Natural Language Engineering **18** (4): 491–519. © Cambridge University Press 2012
doi:10.1017/S1351324912000162

Statistical Translation After Source Reordering: Oracles, Context-Aware Models, and Empirical Analysis

MAXIM KHALILOV* and KHALIL SIMA'AN

Institute for Logic, Language and Computation
University of Amsterdam P.O. Box 94242
1090 GE Amsterdam, The Netherlands
e-mails: maxim@tauslabs.com,k.simaan@uva.nl

(*Received 31 July 2011; revised 9 April 2012; accepted 12 April 2012; first published online 14 May 2012*)

Abstract

In *source reordering* the order of the source words is permuted to minimize word order differences with the target sentence and then fed to a translation model. Earlier work highlights the benefits of resolving long-distance reorderings as a pre-processing step to standard phrase-based models. However, the potential performance improvement of source reordering and its impact on the components of the subsequent translation model remain unexplored. In this paper we study both aspects of source reordering. We set up idealized source reordering (oracle) models with/without syntax and present our own syntax-driven model of source reordering. The latter is a statistical model of inversion transduction grammar (ITG)-like tree transductions manipulating a syntactic parse and working with novel conditional reordering parameters. Having set up the models, we report translation experiments showing significant improvement on three language pairs, and contribute an extensive analysis of the impact of source reordering (both oracle and model) on the translation model regarding the quality of its input, phrase-table, and output. Our experiments show that oracle source reordering has untapped potential in improving translation system output. Besides solving difficult reorderings, we find that source reordering creates more monotone parallel training data at the back-end, leading to significantly larger phrase tables with higher coverage of phrase types in unseen data. Unfortunately, this nice property does not carry over to tree-constrained source reordering. Our analysis shows that, from the string-level perspective, tree-constrained reordering might selectively permute word order, leading to larger phrase tables but without increase in phrase coverage in unseen data.

1 Motivation

A major focus of current work in machine translation is the word order differences between source language and target language sentences. The main stream statistical machine translation (SMT) models (Brown *et al.* 1993; Zens, Och and Ney 2002; Koehn, Och and Marcu 2003; Och and Ney 2004) deal with word order differences

*Present address: TAUS Labs, Oudeschans 85, 1011KW Amsterdam, The Netherlands

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

alongside the target lexical choice (lexicon and phrase tables), albeit in two different statistical sub-models. Dynamic models integrated into the decoding process may reorder target phrases relative to their source sides, e.g., as a phrase sequence generation process with the possibility of dynamically swapping the order of neighboring phrases (Tillman 2004).

Similarly, hierarchical models of phrase reordering employ template phrases represented as synchronous grammars or tree transducers (e.g., Wu and Wong 1998; Chiang 2005), thereby accounting for hierarchical reordering patterns in data alongside lexical choice. These models explore more varied ranges of reordering phenomena, e.g., defined by at most inverting the order of sibling subtrees under each node in binary source/target trees (akin to inversion transduction grammar (ITG); Wu and Wong 1998).

Undoubtedly, the choice of source-target reordering patterns in machine translation is intertwined with the source-target lexical choice pairing. Statistically speaking, however, a generative story could first select a target word order, given the source sentence only, and then choose the target words, given the selected target word order and source words. One gross simplification of this statistical generative story is known as *source reordering* (or source permutation), where the reordered source string is fixated upon before translation starts (e.g., Xia and McCord 2004; Collins, Koehn and Kučerová 2005; Wang, Collins and Koehn 2007; Tromble and Eisner 2009; Khalilov and Sima'an 2010; Visweswariah *et al.* 2010). In source reordering, the words of the source string s are reordered to minimize word order differences with the target string t, leading to the source permuted string $\grave{s}$. Presumably, a standard Phrase-Based Statistical Machine Translation (PBSMT) system (Zens *et al.* 2002; Koehn *et al.* 2003; Och and Ney 2004) trained to translate from $\grave{s}$ to t should have an easier task than translating directly from s to t. The source reordering part, s-to-$\grave{s}$, can be implemented in various learning frameworks, possibly manipulating morpho-syntactic parses of s (e.g., Xia and McCord 2004; Collins *et al.* 2005; Khalilov and Sima'an 2010). Once a source reordering model exists, given s it will output a guess $g\grave{s}$ that might be different from the ideal permutation $\grave{s}$. The training of the translation model on a parallel corpus $g\grave{s}$-to-t implicitly conditions the target lexical choice on the source words and the kind of chosen order in $g\grave{s}$.

