Shallow Local Multi Bottom-up Tree Transducers in Statistical Machine Translation

Fabienne Braune and **Nina Seemann** and **Daniel Quernheim** and **Andreas Maletti**
Institute for Natural Language Processing, University of Stuttgart
Pfaffenwaldring 5b, 70569 Stuttgart, Germany
{braunefe,seemanna,daniel,maletti}@ims.uni-stuttgart.de

Abstract

We present a new translation model integrating the shallow local multi bottom-up tree transducer. We perform a large-scale empirical evaluation of our obtained system, which demonstrates that we significantly beat a realistic tree-to-tree baseline on the WMT 2009 English → German translation task. As an additional contribution we make the developed software and complete tool-chain publicly available for further experimentation.

1 Introduction

Besides phrase-based machine translation systems (Koehn et al., 2003), syntax-based systems have become widely used because of their ability to handle non-local reordering. Those systems use synchronous context-free grammars (Chiang, 2007), synchronous tree substitution grammars (Eisner, 2003) or even more powerful formalisms like synchronous tree-sequence substitution grammars (Sun et al., 2009). However, those systems use linguistic syntactic annotation at different levels. For example, the systems proposed by Wu (1997) and Chiang (2007) use no linguistic information and are syntactic in a structural sense only. Huang et al. (2006) and Liu et al. (2006) use syntactic annotations on the source language side and show significant improvements in translation quality. Using syntax exclusively on the target language side has also been successfully tried by Galley et al. (2004) and Galley et al. (2006). Nowadays, open-source toolkits such as Moses (Koehn et al., 2007) offer syntax-based components (Hoang et al., 2009), which allow experiments without expert knowledge. The improvements observed for systems using syntactic annotation on either the source or the target language side naturally led to experiments with models that use syntactic annotations on *both* sides. However, as noted by Lavie et al. (2008), Liu et al. (2009), and Chiang (2010), the integration of syntactic information on both sides tends to decrease translation quality because the systems become too restrictive. Several strategies such as (i) using parse forests instead of single parses (Liu et al., 2009) or (ii) soft syntactic constraints (Chiang, 2010) have been developed to alleviate this problem. Another successful approach has been to switch to more powerful formalisms, which allow the extraction of more general rules. A particularly powerful model is the non-contiguous version of synchronous tree-sequence substitution grammars (STSSG) of Zhang et al. (2008a), Zhang et al. (2008b), and Sun et al. (2009), which allows sequences of trees on both sides of the rules [see also (Raoult, 1997)]. The multi bottom-up tree transducer (MBOT) of Arnold and Dauchet (1982) and Lilin (1978) offers a middle ground between traditional syntax-based models and STSSG. Roughly speaking, an MBOT is an STSSG, in which all the discontinuities must occur on the target language side (Maletti, 2011). This restriction yields many algorithmic advantages over both the traditional models as well as STSSG as demonstrated by Maletti (2010). Formally, they are expressive enough to express all sensible translations (Maletti, 2012)[1]. Figure 2 displays sample rules of the MBOT variant, called ℓMBOT, that we use (in a graphical representation of the trees and the alignment).

In this contribution, we report on our novel statistical machine translation system that uses an ℓMBOT-based translation model. The theoretical foundations of ℓMBOT and their integration into our translation model are presented in Sections 2 and 3. In order to empirically evaluate the ℓMBOT model, we implemented a machine trans-

[1] A translation is sensible if it is of linear size increase and can be computed by some (potentially copying) top-down tree transducer.

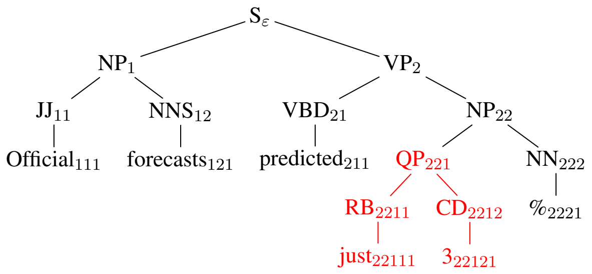

Figure 1: Example tree t with indicated positions. We have $t(21) = \text{VBD}$ and $t|_{221}$ is the subtree marked in red.

lation system that we are going to make available to the public. We implemented ℓMBOT inside the syntax-based component of the Moses open source toolkit. Section 4 presents the most important algorithms of our ℓMBOT decoder. We evaluate our new system on the WMT 2009 shared translation task English → German. The translation quality is automatically measured using BLEU scores, and we confirm the findings by providing linguistic evidence (see Section 5). Note that in contrast to several previous approaches, we perform large scale experiments by training systems with approx. 1.5 million parallel sentences.

2 Theoretical Model

In this section, we present the theoretical generative model used in our approach to syntax-based machine translation. Essentially, it is the local multi bottom-up tree transducer of Maletti (2011) with the restriction that all rules must be shallow, which means that the left-hand side of each rule has height at most 2 (see Figure 2 for shallow rules and Figure 4 for rules including non-shallow rules). The rules extracted from the training example of Figure 3 are displayed in Figure 4. Those extracted rules are forcibly made shallow by removing internal nodes. The application of those rules is illustrated in Figures 5 and 6.

For those that want to understand the inner workings, we recall the principal model in full detail in the rest of this section. Since we utilize syntactic parse trees, let us introduce trees first. Given an alphabet Σ of labels, the set T_Σ of all Σ-*trees* is the smallest set T such that $\sigma(t_1, \dots, t_k) \in T$ for all $\sigma \in \Sigma$, integer $k \geq 0$, and $t_1, \dots, t_k \in T$. Intuitively, a tree t consists of a labeled root node σ followed by a sequence $t_1, \dots, t_k$ of its children. A tree $t \in T_\Sigma$ is *shallow* if $t = \sigma(t_1, \dots, t_k)$ with $\sigma \in \Sigma$ and $t_1, \dots, t_k \in \Sigma$.

