Non-isomorphic Forest Pair Translation

Hui Zhang[1, 2, 3] **Min Zhang**[1] **Haizhou Li**[1] **Eng Siong Chng**[2]

[1]Institute for Infocomm Research
[2]Nanyang Technological University
[3]USC Information Science Institute
huizhang.fuan@gmail.com {mzhang, hli}@i2r.a-star.edu.sg aseschng@ntu.edu.sg

Abstract

This paper studies two issues, non-isomorphic structure translation and target syntactic structure usage, for statistical machine translation in the context of forest-based tree to tree sequence translation. For the first issue, we propose a novel non-isomorphic translation framework to capture more non-isomorphic structure mappings than traditional tree-based and tree-sequence-based translation methods. For the second issue, we propose a parallel space searching method to generate hypothesis using tree-to-string model and evaluate its syntactic goodness using tree-to-tree/tree sequence model. This not only reduces the search complexity by merging spurious-ambiguity translation paths and solves the data sparseness issue in training, but also serves as a syntax-based target language model for better grammatical generation. Experiment results on the benchmark data show our proposed two solutions are very effective, achieving significant performance improvement over baselines when applying to different translation models.

1 Introduction

Recently syntax-based methods have achieved very promising results and attracted increasing interests in statistical machine translation (SMT) research community due to their ability to provide informative context structure information and convenience in carrying out word transformation and sub-span reordering. Fundamentally, syntax-based SMT views translation as a structural transformation process. Generally speaking, from modeling viewpoint, a syntax-based model tries to convert the source structures into target structures iteratively and recursively while from decoding viewpoint a syntax-based system segments an input tree/forest into many sub-fragments, translates each of them separately, combines the translated sub-fragments and then finds out the best combinations. Therefore, from bilingual viewpoint, we face two fundamental problems: the mapping between bilingual structures and the way of carrying out the target structures combination.

For the first issue, a number of models have been proposed to model the structure mapping between tree and string (Galley *et al.*, 2004; Liu *et al.*, 2006; Yamada and Knight, 2001; DeNeefe and Knight, 2009) and between tree and tree (Eisner, 2003; Zhang *et al.*, 2007 & 2008; Liu *et al.*, 2009). However, one of the major challenges is that all the current models only allow one-to-one mapping from one source frontier non-terminal node (Galley *et al.*, 2004) to one target frontier non-terminal node in a bilingual translation rule. Therefore, all those translation equivalents with one-to-many frontier non-terminal node mapping cannot be covered by the current state-of-the-art models. This may largely compromise the modeling ability of translation rules.

For the second problem, currently, the combination is driven by only the source side (both tree-to-string model and tree-to-tree model only check the source span compatibility when combining different target structures in decoding) or only the

Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, pages 440–450,
MIT, Massachusetts, USA, 9-11 October 2010.

target side (string to tree model). There is no well study in considering both the source side information and the compatibility between different target syntactic structures during combination. In addition, it is well known that the traditional tree-to-tree models suffer heavily from the data sparseness issue in training and the spurious-ambiguity translation path issue (the same translation with different syntactic structures) in decoding.

In addition, because of the performance limitation of automatic syntactic parser, researchers propose using packed forest (Tomita, 1987; Klein and Manning, 2001; Huang, 2008)[1] instead of 1-best parse tree to carry out training (Mi and Huang, 2008) and decoding (Mi *et al.*, 2008) in order to reduce the side effect caused by parsing errors of the one-best tree. However, when we apply the tree-to-tree model to the bilingual forest structures, both training and decoding become very complicated.

In this paper, to address the first issue, we propose a framework to model the non-isomorphic translation process from source tree fragment to target tree sequence, allowing any one source frontier non-terminal node to be translated into any number of target frontier non-terminal nodes. For the second issue, we propose a technology to model the combination task by considering both sides' syntactic structure information. We evaluate and integrate the two technologies into forest-based tree to tree sequence translation. Experimental results on the NIST-2003 and NIST-2005 Chinese-English translation tasks show that our methods significantly outperform the forest-based tree to string and previous tree to tree models as well as the phrase-based model.

The remaining of the paper is organized as following. Section 2 reviews the related work. In section 3 and section 4, we discuss the proposed forest-based rule extraction (non-isomorphic mapping) and decoding algorithms (target syntax information usage). Finally we report the experimental results in section 5 and conclude the paper in section 6.

2 Related Work

Much effort has been done in the syntax-based translation modeling. Yamada and Knight (2001) propose a string to tree model. Galley *et al.* (2004) propose the GHKM scheme to model the string-to-tree mapping. Liu *et al.* (2006) propose a tree-to-string translation model. Liu *et al.* (2007) propose the tree sequence to string model to capture rules covered by continuous sequence of trees. Shieber (2007), DeNeefe and Knight (2009) and Carreras and Collins (2009) propose synchronous tree adjoin grammar to capture more tree-string mapping beyond the GHKM scheme. Zhang *et al.* (2009a) propose the concept of virtual node to reform a tree sequence as a tree, and design efficient algorithms for tree sequence model in forest context. All these works only consider either the source side or the target side syntax information.

