forms:

- $A \rightarrow \alpha$ where α is a sequence of nonterminals (syntactic categories).
- $B \rightarrow \gamma$ where γ is a terminal symbol (or a word in this case).

We consider an example of a syntactic tree and a simple CFG extracted from that tree.

Sentence: *"I am a student"*
Syntactic tree: *(S (NP (NN I)) (VP (VBP am) (NP (DT a) (NN student))))*
Rule set: *S → NP VP; VP → VBP NP; NP → NN | DT NN; NN → I | student;*
VBP → am; DT → a

However, we are considering phrase-based translation. Therefore the right hand side of the second rule form must be a sequence of terminal symbols (or a phrase) but not a single symbol (a word). Suppose that the phrase table contains a phrase *"am a student"* which leads to the following possible tree structure:

Phrase segmentation: *"I | am a student"*
Syntactic tree: *(S (NP (NN I)) (VP (VBP am a student)))*
Rule set: *S → NP VP; VP → VBP; NP → NN; NN → I; VBP → am a student*

We have to find out some way to transform a CFG tree into a tree with phrases at leaves. In the next subsection we propose such an algorithm.

3.2 An Algorithm for Word-to-Phrase Tree Transformation

Table 1 represents our algorithm to transform a CFG tree to a phrase CFG tree. When designing this algorithm, our criterion is to preserve the original structure as much as possible. This algorithm includes two steps. There are a number of notions concerning this algorithm:

- A CFG rule has a head symbol on the right hand side. Using this information, head child of a node on a syntactic tree can be determined.

+ Input:	A CFG tree, a phrase segmentation
+ Output:	A phrase CFG tree
+ Step 1:	Allocate phrases to leaf nodes in a top-down manner: A phrase is allocated to head word of a node if the phrase contains the head word. This head word is then considered as the phrase head.
+ Step 2:	Transform the syntactic tree by replacing leaf nodes by their allocated phrase and removing all nodes whose span is a substring of phrases.

Table 1: An algorithm to transform a CFG tree to a phrase CFG tree.

- If a node is a pre-terminal node (containing POS tag), its head word is itself. If a node is an inner node (containing syntactic constituent tag), its head word is retrieved through the head child.
- Word span of a node is a string of its leaves. For instance, word span of subtree (NP (PRP$ your) (NN class)) is ”your class”.

Now we consider an example depicted in Figure 3 and 4. Head children are tagged with functional label H. There are two phrases: ”is a” and ”in your class”. After the Step 1, the phrase ”is a” is attached to (VBZ is). The phrase ”in your class” is attached to (IN in). In Step 2, the node (V is) is replaced by (V ”is a”) and (DT a) is removed from its father NP. Similarly, (IN in) is replaced by (IN ”in your class”) and the subtree NP on the right is removed.

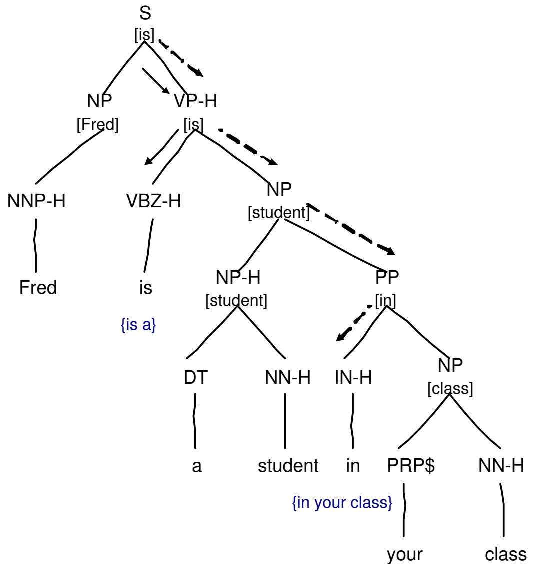

Figure 3: Tree transformation - step 1. Solid arrows show the allocation process of ”is a”. Dotted arrows demonstrate the allocation process of ”in your class”

The proposed algorithm has some properties. We state these properties without presenting proof[2].