Earlier work highlights the benefits of source reordering as a pre-processing step, mainly for dealing with long-range reorderings, e.g., in Dutch–English the Dutch verb in final position of a relative clause may move far forward in the English counterpart. While pre-processing the input may represent an important utility of source reordering, there remains a major side effect largely unexplored: the impact of source reordering on the back end translation model. Once the latter model is trained on parallel data $\{\langle g\grave{s}, t\rangle\}$, with $g\grave{s}$ and t having far more monotone alignment between them than s and t have, the model acquires a rather different phrase table. How does the resulting phrase table differ from the original table, and what properties of this table could predict improvement in translation quality, i.e., besides the quality of the input $g\grave{s}$ relative to the ideal permutation $\grave{s}$?

In this paper we study the effects of source reordering on the back end translation model. To conduct this empirical study we first set up source reordering oracle

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

systems (ideal source reordering) and our own version of the currently popular syntax-driven models and observe how each of these source reordering choices affects the translation model.

After presenting the baseline (Section 2), the source reordering framework (Section 3), and the related work (Section 4), Section 5 reviews the data and the experimental setting for all subsequent empirical experiments. Section 6 presents experiments measuring oracle performance improvement for three language pairs (English to Dutch/Spanish/Chinese). Beside string-driven oracle results, we report results using ITG-like transductions over a single syntactic parse tree of *s*. While the literature on source reordering reports mixed performance improvements for different language pairs (e.g., Xia and McCord 2004; Collins *et al.* 2005; Wang *et al.* 2007; Khalilov and Sima'an 2010), our oracle experiments show great potential for source reordering. Our results confirm other reports that reordering a single syntactic tree could be rather suboptimal (e.g., Huang *et al.* 2009).

Subsequently, in Section 7 we present a source reordering model that manipulates the source parse tree by a cascaded sequence of (ITG-like) permutation transductions. The probability of each individual transduction in the sequence is conditioned on contextual features from *the outcome of all preceding transductions*. The conditioning on the outcome of preceding transductions is a departure from earlier approaches that learn statistically independent source permutation steps (e.g., Tromble and Eisner 2009; Visweswariah *et al.* 2010), leading to individual local reorderings, being unaware of reorderings done elsewhere in the tree.

In Section 8 we report experiments both with the oracle systems and our own model. We show that the present source reordering model succeeds to obtain significant improvements over a strong baseline, yet this improvement is always only the lesser share of the potential improvement exhibited by the oracle counterparts. Following this, in Section 9, we analyze the effects of source reordering on the back end translation model by comparing the quality of its input, its phrase table, and its output to the standardly trained PBSMT baseline. We find that source reordering leads to larger phrase tables than the baseline with better coverage of phrases in held-out data, and that its potential improvement over standard PBSMT system remains largely untapped. Remarkably, tree-based source reordering (oracle as well as model) leads to larger phrase tables than the baselines that do not have better coverage, suggesting that tree-based source reordering leads to somewhat noisier phrase tables. Our main conclusion, Section 10, is that it might be worthwhile to invest in novel tree-based source reordering approaches that lead to phrase tables with better coverage of unseen phrases, because this is where source reordering may offer most of its added value over standard phrase-based systems.

2 Translation model baseline: phrase-based SMT

The baseline is a standard phrase-based SMT system based on Och and Ney (2002) and Koehn *et al.* (2003). The phrase-based model works with a table of statistics over phrase pairs acquired from word-aligned parallel data under a simple definition of translational equivalence (Zens *et al.* 2002). A bilingual phrase is any aligned pair

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

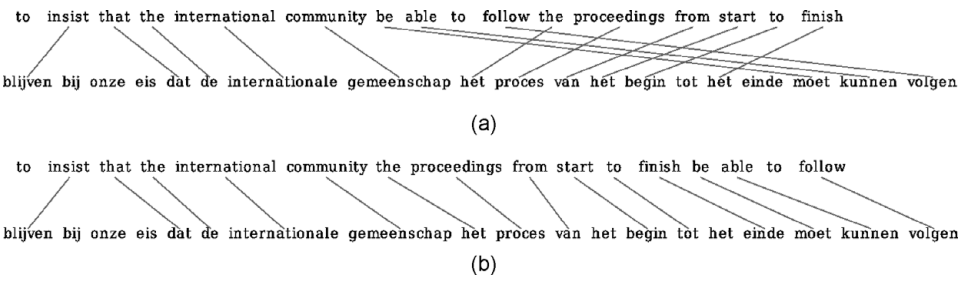


Fig. 1. Example of *Oracle string* unfolding. (a) Original bilingual phrase; (b) reordered bilingual phrase.