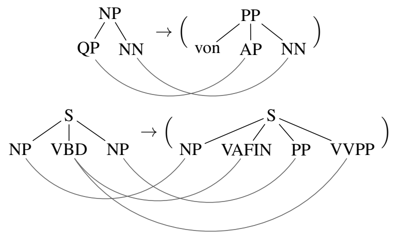

Figure 2: Sample ℓMBOT rules.

To address a node inside a tree, we use its position, which is a word consisting of positive integers. Roughly speaking, the root of a tree is addressed with the position ε (the empty word). The position iw with $i \in \mathbb{N}$ addresses the position w in the i^{th} direct child of the root. In this way, each node in the tree is assigned a unique position. We illustrate this notion in Figure 1. Formally, the *positions* $\text{pos}(t) \subseteq \mathbb{N}^*$ of a tree $t = \sigma(t_1, \dots, t_k)$ are inductively defined by $\text{pos}(t) = \{\varepsilon\} \cup \text{pos}^{(k)}(t_1, \dots, t_k)$, where

$$\text{pos}^{(k)}(t_1, \dots, t_k) = \bigcup_{1 \leq i \leq k} \{iw \mid w \in \text{pos}(t_i)\} \ .$$

Let $t \in T_\Sigma$ and $w \in \text{pos}(t)$. The label of t at position w is $t(w)$, and the subtree rooted at position w is $t|_w$. These notions are also illustrated in Figure 1. A position $w \in \text{pos}(t)$ is a *leaf* (in t) if $w1 \notin \text{pos}(t)$. In other words, leaves do not have any children. Given a subset $N \subseteq \Sigma$, we let

$$\text{leaf}_N(t) = \{w \in \text{pos}(t) \mid t(w) \in N,\ w \text{ leaf in } t\}$$

be the set of all leaves labeled by elements of N. When N is the set of nonterminals, we call them *leaf nonterminals*. We extend this notion to sequences $t_1, \dots, t_k \in T_\Sigma$ by

$$\text{leaf}_N^{(k)}(t_1, \dots, t_k) = \bigcup_{1 \leq i \leq k} \{iw \mid w \in \text{leaf}_N(t_i)\}.$$

Let $w_1, \dots, w_n \in \text{pos}(t)$ be (pairwise prefix-incomparable) positions and $t_1, \dots, t_n \in T_\Sigma$. Then $t[w_i \leftarrow t_i]_{1 \leq i \leq n}$ denotes the tree that is obtained from t by replacing (in parallel) the subtrees at w_i by t_i for every $1 \leq i \leq n$.

Now we are ready to introduce our model, which is a minor variation of the local multi bottom-up tree transducer of Maletti (2011). Let Σ and Δ be the input and output symbols, respectively, and let $N \subseteq \Sigma \cup \Delta$ be the set of nonterminal symbols. Essentially, the model works on pairs $\langle t, (u_1, \dots, u_k)\rangle$ consisting of an input tree $t \in T_\Sigma$

and a sequence $u_1, \dots, u_k \in T_\Delta$ of output trees. Such pairs are *pre-translations* of rank k. The pre-translation $\langle t, (u_1, \dots, u_k)\rangle$ is *shallow* if all trees $t, u_1, \dots, u_k$ in it are shallow.

Together with a pre-translation we typically have to store an alignment. Given a pre-translation $\langle t, (u_1, \dots, u_k)\rangle$ of rank k and $1 \leq i \leq k$, we call u_i the i^{th} translation of t. An *alignment* for this pre-translation is an injective mapping $\psi\colon \operatorname{leaf}_N^{(k)}(u_1, \dots, u_k) \to \operatorname{leaf}_N(t) \times \mathbb{N}$ such that if $(w, j) \in \operatorname{ran}(\psi)$, then also $(w, i) \in \operatorname{ran}(\psi)$ for all $1 \leq j \leq i$.[2] In other words, if an alignment requests the i^{th} translation, then it should also request all previous translations.

Definition 1 *A shallow local multi bottom-up tree transducer (ℓMBOT) is a finite set R of* rules *together with a mapping $c\colon R \to \mathbb{R}$ such that every rule, written $t \to_\psi (u_1, \dots, u_k)$, contains a shallow pre-translation $\langle t, (u_1, \dots, u_k)\rangle$ and an alignment ψ for it.*

The components t, $(u_1, \dots, u_k)$, ψ, and $c(\rho)$ are called the *left-hand side*, the *right-hand side*, the *alignment*, and the *weight* of the rule $\rho = t \to_\psi (u_1, \dots, u_k)$. Figure 2 shows two example ℓMBOT rules (without weights). Overall, the rules of an ℓMBOT are similar to the rules of an SCFG (synchronous context-free grammar), but our right-hand sides contain a sequence of trees instead of just a single tree. In addition, the alignments in an SCFG rule are bijective between leaf nonterminals, whereas our model permits multiple alignments to a single leaf nonterminal in the left-hand side (see Figure 2).

Our ℓMBOT rules are obtained automatically from data like that in Figure 3. Thus, we (word) align the bilingual text and parse it in both the source and the target language. In this manner we obtain sentence pairs like the one shown in Figure 3. To these sentence pairs we apply the rule extraction method of Maletti (2011). The rules extracted from the sentence pair of Figure 3 are shown in Figure 4. Note that these rules are not necessarily shallow (the last two rules are not). Thus, we post-process the extracted rules and make them shallow. The shallow rules corresponding to the non-shallow rules of Figure 4 are shown in Figure 2.