To capture both side syntax contexts, Eisner (2003) studies the bilingual dependency tree-to-tree mapping in conceptual level. Zhang *et al.* (2008) propose tree sequence-based tree-to-tree modeling. Liu *et al.* (2009) propose efficient algorithms for tree-to-tree model in the forest-based training and decoding scheme. One common limitation of the above works is they only allow the one-to-one mapping between each non-terminal frontier node, and thus they suffer from the issue of rule coverage. On the other hand, due to the data sparseness issue and model coverage issue, previous tree-to-tree (Zhang *et al.*, 2008; Liu *et al.*, 2009) decoder has to rely solely on the span information or source side information to combine the target syntactic structures, without checking the compatibility of the merging nodes, in order not to fail many translation paths. Thus, this solution fails to effectively utilize the target structure information.

To address this issue, tree sequence (Liu *et al.*, 2007; Zhang *et al.*, 2008) and virtual node (Zhang *et al.*, 2009a) are two concepts with promising results reported. In this paper, with the help of these two concepts, we propose a novel framework to solve the one-to-many non-isomorphic mapping issue. In addition, our proposed solution of using target syntax information enables our forest-based tree-to-tree sequence translation decoding algorithm to not only capture bilingual forest information but also have almost the same complexity as forest-based tree-to-string translation. This reduces the time/space complexity exponentially.

3 Tree to Tree Sequence Rules

The motivation of introducing tree to tree sequence rules is to add target syntax information to tree-to-string rules. Following, we first briefly review the definition of tree-to-string rules, and then describe the tree-to-tree sequence rules.

[1] A packed forest is a compact representation of a set of trees with sharing substructures; formally, it is defined as a triple a triple$< V, E, S >$, where V is non-terminal node set, E is hyper-edge set and S is leaf node set (i.e. all sentence words). Every node in V covers a consecutive sequence of leaf, every hyper-edge in E connect the father node to its children nodes as in a tree. Figure 8 is a packed forest contains two trees.

3.1 Tree to String Rules

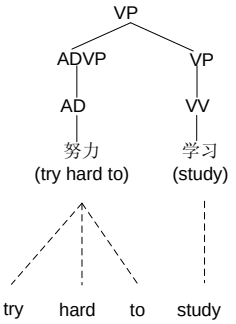

Fig. 1. A word-aligned sentence pair with source tree

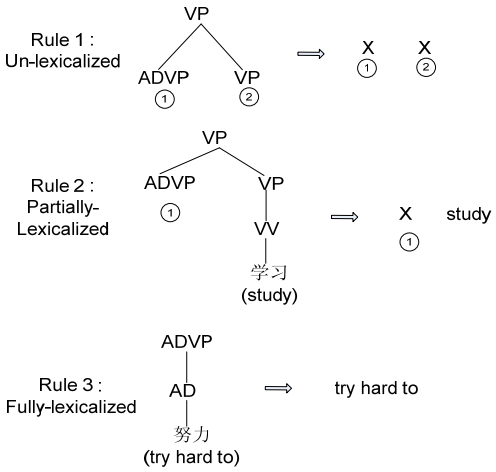

Fig. 2 Examples of tree to string rules

Fig. 2 illustrates the examples of tree to string rules extracted from Fig. 1. The tree-to-string rule is very simple. Its source side is a sub-tree of source parse tree and its target side is a string with only one variable/non-terminal X. The source side and the target side is translation of each other with the constraint of word alignments. Please note that there is no any target syntactic or linguistic information used in the tree-to-string model.

3.2 Tree to Tree Sequence Rules

It is more challenging when extracting rules with target tree structure as constraint. Fig. 3 extends Fig. 1 with target tree structure. The problem is that, given a source tree node, we are able to find its target string translation, but these target string may not form a linguistic sub-tree. For example, in Fig. 3, the source tree node "ADVP" in solid eclipse is translated to "try hard to" in the target sentence, but there is no corresponding sub-tree covering and only covering it in the target side.

Given the example rules in Fig. 2, what are their corresponding rules with target syntax information? The answer is that the previous tree or tree sequence-based models fail to model the Rule 1 and Rule 2 at Fig. 2, since at frontier node level they only allow one-to-one node mapping but the solution is one-to-many non-terminal frontier node mapping. The concept of "virtual node" (Zhang *et al*. 2009a) is a solution to this issue. To facilitate discussion, we first introduce three concepts.