of m source words and n target words that satisfies two basic constraints: (1) words are consecutive along both sides of the bilingual phrase, and (2) no word on either side of the phrase is aligned to a word outside the phrase (Och and Ney 2004). The conditional probabilities of one phrase given its counterpart are interpolated linearly together with a set of other model estimates:

$$\hat{e}_1^I = \arg\max_{e_1^I} \left\{ \sum_{m=1}^{M} \lambda_m h_m\left(e_1^I, f_1^J\right) \right\} \tag{1}$$

where a feature function h_m refers to a system model, and the corresponding λ_m refers to the relative weight given to this model. A phrase-based system employs feature functions for a phrase pair translation model, a language model, a reordering model, and a model to score translation hypothesis according to length. The weights λ_m are usually optimized for system performance (Och 2003) as measured by bilingual evaluation understudy (BLEU) (Papineni *et al.* 2002).

Two phrase reordering methods are widely used in phrase-based systems. A distance-based reordering model provides the decoder with cost linear to the distance between words that are being reordered. This model constitutes the default for the Moses system (Koehn *et al.* 2007). And, a lexicalized block-oriented data-driven reordering model (Tillman 2004) considers three orientations: monotone (M), swap (S), and discontinuous (D) (MSD), while the reordering probabilities are conditioned on the lexical context of each phrase pair.

3 Source reordering: framework

We start out from a word-aligned parallel corpus, consisting of triples $\langle s, a, t\rangle$, a source s, target t, and word alignment a. Source reordering assumes that a permutation of s, called $\grave{s}$, is first generated with a model $P_r(\grave{s} \mid s)$ followed by a phrase translation model $P_t(t \mid \grave{s})$. The desired permutation $\grave{s}$ is one that has minimum word order divergence from t, i.e., the alignment of $\grave{s}$ with t would contain the least number of crossing alignments. Figure 1 exhibits a reordering at string level that resolves the crossing alignments.

Practically, the original parallel corpus $\{\langle s, a, t\rangle\}$ is split to two parallel corpora: (1) a source-to-permutation parallel corpus (consisting of $\langle s, a, \grave{s}\rangle$), and (2) a permutation-to-target parallel corpus (consisting of $\langle g\grave{s}, \grave{a}, t\rangle$), where $g\grave{s}$ is the output

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

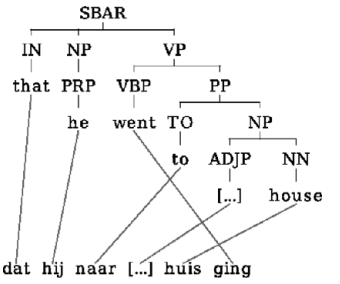


Fig. 2. Example of crossing alignments and long-distance reordering using a source parse tree.

of a source reordering model $P_r(\grave{s} \mid s)$, and $\grave{a}$ results from automatically word aligning $\langle g\grave{s}, t\rangle$. The latter parallel corpus is used for training a phrase-based translation system $P_t(t \mid g\grave{s})$, while the former corpus is used for training a source reordering model $P_r(\grave{s} \mid s)$.

The problem of finding the optimal permutation of the source string is computationally intractable (see Tromble and Eisner 2009). However, various constraints can be made on unfolding the crossing alignments in *a*. A common approach is to assume a binary parse tree for the source string, and define a set of eligible permutations by binary ITG transductions. This defines permutations resulting from at most inverting pairs of children under nodes of the source tree. Figure 2 exhibits a long-range reordering of verb in English-to-Dutch translation: Inverting the order of the children of the VP node would unfold the crossing alignment. However, crossing alignments represented as non-constituents cannot be resolved. This difficulty can be circumvented by employing multiple alternative parse trees, by applying heuristic transforms (e.g., binarization) to the tree to fit the alignments (Wang *et al.* 2010), or by defining new local transductions on top of child permutation (ITG).

4 Existing work on source reordering

As mentioned earlier, source reordering is but one of the existing approaches to deal with word order differences during translation. Alternatives include phrase reordering methods employed in phrase-based models (Zens and Ney 2003; Tillman 2004; Koehn *et al.* 2007), as well as hierarchical, syntax-augmented phrase-based systems often working with synchronous grammars (e.g., Wu 1997; Galley *et al.* 2004; Zollmann and Venugopal 2006; Chiang 2007; DeNeefe *et al.* 2007; Mylonakis and Sima'an 2011).