Next, we define how to combine rules to form derivations. In contrast to most other models, we only introduce a derivation semantics that does not collapse multiple derivations for the same input-output pair.[3] We need one final notion. Let $\rho = t \to_\psi (u_1, \dots, u_k)$ be a rule and $w \in \operatorname{leaf}_N(t)$ be a leaf nonterminal (occurrence) in the left-hand side. The w-rank $\operatorname{rk}(\rho, w)$ of the rule ρ is

$$\operatorname{rk}(\rho, w) = \max \{i \in \mathbb{N} \mid (w, i) \in \operatorname{ran}(\psi)\} \ .$$

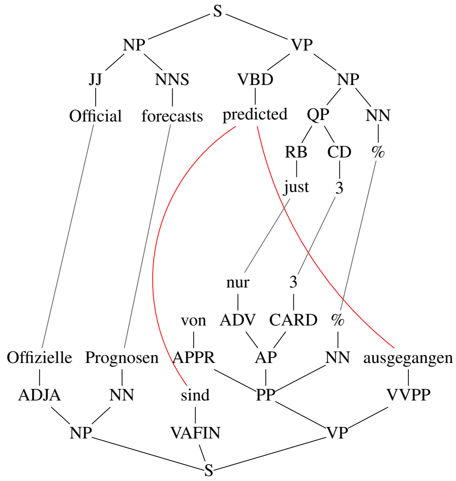

Figure 3: Aligned parsed sentences.

For example, for the lower rule ρ in Figure 2 we have $\operatorname{rk}(\rho, 1) = 1$, $\operatorname{rk}(\rho, 2) = 2$, and $\operatorname{rk}(\rho, 3) = 1$.

Definition 2 *The set $\tau(R, c)$ of* weighted pre-translations *of an ℓMBOT (R, c) is the smallest set T subject to the following restriction: If there exist*

- *a rule $\rho = t \to_\psi (u_1, \dots, u_k) \in R$,*
- *a weighted pre-translation*

  $$\langle t_w, c_w, (u_1^w, \dots, u_{k_w}^w)\rangle \in T$$

  for every $w \in \operatorname{leaf}_N(t)$ with
  - $\operatorname{rk}(\rho, w) = k_w$,[4]
  - $t(w) = t_w(\varepsilon)$,[5] *and*
  - *for every $iw' \in \operatorname{leaf}_N^{(k)}(u_1, \dots, u_k)$,*[6]

    $$u_i(w') = u_j^v(\varepsilon) \text{ with } \psi(iw') = (v, j),$$

then $\langle t', c', (u'_1, \dots, u'_k)\rangle \in T$ is a weighted pre-translation, where

- $t' = t[w \leftarrow t_w \mid w \in \operatorname{leaf}_N(t)]$,

[2] $\operatorname{ran}(f)$ for a mapping $f\colon A \to B$ denotes the range of f, which is $\{f(a) \mid a \in A\}$.

[3] A standard semantics is presented, for example, in (Maletti, 2011).

[4] If w has n alignments, then the pre-translation selected for it has to have suitably many output trees.

[5] The labels have to coincide for the input tree.

[6] Also the labels for the output trees have to coincide.

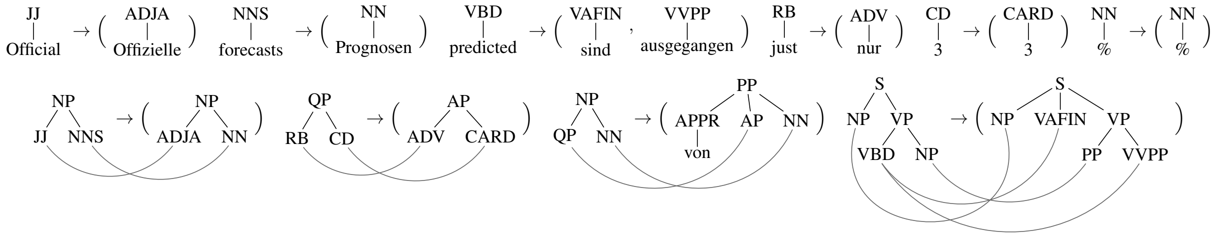

Figure 4: Extracted (even non-shallow) rules. We obtain our rules by making those rules shallow.

- $c' = c(\rho) \cdot \prod_{w \in \mathrm{leaf}_N(t)} c_w$, *and*
- $u'_i = u_i[iw' \leftarrow u^v_j \mid \psi(iw') = (v, j)]$ *for every* $1 \leq i \leq k$.

Rules that do not contain any nonterminal leaves are automatically weighted pre-translations with their associated rule weight. Otherwise, each nonterminal leaf w in the left-hand side of a rule ρ must be replaced by the input tree t_w of a pre-translation $\langle t_w, c_w, (u^w_1, \dots, u^w_{k_w})\rangle$, whose root is labeled by the same nonterminal. In addition, the rank $\mathrm{rk}(\rho, w)$ of the replaced nonterminal should match the number k_w of components in the selected weighted pre-translation. Finally, the nonterminals in the right-hand side that are aligned to w should be replaced by the translation that the alignment requests, provided that the nonterminal matches with the root symbol of the requested translation. The weight of the new pre-translation is obtained simply by multiplying the rule weight and the weights of the selected weighted pre-translations. The overall process is illustrated in Figures 5 and 6.

