Multi-Word Unit Dependency Forest-based Translation Rule Extraction

Hwidong Na **Jong-Hyeok Lee**
Department of Computer Science and Engineering
Pohang University of Science and Technology (POSTECH)
San 31 Hyoja Dong, Pohang, 790-784, Republic of Korea
{leona,jhlee}@postech.ac.kr

Abstract

Translation requires non-isomorphic transformation from the source to the target. However, non-isomorphism can be reduced by learning multi-word units (MWUs). We present a novel way of representating sentence structure based on MWUs, which are not necessarily continuous word sequences. Our proposed method builds a simpler structure of MWUs than words using words as vertices of a dependency structure. Unlike previous studies, we collect many alternative structures in a packed forest. As an application of our proposed method, we extract translation rules in form of a source MWU-forest to the target string, and verify the rule coverage empirically. As a consequence, we improve the rule coverage compare to a previous work, while retaining the linear asymptotic complexity.

1 Introduction

Syntax is the hierarchical structure of a natural language sentence. It is generally represented with tree structures using phrase structure grammar (PSG) or dependency grammar (DG). Although the state-of-the-art statistical machine translation (SMT) paradigm is phrase-based SMT (PBSMT), many researchers have attempted to utilize syntax in SMT to overcome the weaknesses of PBSMT. An emerging paradigm alternative to PBSMT is syntax-based SMT, which embeds the source and/or target syntax in its translation model (TM). Utilizing syntax in TM has two advantages over PBSMT.

Figure 1: A pair of sentences that require long distance reordering (dashed line) and discontinuous translation (thick line)

The first advantage is that syntax eases global reordering between the source and the target language. Figure 1 shows that we need global reordering in a complex real situation, where a verbal phrase requires a long distance movement. PBSMT often fails to handle global reordering, for example, from subject-verb-object (SVO) to SOV transformation where V should be moved far away from the original position in

Proceedings of SSST-5, Fifth Workshop on Syntax, Semantics and Structure in Statistical Translation, pages 41–51,
ACL HLT 2011, Portland, Oregon, USA, June 2011.

Table 1: Statistics of the corresponding target words for the continuous word sequences in the source language, or vice versa. C denotes consistent, O overlapped, D discontinuous, and N null.

Word Alignment	C	O	D	N
Manual	25	60	10	5
Automatic	20	55	15	5

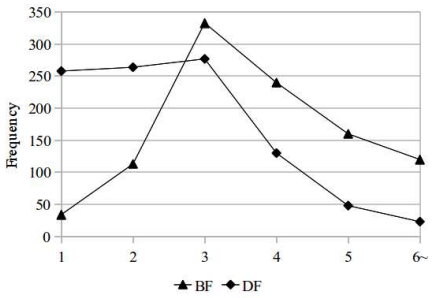

Figure 2: The maximum branching factor (BF) and depth factor (DF) in a dependency tree in our corpus

the source language. This is because of the two distance-based constraints in PBSMT: the distortion model cost and the distortion size limit. For the distortion model cost, PBSMT sets zero cost to the monotone translation and penalizes the distorted translations as the distortion grows larger. For the distortion size limit, a phrase can only be moved from its original position within a limit. Therefore, PBSMT fails to handle long distance reordering. Syntax-based SMT manages global reordering as structural transformation. Because reordering occurs at the substructure level such as constituents or treelets in syntax-based SMT, the transformation of the sub-structure eventually yields the reordering of the whole sentence.

The second advantage of using syntax in TM is that syntax guides us to discontinuous translation patterns. Because PBSMT regards only a continuous sequence of words as a translation pattern, it often fails to utilize many useful discontinuous translation patterns. For example, two discontinuous source words correspond to a target word in Figure 1. In our inspection of the training corpus, a continuous word sequence often corresponds to a set of discontinuous words in the target language, or vice versa (Table 1). Discontinuous translation patterns frequently appear in many languages (Søgaard and Kuhn, 2009). Syntax-based SMT overcomes the limitations of PBSMT because it finds discontinuous patterns along with the hierarchical structure. For example, the two discontinuous source words have a head-dependent relation (Figure 3). Especially with the dependency tree, we can easily identify patterns that have non-projectivity (Na et al., 2010). However, syntax-based patterns such as constituents or treelets do not sufficiently cover various useful patterns, even if we have the correct syntactic analysis (Chiang, 2010). For this reason, many researchers have proposed supplementary patterns such as an intra/inter constituent or sequence of treelets (Galley et al., 2006; Shen et al., 2008).