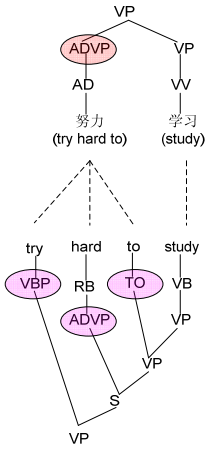

Fig. 3. A word-aligned bi-parsed tree

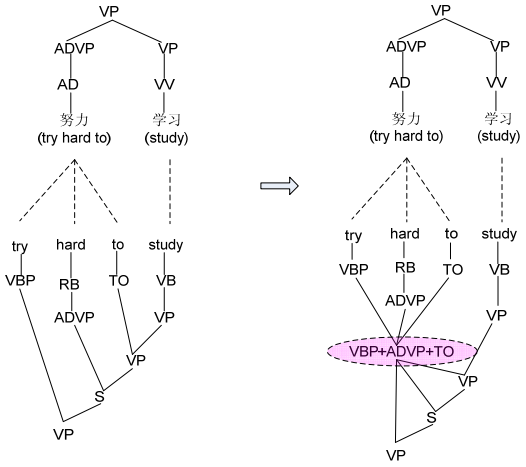

Fig. 4. A restructured tree with a virtual span root

- **Def. 1**. The ***"node sequence"*** is a sequence of nodes (either leaf or internal nodes) covering a consecutive span. For example, in Fig 3, "VBP RB TO" and "VBP ADVP TO" are two "node sequence" covering the same span "try hard to".

- **Def. 2**. The "***root node sequence***" of a span is such a node sequence that any node in this sequence could not be a child of a node in other node sequence of the span. Intuitively, the "*root node sequence*" of a span is the node sequence with the highest topology level. For example, "VBP ADVP TO" is the "root node sequence" of the span of "try hard to". It is easy to prove that given any span, there exist one and only one "*root node sequence*".

- **Def. 3**. The "***span root***" of a span is such a node that if the "*root node sequence*" contains only one tree node, then the "*span root*" is this tree node; otherwise, the "*span root*" is the virtual father node (Zhang *et al*., 2009a) of the "*root node sequence*". Fig. 4 illustrates the reformed Fig. 3 by introducing the virtual node "VBP+ADVP+TO" as the "span root" of the span of "try hard to".

The "*span root*" facilitates us to extract rules with target side structure information. Given a sub-tree of the source tree, we have a set of non-terminal frontier nodes. For each such frontier node, we can find its corresponding target "*span root*". If the "*span root*" is a virtual node, then we add it into the target tree as a virtual segmentation joint point. After adding the "span root" as joint point, we are able to ensure that each frontier source node has only one corresponding target node, then we can use any traditional rule extraction algorithm to extract rules, including those rules with one-to-many non-terminal frontier mappings.

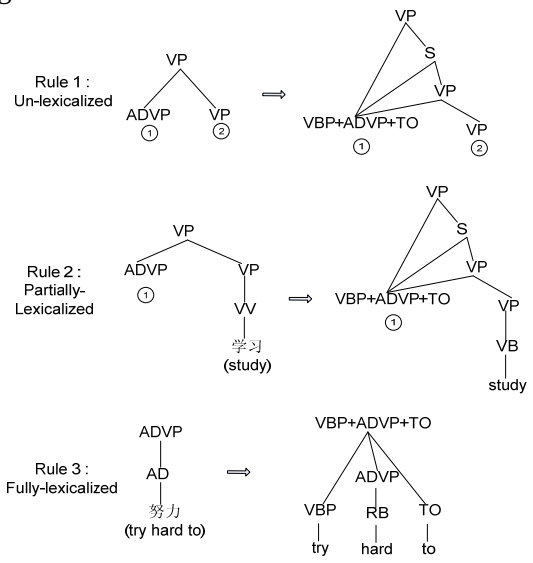

Fig. 5. Tree-to-tree sequence rules

Fig. 5 lists the corresponding rules with target structure information of the tree-to-string rules in Fig 2. All the three rules cannot be extracted by previous tree-to-tree mapping methods (Liu *et al*., 2009). The previous tree-sequence-based methods (Zhang *et al*., 2008; Zhang *et al*., 2009a) can extracted rule 3 since they allow one-to-many mapping in root node level. But they cannot extract rule 1 and rule 2. Therefore, for any tree-to-string rule, our method can always find the corresponding tree-to-tree sequence rule. As a result, our rule coverage is the same as tree-to-string framework while our rules contain more informative target syntax information. Later we will show that using our decoding algorithm the tree-to-tree sequence search space is exponentially reduced to the same as tree-to-string search space. That is to say, we do not need to worry about the exponential search space issue of tree-to-tree sequence model existing in previous work.

3.3 Rule Extraction in Tree Context

Given a word aligned tree pair, we first extract the set of minimum tree to string rules (Galley *et al*. 2004), then for each tree-to-string rule, we can easily extract its corresponding tree-to-tree sequence rule by introducing the virtual span root node. After that, we generate the composite rules by iteratively combining small rules.