3 Translation Model

Given a source language sentence e, our translation model aims to find the best corresponding target language translation $\hat{g}$;[7] i.e.,

$$\hat{g} = \arg\max_g p(g|e) \ .$$

We estimate the probability $p(g|e)$ through a log-linear combination of component models with parameters λ_m scored on the pre-translations $\langle t, (u)\rangle$ such that the leaves of t concatenated read e.[8]

$$p(g|e) \propto \prod_{m=1}^{7} h_m\big(\langle t, (u)\rangle\big)^{\lambda_m}$$

Our model uses the following features $h_m(\langle t, (u_1, \dots, u_k)\rangle)$ for a general pre-translation $\tau = \langle t, (u_1, \dots, u_k)\rangle$:

(1) The forward translation weight using the rule weights as described in Section 2
(2) The indirect translation weight using the rule weights as described in Section 2
(3) Lexical translation weight source → target
(4) Lexical translation weight target → source
(5) Target side language model
(6) Number of words in the target sentences
(7) Number of rules used in the pre-translation
(8) Number of target side sequences; here k times the number of sequences used in the pre-translations that constructed τ (gap penalty)

The rule weights required for (1) are relative frequencies normalized over all rules with the same left-hand side. In the same fashion the rule weights required for (2) are relative frequencies normalized over all rules with the same right-hand side. Additionally, rules that were extracted at most 10 times are discounted by multiplying the rule weight by 10^{-2}. The lexical weights for (2) and (3) are obtained by multiplying the word translations $w(g_i|e_j)$ [respectively, $w(e_j|g_i)$] of lexically aligned words (g_i, e_j) accross (possibly discontiguous) target side sequences.[9] Whenever a source word e_j is aligned to multiple target words, we average over the word translations.[10]

$$h_3(\langle t, (u_1, \dots, u_k)\rangle) = \prod_{\substack{\text{lexical item} \\ e \text{ occurs in } t}} \text{average } \{w(g|e) \mid g \text{ aligned to } e\}$$

The computation of the language model estimates for (6) is adapted to score partial translations consisting of discontiguous units. We explain the details in Section 4. Finally, the count c of target sequences obtained in (7) is actually used as a score 100^{1-c}. This discourages rules with many target sequences.

[7] Our main translation direction is English to German.

[8] Actually, t must embed in the parse tree of e; see Section 4.

[9] The lexical alignments are different from the alignments used with a pre-translation.

[10] If the word e_j has no alignment to a target word, then it is assumed to be aligned to a special NULL word and this alignment is scored.

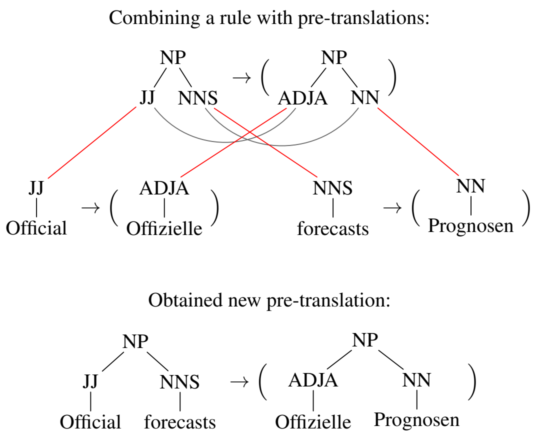

Figure 5: Simple rule application.

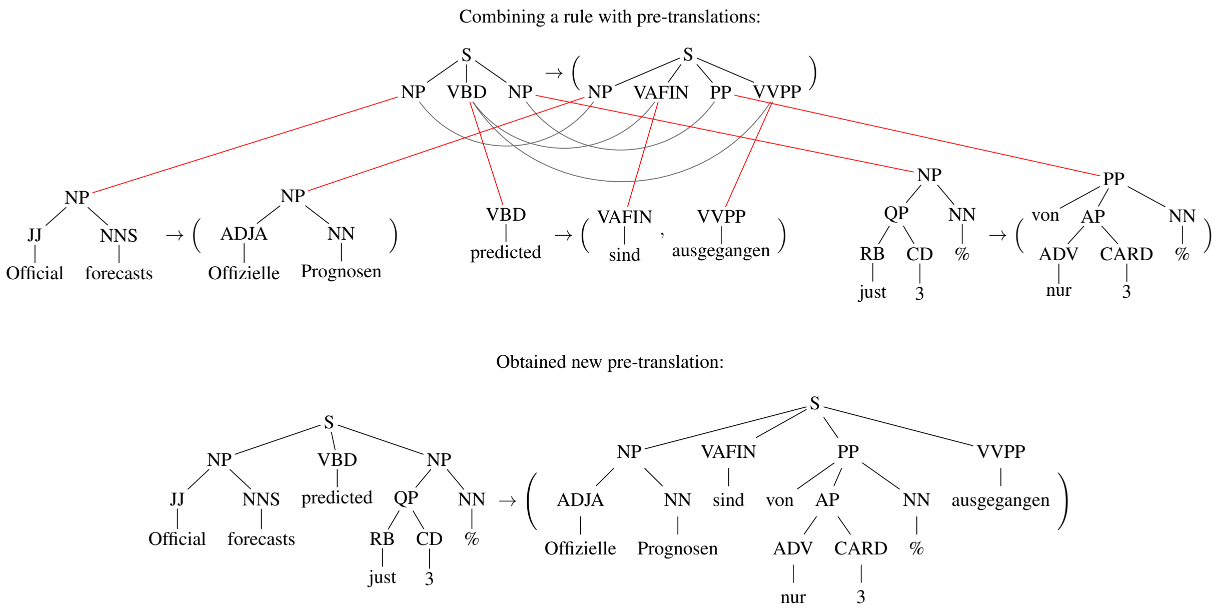

Figure 6: Complex rule application.

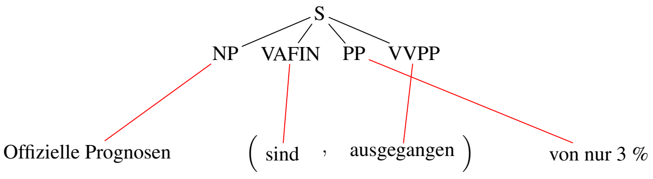

Figure 7: Illustration of LM scoring.