- **Uniqueness:** Given a CFG tree and a phrase segmentation, by applying Algorithm 1, one and only one phrase tree is generated.

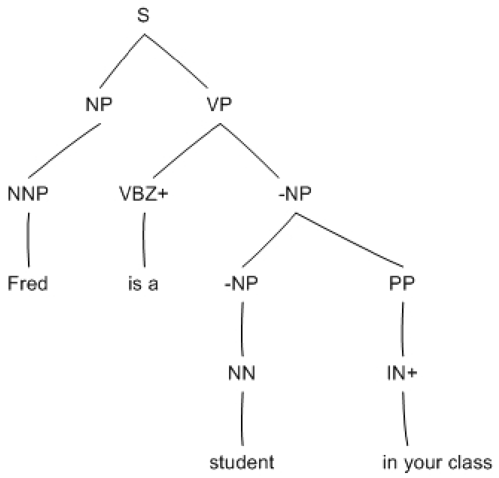

Figure 4: Tree transformation - step 2.

- **Constituent subgraph:** A phrase CFG tree is a connected subgraph of input tree if leaves are ignored.
- **Flatness:** A phrase CFG tree is flatter than input tree.
- **Outside head:** The head of a phrase is always a word whose head outside the phrase. If there is more than one word satisfying this condition, the word at the highest level is chosen.
- **Dependency subgraph:** Dependency graph of a phrase CFG tree is a connected subgraph of input tree’s dependency graph if there exist no detached nodes.

The meaning of uniqueness property is that our algorithm is a deterministic procedure. The constituent-subgraph property will be employed in the next section for an efficient decoding algorithm. When a syntactic tree is transformed, a number of subtrees are replaced by phrases. The head word of a phrase is the contact point of that phrase with the remaining part of a sentence. From the dependency point of view, a head word should depend on an outer word rather than an inner word. About dependency-subgraph property, when there is a detached node, an indirect dependency will become a direct one. In any cases, there is no

[2] Proofs are simple.

change in dependency direction. We can observe dependency trees in Figure 5. The first two trees are source dependency tree and phrase dependency tree of the previous example. The last one corresponds to the case in which a detached node exists.

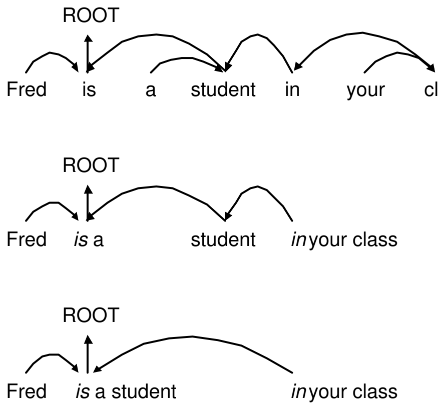

Figure 5: Dependency trees. The third tree corresponds with phrase segmentation: ”Fred | is a student | in your class”

3.3 Probabilistic Word-to-Phrase Tree Transformation

We have proposed an algorithm to create a phrase CFG tree from a pair of CFG tree and phrase segmentation. Two questions naturally arise: ”is there a way to evaluate how good a phrase tree is?” and ”is such an evaluation valuable?” Note that phrase trees are the means to reorder the source sentence represented as phrase segmentations. Therefore a phrase tree is surely not good if no right order can be generated. Now the answer to the second question is clear. We need an evaluation method to prevent our program from generating bad phrase trees. In other words, good phrase trees should be given a higher priority.

We define the phrase tree probability as the product of its rule probability given the original CFG rules:

$$P(T') = \prod_i P(LHS_i \rightarrow RHS'_i | LHS_i \rightarrow RHS_i) \tag{3}$$

where T' is a phrase tree whose CFG rules are $LHS_i \rightarrow RHS'_i$. $LHS_i \rightarrow RHS_i$ are original CFG rules. RHS'_i are subsequences of RHS_i. Since phrase tree rules should capture changes made by the transformation from word to phrase, we use ’+’ to represent an expansion and ’-’ to show an overlap. These symbol will be added to a nonterminal on the side having a change. In the previous example, since a head noun in the word tree has been expanded on the right, the corresponding symbol in phrase tree is NN-H+. A nonterminal X can become one of the following symbols $X, -X, +X, X-, X+, -X-, -X+, +X-, +X+$.