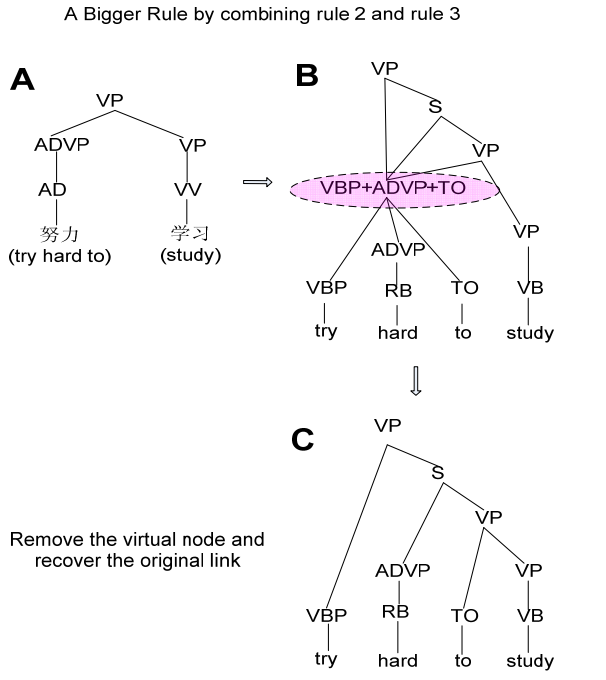

Fig. 6. Rule combination and virtual node removing

Please note that in generating composite rules, if the joint node is a virtual node, we have to recover the original link and remove this virtual node to avoid unnecessary ambiguity. Fig. 6 illustrates the combination process of rule 2 and rule 3 in Fig. 5. As a result, all of our extract rules do not contain any internal virtual nodes.

3.4 Rule Extraction in Forest Context

In forest pair context, we also first generate the minimum tree-to-string rule set as Mi *et al*. (2008), and for each tree-to-string rule, we find its corresponding tree-to-tree sequence rules, and then do rule composition.

In tree pair context, given a tree-to-string rule, there is one and only one corresponding tree-to-tree sequence rule. But in forest pair context, given one such tree-to-string rule, there are many corresponding tree-to-tree sequence rules. All these sub-trees form one or more sub-forests[2] of the entire big target forest. If we can identify the sub-forests, i.e., all of the hyper-edges of the sub-forests, we can retrieve all the sub-trees from the sub-forests as the target sides of the corresponding tree-to-tree sequence rules.

Given a source sub-tree, we can obtain the target root span where the target sub-forests start and the frontier spans where the target sub-forests stop. To indentify all the hyper-edges in the sub-forests, we start from every node covering the root span, traverse from top to down, mark all the hyper-edges visited and stop at the node if its span is a sub-span of one of the forest frontier spans or if it is a word node. The reason we stop at the node once it fell into a frontier span (i.e. the span of the node is a sub-span of the frontier span) is to guarantee that given any frontier span, we could stop at the "*root node sequence*" of this span by **Def.** 2.

For example, Fig. 7 is a source sub-tree of rule 2 in Fig. 5 and the circled part in Fig. 8 is one of its corresponding target sub-forests. Its corresponding target root span is [1,4] (corresponding to source root "VP") and its corresponding target frontier span is {[1,3], study[4,4]}. Now given the target forest, we start from node VP[1,4] and traverse from top to down, finally stop at following nodes: VBP[1,1], ADVP[2,2], TO[3,3], study .

[2] All the sub-forests cover the same span. But their roots have different grammar tags as the roots' names. The root may be a virtual span root node in the case of the one-to-many frontier non-terminal node mappings.

Please note that the starting root node must be a single node, being either a normal forest node or a virtual "span root" node. The virtual "span root" node serves as the frontier node of upper rules and root node of the currently being extracted rules. Because we extract rules in a top-to-down manner, the necessary virtual "span root" node for current sub-forest has already been added into the global forest when extracting upper level rules.

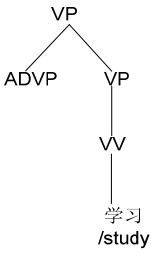

Figure 7. A source sub-tree in rule 2

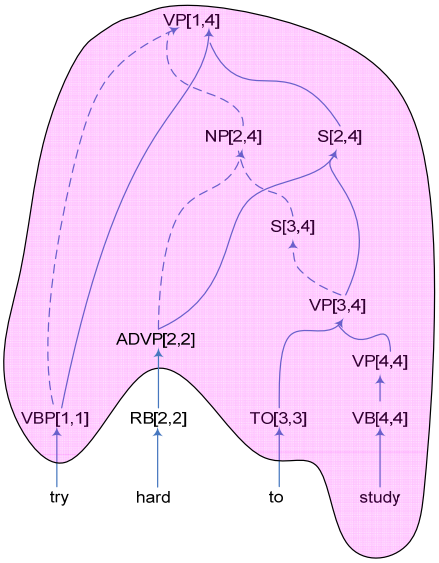

Fig. 8. The corresponding target sub-forest for the tree of Figure 7.

3.5 Fractional Count of Rule

Following Mi and Huang (2008) and Liu *et al*. (2009), we assign a fractional count to a rule to measure how likely it appears given the context of the forest pair. In following equation, "S" means source sub-tree, "T" means target sub-tree, "SF" is source forest and "TF" is the target forest.

$$P(S,T\,|SF,TF) \cong P(S|SF,TF) * P(T|SF,TF)$$
$$\cong P(S|SF) * P(T|TF)$$

The above equation means the fractional count of a source-target tree pair is just the product of each of their fractional count in corresponding forest context in following equation.

$$P(subtree\ |Forest) = \frac{\alpha\beta(subtree)}{\alpha\beta(forest\ root)}$$
$$= \frac{\alpha(subtree\ root) * \prod_{h \in subtree} P(h) * \prod_{v \in leaves(subtree)} \beta(v)}{\alpha\beta(forest\ root)}$$

where α and β are the outside and inside probabilities. In addition, if a sub-tree root is a virtual node (formed by a root node sequence), then we use following equation to approximate the outside probability of the virtual node.

$$\alpha(node\ sequence\ ns) = \sqrt[\#\ of\ nodes\ in\ ns]{\prod_{n \in ns} \alpha(n)}$$

4 Decoding

4.1 Traditional Forest-based Decoding

A typical translation process of a forest-based system is to first convert the source packed forest into a target translation forest, and then apply search algorithm to find the best translation result from this target translation forest (Mi *et al.*, 2008).

