Dependency Graph-to-String Translation

Liangyou Li Andy Way Qun Liu
ADAPT Centre, School of Computing
Dublin City University
{liangyouli,away,qliu}@computing.dcu.ie

Abstract

Compared to tree grammars, graph grammars have stronger generative capacity over structures. Based on an edge replacement grammar, in this paper we propose to use a synchronous graph-to-string grammar for statistical machine translation. The graph we use is directly converted from a dependency tree by labelling edges. We build our translation model in the log-linear framework with standard features. Large-scale experiments on Chinese–English and German–English tasks show that our model is significantly better than the state-of-the-art hierarchical phrase-based (HPB) model and a recently improved dependency tree-to-string model on BLEU, METEOR and TER scores. Experiments also suggest that our model has better capability to perform long-distance reordering and is more suitable for translating long sentences.

1 Introduction

Compared to trees, which have dominated the field of natural language processing (NLP) for decades, graphs are more general for modelling natural languages. The corresponding grammars for recognizing and producing graphs are more flexible and powerful than tree grammars. However, because of their high complexity, graph grammars have not been widely used in NLP.

Recently, along with progress on graph-based meaning representation, hyperedge replacement grammars (HRG) (Drewes et al., 1997) have been revisited, explored and used for semantic-based machine translation (Jones et al., 2012). However, the translation process is rather complex and the resources it relies on, namely abstract meaning corpora, are limited as well.

As most available syntactic resources and tools are tree-based, in this paper we propose to convert dependency trees, which are usually taken as a kind of shallow semantic representation, to dependency graphs by labelling edges. We then use a synchronous version of edge replacement grammar (ERG) (Section 2), a special case of HRG, to translate these graphs. The resulting translation model has the same order of magnitude in terms of time complexity with the hierarchical phrase-based model (HPB) (Chiang, 2005) under a certain restriction (Section 3).

Compared to dependency tree-to-string models, using ERG for graph-to-string translation brings some benefits (Section 3). Thanks to the stronger generative capacity of the grammar, our model can naturally translate siblings in a tree structure, which are usually treated as non-syntactic phrases and handled by other techniques (Huck et al., 2014; Xie et al., 2014). Furthermore, compared to the known treelet approach (Quirk et al., 2005) and Dep2Str (Xie et al., 2011), our method not only uses treelets but also has a full capacity of reordering.

We define our translation model (Section 4) in the log-linear framework (Och and Ney, 2002). Large-scale experiments (Section 5) on Chinese–English and German–English, two language pairs that have a high degree of syntactic reordering, show that our method significantly improves translation quality over both HPB and Dep2Str, as measured by BLEU (Papineni et al., 2002), TER (Snover et al., 2006) and METEOR (Denkowski and Lavie, 2011). We also find that the rules in our model are more suitable for long-distance reordering and translating long sentences.

2 Edge Replacement Grammar

As a special case of HRG, ERG is also a context-free rewriting grammar to recognize and produce graphs. Following HRG, the graph we use in this

Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 33–43,
Lisbon, Portugal, 17-21 September 2015. ©2015 Association for Computational Linguistics.

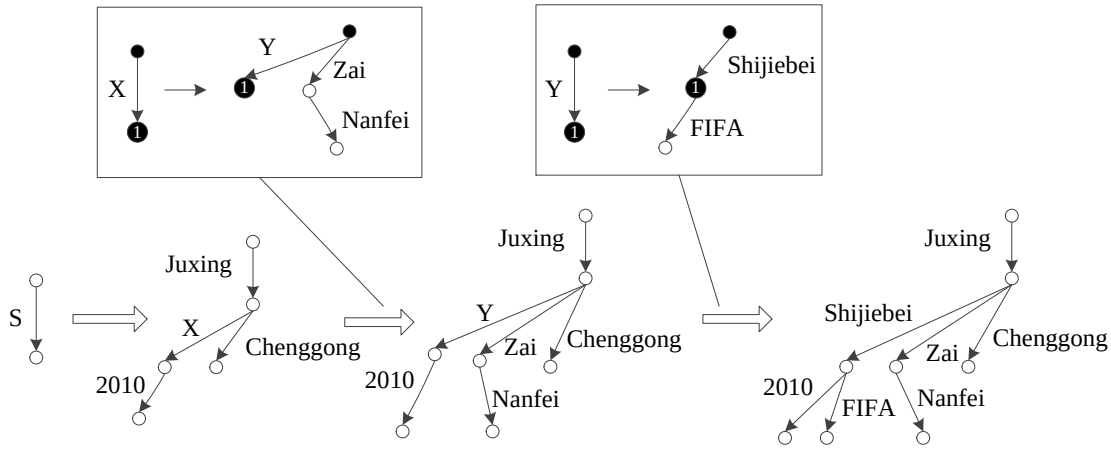


Figure 1: An example of a derivation in an ERG. Dark circles are external nodes.

paper is connected, nodes ordered, acyclic and has edge labels but no node labels (Chiang et al., 2013). We provide some formal definitions on ERG.