Conditional probabilities are computed in a separate training phase using a source-parsed and word-aligned bitext. First, all phrase pairs consistent with the word alignment are collected. Then using this phrase segmentation and syntactic trees we can generate phrase trees by word-to-phrase tree transformation and extract rules.

4 Phrase Reordering Model

Reordering rules are represented as SCFG rules which can be un-lexicalized or source-side lexicalized (Nguyen and Shimazu, 2006). In this paper, we used un-lexicalized rules. We used a learning algorithm as in (Nguyen and Shimazu, 2006) to learn weighted SCFGs. The training requirements include a bilingual corpus, a word alignment tool, and a broad coverage parser of the source language. The parser is a constituency analyzer which can produce parse tree in Penn Tree-bank’s style. The model is applicable to language pairs in which the target language is poor in resources. We used phrase reorder rules whose ’+’ and ’-’ symbols are removed.

5 Decoding

A source sentence can have many possible phrase segmentations. Each segmentation in combination with a source tree corresponds to a phrase tree. A phrase-tree forest is a set of those trees. A naive decoding algorithm is that for each segmentation, a phrase tree is generated and then the sentence is translated. This algorithm is very slow or even intractable. Based on the constituent-subgraph property of the tree transformation algorithm, the forest of phrase trees will be packed into a tree-structure container whose backbone is the original CFG tree.

5.1 Translation Options

A translation option encodes a possibility to translate a source phrase (at a leaf node of a phrase tree) to another phrase in target language. Since our decoder uses a log-linear translation model, it can exploit various features of translation options. We use the same features as (Koehn et al., 2003). Basic information of a translation option includes:

- source phrase
- target phrase
- phrase translation score (2)

- lexical translation score (2)
- word penalty

Translation options of an input sentence are collected before any decoding takes place. This allows a faster lookup than consulting the whole phrase translation table during decoding. Note that the entire phrase translation table may be too big to fit into memory.

5.2 Translation Hypotheses

A translation hypothesis represents a partial or full translation of an input sentence. Initial hypotheses correspond to translation options. Each translation hypothesis is associated with a phrase-tree node. In other words, a phrase-tree node has a collection of translation hypotheses. Now we consider basic information contained in a translation hypothesis:

- the cost so far
- list of child hypotheses
- left language model state and right language model state

5.3 Decoding Algorithm

First we consider structure of a syntactic tree. A tree node contains fields such as syntactic category, child list, and head child index. A leaf node has an additional field of word string. In order to extend this structure to store translation hypotheses, a new field of hypothesis collection is appended. A hypothesis collection contains translation hypotheses whose word spans are the same. Actually, it corresponds to a phrase-tree node. A hypothesis collection whose word span is $[i_1, i_2]$ at a node whose tag is X expresses that:

- There is a phrase-tree node (X, i_1, i_2).
- There exist a phrase $[i_1, i_2]$ or
- There exist a subsequence of X's child list: (Y_1, j_0, j_1), (Y_2, j_1+1, j_2), ..., $(Y_n, j_{n-1}+1, j_n)$ where $j_0 = i_1$ and $j_n = i_2$
- Suppose that $[i, j]$ is X's span, then $[i_1, i_2]$ is a valid phrase node's span if and only if: $i_1 <= i$ or $i < i_1 <= j$ and there exist a phrase $[i_0, i_1 - 1]$ overlapping X's span at $[i, i_1 - 1]$. A similar condition is required of j.