Unlike PSG, DG does not include nonterminal symbols, which represent constituent information. This makes DG simpler than PSG. For instance, it directly associates syntatic role with the structure, but introduces a difficulty in syntax-based SMT. The branching factor of a dependency tree becomes larger when a head word dominates many dependents. We observe that the maximum branching factor of an automatically parsed dependency tree ranges widely, while most trees have depth under a certain degree (Figure 2). This indicates that we have a horizontally flat dependency tree structure. The translation patterns extracted from the

flat dependency tree are also likely to be flat. Unfortunately, the flat patterns are less applicable at the decoding stage. When one of the modifiers does not match, for instance, we fail to apply the translation pattern. Therefore, we need a more generally applicable representation for syntax-based SMT using DG.

We propose a novel representation of DG that regards a set of words as a unit of the dependency relations, similar to (Ding, 2006; Wu et al., 2009; Na et al., 2010). Unlike their work, we consider many alternatives without predefined units, and construct a packed forest of the multi-word units (MWUs) from a dependency tree. For brevity, we denote the forest based on MWUs as an MWU-forest. Because all possible alternatives are exponentially many, we give an efficient algorithm that enumerates the k-best alternatives in section 3. As an application, we extract translation patterns in form of a source MWU-forest to the target string in order to broaden the coverage of the extracted patterns for syntax-based SMT in section 4. We also report empirical results related to the usefulness of the extracted pattern in section 5. The experimental results show that the MWU-forest representation gives more applicable translation patterns than the original word-based tree.

2 Related Work

Previous studies have proposed merging alternative analyses to deal with analysis errors for two reasons: 1) the strongest alternative is not necessarily the correct analysis, and 2) most alternatives contain similar elements such as common sub-trees. For segmentation alternatives, Dyer et al. (2008) proposed a word lattice that represents exponentially large numbers of segmentations of a source sentence, and integrates reordering information into the lattice as well. For parsing alternatives, Mi et al. (2008) suggested a packed forest that encodes alternative PSG derivations. Futher, Mi et al. (2010) combined the two approaches in order to benefit from both.

The translation literature also shows that translation requires non-isomorphic transformation from the source to the target. This yields translation divergences such as head-switching (Dorr, 1994). Ding and Palmer (2005) reported that the percentage of the head-swapping cases is 4.7%, and that of broken dependencies is 59.3% between Chinese and English. The large amount of non-isomorphism, however, will be reduced by learning MWUs such as elementary trees (Eisner, 2003).

There are few studies that consider a dependency structure based on MWUs. Ding (2006) suggested a packed forest which consists of the elementary trees, and described how to find the best decomposition of the dependency tree. However, Ding (2006) did not show how to determine the MWUs and restrict them to form a subgraph from a head. For opinion mining, Wu et al. (2009) also utilized a dependency structure based on MWUs, although they restricted MWUs with predefined relations. Na et al. (2010) proposed an MWU-based dependency tree-to-string translation rule extraction, but considered only one decomposition for efficiency. Our proposed method includes additional units over Ding's method, such as a sequence of subgraphs within a packed forest. It is also more general than Wu et al.'s method because it does not require any predefined relations. We gain much better rule coverage against Na et al.'s method, while retaining linear asymptotical computational time.

Figure 3: A dependency tree of the source sentence in Figure 1

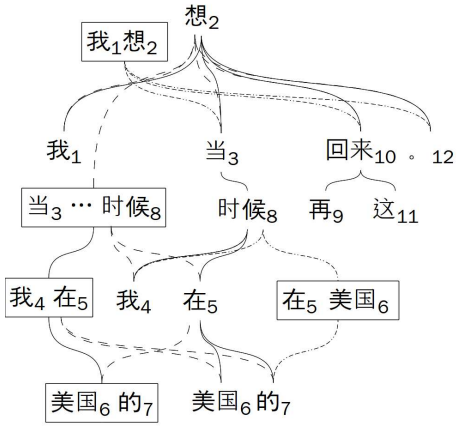

Figure 4: An MWU-forest of Figure 3. The dashed line indicates the alternative hyperedges.