Source reordering has been reported to be useful for PBSMT for a variety of language pairs with high mutual word order disparity (Xia and McCord 2004; Collins *et al.* 2005; Costa-jussà and Fonollosa 2006; Popovic' and Ney 2006; Li *et al.* 2007; Zwarts and Dras 2007; Dyer and Resnik 2010; Genzel 2010; Visweswariah *et al.* 2010).

A rather popular class of source-reordering algorithms involves syntactic information and aims at minimizing the need for reordering during translation by permuting the source sentence (Collins *et al.* 2005; Li *et al.* 2007; Wang *et al.* 2007;

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

Zwarts and Dras 2007; Genzel 2010; Khalilov and Sima'an 2010). Some systems perform source permutation using a set of hand-crafted rules (Collins *et al.* 2005; Wang *et al.* 2007; Ramanathan *et al.* 2008; Isozaki *et al.* 2010), others make use of automatically learned reordering patterns extracted from the plain training data, the corresponding parse or dependency trees, and the alignment matrix (Li *et al.* 2007; Dyer and Resnik 2010; Genzel 2010; PVS 2010; Visweswariah *et al.* 2010; Khalilov and Sima'an 2011; Mylonakis and Sima'an 2011). While the majority of systems operate on the source side of parallel corpus, there has been some algorithms that require access to source and target tree banks (Xia and McCord 2004; Khalilov 2009). Only a few models (Visweswariah *et al.* 2011) work non-hierarchically at the word alignment level.

Following Li et al. (2007), Tromble and Eisner (2009), Genzel (2010), and Khalilov and Sima'an (2010), in this work source reordering is viewed as a word permutation learning problem. The space of permutations is approached efficiently using a binary ITG-like synchronous context-free grammar put on the parallel data. Similarly, a local ITG-based tree transducer with contextual conditioning is used in Li *et al.* (2007) and Khalilov and Sima'an (2010), and preliminary experiments on a single language pair show improved performance. The latter two are limited to ITG-like permutations, employ limited context conditioning, and are explored either on small training data or on a single language pair. Particularly, the model in Li *et al.* (2007) is explicitly aimed at long-distance reorderings (English–Chinese), prunes the alignment matrix gradually to fit the source syntactic parse, and employs Maximum-Entropy modeling to choose the optimal local ITG-like permutation step of sister subtrees but interleaves that step with a translation step. The latter property implies that Li *et al.* (2007) has a unique status because it lies somewhere between pure source reordering and models that conduct reordering dynamically during decoding. Apart from this major difference, the model presented in Section 7 differs substantially from Li *et al.* (2007) and other earlier work because it (1) incorporates other kinds of tree transduction operations than those promoted by ITG, and (2) works with the unmodified alignment matrix but learns reorderings only from those alignments that are consistent with the tree, thereby avoiding the effects of heuristics for pruning alignments to fit the tree structure (e.g., Li *et al.* (2007)). These properties allow us to study and analyze source reordering with/without syntax using a reasonably clean but well-equipped transduction model.

Source reordering is learned over binary synchronous trees in DeNero and Uszkoreit (2011) and pipe-lined with a phrase-based SMT system. The reordering model learns a binary choice over all binary trees that fit with word alignment independently from one another. Learning binary synchronous trees for reordering as independent local binary decisions contrasts with our choice for syntactic trees for learning a cascade of local context-dependent source reorderings.

An elegant approach is described in Dyer and Resnik (2010), where the source reordering derivations ($s \rightarrow \grave{s}$) constitute a latent component modeled by local child-order permutation under the nodes of a source syntactic tree. The reordering parameters are defined over individual context-free grammar (CFG) productions in these derivations without context and cast together with the standard translation

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

model parameters in a log-linear interpolation, estimated by maximizing the posterior probability. The fact that the child-order permutations are cast as a CFG, contrasts sharply with our use of reordering patterns collected from the training data and applied as individual transducers. While the translation model in Dyer and Resnik (2010) is used during training the latent source reordering model parameters, the parameters of the two models are formulated independently of each other.

A large majority of existing works report encouraging performance improvements by source reordering but none tries to quantify the range of potential improvements with/without syntax-based reordering, and none provides an analysis of why it sometimes fails to return the expected performance improvements. The present paper constitutes a harmonized, significantly extended write-up of the work described in Khalilov and Sima'an (2011). The latter publication briefly and informally presents oracle results and the source reordering model (Section 7). In the present paper we concentrate on a *novel* empirical study and analysis of statistical translation after source reordering with *fresh insights* into the potential weaknesses of syntax-driven source reordering before translation.