For the tree-to-string model, the forest conversion process is as following: given an input packed forest, we do pattern matching (Zhang *et al.*, 2009b) with the source side structures in the rule set. For each matched rule, we establish its target side as a hyper-edge in the target forest.

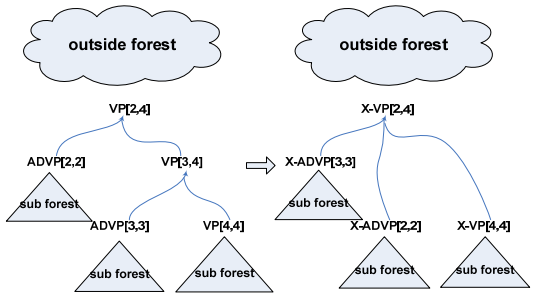

Fig. 9. A forest conversion step in a tree to string model

Fig. 9 exemplifies a conversion step in the tree to string model. A sub-tree structure with two hyper-edge "*VP[2,4] => ADVP[2,2] VP[3,4]*" and "*VP[3,4] => ADVP[3,4] VP[4,4]*" is converted into a target hyper-edge "*X-VP[2,4] => X-ADVP[3,3] X-ADVP[2,2] X-VP[4,4] "*. The node "*X-VP[4,4]*" in the target forest means that its syntactic label in target forest is "X" and it is translated from the source node "VP[4,4]" in the source forest. In this target hyper-edge, "*X-ADVP[3,3] X-ADVP[2,2]*" means the translation from source node "*ADVP[3,3]*" is put before the translation from "*ADVP[2,2]*", representing a structure reordering.

4.2 Toward Bilingual Syntax-aware Translation Generation

As we could see in section 4.1, there is only one kind of non-terminal symbol "X" in the target side. It is a big challenge to rely on such a coarse label to generate a translation with fine syntactic quality. For example, a source node may be translated into a "NP" (noun phrase) in target side. However, in this rule set with the only symbol "X", it may be merged with upper structure as a "VP" (verb phrase) instead, because there is no way to favor one over another. In this case, the target tree does not well model the translation syntactically. In addition, all of the internal structure information in the target side is ignored by the tree-to-string rules.

One natural solution to the above issue is to use the tree to tree/tree sequence model, which have richer target syntax structures for more discriminative probability and finer labels to guide the combination process. However, the tree to tree/tree sequence model may face very severe computational problem and so-called "spurious ambiguities" issue.

Theoretically, if in the tree-to-tree sequence model-based decoding, we just give a penalty to the incompatible-node combinations instead of pruning out the translation paths, then the set of sentences generated by the tree-to-tree sequence model is identical to that of the tree-to-string model since every tree-to-tree sequence rule can be projected into a tree-to-string rule. Motivated by this, we propose a solution call parallel hypothesis spaces searching to solve the computational and "spurious ambiguities" issues mentioned above. In the meanwhile, we can fully utilize the target structure information to guide translation.

We restructure the tree-to-tree sequence rule set by grouping all the rules according to their corresponding tree-to-string rules. This behaves like a "tree-to-forest" rule. The "forest" encodes all the tree sequences with same corresponding string. With the re-constructed rule set, during decoding, we generate two target translation hypothesis spaces (in the form of packed forests) synchronously by the tree-to-string

rules and tree-to-tree sequence rules, and maintain the projection between them. In other words, we generate hypothesis (searching) from the tree-to-string forest and calculate the probability (evaluating syntax goodness) for each hypothesis by the hyper-edges in the tree-to-tree sequence forest.

4.3 Parallel Hypothesis Spaces

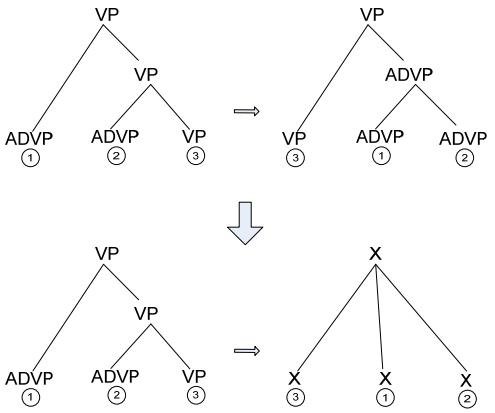

Fig. 10. Mapping from tree-to-tree sequence into tree-to-string rule

In this subsection, we describe what the parallel search spaces are and how to construct them. As shown at Fig. 10, given a tree-to-tree sequence rule, it is easy to find its corresponding tree-to-string rule by simply ignoring the target inside structure and renaming the root and leaves non-terminal labels into "X". We iterate through the tree-to-tree sequence rule set, find its corresponding tree-to-string rule and then group those rules with the same tree-to-string projection. After that, the original tree-to-tree sequence rule set becomes a set of smaller rule sets. Each of them is indexed by a unique tree-to-string rule.