4 Decoding

We implemented our model in the syntax-based component of the Moses open-source toolkit by Koehn et al. (2007) and Hoang et al. (2009). The standard Moses syntax-based decoder only handles SCFG rules; i.e, rules with contiguous components on the source and the target language side. Roughly speaking, SCFG rules are ℓMBOT rules with exactly one output tree. We thus had to extend the system to support our ℓMBOT rules, in which arbitrarily many output trees are allowed.

The standard Moses syntax-based decoder uses a CYK+ chart parsing algorithm, in which each source sentence is parsed and contiguous spans are processed in a bottom-up fashion. A rule is applicable[11] if the left-hand side of it matches the nonterminal assigned to the full span by the parser and the (non-)terminal assigned to each subspan.[12] In order to speed up the decoding, cube pruning (Chiang, 2007) is applied to each chart cell in order to select the most likely hypotheses for subspans. The language model (LM) scoring is directly integrated into the cube pruning algorithm. Thus, LM estimates are available for all considered hypotheses. To accommodate ℓMBOT rules, we had to modify the Moses syntax-based decoder in several ways. First, the rule representation itself is adjusted to allow *sequences* of shallow output trees on the target side. Naturally, we also had to adjust hypothesis expansion and, most importantly, language model scoring inside the cube pruning algorithm. An overview of the modified pruning procedure is given in Algorithm 1.

Algorithm 1 Cube pruning with ℓMBOT rules

```
Data structures:
- $r[i,j]$: list of rules matching span $e[i \dots j]$
- $h[i,j]$: hypotheses covering span $e[i \dots j]$
- $c[i,j]$: cube of hypotheses covering span $e[i \dots j]$

1:  for all $\ell$MBOT rules $\rho$ covering span $e[i \dots j]$ do
2:      Insert $\rho$ into $r[i,j]$
3:  Sort $r[i,j]$
4:  for all $(l \to_\psi r) \in r[i,j]$ do
5:      Create $h[i,j]$ by expanding all nonterminals in $l$ with
        best scoring hypotheses for subspans
6:      Add $h[i,j]$ to $c[i,j]$
7:  for all hypotheses $h \in c[i,j]$ do
8:      Estimate LM score for $h$                  // see Algorithm 2
9:      Estimate remaining feature scores
10: Sort $c[i,j]$
11: Retrieve first $\alpha$ elements from $c[i,j]$   // we use $\alpha = 10^3$
```

The most important modifications are hidden in lines 5 and 8. The expansion in Line 5 involves matching all nonterminal leaves in the rule as defined in Definition 2, which includes matching all leaf nonterminals in all (discontiguous) output trees. Because the output trees can remain discontiguous after hypothesis creation, LM scoring has to be done individually over all output trees. Algorithm 2 describes our LM scoring in detail. In it we use k strings $w_1, \dots, w_k$ to collect the lexical information from the k output components $(u_1, \dots, u_k)$ of a rule. These strings can later be rearranged in any order, so we LM-score all of them separately. Roughly speaking, we obtain w_i by traversing u_i depth-first left-to-right. If we meet a lexical element (terminal), then we add it to the end of w_i. On the other hand, if we meet a nonterminal, then we have to consult the best pre-translation $\tau' = \langle t', (u'_1, \dots, u'_{k'}) \rangle$, which will contribute the subtree at this position. Suppose that u'_j will be substituted into the nonterminal in question. Then we first LM-score the pre-translation τ' to obtain the string w'_j corresponding to u'_j. This string w'_j is then appended to w_i. Once all the strings are built, we score them using our 4-gram LM. The overall LM score for the pre-translation is obtained by multiplying the scores for $w_1, \dots, w_k$. Clearly, this treats $w_1, \dots, w_k$ as k separate strings, although they eventually will be combined into a single string. Whenever such a concatenation happens, our LM scoring will automatically compute n-gram LM scores based on the concatenation, which in particular means that the LM scores get more accurate for larger spans. Finally, in the final rule only one component is allowed, which yields that the LM indeed scores the complete output sentence.

Figure 7 illustrates our LM scoring for a pre-translation involving a rule with two (discontiguous) target sequences (the construction of the pre-translation is illustrated in Figure 6). When processing the rule rooted at S, an LM estimate is computed by expanding all nonterminal leaves. In our case, these are NP, VAFIN, PP, and VVPP. However, the nodes VAFIN and VVPP are assembled from a (discontiguous) tree sequence. This means that those units have been considered as in-

[11] Note that our notion of applicable rules differs from the default in Moses.

[12] Theoretically, this allows that the decoder ignores unary parser nonterminals, which could also disappear when we make our rules shallow; e.g., the parse tree left in the pre-translation of Figure 5 can be matched by a rule with left-hand side NP(Official, forecasts).

Algorithm 2 LM scoring

```
Data structures:
- $(u_1, \ldots, u_k)$: right-hand side of a rule
- $(w_1, \ldots, w_k)$: $k$ strings all initially empty

score $= 1$
for all $1 \leq i \leq k$ do
    for all leaves $\ell$ in $u_i$ (in lexicographic order) do
        if $\ell$ is a terminal then
            Append $\ell$ to $w_i$
        else
            LM score the best hypothesis for the subspan
            Expand $w_i$ by the corresponding $w'_j$
    score $=$ score $\cdot$ LM$(w_i)$
```

dependent until now. So far, the LM scorer could only score their associated unigrams. However, we also have their associated strings w'_1 and w'_2, which can now be used. Since VAFIN and VVPP now become parts of a single tree, we can perform LM scoring normally. Assembling the string we obtain

Offizielle Prognosen sind von nur 3 %
ausgegangen

which is scored by the LM. Thus, we first score the 4-grams "Offizielle Prognosen sind von", then "Prognosen sind von nur", etc.