3 MWU-based Dependency Forest

There are two advantages when we use the MWU-forest representaion with DG. First, we express the discontinuous patterns in a vertex, so that we can extract more useful translation patterns beyond continuous ones for syntax-based SMT. Second, an MWU-forest contains many alternative structures which may be simpler structures than the original tree in terms of the branching factor and the maximum depth. Wu et al. (2009) utilized an MWU-tree to identify the product features in a sentence easily.

As in previous literature in syntax-based SMT using DG, we only consider the **well-formed** MWUs where an MWU is either a treelet (a connected sub-graph), or a sequence of treelets under a common head. In other words, each vertex in an MWU-forest is either "fixed on head" or "floating with children". The formal definitions can be found in (Shen et al., 2008).

We propose encoding multiple dependency structures based on MWUs into a hypergraph. A hypergraph is a compact representation of exponetially many variations in a polynominal space. Unlike PSG, DG does not have non-terminals that represent the linguistically motivated, intermediate structure such as noun phrases and verb phrases. For this simplicity, Tu et al. (2010) proposed a dependency forest as a hypergraph, regarding a word as a vertex with a span that ranges for all its descendants. The dependency forest offers tolerence of parsing errors.

Our representation is different from the dependency forest of Tu et al. (2010) since a vertex corresponds to multiple words as well as words. Note that our representation is also capable of incorporating multiple parse trees. Therefore, MWU-forests will also be tolerant of the parsing error if we provide multiple parse trees. In this work, we concentrate on the effectiveness of MWUs, and hence utilize the best dependency parse tree. Figure 4 shows an MWU-forest of the dependency tree in Figure 3.

More formally, a hypergraph $H = \langle V, E \rangle$ consists of the vertices V and hyperedges E. We assume that a length-J sentence has a dependency graph which is single-headed,

acyclic, and rooted, i.e. h_j is the index of the head word of the j-th word, or 0 if the word is the root. Each vertex $v = \{j | j \in [1, J]\}$ denotes a set of the indices of the words that satisfies the well-formed constraint. Each hyperedge $e = \langle tails(e), head(e) \rangle$ denotes a set of the dependency relations between $head(e)$ and $\forall v \in tails(e)$. We include a special node $v_0 \in V$ that denotes the dummy root of an MWU-forest. Note that v_0 does not appear in $tails(e)$ for all hyperedges. We denote $|e|$ is the arity of hyperedge e, i.e. the number of tail nodes, and the arity of a hypergraph is the maximum arity over all hyperedges. Also, let $\sigma(v)$ be the indices of the words that the head lays out of the vertex, i.e. $\sigma(v) = \{j | h_j \notin v \wedge j \in v\}$, and $\tau(v)$ be the indices of the direct dependent words of the vertex, i.e. $\tau(v) = \{j | h_j \in v \wedge j \notin v\}$. Let $OUT(v)$ and $IN(v)$ be the outgoing and incoming hyperedges of a vertex v, respectively.

It is challenging to weight the hyperedges based on dependency grammar because a dependency relation is a binary relation from a head to a dependent. Tu et al. (2010) assigned a probability for each hyperedge based on the score of the binary relation. We simply prefer the hyperedges that have lower arity by scoring as follows:

$$c(e) = \frac{\sum_{v \in tails(e)} |v|}{|e|}$$

$$p(e) = \frac{c(e)}{\sum_{e' \in IN(head(e))} c(e')}$$

We convert a dependency tree into a hypergraph in two steps using the Inside-Outside algorithm. Algorithm 1 shows the pseudo code of our proposed method. At the first step, we find the k-best incoming hyperedges for each vertex (line 3-8), and compute the inside probability (line 9), in bottom-up order. At the second step, we compute the outside probability

Algorithm 1 Build Forest

```
1:  Initialize V
2:  for v ∈ V in bottom-up order do
3:      Create a chart C = |τ(v)|^2
4:      for chart span [p, q] do
5:          Initialize C[p, q] if ∃v s.t. [p, q] = v or
            σ(v)
6:          Combine C[p, i] and C[i + 1, q]
7:      end for
8:      Set IN(v) to the k-best in C[TOP]
9:      Set β(v) as in Eq. 1
10: end for
11: for v ∈ V in top-down order do
12:     Set α(v) as in Eq. 2
13: end for
14: Prune out e if p(e) ≤ δ
15: return v_0
```