5 Data and experimental setting

5.1 Word order difference between languages under consideration

Systematic differences between the word order of the source and the target languages is an important input parameter to an SMT system. The three target languages (Spanish, Dutch, and Chinese) employed in our experiments exhibit different kinds of word order disparity with English (the fixed source language).

Spanish and English are both subject–verb–object (SVO) language and follow a similar but slightly different word order and distinct inflectional characteristics. While adjectives usually appear before nouns in English, the majority of nouns precede adjectives in Spanish.

In contrast to English, the verbs in a Dutch relative clause always appear in the final position. This could lead to long-range dependencies with subjects and objects. Another noticeable difference is that in Dutch adverbials may appear between a verb and its object, while this is irregular in English.

Chinese and English exhibit major word order differences, e.g., the order within a clause (Chinese relative clauses appear before the noun being modified), and the position of pre- and post-modifiers of a noun or pronoun (prepositional phrases often appear before the head they modify) (cf. Li 1998). These differences complicate the reordering decisions in SMT and could result in increased translation model that slows down the decoding.

5.2 Experimental setting

Data. In our experiments we used English–Dutch and English–Spanish European Parliament data and an extraction from the English–Chinese Hong Kong Parallel Corpus. All the sets were provided with one reference translation. Basic statistics of the training, development, and test data can be found in Table 1.

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

Table 1. *Statistics of the training, development and test corpora*

Parameter	Dutch	English	Spanish	English	Chinese	English
			Training corpus			
Sentences	1.2 M	1.2 M	1.4 M	1.4 M	1.5 M	1.5 M
Words	32.9 M	33.0 M	40.1 M	38.54 M	35.35 M	35.00 M
Vocabulary	228 K	104 K	168 K	119 K	136 K	245 K
Average sentence length	27.20	27.28	28.80	27.67	24.06	23.83
			Development corpus			
Sentences	500	500	1.9 K	1.9 K	500	500
Words	13.2 K	13.2 K	60.1 K	58.2 K	14.0 K	12.9 K
Vocabulary	3.2 K	2.9 K	8.2 K	6.4 K	2.3 K	3.1 K
Average sentence length	26.43	26.48	30.24	29.31	28.11	25.77
			Test corpus			
Sentences	1 K	1 K	1.9 K	1.9 K	500	500
Words	27.7 K	27.8 K	58.0 K	59.5 K	13.5 K	12.2 K
Vocabulary	5.1 K	4.6 K K	6.4 K	8.1 K	2.1 K	2.8 K
Average sentence length	27.71	27.82	29.32	30.07	26.92	24.36

Experimental setup. The word alignment was obtained by running GIZA++[1] (Och 2003), supported by mkcls[2] (Och 1999) tool. All phrase translation systems in this paper (including the back-end systems for oracle and source reordering) are trained on word alignments obtained by the grow-diag-final-and heuristic for symmetrizing alignments produced from both translation directions (cf. Och and Ney 2003). The source reordering models (before translation) are trained on simplified word alignments as explained in relevant sections.

The PBSMT system that we consider in this study employs Moses toolkit (Koehn *et al.* 2007). We followed the guidelines provided on the Moses web page[3] to train and use the system.

All language models were trained with SRI LM toolkit (Stolcke 2002). Target-side language models (for Dutch, Spanish, and Chinese) use 5-grams, while the idealized English ($\grave{s}$) is modeled using 3-grams.

We use Stanford parser[4] (Klein and Manning 2003) as a source-side parsing engine. The parser was trained on the WSJ Penn treebank (Marcus, Santorini and Marcinkiewicz 1993) provided with fourteen syntactic categories and forty-eight part-of-speech (POS) tags. The evaluation conditions were case-sensitive and included punctuation marks. For Maximum-Entropy modeling we used the maxent toolkit[5].

[1] code.google.com/p/giza-pp/
[2] http://www.fjoch.com/mkcls.html
[3] http://www.statmt.org/moses/
[4] http://nlp.stanford.edu/software/lex-parser.shtml
[5] http://homepages.inf.ed.ac.uk/lzhang10/maxent_toolkit.html

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

6 Idealized source reordering: oracles

Work on source reordering for PBSMT reports that permuting the source words to minimize the order differences with the target sentence could improve translation performance. The reported improvements often vary considerably, both due to language pair and choice of source reordering approach. In this section we attempt gaging the performance improvement of various idealized (oracle) source reordering systems. The oracle task is simply to perform unfolding operations on the crossing links in alignment a (estimated between corpora s and t) that leads to as monotone alignment $\grave{a}$ as possible. We introduce a set of tree-based constraints that control the unfolding of alignment crossings. We measure the impact of (un)folded alignment crossings on the final performance of the PBSMT system (see Table 2). In Section 9 we analyze the impact of these oracles on the subsequent translation system.