Definition 1. A *connected, edge-labeled, ordered graph* is a tuple $H = \langle V, E, \phi \rangle$, where

- V is a finite set of nodes.
- $E \subseteq V^2$ is a finite set of edges.
- $\phi : E \rightarrow C$ assigns a label (drawn from C) to each edge.

In ERG, the elementary unit is a graph fragment, which is also the right-hand side of a production in the grammar. Its definition is as follows.

Definition 2. A *graph fragment* is a tuple $H = \langle V, E, \phi, X \rangle$, where $\langle V, E, \phi \rangle$ is a graph and $X \in (V \cup V^2)$ is a list of distinct nodes. Following Chiang et al. (2013), we call these *external nodes*.

The external nodes indicate how to integrate a graph into another one during a derivation. Different to HRG, ERG limits the number of external nodes to 2 at most to make sure hyperedges do not exist during a derivation. Now we define the ERG.

Definition 3. An *edge replacement grammar* is a tuple $\langle N, T, P, S \rangle$, where

- N and T are disjoint finite sets of nonterminal symbols and terminal symbols, respectively.
- P is a finite set of productions of the form $A \rightarrow R$, where $A \in N$ and R is a graph fragment, where edge-labels are from $N \bigcup T$.
- $S \in N$ is the start symbol.

Figure 1 shows an example of a derivation in an ERG to produce a graph. Starting from the start symbol S, when a rule $(A \rightarrow R)$ is applied to an edge e, the edge is replaced by the graph fragment R. Just like in HRG, the ordering of nodes V_e in e and external nodes X_R in R implies the mapping from V_e to X_R (Chiang et al., 2013).

3 Graph-to-String Grammar

In SMT, we need a synchronous grammar to simultaneously parse an input graph and produce translations. The graph we use in this paper is from a dependency structure which is capable of modelling long-distance relations in a sentence.

3.1 The Grammar

Before defining the synchronous grammar, we firstly define a dependency graph which is a special case of a graph.

Definition 4. A *dependency graph* is a tuple $\langle V, E, \phi, \Delta \rangle$, where $\langle V, E, \phi \rangle$ is a graph and Δ is a restriction: edges are ordered.

A dependency graph is directly derived from a dependency tree by labeling edges with words, as shown in Figure 2. Although in general graph edges are unordered, in Definition 4 we keep word order by ordering edges, because the word order is an important piece of information for translation.

Similar to the graph fragment, a dependency-graph fragment is defined as below.

Definition 5. A *dependency-graph fragment* is a tuple $\langle V, E, \phi, \Delta, X \rangle$, where $\langle V, E, \phi, \Delta \rangle$ is a dependency graph, $X \in (V \cup V^2)$ is a list of external nodes.

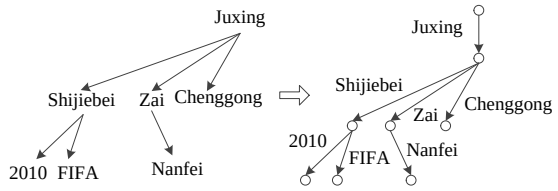


Figure 2: An example of deriving a dependency graph from a dependency tree by labelling edges with words.

In this paper, we define a synchronous ERG over dependency graphs as a dependency graph-to-string grammar, which can be used for MT.

Definition 6. A *dependency graph-to-string grammar* (DGSG) is a tuple $\langle N, T, T', P, S \rangle$, where

- N is a finite set of non-terminal symbols.
- T and T' are finite sets of terminal symbols.
- $S \in N$ is the start symbol.
- P is a finite set of productions of the form $\langle A \rightarrow R, A' \rightarrow R', \sim \rangle$, where $A, A' \in N$, R is a dependency-graph fragment over $N \bigcup T$ and R' is a string over $N \bigcup T'$. $\sim$ is a one-to-one mapping between non-terminal symbols in R and R'.

Figure 3 shows a derivation simultaneously producing a Chinese dependency graph and an English string using a DGSG. Each time a rule is applied, the dependency-graph fragment in the rule replaces an edge in the source graph, and the string in the rule replaces a non-terminal in the target string.

Proposition 1. DGSG has stronger generative capacity over graph-string pairs than both SCFG and synchronous tree substitution grammar (STSG).

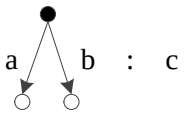
Proof. STSG has stronger generative capacity over structures than SCFG (Chiang, 2012).[1]

Any STSG can easily be converted into a DGSG by labelling edges in tree structures.

[1]The following STSG generates a trivial example of a tree-string pair that no SCFG can generate, as SCFG must always have an equal number of non-terminal symbols.

$$\begin{array}{c} X \\ X \quad | \\ | \ : X \\ \epsilon \quad | \\ \epsilon \end{array}$$

The following DGSG generates a trivial example of a graph-string pair, which no STSG can generate, as the left-head side has no head nodes while STSG always requires one to form a tree.



□

This proof is also verified in Figure 3 where the third rule is used to translate a non-syntactic phrase, which can be a problem for dependency tree-to-string methods. In addition, the second rule translates a treelet and the first rule encodes reordering information inside. All these three aspects are uniformly modeled in our grammar, which makes it more powerful than other methods, such as the treelet approach and the Dep2Str.