(line 12) for each vertex in a top-down manner. Finally we prune out less probable hyperedges (line 14) similar to (Mi et al., 2008). The inside and outside probabilities are defined as follows:

$$\beta(v) = \sum_{e \in IN(v)} p(e) \prod_{d \in tails(e)} \beta(d) \quad (1)$$

where $\beta(v) = 1.0$ if $IN(v) = \emptyset$, and

$$\alpha(v) = \sum_{\substack{h \in OUT(v) \\ e \in IN(head(h))}} \frac{\alpha(head(e))p(e)}{|OUT(v)|} \cdot \prod_{d \in tails(e) \setminus \{v\}} \beta(d) \quad (2)$$

where $\alpha(v) = 1.0$ if $OUT(v) = \emptyset$.

In practice, we restrict the number of words in a vertex in the initialization (line 1). We approximate all possible alternative MWUs that include each word as follows:

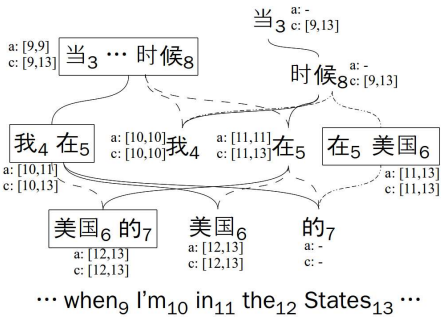

Figure 5: A sub-forest of Figure 4 with annotation of $aspan$ and $cspan$ for each vertex. We omit the span if it is not consistent.

- A horizontal vertex is a sequence of modifiers for a common head word, and
- A vertical vertex is a path from a word to one of the ancestors, and
- A combination of the horizontal vertices and the vertical vertices, and
- A combination of the vertical vertices and the vertical vertices.

The computational complexity of the initializaion directly affects the complexity of the entire procedure. For each word, generating the horizontal vertices takes $O(b^2)$, and the vertical vertices take $O(b^{d-1})$, where b is the maximum branching factor and d is the maximum depth of a dependency tree. The two combinations take $O(b^{d+1})$ and $O(b^{2(d-1)})$ time to initialize the vertices. However, it takes $O(m^{m+1})$ and $O(m^{2(m-1)})$ if we restrict the maximum number of the words in a vertex to a constant m.

Ding and Palmer (2005) insisted that the Viterbi decoding of an MWU-forest takes linear time. In our case, we enumerate the k-best incoming hyperedeges instead of the best one. Because each enumeration takes $O(k^2|\tau(v)|^3)$, the total time complexity also becomes linear to the length of the sentence n similar to Ding and Palmer (2005), i.e. $O(|V|k^2|\tau(v)|^3)$, where $|V| = O(na^{2(a-1)})$ and $a = \min(m, b, d)$.

Table 2: The extracted rules in Figure 5. N denotes the non-lexicalized rules with variables x_i for each $v \in tails(e)$, and L denotes the lexicalized rule.

	$head(e)$	$tails(e)$	$rhs(\gamma)$
N	$\{3\}$	$\{8\} : x_1$	x_1
	$\{8\}$	$\{4\} : x_1, \{5\} : x_2$	when x_1 x_2
	$\{3, 8\}$	$\{4, 5\} : x_1$	when x_1
	$\{3, 8\}$	$\{4\} : x_1, \{5\} : x_2$	when x_1 x_2
	$\{4, 5\}$	$\{6, 7\} : x_1$	I'm in x_1
	$\{5\}$	$\{6, 7\} : x_1$	in x_1
L	$\{6, 7\}$	N/A	the States
	$\{4\}$		I'm
	$\{5\}$		in
	$\{4, 5\}$		I'm in
	$\{5, 6\}$		in the State
	$\{3, 8\}$		When

4 MWU-Forest-to-String Translation Rule Extraction

As an application of our proposed MWU-forest, we extract translation rules for syntax-based SMT. Forest-based translation rule extraction has been suggested by Mi and Huang (2008) although their forest compacts the k-best PSG trees. The extraction procedure is essentially the same as Galley et al. (2004), which identifies the cutting points (frontiers) and extracts the sub-structures from a root to frontiers.