6.1 Oracle models

The oracle models discussed here are greedy in nature in that they execute a *sequence of local reordering* operations unfolding crossing alignments in a given word alignment. It is possible to generalize this greedy approach into a dynamic search-based approach by defining a figure-of-merit for every possible sequence of local reorderings based on independent, local figures-of-merit (cf. Tromble and Eisner 2009). The definition of an adequate, decomposable figure-of-merit, particularly for tree-based systems, is difficult and will necessarily involve other kinds of simplifying assumptions.

Crucially, the word alignments used for preparing the source-permuted strings for oracles are obtained using Giza++ with intersection heuristic. In contrast, the translation systems, including all back-end models for source reordering, are trained on word alignments obtained by Giza++ with the grow-diag-final-and heuristic.

Oracle string. The objective that this algorithm optimizes is to obtain the source permutation $\grave{s}$ that unfolds all crossing alignments in a, the alignment between source and target strings. This algorithm (*Oracle string*) traverses the alignment a from left-to-right and for each crossing link it comes across, it moves the source word involved in the crossing to the nearest source position that unfolds that crossing. Figure 3 shows the pseudo-code for this algorithm. The pseudo-code produces the permutation $\grave{s}$ of s that contains no crossing alignments. Figure 1 shows an example of word reordering done at the string level.

Oracle parse tree with permute siblings. The oracle tree system employs a source syntactic tree τ *to constrain* Oracle string. Let the *current crossing alignment pair* with source side positions be given by i and k (underlined in Figure 3). Let the current parse tree at the mth step of this algorithm be τ^m, obtained after m preceding steps and initialized at the beginning with $\tau_0 = \tau$. Having a crossing alignment in

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

Function Oracle-String
Input: $s = w_1 \dots w_n$ and a word alignment a
Output: $\grave{s}$

BEGIN
Initialize: $\acute{a} = \{\langle i, j, i\rangle \mid \langle i, j\rangle \in a\}$
Repeat
  For $i = 1$ to $i = n$
    For $k = i + 1$ to $k = n$
      If exist $\langle i_0, j, i\rangle \in \acute{a}$ and $\langle k_0, l, k\rangle \in \acute{a}$
        If $\langle i, j\rangle$ and $\langle k, l\rangle$ are crossing
          $\acute{a} =$ **Move**$(\langle i_0, j, i\rangle, \acute{a}, k)$
Until $\acute{a}$ does not change any more
Let $\grave{s} = \grave{w}_1 \dots \grave{w}_n$ where:
  $\forall 1 \le i \le n$: $\grave{w}_i := w_k$ iff $\langle k, \star, i\rangle \in \acute{a}$.
Return $\grave{s}$
END

#Exchange a link $\langle i, j\rangle \in \acute{a}$ with $\langle k, j\rangle$
Function Move
Input: $\langle i_0, j, i\rangle$, $\acute{a}$ and k
Output: $\acute{a}$ adapted

BEGIN
Remove link $\langle i_0, j, i\rangle$ from $\acute{a}$
#get rid of the gap source position i
For $x = 1$ to $x = k - i$
  replace $\langle \star, \star, i + x\rangle \in \acute{a}$ with $\langle \star, \star, i + x - 1\rangle$
Add $\langle i_0, j, k\rangle$ to $\acute{a}$
Return $\acute{a}$
END

Fig. 3. Pseudo code Oracle string. The algorithm creates the $\grave{s}$ permutation by resolving crossing alignments in a. A triple $\langle i_0, j, i_n\rangle$ in $\acute{a}$ consists of the original source position i_0 for word w_{i_0}, the aligned target position j, and the current source position i_n for word w_{i_0}. The wild-card $\star$ in a triple stands for 'any existing value' and is used when the value is irrelevant at that point.

hand, we search bottom-up in τ_m for the *largest* subtree[6] with root address μ that fulfills the following three requirements:

(R1) Node μ covers both positions i and k,
(R2) The crossing alignment given by source-sides i and k can be resolved by *permuting the positions of the child nodes of* μ, and
(R3) Swapping the positions of these child nodes does not introduce new crossing alignments.

If and when a suitable node μ is found, this oracle unfolds this crossing alignment pair by conducting swapping in τ_m leading to a new parse tree τ_{m+1}, else the alignment crossing is skipped to the next one and τ_{m+1} is equivalent to τ_m. In practice this adds a constraint on unfolding steps (1)–(3) in Oracle string (Figure 3). NULL-aligned words do not prevent unfolding crossings because we include them with the adjacent words that are involved in the crossings. We call this configuration *Oracle tree*.