3.2 Time Complexity and a Restriction

Given a dependency graph, training and decoding time using DGSG depends on the number of dependency-graph fragments. For example, for a graph where the degree of a node is k, the number of all possible fragments starting from the node is $O(2^k)$. Therefore, the time complexity would be exponential if we consider them all.

It is easy to find that the high complexity of DGSG comes from the free combination of edges. That means that a dependency-graph fragment can cover discontinuous words of an input sentence. However, this is not the convention in the field of SMT.

For efficient training and decoding, we add a restriction to DGSG: each dependency-graph fragment covers a continuous span of the source sentence. This reduces the complexity from exponential time to cubic time.

3.3 Non-terminal Symbols

In this paper we build a dependency graph-to-string model, so we only use one non-terminal symbol X as in HPB on the target side. However, on the source side we define non-terminal symbols over Part-of-Speech (POS) tags, which can be easily obtained as a by-product of dependency parsing.

We define the *head* of a dependency-graph fragment H as a list of edges, the dependency head of each of which is not in this fragment. Then the

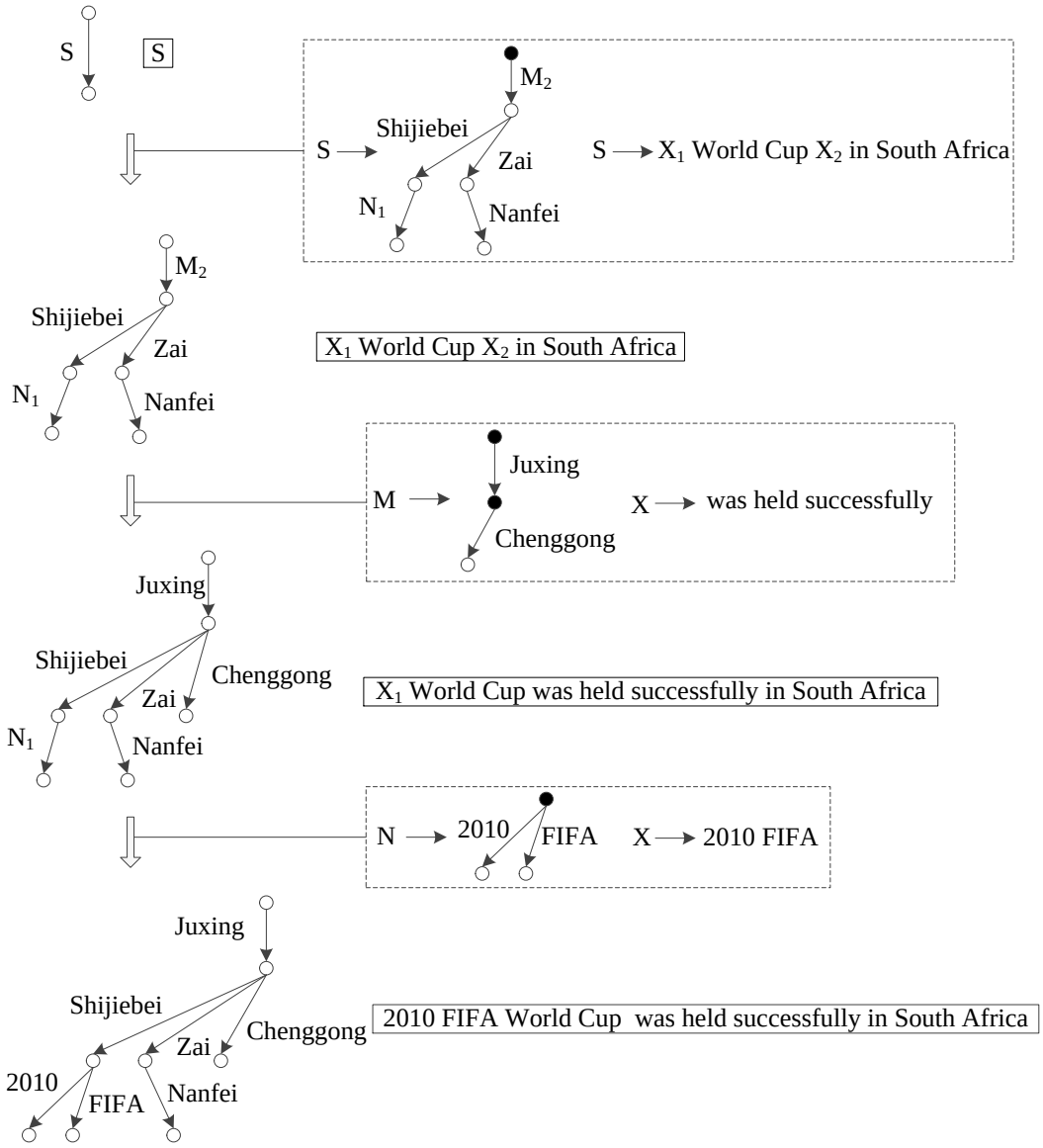


Figure 3: An example of a derivation in dependency graph-to-string grammar to produce a Chinese dependency graph and an English string. Rules are included in dashed rectangles. Target strings are in solid rectangles. External nodes are dark circles. This example is under the restriction in Section 3.2. In addition to the start symbol S, non-terminal symbols for the source side are M and N, while the target side only has one non-terminal X. The index in each non-terminal of a rule indicates the mapping.