5 Experiments

5.1 Setup

The baseline system for our experiments is the syntax-based component of the Moses open-source toolkit of Koehn et al. (2007) and Hoang et al. (2009). We use linguistic syntactic annotation on both the source and the target language side (tree-to-tree). Our contrastive system is the ℓMBOT-based translation system presented here. We provide the system with a set of SCFG as well as ℓMBOT rules. We do not impose any maximal span restriction on either system.

The compared systems are evaluated on the English-to-German[13] news translation task of WMT 2009 (Callison-Burch et al., 2009). For both systems, the used training data is from the 4th version of the *Europarl Corpus* (Koehn, 2005) and the *News Commentary* corpus. Both translation models were trained with approximately 1.5 million bilingual sentences after length-ratio filtering. The word alignments were generated by GIZA++ (Och and Ney, 2003) with the grow-diag-final-and heuristic (Koehn et al., 2005). The English side of the bilingual data was parsed using the Charniak parser of Charniak and Johnson (2005), and the German side was parsed using BitPar (Schmid, 2004) without the function and morphological annotations. Our German 4-gram language model was trained on the German sentences in the training data augmented by the Stuttgart SdeWaC corpus (Web-as-Corpus Consortium, 2008), whose generation is detailed in (Baroni et al., 2009). The weights λ_m in the log-linear model were trained using minimum error rate training (Och, 2003) with the News 2009 development set. Both systems use glue-rules, which allow them to concatenate partial translations without performing any reordering.

System	**BLEU**
Baseline	12.60
ℓMBOT	*13.06
Moses t-to-s	12.72

Table 1: Evaluation results. The starred results are statistically significant improvements over the Baseline (at confidence $p < 0.05$).

5.2 Results

We measured the overall translation quality with the help of 4-gram BLEU (Papineni et al., 2002), which was computed on tokenized and lower-cased data for both systems. The results of our evaluation are reported in Table 1. For comparison, we also report the results obtained by a system that utilizes parses only on the source side (Moses tree-to-string) with its standard features.

We can observe from Table 1 that our ℓMBOT-based system outperforms the baseline. We obtain a BLEU score of 13.06, which is a gain of 0.46 BLEU points over the baseline. This improvement is statistically significant at confidence $p < 0.05$, which we computed using the pairwise bootstrap resampling technique of Koehn (2004). Our system is also better than the Moses tree-to-string system. However this improvement (0.34) is not statistically significant. In the next section, we confirm the result of the automatic evaluation through a manual examination of some translations generated by our system and the baseline.

In Table 2, we report the number of ℓMBOT rules used by our system when decoding the test set. By *lex* we denote rules containing only lexical

[13] Note that our ℓMBOT-based system can be applied to any language pair as it involves no language-specific engineering.

	lex	**non-term**	**total**
contiguous	23,175	18,355	41,530
discontiguous	315	2,516	2,831

Table 2: Number of rules used in decoding test (lex: only lexical items; non-term: at least one nonterminal).

2-dis	**3-dis**	**4-dis**
2,480	323	28

Table 3: Number of k-discontiguous rules.

items. The label *non-term* stands for rules containing at least one leaf nonterminal. The results show that approx. 6% of all rules used by our ℓMBOT-system have discontiguous target sides. Furthermore, the reported numbers show that the system also uses rules in which lexical items are combined with nonterminals. Finally, Table 3 presents the number of rules with k target side components used during decoding.

5.3 Linguistic Analysis

In this section we present linguistic evidence supporting the fact that the ℓMBOT-based system significantly outperforms the baseline. All examples are taken from the translation of the test set used for automatic evaluation. We show that when our system generates better translations, this is directly related to the use of ℓMBOT rules.

Figures 8 and 9 show the ability of our system to correctly reorder multiple segments in the source sentence where the baseline translates those segments sequentially. An analysis of the generated derivations shows that our system produces the correct translation by taking advantage of rules with discontiguous units on target language side. The rules used in the presented derivations are displayed in Figures 10 and 11. In the first example (Figure 8), we begin by translating “((*smuggle*)$_{\text{VB}}$ (*eight projectiles*)$_{\text{NP}}$ (*into the kingdom*)$_{\text{PP}}$)$_{\text{VP}}$” into the discontiguous sequence composed of (i) “(*acht geschosse*)$_{\text{NP}}$” ; (ii) “(*in das königreich*)$_{\text{PP}}$” and (iii) “(*schmuggeln*)$_{\text{VP}}$”. In a second step we assemble all sequences in a rule with contiguous target language side and, at the same time, insert the word “(*zu*)$_{\text{PTKZU}}$” between “(*in das königreich*)$_{\text{PP}}$” and “(*schmuggeln*)$_{\text{VP}}$”.