Figure 2 shows an example. According to the *Oracle tree* constraint, the word 'went' can be placed in the end of the sentence, since the replacement can be done as a swapping of 'VBP' and 'PP' categories. The same happens for the word 'reflect'

[6] The largest subtree fulfilling the requirements will also contain the longest contiguous sequence of crossing alignments that can be resolved in a single step of sibling permutation. In other words, two whole blocks/phrases of crossing alignments (contiguous on source and target sides) dominated by two sibling nodes swap source positions. The demand that the swapping does not introduce new crossings is interpreted as search for two such 'crossing blocks'.

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

swapping with 'S' constituent in Figure 2, but not for the chunks 'the positions' and 'not properly': This crossing cannot be resolved under the tree constraint, since they are not dominated by sibling syntactic categories.

Oracle tree with delete descendants, permute siblings. This oracle implements an additional mechanism of tree modification to increase the number of reordering permutations in comparison with *Oracle tree* algorithm. Here we allow for a tree transduction operation that deletes a maximum number d of intervening node layers in the source tree before applying sibling permutation as in *Oracle tree*. This oracle can be seen as an attempt to bridge the difference between unconstrained string-level and tree-constrained unfolding of crossing alignments. Tree layers that intervene between a crossing alignment pair are removed to relax tree constraints.

This algorithm consists of a single refinement when the three requirements (R1, R2, and R3) of *Oracle tree* above are not fulfilled. If these requirements are not fulfilled by any node μ in τ_m, then this algorithm enters *a loop for deleting the smallest number of layers (at most d) before attempting to unfold the alignment crossing by standard permute siblings as in Oracle tree*. At every round in this loop it conducts two steps:

(1) Seek the *smallest subtree* covering both source sides of the alignment crossing pair (with source side i and k in Figure 3). Let the root of this subtree have address μ_p, and suppose the address of the child node that covers source side positions i and k are μ_p^l and μ_p^r, respectively.
(2) For each $\mu_p^x \in \{\mu_p^l, \mu_p^r\}$ if the alignments covered by the subtree under μ_p^x do *not* consist of a contiguous span (i.e., block) on the target side, then node μ_p^x is deleted from τ_m and the child subtrees of μ_p^x are pulled up to become the direct children of μ_p, leading to tree $\grave{\tau}_m$. When this deletion has been attempted/done for both child nodes, the number of deleted layers is incremented by one.

If the *Oracle tree* still cannot be applied to resolve the current alignment crossing by permuting the children of μ_p after the deletion, then the loop is re-entered to attempt deleting another layer from $\grave{\tau}_m$. If the crossing alignment at hand is resolved, the algorithm continues to the next one with the modified tree (after all deletions and permute-siblings) renamed as the current tree τ_{m+1}, otherwise it proceeds to treat the next crossing alignment with the trees obtained after deletions as the current tree τ_{m+1}.

This oracle is illustrated in Figure 4. The word 'must' cannot be moved to the beginning of the sentence in Figure 4(a) by *Oracle tree*. Instead, this is done in two steps. Firstly, the VP dominating the words 'must' and 'apply' (a node dominating a source side with a non-contiguous target side) is deleted under the current node S (the root of the smallest subtree dominating the crossing alignments), the transformed tree is shown in Figure 4(b). Subsequently, the siblings under S in the resulting tree are permuted, 'must' is reordered across the whole clause and placed to the first position (see Figure 4(c)). We call this system *Oracle mod*. Figure 5 shows an example in which crossing alignment links cannot be unfolded without deleting four intervening layers.

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

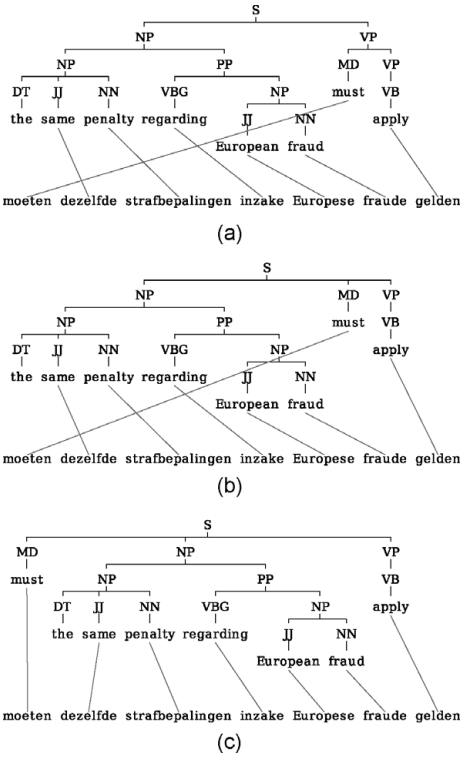


Fig. 4. Example of text monotonization with tree transformation. (a) Original parse tree; (b) parse tree with deleted VP category; (c) reordered parse tree.