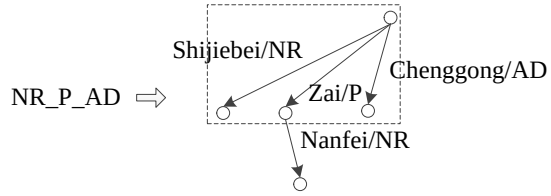


Figure 4: An example inducing a non-terminal symbol (left side) for a dependency-graph fragment (right side). Each edge is labeled by a word associated with its POS tag. The head of this fragment includes three edges which are in the rectangle.

non-terminal symbol for H is defined as the joining of POS tags of its head (Li et al., 2012). Figure 4 shows an example.

3.4 Rule Extraction

As well as the restriction defined in Section 3.2 making the grammar much smaller, it also results in a similar way of extracting rules as in HPB. Inspired by HPB, we define the rule set over *initial pairs*.

Given a word-aligned dependency graph-string pair $P = \langle G, e, \sim \rangle$, let G_i^j stand for the sub-graph (it may not be connected) covering words from position i to position j. Then a rule $\langle G_i^j, e_{i'}^{j'} \rangle$ is an initial pair of P, iff:

1. G_i^j is a dependency-graph fragment. That means it is a connected sub-graph and has at most two external nodes, nodes which connect with nodes outside or are the root.
2. It is consistent with the word alignment $\sim$ (Och and Ney, 2004).

The set of rules from P satisfies the following:

1. If $\langle G_i^j, e_{i'}^{j'} \rangle$ is an initial pair, then
$$\langle N(G_i^j) \rightarrow G_i^j, X \rightarrow e_{i'}^{j'} \rangle$$
is a rule, where $N(G)$ defines the non-terminal symbol for G.
2. If $\langle N(R) \rightarrow R, X \rightarrow R' \rangle$ is a rule of P and $\langle G_i^j, e_{i'}^{j'} \rangle$ is an initial pair such that G_i^j is a sub-graph of R and $R' = r_1 e_{i'}^{j'} r_2$, then
$$\langle N(R) \rightarrow R \backslash G_{i\,k}^j, X \rightarrow r_1 X_k r_2 \rangle$$
is a rule of P, where $\backslash$ means replacing G_i^j in R with an edge labelled with $N(G_i^j)$ and k is a unique index for a pair of non-terminal symbols.

As in HPB, in addition to rules extracted from the parallel corpus, we also use glue rules to combine fragments and translations when no matched rule can be found.

Furthermore, we can use the same rule extraction algorithm as that in HPB, except that we need to check if a span of a source sentence indicates a dependency-graph fragment, in which case we keep the dependency structure and induce a non-terminal for the fragment.

4 Model and Decoding

We define our model in the log-linear framework over a derivation d, as in Equation (1):

$$P(d) \propto \prod_i \phi_i(d)^{\lambda_i} \quad (1)$$

where ϕ_i are features defined on derivations and λ_i are feature weights. In our experiments, we use 9 features:

- translation probabilities $P(s|t)$ and $P(t|s)$, where s is the source graph fragment and t is the target string.
- lexical translation probabilities $P_{lex}(s|t)$ and $P_{lex}(t|s)$.
- language model $lm(e)$ over translation e.
- rule penalty $exp(-1)$.
- word penalty $exp(|e|)$.
- glue penalty $exp(-1)$.
- unknown words penalty $exp(u(g))$, where $u(g)$ is the number of unknown words in a source graph g.

Our decoder is based on the conventional chart parsing CYK algorithm (Kasami, 1965; Younger, 1967; Cocke and Schwartz, 1970). It searches for the best derivation d^* among all possible derivations D, as in Equation (2):

$$d^* = \underset{d \in D}{\operatorname{argmax}} P(d) \quad (2)$$

For each span of an input graph, the decoder checks if it is a dependency-graph fragment. Then

ZH–EN

corpus	#sent.	#words(ZH)	#words(EN)
train	1.5M+	38M+	∼45M
dev	878	22,655	26,905
MT04	1,597	43,719	52,705
MT05	1,082	29,880	35,326

DE–EN

corpus	#sent.	#words(DE)	#words(EN)
train	2M+	52M+	55M+
dev	3,003	72,661	74,753
WMT12	3,003	72,603	72,988
WMT13	3,000	63,412	64,810

Table 1: Chinese–English (ZH–EN) and German–English (DE–EN) corpora. For the English side of dev and test sets, words counts are averaged across all references.

for each fragment, the decoder finds rules to translate it. The translation of a large span can be obtained by combining translations from its sub-span using rules which have non-terminals. Finally, glue rules are used to make sure that at least one translation is produced.

5 Experiment

We conduct experiments on Chinese–English and German–English translation tasks.