We apply the tree-to-string rules to generate an explicit target translation forest to represent the target sentences space. At the same time, whenever a tree-to-string rule is applied, we also retrieve its corresponding tree-to-tree sequence rule set and generate a set of latent hyper-edges with fine-grained syntax information. In this case, we have two parallel forests, one with coarse explicit hyper-edges and the other fine and latent. Given a hyper-edge (or a node) in the coarse forest, there are a group of corresponding latent hyper-edges (or nodes) with finer syntax labels in the fine forest. Accordingly, given a tree in the coarse forest, there is a corresponding sub-forest in the latent fine forest. We can view the latent fine forest as imbedded inside the explicit coarse forest. If an explicit hyper-edge is viewed as a big cable, then the group of its corresponding latent hyper-edges is the small wires inside it.

We rely on the explicit hyper-edges to enumerate possible hypothesis while using the latent hyper-edges to measure its translation probability and syntax goodness. Thus, the complexity of the search space is reduced into the tree-to-string model level, while keeping the target language generation syntactic aware. More importantly, we thoroughly avoid those spurious ambiguities introduced by the tree-to-tree sequence rules.

4.4 Decoding with Parallel Hypothesis Spaces

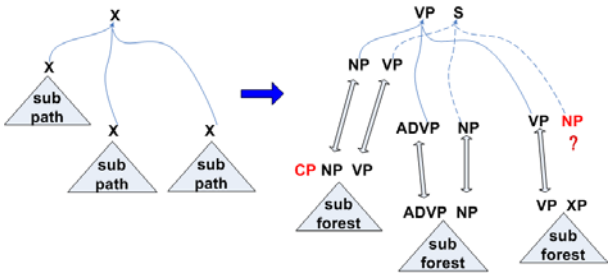

Fig. 11. Derivation path and derivation forest

In this subsection, we show exactly how our decoder finds the best result from the parallel spaces. We generate hypothesis by traversing the coarse forest in the parallel spaces with cube-pruning (Huang and Chiang, 2007). Given a newly generated hypothesis, it is affiliated with a derivation path (tree) in the coarse forest and a group of derivation paths (sub-forest) in the finer forest. As shown in Fig. 11, the left part is the derivation path formed by a coarse hyper-edge, consisting the newly-generated sub-tree "X => X X X" connecting with three previously-generated sub paths while the right part is the derivation forest formed by newly-generated finer hyper-edges rooted at "VP" and "S", and previously-generated sub-forests.

In this paper, we use the sum of probabilities of all the derivation paths in the finer forest to measure the quality of the candidate translation suggested by the hypothesis. From Fig. 11, we can see there may be more than one corresponding finer forests, it is easy to understand that the sum of all the trees' probabilities in these finer forests is equal to the sum of the inside probability of all these root nodes of these finer forests. We adopt the dynamic programming to compute the probability of the finer forest: whenever we generate a new hypothesis by concatenating a

coarse hyper-edge and its sub-path, we find its corresponding finer hyper-edges and sub-forests, do the combination and accumulate probabilities from bottom to up. For the coarse hyper-edge, because there is only one label "X", any sub-path could be easily concatenated with upper structure covering the same sub-span without the need of checking label compatibility. While for the finer hyper-edges, we only link the root nodes of sub-forests to upper hyper-edges with the same linking node label. This is to guarantee syntactic goodness. In case there are some leaf nodes of the upper hyper-edges fail to find corresponding sub-forest roots with the same label (e.g. the "NP" in red color in the rightmost of Fig 11), we simply link it into the nodes with the least inside probability (among these sub-forests), and at the same time give a penalty score to this combination. If some root nodes of some sub-forest still cannot find upper leaf nodes to concatenate (e.g. the "CP" in red color in Fig. 11), we simply ignore them. After the combination process, it is straightforward to accumulate the inside probability dynamically from bottom up.

5 Experiment

5.1 Experimental Settings

We evaluate our method on the Chinese-English translation task. We first carry out a series empirical study on a set of parallel data with 30K sentence pairs, and then do experiment on a larger data set to ensure that the effectiveness of our method is consistent across data set of different size. We use the NIST 2002 test set as our dev set, and NIST 2003 and NIST 2005 test sets as our test set. A 3-gram language model is trained on the target side of the training data by the SRILM Toolkits (Stolcke, 2002) with modified Kneser-Ney smoothing (Kneser and Ney, 1995). We train Charniak's parser (Charniak, 2000) on CTB5.0 for Chinese and ETB3.0 for English and modify it to output packed forest. GIZA++ (Och and Ney, 2003) and the heuristics "grow-diag-final-and" are used to generate *m-to-n* word alignments. For the MER training (Och, 2003), Koehn's MER trainer (Koehn, 2007) is modified for our system. For significance test, we use Zhang *et al.*'s implementation (Zhang *et al*, 2004). Our evaluation metrics is *case-sensitive closest* BLEU-4 (Papineni *et al.*, 2002). We use following features in our systems: 1) bidirectional tree-to-tree sequence probability, 2) bidirectional tree-to-string probability, 3) bidirectional lexical translation probability, 4) target language model, 5) source tree probability 6) the average number of unmatched nodes in the target forest. 7) the length of the target translation, 8) the number of glue rules used.