Table 2 shows our decoding algorithm. Step 1 distributes translation options to leaf nodes using a procedure similar to Step 1 of algorithm in Table 1. Step 2 helps check valid subsequences in Step 3 fast. Step 3 is a bottom-up procedure, a node is translated if all of its child nodes have been translated. Step 3.1 calls syntactic transformation models. After reordered in Step 3.2, a subsequence will be translated in Step 3.3 using a simple monotonic decoding procedure resulting in new translation hypotheses. We used a beam pruning technique to reduce the memory cost and to accelerate the computation.

Corpus	Size	Training	Development	Testing
Conversation	16,809	15,734	403	672
Reuters	57,778	55,757	1,000	1,021

Table 3: Corpora and data sets.

	English		Vietnamese
Sentences		16,809	
Average sent. len.	8.5		8.0
Words	143,373		130,043
Vocabulary	9,314		9,557
	English		Japanese
Sentences		57,778	
Average sent. len.	26.7		33.5
Words	1,548,572		1,927,952
Vocabulary	31,702		29,406

Table 4: Corpus statistics of translation tasks.

6 Experimental Results

6.1 Experimental Settings

We used Reuters[3], an English-Japanese bilingual corpus, and Conversation, an English-Vietnamese corpus (Table 4). These corpora were split into data sets as shown in Table 3. Japanese sentences were analyzed by ChaSen[4], a word-segmentation tool.

A number of tools were used in our experiments. Vietnamese sentences were segmented using a word-segmentation program (Nguyen et al., 2003). For learning phrase translations and decoding, we used Pharaoh (Koehn, 2004), a state-of-the-art phrase-based SMT system which is available for research purpose. For word alignment, we used the GIZA++ tool (Och and Ney, 2000). For learning language models, we used SRILM toolkit (Stolcke, 2002). For MT evaluation, we used BLEU measure (Papineni et al., 2001) calculated by the NIST script version 11b. For the parsing task, we used Charniak's parser (Charniak, 2000). For experiments with chunking (or shallow parsing), we used a CRFs-based chunking tool [5] to split a source sentence into syntactic chunks. Then a pseudo CFG rule over chunks is built to generate a two-level syntactic tree. This tree can be used in the

[3] http://www2.nict.go.jp/x/x161/members/mutiyama/index.html

[4] http://chasen.aist-nara.ac.jp/chasen/distribution.html.en

[5] http://crfpp.sourceforge.net/

+ Input:	A source CFG tree, a translation-option collection
+ Output:	The best target sentence
+ Step 1:	Allocate translation options to hypothesis collections at leaf nodes.
+ Step 2:	Compute overlap vector for all nodes.
+ Step 3:	For each node, if all of its children have been translated, then for each valid sub-sequence of child list, carry out the following steps:
+ Step 3.1:	Retrieve transformation rules
+ Step 3.2:	Reorder the sub-sequence
+ Step 3.3:	Translate the reordered sub-sequence and update corresponding hypothesis collections

Table 2: A bottom-up dynamic-programming decoding algorithm.

Corpus	CFG	PhraseCFG	W2PTT	Reorder
Conversation	2,784	2,684	8,862	2,999
Reuters	7,668	5,479	13,458	7,855

Table 5: Rule induction statistics.

Corpus	Pharaoh	PB system	SD system (chunking)	SD system (full-parsing)
Conversation	35.47	35.66	36.85	37.42
Reuters	24.41	24.20	20.60	25.53

Table 6: BLEU score comparison between phrase-based SMT and syntax-directed SMT. PB=phrase-based; SD=syntax-directed

same way as trees produced by Charniak's parser.

We built a SMT system for phrase-based log-linear translation models. This system has two decoders: beam search and syntax-based. We implemented the algorithm in Section 5 for the syntax-based decoder. We also implemented a rule induction module and a module for minimum error rate training. We used the system for our experiments reported later.