5.1 Datasets

The Chinese–English training corpus is from LDC, including LDC2002E18, LDC2003E07, LDC2003E14, LDC2004T07, the Hansards portion of LDC2004T08 and LDC2005T06. NIST 2002 is taken as a development set to tune weights, and NIST 2004 (MT04) and NIST 2005 (MT05) are two test sets to evaluate systems. Table 1 provides a summary of this corpus. The Stanford Chinese word segmenter (Chang et al., 2008) is used to segment Chinese sentences. The Stanford dependency parser (Chang et al., 2009) parses a Chinese sentence into a projective dependency tree which is then converted to a dependency graph in our model.

The German–English training corpus is from WMT 2014, including Europarl V7 and News Commentary. News-test 2011 is taken as a development set, while News-test 2012 (WMT12) and News-test 2013 (WMT13) are our test sets. Table 1 provides a summary of this corpus. We use mate-tools[2] to perform morphological analysis and parse German sentences (Bohnet, 2010). Then MaltParser[3] converts a parse result into a projective dependency tree (Nivre and Nilsson, 2005).

5.2 Settings

In this paper, we mainly compare our system (**DGST**) with HPB in Moses (Koehn et al., 2007). We implement our model in Moses and take the same settings as Moses HPB in all experiments. In addition, translation results from a recently open-source dependency tree-to-string system, Dep2Str[4] (Li et al., 2014), which is implemented in Moses and improves the dependency-based model in Xie et al. (2011), are also reported. All systems use the same sets of features defined in Section 4.

In all experiments, word alignment is performed by GIZA++ (Och and Ney, 2003) with the heuristic function *grow-diag-final-and*. We use SRILM (Stolcke, 2002) to train a 5-gram language model on the Xinhua portion of the English Gigaword corpus 5th edition with modified Kneser-Ney discounting (Chen and Goodman, 1996). Minimum Error Rate Training (MERT) (Och, 2003) is used to tune weights.

To obtain more reliable results, in each experiment, we run MERT three times and report average scores. These scores are calculated by three widely used automatic metrics in case-insensitive mode: BLEU, METEOR and TER.

5.3 Results

Table 2 shows the scores of all three metrics on all systems. Similar to Li et al. (2014), in our experiments Dep2Str has on average a comparable result with Moses HPB in terms of BLEU and METEOR scores. However, it obtains a significantly higher (i.e. worse) TER score on the Chinese–English task. This may suggest that translations produced by Dep2Str need more post-editing effort (He et al., 2010).

By contrast, on all test sets, measured by all metrics, our system is significantly better than Moses HPB. On the Chinese–English task, our system achieves an average gain of 1.25 (absolute, 3.6% relative) BLEU score and 0.55 (absolute, 1.7% relative) METEOR score while also ob-

[2] http://code.google.com/p/mate-tools/
[3] http://www.maltparser.org/
[4] http://computing.dcu.ie/~liangyouli/dep2str.zip

Metric	System	ZH–EN		DE–EN	
		MT04	MT05	WMT12	WMT13
BLEU ↑	Moses HPB	35.6	33.8	20.2	22.7
	Dep2Str	35.4	33.9	20.3	22.8
	DGST	**36.6**	**35.3**	**20.7**	**23.3**
METEOR ↑	Moses HPB	31.6	31.9	28.6	29.7
	Dep2Str	**31.8**	31.9	28.5	29.5*
	DGST	**32.1**	**32.5**	**28.7**	**29.8**
TER ↓	Moses HPB	57.0	58.3	63.2	59.5
	Dep2Str	58.2*	59.6*	63.1	59.6
	DGST	**56.1**	**57.0**	**62.6**	**59.0**

Table 2: Metric scores for all systems on Chinese–English (ZH–EN) and German–English (DE–EN) corpus. Each score is the average score over three MERT runs. Bold figures mean a system is significantly better than Moses HPB at $p \leq 0.01$. Moses HPB is significantly better than systems with * at $p \leq 0.01$.

Length	Percentage			
	MT04	MT05	WMT12	WMT13
(0, 10]	7.6%	8.6%	15.0%	19.2%
(10, 20]	28.2%	26.0%	31.4%	37.2%
(20, 30]	28.2%	26.5%	26.3%	24.5%
(30, 40]	20.2%	23.8%	14.4%	12.0%
(40, ∞)	15.7%	15.2%	12.9%	7.2%

Table 3: Statistics of sentence length on four test sets.

taining a reduction of 1.1 (absolute, 1.91% relative) TER score on average.

On the German–English task, our system achieves an average gain of 0.55 (absolute, 2.56% relative) BLEU score and 0.1 (absolute, 0.35% relative) METEOR score and also obtains a reduction of 0.55 (absolute, 0.89% relative) TER score on average.

5.4 Analysis

As shown in Table 2, compared to Moses HPB and Dep2Str, our system achieves higher translation quality as measured by three automatic metrics. In this section, we investigate whether dependency structures bring benefits as expected on long-distance reordering. Table 3 provides the statistics on sentence length of our four test sets.