The situation changes in DG because DG does not have intermediate representation. At the dependency structure, a node corresponds to two kinds of target spans. We borrow the definitions of the aligned span ($aspan$), and the covered span ($cspan$) from Na et al. (2010), i.e.

- $aspan(v) = [\min(a_v), \max(a_v)]$, and
- $cspan(v) = aspan(v) \bigcup_{\substack{d \in tails(e) \\ e \in IN(v)}} cspan(d)$

, where $a_v = \{i | j \in v \wedge (i,j) \in A\}$. Figure 5 shows $aspan$s and $cspan$s of a sub-forest of of the MWU-forest in the previous example.

Each span type yields a different rule type: $aspan$ yields a lexicalized rule without any variables, and $cspan$ yields a non-lexicalized rule with variables for the dependents of the head word. For example, Table 2 shows the extracted rule in Figure 5.

In our MWU-forest, the rule extraction procedure is almost identical to a dependency tree-to-string rule extraction except we regard MWUs as vertices. Let f_j and e_i be the j-th source and i-th target word, respectively. As an MWU itself has a internal structure, a lexical rule is a tree-to-string translation rule. Therefore, a lexicalized rule is a pair of the source words s and the target words t as follows:

$$\begin{aligned} s(v) &= \{f_j | j \in v\} \\ t(v) &= \{e_i | i \in aspan(v)\} \end{aligned} \quad (3)$$

In addition, we extract the non-lexicalized rules from a hyperedge e to $cspan$ of the $head(e)$. A non-lexicalized rule is a pair of the source words in the vertices of a hyperedge and the $cspan$ of the target words with substitutions of $cspan(d)$ for each $d \in tails(e)$. We abstract d on the source with $\sigma(d)$ for non-lexicalized rules (row 2 in Table 2). We define the source words s and the target words t as follows:

$$\begin{aligned} s(e) &= \{f_j | j \in head(e) \vee j \in \sigma(d)\} \\ t(e) &= \{e_i | i \in cspan(v) \wedge i \notin cspan(d)\} \\ &\quad \cup \{x_i | d \leftrightarrow x_i\} \end{aligned} \quad (4)$$

Algorithm 2 Extract Rules($H = \langle V, E \rangle$)

```
Γ = ∅
for v ∈ V do
    if aspan(v) is consistent then
        Γ ← Γ ∪ ⟨ s(v) , t(v) ⟩ as in Eq. 3
    end if
    if cspan(v) is consistent then
        for e ∈ IN(v) do
            if cspan(d) ∀d ∈ tails(e) then
                Γ ← Γ ∪ ⟨ s(e), t(e) ⟩ as in Eq. 4
            end if
        end for
    end if
end for
return Γ
```

where $d \in tails(e)$.

More formally, we extract a synchronous tree substitution grammar (STSG) which regards the MWUs as non-terminals.

Definition 1 *A STSG using MWU (**STSG-MWU**) is a 6-tuple $G = \langle \Sigma_S, \Sigma_T, \Delta, \Gamma, S, \phi \rangle$, where:*

- Σ_S and Σ_T are finite sets of terminals (words, POSs, etc.) of the source and target languages, respectively.
- Δ is a finite set of MWUs in the source language, i.e. $\Delta = \{\Sigma_S\} +$
- Γ is a finite set of production rules where a production rule $\gamma : X \rightarrow \langle\, lhs(\gamma)\,,\, rhs(\gamma),\, \phi\,\rangle$, which is a relationship from Δ to $\{x \cup \Sigma_T\} *$, where ϕ is the bijective function from the source vertices to the variables x in $rhs(\gamma)$. The asterisk represents the Kleenstar operation, and
- S is the start symbol used to represent the whole sentence, i.e. $\gamma_0 : S \rightarrow \langle\, X\,,\, X\,\rangle$.

For each type of span, we only extract the rules if the target span has consistent word alignments, i.e. $span \neq \emptyset \wedge \forall i \in span$, $\{j|(i,j) \in A \cap (i',j) \in A \text{ s.t. } i' \notin span\} = \emptyset$. Algorithm 2 shows the pseudo code of the extraction. Because a vertex has $aspan$ and $csapn$, we extract a lexicalized rule (line 3-5) and/or non-lexicalized rules (line 6-12) for each vertex.