The second example (Figure 9) illustrates a more complex reordering. First, we translate “((*again*)$_{\text{ADV}}$ *commented on* (*the problem of global warming*)$_{\text{NP}}$)$_{\text{VP}}$” into the discontiguous sequence composed of (i) “(*das problem der globalen erwärmung*)$_{\text{NP}}$”; (ii) “(*wieder*)$_{\text{ADV}}$” and (iii) “(*kommentiert*)$_{\text{VPP}}$”. In a second step, we translate the auxiliary “(*has*)$_{\text{VBZ}}$” by inserting “(*hat*)$_{\text{VAFIN}}$” into the sequence. We thus obtain, for the input segment “((*has*)$_{\text{VBZ}}$ (*again*)$_{\text{ADV}}$ *commented on* (*the problem of global warming*)$_{\text{NP}}$)$_{\text{VP}}$”, the sequence (i) “(*das problem der globalen erwärmung*)$_{\text{NP}}$”; (ii) “(*hat*)$_{\text{VAFIN}}$”; (iii) “(*wieder*)$_{\text{ADV}}$”; (iv) “(*kommentiert*)$_{\text{VVPP}}$”. In a last step, the constituent “(*president václav klaus*)$_{\text{NP}}$” is inserted between the discontiguous units “(*hat*)$_{\text{VAFIN}}$” and “(*wieder*)$_{\text{ADV}}$” to form the contiguous sequence “((*das problem der globalen erwärmung*)$_{\text{NP}}$ (*hat*)$_{\text{VAFIN}}$ (*präsident václav klaus*)$_{\text{NP}}$ (*wieder*)$_{\text{ADV}}$ (*kommentiert*)$_{\text{VVPP}}$)$_{\text{TOP}}$”.

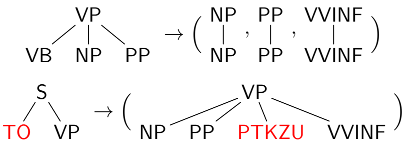

Figure 10: Used ℓMBOT rules for verbal reordering

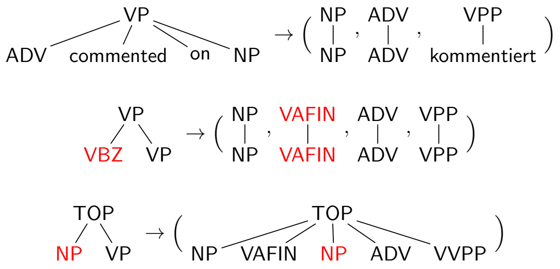

Figure 11: Used ℓMBOT rules for verbal reordering

Figures 12 and 13 show examples where our system generates complex words in the target language out of a simple source language word. Again, an analysis of the generated derivation shows that ℓMBOT takes advantage of rules having several target side components. Examples of such rules are given in Figure 14. Through its ability to use these discontiguous rules, our system correctly translates into reflexive or particle verbs such as “*konzentriert sich*” (for the English “*focuses*”) or “*besteht darauf*” (for the English “*insist*”). Another phenomenon well handled by our system are relative pronouns. Pronouns such as “that” or “whose” are systematically translated

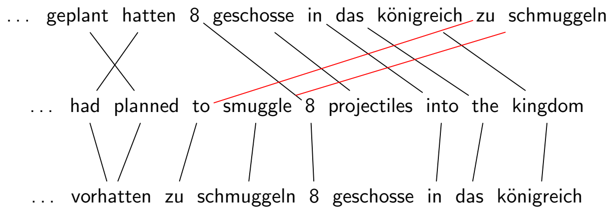
. . . geplant hatten 8 geschosse in das königreich zu schmuggeln
. . . had planned to smuggle 8 projectiles into the kingdom
. . . vorhatten zu schmuggeln 8 geschosse in das königreich

Figure 8: Verbal Reordering (top: our system, bottom: baseline)

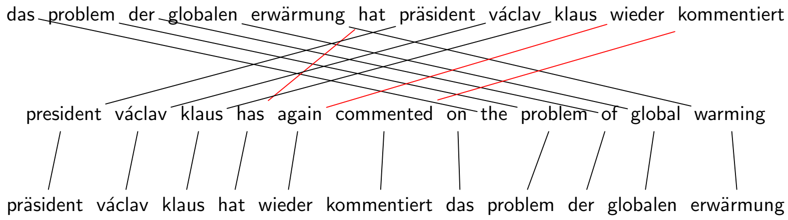
das problem der globalen erwärmung hat präsident václav klaus wieder kommentiert
president václav klaus has again commented on the problem of global warming
präsident václav klaus hat wieder kommentiert das problem der globalen erwärmung

Figure 9: Verbal Reordering (top: our system, bottom: baseline)

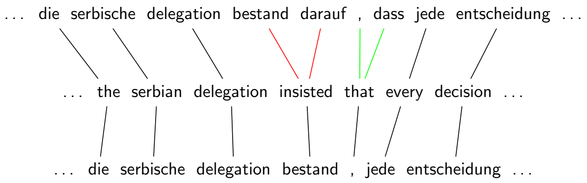
. . . die serbische delegation bestand darauf , dass jede entscheidung . . .
. . . the serbian delegation insisted that every decision . . .
. . . die serbische delegation bestand , jede entscheidung . . .

Figure 12: Relative Clause (top: our system, bottom: baseline)

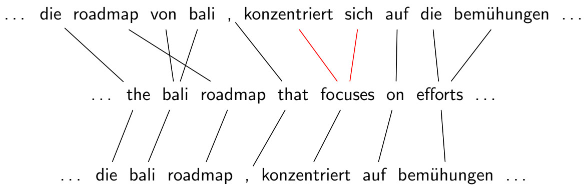
. . . die roadmap von bali , konzentriert sich auf die bemühungen . . .
. . . the bali roadmap that focuses on efforts . . .
. . . die bali roadmap , konzentriert auf bemühungen . . .

Figure 13: Reflexive Pronoun (top: our system, bottom: baseline)

into both both, "," and "*dass*" or "," and "*deren*" (Figure 12).

6 Conclusion and Future Work

We demonstrated that our ℓMBOT-based machine translation system beats a standard tree-to-tree system (Moses tree-to-tree) on the WMT 2009 translation task English → German. To achieve this we implemented the formal model as described in Section 2 inside the Moses machine translation toolkit. Several modifications were necessary to obtain a working system. We publicly release all our developed software and our complete tool-chain to allow independent experiments and evaluation. This includes our ℓMBOT decoder presented in Section 4 and a separate C++ module that we use for rule extraction (see Section 3).