In both HPB and our model, the length range of a reordering performed on an input sentence is related to the use of glue grammars which bring two benefits during decoding. When no matched rule is found in the models, glue grammars are applied to make sure a translation is produced. In addition, because of the generalization capability of rules, which typically are learned under a length limitation, using them on long sentences could cause translation quality to deteriorate. Therefore, when the length of a phrase is greater than a certain value, glue grammars are also applied. Therefore, our experiment of analysis is based on the length limitation that a rule can cover (max. phrase length) during decoding.

We set this max. phrase length to different values, including 10, 20 (default), 30, 40 and 50. Figure 5 gives the BLEU scores on all test sets. We find that on all different values, our system achieves higher BLEU scores than Moses HPB. In addition, when the max. phrase length becomes larger, Moses HPB shows a declining trend in most cases, especially on the German–English task (WMT12 and WMT13). However, our system is less sensitive to this value. We hypothesize that this is because rules from dependency graphs have better generalization for translating longer phrases and are more suitable for translating long sentences.

5.5 Case Study

On a manual check, we find that translations produced by our system are more fluent than those of both Moses HPB and Dep2Str. Figure 6 gives an example comparing translations produced by three systems on the Chinese–English task.

We first find a case of long-distance relation, i.e. the subject-verb-object (SVO) structure in the source sentence. In this example, this relation implies a long-distance reordering, which moves the translation of the object to the front of its modifiers, as shown in the given reference. Com-

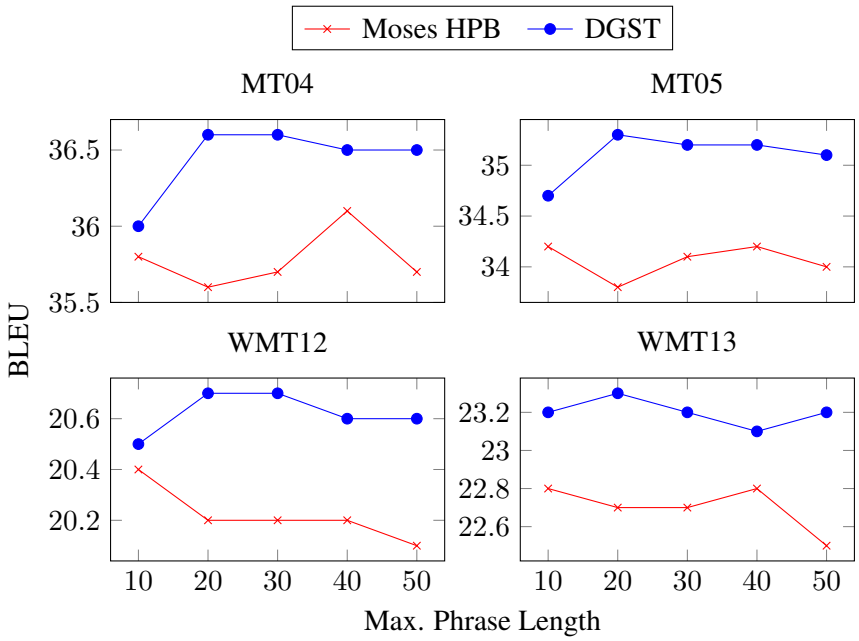
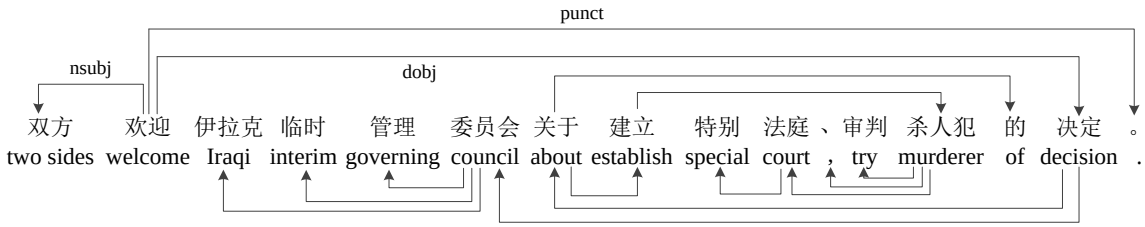


Figure 5: BLEU scores of Moses HPB and DGST (our system) when the length of maximum phrase that a rule can cover during decoding is set to different values.



Ref: The two sides welcomed the decision by the Iraqi Interim Governing Council to establish a special court to try the murderers.

HPB: the two sides welcomed the interim iraqi authority on establishing a special court, trial of the murderer.

Dep2Str: the two sides welcomed the decision on the Establishment of a special court, justice murderers of the provisional governing council of iraq.

DGST: the two sides welcomed the decision of the iraqi interim governing council on the establishment of a special court, justice murderers.

Figure 6: An example of comparing translations produced by three systems on the Chinese–English task. The source sentence is parsed into a dependency structure. Each source word is annotated by a corresponding English word (or phrase).