6.2 Rule Induction

In Table 5, we report statistics of CFG rules, phrase CFG rules, word-to-phrase tree transformation (W2PTT) rules, and reordering rules. All counted rules were in un-lexicalized form. Those numbers are very small in comparison with the number of phrasal translations (up to hundreds of thousands on our corpora). There were a number of "un-seen" CFG rules which did not have a corresponding reordering rule. A reason is that those rules appeared once or several times in the training corpus; however, their hierarchical alignments did not satisfy the conditions for inducing a reordering rule since word alignment is not perfect (Nguyen and Shimazu, 2006). Another reason is that there were CFG rules which required nonlocal reordering. This may be an issue for future research: a Markovization technique for SCFGs.

6.3 BLEU Scores

Table 6 shows a comparison of BLEU scores between Pharaoh, our phrase-based SMT system, and our syntax-directed (SD) SMT system with chunking and full parsing respectively. On both Conversation corpus and Reuters corpus: The BLEU score of our phrase-based SMT system is comparable to that of Pharaoh; The BLEU score of our SD system with full parsing is higher than that of our phrase-based system. On Conversation corpus, our SD system with chunking has a higher performance in terms of BLEU score than our phrase-based system. Using sign test (Lehmann, 1986), we verified the improvements are statistically significant. However, on Reuters corpus, performance of the SD system with chunking is much lower than the phrase-based system's. The reason is that in English-Japanese translation, chunk is a too shallow syntactic structure to capture word order information. For example, a prepositional chunk often includes only preposition and adverb, therefore such information does not help reordering prepositional phrases.

6.4 The Effectiveness of the W2PTT Model

Without this feature, BLEU scores decreased around 0.5 on both corpora. We now consider a linguistically motivated example of English-Vietnamese translation to show that phrase segmentation can be evaluated through phrase tree scoring. This example was extracted from Conversation test set.

English sentence: *for my wife 's mother*

Vietnamese word order: *for mother 's wife my*

Phrase segmentation 1: *for my wife | 's | mother*

P1=P(PP→IN+ -NP | PP→IN NP)xP(-NP→-NP NN | NP→NP NN)xP(-NP→POS | NP→PRP$ NN POS)=log(0.00001)+log(0.14)+log(0.048)=-5-0.85-1.32=-7.17

Phrase segmentation 2: *for | my wife 's | mother*

P2=P(PP→IN NP | PP→IN NP)xP(NP→NP NN | NP→NP NN) xP(NP→POS | NP→PRP$ NN POS) =log(0.32)+log(0.57)+log(0.048)=-0.5-0.24-1.32=-2.06

The first phrase segmentation is bad (or even unacceptable) since the right word order can not be achieved from this segmentation by phrase reordering and word reordering within phrases. The second phrase segmentation is much better. Source syntax tree and phrase trees are shown in Figure 6. The first phrase tree has a much smaller probability (P1=-7.17) than the second (P2=-2.06).

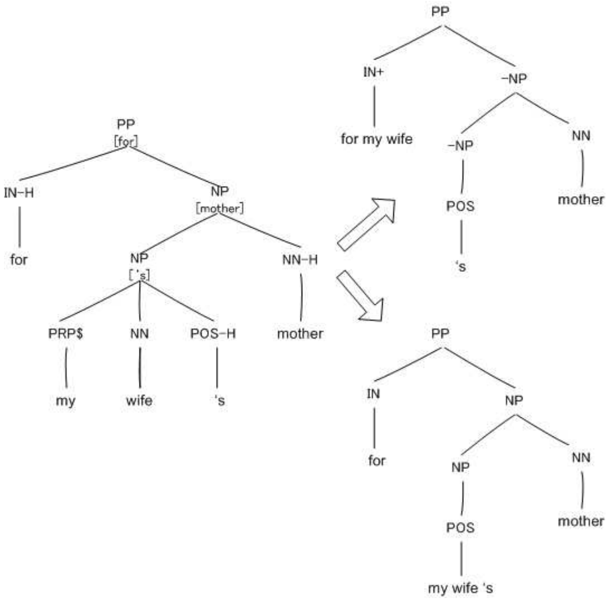

Figure 6: Two phrase trees.