5 Experiment

We used the training corpus provided for the DIALOG Task in IWSLT10 between Chinese and English . The corpus is a collection of 30,033 sentence pairs and consists of dialogs in travel situations (10,061) and parts of the BTEC corpus (19,972). Details about the provided corpus are described in (Paul, 2009). We used the Stanford Parser [1] to obtain word-level dependency structures of Chinese sentences, and GIZA++ [2] to obtain word alignments of the biligual corpus.

We extracted the SCFG-MWU from the biligual corpus with word alignment. In order to investigate the coverage of the extracted rule, we counted the number of the recovered sentences, i.e. counted if the extracted rule for each sentence pair generates the target sentence by combining the extracted rules. As we collected many alternatives in an MWU-forest, we wanted to determine the importance of each source fragment. Mi and Huang (2008) penalized a rule γ by the posterior probability of its tree fragment $lhs(\gamma)$. This posterior probability is also computed in the Inside-Outside fashion that we used in Algorithm 1. Therefore, we regarded the fractional count of a rule γ as

[1]http://nlp.stanford.edu/software/lex-parser.shtml, Version 1.6.4

[2]http://code.google.com/p/giza-pp/

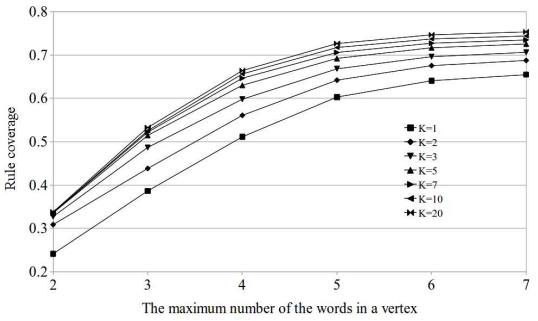

Figure 6: The rule coverage according to the number of the words in a vertex.

$$c(\gamma) = \frac{\alpha\beta(lhs(\gamma))}{\alpha\beta(v_0)}$$

We prioritized the rule according to the fractional count. The priority is used when we combine the rules to restore the target sentence using the extracted rule for each sentence. We varied the maximum size of a vertex m, and the number of incoming hyperedges k. Figure 6 shows the emprical result.

6 Discussion

Figure 6 shows that we need MWU to broaden the coverage of the extracted translation rules. The rule coverage increases as the number of words in an MWU increases, and almost converges at $m = 6$. Our proposed method recover around 75% of the sentences in the corpus when we properly restrict m and k. This is a great improvement over Na et al. (2010), who reported around 60% of the rule coverage without the limitaion of the size of MWUs. They only considered the best decomposition of the dependency tree, while our proposed method collects many alternative MWUs into an MWU-forest. When we considered the best decomposition ($k = 1$), the rule coverage dropped to

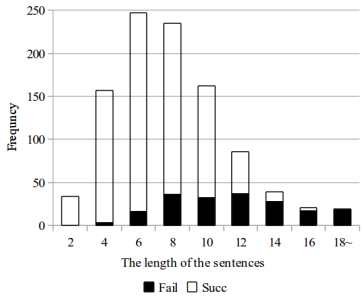

Figure 7: The frequency of the recovery according to the length of the sentences in 1,000 sentences

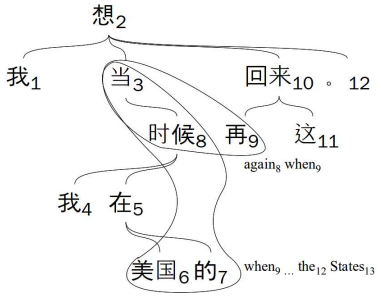

Figure 8: An illustration of ill-formed MWUs

around 65%. This can be viewed as an indirect comparison between Na et al. (2010) and our proposed method in this corpus.

Figure 7 shows that the frequency of success and failure in the recovery depends on the length of the sentences. As the length of sentences increase, the successful recovery occurs less frequently. We investigated the reason of failure in the longer sentences. As a result, the two main sources of the failure are the word alignment error and the dependency parsing error.