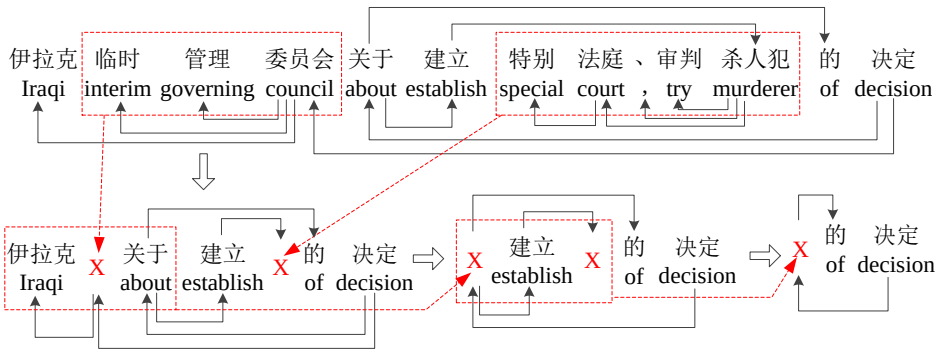


Figure 7: An example of inducing a dependency structure in Figure 6 to ”X 的(of) X” structure in our system by using treelets and non-syntactic phrases. ➪ denotes one or more steps. All non-terminals are simply represented by X.

pared to Moses HPB, both Dep2Str and our system, which rely on dependency structures, are capable of dealing with this. This also suggests that dependency structures are useful for long-distance reordering.

Furthermore, compared to Dep2Str, our system produces a better translation for the "X 的(of) X" expression, which is not explicitly represented in the dependency structure and thus results in a wrong translation in Dep2Str. After looking into the details of the translation process, we find that our system induces the dependency structure to the "X 的(of) X" structure by handling both treelets and non-syntactic phrases. Figure 7 shows the process of this induction.

6 Related Work

Dependency structures have been used in SMT for a few years. Because of its better inter-lingual phrasal cohesion properties (Fox, 2002), it is believed to be beneficial to translation.

Researchers have tried to use dependency structures on both target and source sides. Shen et al. (2010) propose a string-to-dependency model by using dependency fragments of neighbouring words on the target side, which makes the model easier to include a dependency-based language model.

Menezes and Quirk (2005) and Quirk et al. (2005) propose the treelet approach which uses dependency structures on the source side. Xiong et al. (2007) extend this approach by allowing gaps in rules. However, their methods need a separate reordering model to decide the position of translated words (insertion problem). To avoid this problem, Xie et al. (2011) propose to use full head-dependent structures of a dependency tree and build a new dependency-to-string model. However, this model has difficulties in handling non-syntactic phrasal rules and ignores treelets. Meng et al. (2013) and Xie et al. (2014) further augment this model by incorporating constituent phrases and integrating fix/float structures (Shen et al., 2010), respectively, to allow phrasal rules. Li et al. (2014) extend this model by decomposing head-dependent structures into treelets.

Different from these methods, by labelling edges and using the ERG, our model considers the three aspects in a unified way: treelet, reordering and non-syntactic phrase. In addition, the ERG also naturally provides a decision on what kind of treelets and phrases should be used.

7 Conclusion

In this paper, we present a dependency graph-to-string grammar based on a graph grammar, which we call edge replacement grammar. This grammar can simultaneously produce a pair of dependency graph and string. With a restriction of using contiguous edges, our translation model built using this grammar can decode an input dependency graph, which is directly converted from a dependency tree, in cubic time using the CYK algorithm.

Experiments on Chinese–English and German–English tasks show that our model is significantly better than the hierarchical phrase-based model and a recent dependency tree-to-string model (Dep2Str) in Moses. We also find that the rules used in our model are more suitable for long-distance reordering and translating long sentences.

Although experiments show significant improvements over baselines, our model has limitations that can be avenues for future work. The restriction used in this paper reduces the time complexity but at the same time reduces the generative capacity of graph grammars. Without allowing hyperedges or only using at most two external nodes reduces the phrase coverage in our model as well.

Acknowledgments

This research has received funding from the People Programme (Marie Curie Actions) of the European Union's Framework Programme (FP7/2007-2013) under REA grant agreement n° 317471. The ADAPT Centre for Digital Content Technology is funded under the SFI Research Centres Programme (Grant 13/RC/2106) and is co-funded under the European Regional Development Fund. We thank anonymous reviewers for their insightful comments and suggestions.

References

Bernd Bohnet. 2010. Very High Accuracy and Fast Dependency Parsing is Not a Contradiction. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 89–97, Beijing, China.

Pi-Chuan Chang, Michel Galley, and Christopher D. Manning. 2008. Optimizing Chinese Word Segmentation for Machine Translation Performance. In *Proceedings of the Third Workshop on Statistical*

Machine Translation, pages 224–232, Columbus, Ohio.

Pi-Chuan Chang, Huihsin Tseng, Dan Jurafsky, and Christopher D. Manning. 2009. Discriminative Reordering with Chinese Grammatical Relations Features. In *Proceedings of the Third Workshop on Syntax and Structure in Statistical Translation*, pages 51–59, Boulder, Colorado.

Stanley F. Chen and Joshua Goodman. 1996. An Empirical Study of Smoothing Techniques for Language Modeling. In *Proceedings of the 34th Annual Meeting on Association for Computational Linguistics*, ACL '96, pages 310–318, Santa Cruz, California.