Corpus	Level-1	Level-2	Level-3	Level-4	Full
Conversation	36.85	36.91	37.11	37.23	37.42
Reuters	20.60	22.76	24.49	25.12	25.53

Table 7: BLEU score with different syntactic levels. Level-i means syntactic transformation was applied to tree nodes whose level smaller than or equal to i. The level of a pre-terminal node (POS tag) is 0. The level of an inner node is the maximum of its children's levels.

6.5 Levels of Syntactic Analysis

Since in practice, chunking and full parsing are often used, in Table 6, we showed translation quality of the two cases. It is interesting if we can find how syntactic analysis can affect BLEU score at more intermediate levels (Table 7). On the Conversation corpus, using syntax trees of level-1 is effective in comparison with baseline. The increase of syntactic level makes a steady improvement in translation quality. Note that when we carried out experiments with chunking (considered as level-1 syntax) the translation speed (including chunking) of our tree-to-string system was much faster than baseline systems'. This is an option for developing applications which require high speed such as web translation.

7 Related Works

7.1 A Comparison of Syntactic SMT Methods

To advance the state of the art, SMT system designers have experimented with tree-structured translation models. The underlying computational models were synchronous context-free grammars and finite-state tree transducers which conceptually have a better expressive power than finite-state transducers. We create Tables 8 and 9 in order to compare syntactic SMT methods including ours. The first row is a baseline phrasal SMT approach. The second column in Table 8 only describes input types because the output is often string. Syntactic SMT methods are different in many aspects. Methods which make use of phrases (in either explicit or implicit way) can beat the baseline approach (Table 8) in terms of BLEU metric. Two main problems these models aim to deal with are word order and word choice. In order to accomplish this purpose, the underlying formal grammars (including synchronous context-free grammars and tree transducers) can be fully lexicalized or unlexicalized (Table 9).

7.2 Non-constituent Phrasal Translations

Liu et al. (2007) proposed forest-to-string rules to capture non-constituent phrasal translation while our system can naturally make use of such kind of phrasal translation by using word-to-phrase tree transformation. Liu et al. (2007) also discussed about how the phenomenon of non-syntactic bilingual phrases is dealt with in other SMT methods. Galley et al. (2006) handled non-constituent phrasal translation by traversing the tree upwards until reaches a node that subsumes the phrase. Marcu et al. (2006) reported that approximately 28% of bilingual phrases are non-syntactic on their English-Chinese corpus. They proposed using a pseudo nonterminal symbol that subsumes the phrase and corresponding multi-headed syntactic structure. One new xRs rule is required to explain how the new nonterminal symbol can be combined with others. This technique brought a significant improvement in performance to their string-to-tree noisy channel SMT system.

8 Conclusions

We have presented a general tree-to-string phrase-based method. This method employs a syntax-based reordering model in the decoding phase. By word-to-phrase tree transformation, all possible phrases are considered in translation. Our method does not suppose a uniform distribution over all possible phrase segmentations as (Koehn et al., 2003) since each phrase tree has a probability. We believe that other kinds of translation unit such as n-gram (Jos et al., 2006), factored phrasal translation (Koehn and Hoang, 2007), or treelet (Quirk et al., 2005) can be used in this method. We would like to consider this problem as a future study. Moreover we would like to use n-best trees as the input of our system. A number

Method	Input	Theoretical model	Decoding style	Linguistic information	Phrase usage	Performance
Koehn et al. (2003)	string	FSTs	beam search	no	yes	baseline
Yamada and Knight (2001)	string	SCFGs	parsing	target	no	not better
Melamed (2003)	string	SCFGs	parsing	both sides	no	not better
Chiang (2005)	string	SCFGs	parsing	no	yes	better
Quirk et al. (2005)	dep. tree	TTs	parsing	source	yes	better
Galley et al. (2006)	string	TTs	parsing	target	yes	better
Liu et al. (2006)	tree	TTs	tree transf.	source	yes	better
Our work	tree	SCFGs	tree transf.	source	yes	better

Table 8: A comparison of syntactic SMT methods (part 1). FST=Finite State Transducer; SCFG=Synchronous Context-Free Grammar; TT=Tree Transducer.