5.2 Empirical Study on Small Data

We set forest pruning threshold (Mi *et al.*, 2008) to 8 on both source and target forests for rule extraction. For each source sub-tree, we set its height up to 3, width up to 7 and extract up to 10-best target structures. In decoding, we set the pruning threshold to 10 for the input source forest. Table 1 compares the performance in NIST 2003 data set of our method and several state-of-the-art systems as our baseline.

1) **MOSES**: phrase-based system (Koehn *et al.*, 2007)
2) **FT2S**: forest-based tree-to-string system (Mi and Huang, 2008; Mi *et al.*, 2008)
3) **FT2T**: forest-based tree-to-tree system (Liu *et al.*, 2009).
4) **FT2TS (1to1)**: our forest-based tree-to-tree sequence system, where 1to1 means only one-to-one frontier non-terminal node mapping is allowed, thus the system does not follow our non-isomorphic mapping framework.
5) **FT2TS (1toN)**: our forest-based tree-to-tree sequence system that allows one-to-many frontier non-terminal node mapping by following our non-isomorphic mapping framework

In addition, our proposed parallel searching space (**PSS**) technology can be applied to both tree to tree and tree to sequence systems. Thus in table 1, for the tree-to-tree/tree sequence systems, we report two BLEU scores, one uses this technology (withPSS) and one does not (noPSS).

Model		**BLEU-4**
MOSES		23.39
FT2S		26.10
FT2T	noPSS	23.40
	withPSS	24.46
FT2TS (1to1)	noPSS	25.39
	withPSS	26.58
FT2TS (1toN)	noPSS	26.30
	withPSS	**27.70**

Table 1. Performance comparison of different methods

From Table 1, we can see that:

1) All the syntax-based systems (except FT2T (noPSS) (23.40)) consistently outperform the phrase-based system MOSES significantly ($p < 0.01$), indicating that syntactic knowledge is very useful to SMT.
2) The PSS technology shows significant performance improvement ($p < 0.01$) in all models, which clearly shows effectiveness of the PSS technology in utilizing target structures for target language generation.
3) FT2TS (1toN) significantly outperforms ($p < 0.01$) FT2TS (1to1) in both cases (noPSS and withPSS). This convincingly shows the effectiveness of our non-isomorphic mapping framework in capturing the non-isomorphic structure translation equivalences.
4) Both FT2TS systems significantly outperform FT2T($p < 0.01$). This verifies the effectiveness of tree sequence rules.
5) FT2TS shows different level of performance improvements over FT2S with the best case having 1.6 (27.70-26.10) BLEU score improvement over FT2S. This suggests that the target structure information is very useful, but we need to find a correct way to effectively utilize it.

1to1	1toN	ratio
1735871	2363771	1:1.36

Table 2. Statistics on node mapping in forest, where "1to1" means the number of nodes in source forest that can be translated into one node in target forest and "1toN" means the number of nodes in source forest that have to be translated into more than one node in target forest, where the node refers to non-terminal nodes only

Model	# of rules	T2S covered
FT2T	295732	26.8%
FT2TS(1to1)	631487	57.1%
FT2TS (1toN)	1945168	100%

Table 3. Statistics of rule coverage, where "T2S covered" means the percentage of tree-to-string rules that can be covered by the model

Table 2 studies the node isomorphism between bilingual forest pair. We can see that the non-isomorphic node translation mapping (1toN) accounts for 57.6% (=1.36/(1+1.36)) of all the forest non-terminal nodes with target translation. This means that the one-to-many node mapping is a major issue in structure transformation. It also empirically justifies the importance of our non-isomorphic mapping framework.

Table 3 shows the rule coverage of different bilingual structure mapping model. FT2T only covers 26.8% tree-to-string rules, so it performs worse than FT2S as shown in Table 1. FT2TS (1to1) does not allow one-to-many frontier node mapping, so it could only recover the non-isomorphic node mapping in the root level, while FT2TS (1toN) could make it at both root and leaf levels. Therefore, it is not surprising that in Table 3, FT2TS (1toN) cover many more rules than FT2TS (1to1) because given a source tree, there are many leaves, if any one of them is non-isomorphic, then it could not be covered by the FT2TS (1to1).