Our proposed method does not include all translation rules in PBSMT because of the syntactic constraint. Generally speaking, our proposed method cannot deal with MWUs that do not satisfy the well-formed constraint. However, ill-formed MWUs seems to be useful as well. For example, our proposed method dose not allow ill-formed vertices in an MWU-forest as shown in Figure 8. This would be problematic when we use an erroneuos parsing result. Because dealing with parsing error has been studied in literature, our proposed method has the potential to improve thought future work.

7 Conclusion

We have presented a way of representing sentence structure using MWUs on DG. Because of the absence of the intermdiate representation in DG, we built a simpler structure of MWUs than words using words as vertices of a dependency structure. Unlike previous studies, we collected many alternative structures using MWUs in a packed forest, which is novel. We also extracted MWU-forest-to-string translation rules, and verified the rule coverage empirically. As a consequence, we improvemed the rule coverage compared with a previous work, while retaining the linear asymptotic complexity. We will expand our propose method to develop a syntax-based SMT system in the future, and incoporate the parsing error by considering multiple syntactic analyses.

Acknowledgments

We appreciate the three anonymous reviewers. This work was supported in part by the Korea Science and Engineering Foundation (KOSEF) grant funded by the Korean government (MEST No. 2011-0003029), and in part by the BK 21 Project in 2011.

References

David Chiang. 2010. Learning to translate with source and target syntax. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1443–1452, Uppsala, Sweden, July. Association for Computational Linguistics.

Yuan Ding and Martha Palmer. 2005. Machine translation using probabilistic synchronous dependency insertion grammars. In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 541–548, Morristown, NJ, USA. Association for Computational Linguistics.

Yuan Ding. 2006. *Machine Translation Using Probabilistic Synchronous Dependency Insertion Grammars*. Ph.D. thesis, August.

Bonnie J. Dorr. 1994. Machine translation divergences: a formal description and proposed solution. *Comput. Linguist.*, 20:597–633, December.

Christopher Dyer, Smaranda Muresan, and Philip Resnik. 2008. Generalizing word lattice translation. In *Proceedings of ACL-08: HLT*, pages 1012–1020, Columbus, Ohio, June. Association for Computational Linguistics.

Jason Eisner. 2003. Learning non-isomorphic tree mappings for machine translation. In *ACL '03: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 205–208, Morristown, NJ, USA. Association for Computational Linguistics.

Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What's in a translation rule? In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Main Proceedings*, pages 273–280, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 961–968, Sydney, Australia, July. Association for Computational Linguistics.

Haitao Mi and Liang Huang. 2008. Forest-based translation rule extraction. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 206–214, Honolulu, Hawaii, October. Association for Computational Linguistics.

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

Haitao Mi, Liang Huang, and Qun Liu. 2010. Machine translation with lattices and forests. In *Coling 2010: Posters*, pages 837–845, Beijing, China, August. Coling 2010 Organizing Committee.

Hwidong Na, Jin-Ji Li, Yeha Lee, and Jong-Hyeok Lee. 2010. A synchronous context free grammar using dependency sequence for syntax-base statistical machine translation. In *The Ninth Conference of the Association for Machine Translation in the Americas (AMTA 2010)*, Denver, Colorado, October.

Michael Paul. 2009. Overview of the iwslt 2009 evaluation campaign.

Libin Shen, Jinxi Xu, and Ralph Weischedel. 2008. A new string-to-dependency machine translation algorithm with a target dependency language model. In *Proceedings of ACL-08: HLT*, pages 577–585, Columbus, Ohio, June. Association for Computational Linguistics.

Anders Søgaard and Jonas Kuhn. 2009. Empirical lower bounds on aligment error rates in syntax-based machine translation. In *Proceedings of the Third Workshop on Syntax and Structure in Statistical Translation (SSST-3) at NAACL HLT 2009*, pages 19–27, Boulder, Colorado, June. Association for Computational Linguistics.

Zhaopeng Tu, Yang Liu, Young-Sook Hwang, Qun Liu, and Shouxun Lin. 2010. Dependency forest for statistical machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages

1092–1100, Beijing, China, August. Coling 2010 Organizing Committee.

Yuanbin Wu, Qi Zhang, Xuangjing Huang, and Lide Wu. 2009. Phrase dependency parsing for opinion mining. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1541, Singapore, August. Association for Computational Linguistics.