Method	Rule form	Rule function	Rule lexicalization level
Koehn et al. (2003)	no	no	no
Yamada and Knight (2001)	SCFG rule	reorder and function-word ins./del.	unlexicalized
Melamed (2003)	SCFG rule	reorder and word choice	full
Chiang (2005)	SCFG rule	reorder and word choice	full
Quirk et al. (2005)	Treelet pair	word choice	full
Galley et al. (2006)	xRs rule	reorder and word choice	full
Liu et al. (2006)	xRs rule	reorder and word choice	full
Our work	SCFG rule	reorder	unlexicalized

Table 9: A comparison of syntactic SMT methods (part 2). xRs is a kind of rule which maps a syntactic pattern to a string, for example VP(AUX(does), RB(not),x_0:VB) → ne, x_0, pas. In the column Rule lexicalization level: full=lexicalization using vocabularies of both source language and target language.

of non-local reordering phenomena such as adjunct attachment should be handled in the future.

References

Charniak, E. 2000. A maximum entropy inspired parser. In *Proceedings of HLT-NAACL*.

Galley, M., Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, Ignacio Thayer 2006. Scalable Inference and Training of Context-Rich Syntactic Translation Models. In *Proceedings of ACL*.

Jos B. Mario, Rafael E. Banchs, Josep M. Crego, Adri de Gispert, Patrik Lambert, Jos A. R. Fonollosa, Marta R. Costa-juss. 2006. N-gram-based Machine Translation. *Computational Linguistics*, 32(4): 527–549.

Koehn, P. 2004. Pharaoh: a beam search decoder for phrase-based statistical machine translation models. In *Proceedings of AMTA*.

Koehn, P. and Hieu Hoang. 2007. Factored Translation Models. In *Proceedings of EMNLP*.

Koehn, P., F. J. Och, and D. Marcu. 2003. Statistical phrase-based translation. In *Proceedings of HLT-NAACL*.

Lehmann, E. L. 1986. Testing Statistical Hypotheses (Second Edition). Springer-Verlag.

Liu, Y., Qun Liu, Shouxun Lin. 2006. Tree-to-String Alignment Template for Statistical Machine Translation. In *Proceedings of ACL*.

Liu, Y., Yun Huang, Qun Liu, and Shouxun Lin 2007. Forest-to-String Statistical Translation Rules. In *Proceedings of ACL*.

Marcu, D., Wei Wang, Abdessamad Echihabi, and Kevin Knight. 2006. SPMT: Statistical Machine Translation with Syntactified Target Language Phrases. In *Proceedings of EMNLP*.

Melamed, I. D. 2004. Statistical machine translation by parsing. In *Proceedings of ACL*.

Nguyen, Thai Phuong and Akira Shimazu. 2006. Improving Phrase-Based Statistical Machine Translation with Morphosyntactic Transformation. *Machine Translation*, 20(3): 147–166.

Nguyen, Thai Phuong, Nguyen Van Vinh and Le Anh Cuong. 2003. Vietnamese Word Segmentation Using Hidden Markov Model. In *Proceedings of International Workshop for Computer, Information, and Communication Technologies in Korea and Vietnam*.

Och, F. J. and H. Ney. 2000. Improved statistical alignment models. In *Proceedings of ACL*.

Papineni, K., S. Roukos, T. Ward, W.-J. Zhu. 2001. BLEU: a method for automatic evaluation of machine translation. Technical Report RC22176 (W0109-022), IBM Research Report.

Quirk, C., A. Menezes, and C. Cherry. 2005. Dependency treelet translation: Syntactically informed phrasal SMT. In *Proceedings of ACL*.

Stolcke, A. 2002. SRILM - An Extensible Language Modeling Toolkit. In *Proc. Intl. Conf. Spoken Language Processing*.

Yamada, K. and K. Knight. 2001. A syntax-based statistical translation model. In *Proceedings of ACL*.