Decoding Method	BLEU-4	Speed (sec/sent)
Traditional: FT2TS (1toN) (**noPPS**)	26.30	152.6
Ours: FT2TS (1toN) (**withPPS**)	**27.70**	5.22

Table 4. Performance and speed comparison

Table 4 clearly shows the advantage of our decoder over the traditional one. Ours could not only generate better translation result, but also be 152.6/5.22>30 times faster. This mainly attributes to two reasons: 1) one-to-many frontier node mapping equipments the model with more ability to capture more non-isomorphic structure mappings than traditional models, and 2) "parallel search space" enables the decoder to fully utilize target syntactic information, but keeping the size of search space the same as that a "tree to string" model explores.

5.3 Results on Larger Data Set

We also carry out experiment on a larger dataset consisting of the small dataset used in last section and the FBIS corpus. In total, there are 280K parallel sentence pairs with 9.3M Chinese words and 11.8M English words. A 3-gram language model is trained on the target side of the parallel corpus and the GIGA3 Xinhua portion. We compare our system (FT2TS with 1toN and withPPS) with two state-of-the-art baselines: the phrase-based system MOSES and the forest-based tree-to-string system

implemented by us. Table 5 clearly shows the effectiveness of our method is consistent across small and larger corpora, outperforming FT2S by 1.6-1.8 BLEU and the MOSES by 3.3-4.0 BLEU statistically significantly ($p<0.01$).

Model	BLEU	
	NIST2003	**NIST2005**
MOSES	29.51	27.53
FT2S	31.21	29.72
FT2TS	**32.88**	**31.50**

Table 5. Performance on larger data set

6 Conclusions

In this paper, we propose a framework to address the issue of bilingual non-isomorphic structure mapping and a novel parallel searching space scheme to effectively utilize target syntactic structure information in the context of forest-based tree to tree sequence machine translation. Based on this framework, we design an efficient algorithm to extract tree-to-tree sequence translation rules from word aligned bilingual forest pairs. We also elaborate the parallel searching space-based decoding algorithm and the node label checking scheme, which leads to very efficient decoding speed as fast as the forest-based tree-to-string model does, at the same time is able to utilize informative target structure knowledge. We evaluate our methods on both small and large training data sets and two NIST test sets. Experimental results show our methods statistically significantly outperform the state-of-the-art models across different size of corpora and different test sets. In the future, we are interested in testing our algorithm at forest-based tree sequence to tree sequence translation.

References

Eugene Charniak. 2000. *A maximum-entropy inspired parser*. NAACL-00.

Eugene Charniak, Kevin Knight, and Kenji Yamada. 2003. *Syntax-based language models for statistical machine translation.* MT Summit IX. 40–46.

David Chiang. 2007. *Hierarchical phrase-based translation.*Computational Linguistics, 33(2).

Steve DeNeefe, Kevin Knight. 2009. *Synchronous Tree Adjoining Machine Translation*. EMNLP-2009. 727-736.

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. 273-280.

Liang Huang. 2008. Forest Reranking: *Discriminative Parsing with Non-Local Features.* ACL-HLT-08. 586-594

Liang Huang and David Chiang. 2005. *Better k-best Parsing*. IWPT-05. 53-64

Liang Huang and David Chiang. 2007. *Forest rescoring: Faster decoding with integrated language models*. ACL-07. 144–151

Dan Klein and Christopher D. Manning. 2001. *Parsing and Hypergraphs*. IWPT-2001.

Reinhard Kneser and Hermann Ney. 1995. *Improved backing-off for M-gram language modeling*. ICASSP-95, 181-184

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. 177-180. (poster)

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.

Yang Liu, Yajuan Lü, Qun Liu. 2009. *Improving Tree-to-Tree Translation with Packed Forests.* ACL-09. 558-566

Haitao Mi, Liang Huang, and Qun Liu. 2008. *Forest-based translation*. ACL-HLT-08. 192-199.

Haitao Mi and Liang Huang. 2008. *Forest-based Translation Rule Extraction*. EMNLP-08. 206-214.

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

Franz Josef Och and Hermann Ney. 2003. *A Systematic Comparison of Various Statistical Alignment Models*. Computational Linguistics. 29(1) 19-51.

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

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

Masaru Tomita. 1987. *An Efficient Augmented-Context-Free Parsing Algorithm*. Computational Linguistics 13(1-2): 31-46

Xavier Carreras and Michael Collins. 2009. *Non-projective Parsing for Statistical Machine Translation*. EMNLP-2009. 200-209.

K. Yamada and K. Knight. 2001. *A Syntax-Based Statistical Translation Model.* ACL-01. 523-530.

Hui Zhang, Min Zhang, Haizhou Li, Aiti Aw and Chew Lim Tan. 2009a. *Forest-based Tree Sequence to String Translation Model.* ACL-IJCNLP-09. 172-180.

Hui Zhang, Min Zhang, Haizhou Li, and Chew Lim Tan. 2009b. *Fast Translation Rule Matching for Syntax-based Statistical Machine Translation.* EMNLP-09. 1037-1045.

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

Min Zhang, Hongfei Jiang, Aiti Aw, Haizhou Li, Chew Lim Tan, Sheng Li. 2008. *A Tree Sequence Alignment-based Tree-to-Tree Translation Model.* ACL-HLT-08. 559-567.

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.