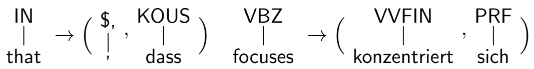

Figure 14: ℓMBOT rules generating a relative clause/reflexive pronoun

Besides the automatic evaluation, we also performed a small manual analysis of obtained translations and show-cased some examples (see Section 5.3). We argue that our ℓMBOT approach can adequately handle discontiguous phrases, which occur frequently in German. Other languages that exhibit such phenomena include Czech, Dutch, Russian, and Polish. Thus, we hope that our system can also successfully be applied for other language pairs, which we plan to pursue as well.

In other future work, we want to investigate full backwards application of ℓMBOT rules, which would be more suitable for the converse translation direction German → English. The current independent LM scoring of components has some negative side-effects that we plan to circumvent with the use of lazy LM scoring.

Acknowledgement

The authors thank Alexander Fraser for his ongoing support and advice. All authors were financially supported by the German Research Foundation (DFG) grant MA 4959 / 1-1.

References

André Arnold and Max Dauchet. 1982. Morphismes et bimorphismes d'arbres. *Theoret. Comput. Sci.*, 20(1):33–93.

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The WaCky Wide Web: A collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43(3):209–226.

Chris Callison-Burch, Philipp Koehn, Christof Monz, and Josh Schroeder. 2009. Findings of the 2009 Workshop on Statistical Machine Translation. In *Proc. 4th Workshop on Statistical Machine Translation*, pages 1–28.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *Proc. 43rd ACL*, pages 173–180.

David Chiang. 2007. Hierarchical phrase-based translation. *Computat. Linguist.*, 33(2):201–228.

David Chiang. 2010. Learning to translate with source and target syntax. In *Proc. 48th ACL*, pages 1443–1452.

Jason Eisner. 2003. Learning non-isomorphic tree mappings for machine translation. In *Proc. 41st ACL*, pages 205–208.

Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What's in a translation rule? In *Proc. HLT-NAACL*, pages 273–280.

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 *Proc. 44th ACL*, pages 961–968.

Hieu Hoang, Philipp Koehn, and Adam Lopez. 2009. A unified framework for phrase-based, hierarchical, and syntax-based statistical machine translation. In *Proc. 6th Int. Workshop Spoken Language Translation*, pages 152–159.

Liang Huang, Kevin Knight, and Aravind Joshi. 2006. Statistical syntax-directed translation with extended domain of locality. In *Proc. 7th Conf. Association for Machine Translation of the Americas*, pages 66–73.

Philip Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proc. HLT-NAACL*, pages 127–133.

Philipp Koehn, Amittai Axelrod, Alexandra Birch Mayne, Chris Callison-Burch, Miles Osborne, and David Talbot. 2005. Edinburgh system description for the 2005 IWSLT Speech Translation Evaluation. In *Proc. 2nd Int. Workshop Spoken Language Translation*.

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. In *Proc. ACL*, pages 177–180.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proc. EMNLP*, pages 388–395.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proc. 10th Machine Translation Summit*, pages 79–86.

Alon Lavie, Alok Parlikar, and Vamshi Ambati. 2008. Syntax-driven learning of sub-sentential translation equivalents and translation rules from parsed parallel corpora. In *Proc. 2nd ACL Workshop on Syntax and Structure in Statistical Translation*, pages 87–95.

Eric Lilin. 1978. *Une généralisation des transducteurs d'états finis d'arbres: les S-transducteurs*. Thèse 3ème cycle, Université de Lille.

Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proc. 44th ACL*, pages 609–616.

Yang Liu, Yajuan Lü, and Qun Liu. 2009. Improving tree-to-tree translation with packed forests. In *Proc. 47th ACL*, pages 558–566.

Andreas Maletti. 2010. Why synchronous tree substitution grammars? In *Proc. HLT-NAACL*, pages 876–884.

Andreas Maletti. 2011. How to train your multi bottom-up tree transducer. In *Proc. 49th ACL*, pages 825–834.

Andreas Maletti. 2012. Every sensible extended top-down tree transducer is a multi bottom-up tree transducer. In *Proc. HLT-NAACL*, pages 263–273.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computat. Linguist.*, 29(1):19–51.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. 41st ACL*, pages 160–167.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proc. 40th ACL*, pages 311–318.

Jean-Claude Raoult. 1997. Rational tree relations. *Bull. Belg. Math. Soc. Simon Stevin*, 4(1):149–176.

Helmut Schmid. 2004. Efficient parsing of highly ambiguous context-free grammars with bit vectors. In *Proc. 20th COLING*, pages 162–168.

Jun Sun, Min Zhang, and Chew Lim Tan. 2009. A non-contiguous tree sequence alignment-based model for statistical machine translation. In *Proc. 47th ACL*, pages 914–922.

Web-as-Corpus Consortium. 2008. SDeWaC — a 0.88 billion word corpus for german. Website: `http://wacky.sslmit.unibo.it/doku.php`.

Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computat. Linguist.*, 23(3):377–403.

Min Zhang, Hongfei Jiang, Aiti Aw, Haizhou Li, Chew Lim Tan, and Sheng Li. 2008a. A tree sequence alignment-based tree-to-tree translation model. In *Proc. 46th ACL*, pages 559–567.

Min Zhang, Hongfei Jiang, Haizhou Li, Aiti Aw, and Sheng Li. 2008b. Grammar comparison study for translational equivalence modeling and statistical machine translation. In *Proc. 22nd International Conference on Computational Linguistics*, pages 1097–1104.