David Chiang, Jacob Andreas, Daniel Bauer, Karl Moritz Hermann, Bevan Jones, and Kevin Knight. 2013. Parsing graphs with hyperedge replacement grammars. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 924–932, Sofia, Bulgaria, August.

David Chiang. 2005. A Hierarchical Phrase-based Model for Statistical Machine Translation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 263–270, Ann Arbor, Michigan.

David Chiang. 2012. *Grammars for Language and Genes: Theoretical and Empirical Investigations*. Springer.

John Cocke and Jacob T. Schwartz. 1970. Programming Languages and Their Compilers: Preliminary Notes. Technical report, Courant Institute of Mathematical Sciences, New York University, New York, NY.

Michael Denkowski and Alon Lavie. 2011. Meteor 1.3: Automatic Metric for Reliable Optimization and Evaluation of Machine Translation Systems. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, WMT '11, pages 85–91, Edinburgh, Scotland.

Frank Drewes, Hans Jörg Kreowski, and Annegret Habel. 1997. Handbook of graph grammars and computing by graph transformation. chapter Hyperedge Replacement Graph Grammars, pages 95–162. World Scientific Publishing Co., Inc., River Edge, NJ, USA.

Heidi J. Fox. 2002. Phrasal Cohesion and Statistical Machine Translation. In *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10*, pages 304–3111, Philadelphia.

Yifan He, Yanjun Ma, Josef van Genabith, and Andy Way. 2010. Bridging SMT and TM with Translation Recommendation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 622–630, Uppsala, Sweden, July.

Matthias Huck, Hieu Hoang, and Philipp Koehn. 2014. Augmenting String-to-Tree and Tree-to-String Translation with Non-Syntactic Phrases. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 486–498, Baltimore, Maryland, USA, June.

Bevan Jones, Jacob Andreas, Daniel Bauer, Karl Moritz Hermann, and Kevin Knight. 2012. Semantics-Based Machine Translation with Hyperedge Replacement Grammars. In *COLING 2012, 24th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers*, pages 1359–1376, Mumbai, India, December.

Tadao Kasami. 1965. An Efficient Recognition and Syntax-Analysis Algorithm for Context-Free Languages. Technical report, Air Force Cambridge Research Lab, Bedford, MA.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ond'ej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 177–180, Prague, Czech Republic.

Junhui Li, Zhaopeng Tu, Guodong Zhou, and Josef van Genabith. 2012. Head-Driven Hierarchical Phrase-based Translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 33–37, Jeju Island, Korea, July.

Liangyou Li, Jun Xie, Andy Way, and Qun Liu. 2014. Transformation and Decomposition for Efficiently Implementing and Improving Dependency-to-String Model In Moses. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, October.

Arul Menezes and Chris Quirk. 2005. Dependency Treelet Translation: The Convergence of Statistical and Example-Based Machine-translation? In *Proceedings of the Workshop on Example-based Machine Translation at MT Summit X*, September.

Fandong Meng, Jun Xie, Linfeng Song, Yajuan L', and Qun Liu. 2013. Translation with Source Constituency and Dependency Trees. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1066–1076, Seattle, Washington, USA, October.

Joakim Nivre and Jens Nilsson. 2005. Pseudo-Projective Dependency Parsing. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 99–106, Ann Arbor, Michigan.

Franz Josef Och and Hermann Ney. 2002. Discriminative Training and Maximum Entropy Models for Statistical Machine Translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 295–302, Philadelphia, PA, USA.

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

Franz Josef Och and Hermann Ney. 2004. The Alignment Template Approach to Statistical Machine Translation. *Computational Linguistics*, 30(4):417–449, December.

Franz Josef Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 160–167, Stroudsburg, PA, USA. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Philadelphia, Pennsylvania.

Chris Quirk, Arul Menezes, and Colin Cherry. 2005. Dependency Treelet Translation: Syntactically Informed Phrasal SMT. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 271–279, Ann Arbor, Michigan, June.

Libin Shen, Jinxi Xu, and Ralph Weischedel. 2010. String-to-Dependency Statistical Machine Translation. *Computational Linguistics*, 36(4):649–671, December.

M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul. 2006. A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of Association for Machine Translation in the Americas*, pages 223–231, Cambridge, Massachusetts, USA, August.

Andreas Stolcke. 2002. SRILM '' An Extensible Language Modeling Toolkit. In *Proceedings of the International Conference Spoken Language Processing*, pages 901–904, Denver, CO.

Jun Xie, Haitao Mi, and Qun Liu. 2011. A Novel Dependency-to-string Model for Statistical Machine Translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 216–226, Edinburgh, United Kingdom.

Jun Xie, Jinan Xu, and Qun Liu. 2014. Augment Dependency-to-String Translation with Fixed and Floating Structures. In *Proceedings of the 25th International Conference on Computational Linguistics*, pages 2217–2226, Dublin, Ireland.

Deyi Xiong, Qun Liu, and Shouxun Lin. 2007. A Dependency Treelet String Correspondence Model for Statistical Machine Translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 40–47, Prague, June.

Daniel H. Younger. 1967. Recognition and Parsing of Context-Free Languages in Time n^3. *Information and Control*, 10(2):189–208.