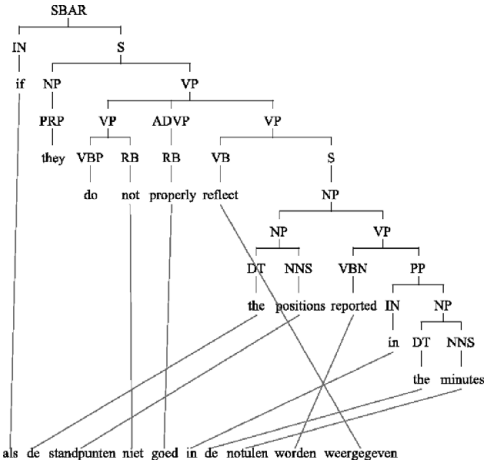


Fig. 5. Example of unfoldable alignment crossings.

6.2 Oracle results

Table 2 contrasts oracle results with the performance shown by standard PBSMT systems. The experimental setup is detailed in Section 8. We consider the following baseline configurations: *PBSMT* – Moses-based PBSMT with distance-based

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

Table 2. *Summary of oracle results*

	EnNl			EnEs			EnZh		
	Dev	Test		Dev	Test		Dev	Test	
System	BLEU	BLEU	NIST	BLEU	BLEU	NIST	BLEU	BLEU	NIST
Baselines									
PBSMT	23.88	24.04	6.29	32.31	31.70	7.48	18.71	22.21	5.28
PBSMT + MSD	24.07	24.04	6.28	32.45	31.85	7.47	18.99	21.18	5.30
Moses-chart	23.94	24.93	6.39	30.58	31.80	7.41	19.93	23.90	5.41
Oracle results with distance-based and MSD reordering									
Oracle tree	24.70	24.80	6.32	32.76	32.21	7.51	20.23	23.44	5.35
Oracle string	26.28	27.02	6.50	34.09	33.52	7.60	23.01	26.08	5.52
Oracle mod + 2lt	25.05	25.04	6.36	32.24	32.18	7.51	20.64	23.75	5.37
Oracle mod + 3lt	25.11	25.27	6.37	32.22	32.34	7.52	20.71	23.59	5.37
Oracle mod + 5lt	25.07	25.23	6.37	32.51	32.37	7.55	20.93	23.93	5.39
Oracle results using monotone translation (without reordering models)									
Oracle tree	24.57	24.74	6.32	32.72	32.20	7.51	20.15	23.24	5.32
Oracle string	26.22	26.85	6.45	34.02	33.50	7.60	22.92	26.00	5.50
Oracle mod+5lt	24.98	25.16	6.35	32.45	32.33	7.54	20.81	23.83	5.37

reordering model; *PBSMT + MSD* – Moses-based PBSMT with distance-based reordering and MSD; and *Moses-chart* and the Hierarchical Moses-chart PBSMT.

For each of the Oracle reordering systems we show results with two settings of reordering in the back-end phrase-based translation systems: in the one case we allow distance-based and MSD reordering (middle rows in Table 2), and in the other with these reordering capabilities deactivated (the last three rows in Table 2). Depending on the number of parse tree levels allowed to be deleted, we consider three *Oracle mod* systems: with two (*2lt*), three (*3lt*), and five (*5lt*) levels of descendants allowed to be deleted for a more flat parse tree structure before sibling permutation. The impact of corpus monotonization on translation system performance is measured using the final point of weight optimization on the development set (*Dev BLEU*), as well as on the test set (*Test BLEU/NIST*).

The main conclusion that can be drawn from the oracle results is that the source reordering defined in terms of parse tree transduction can potentially lead to increased translation quality (up to 1.2 BLEU points for English–Dutch, 0.5 for English–Spanish, and 1.7 for English–Chinese). At the same time, the substantial remaining gap between the performance of *Oracle string* and tree oracle systems (≈2.2 BLEU points for English–Dutch, ≈1.3 BLEU points for English–Spanish, and ≈2.5 BLEU points for English–Chinese) shows that there are many crossing alignments that cannot be unfolded using simple, local transforms over a *single* source-side syntactic tree. Interestingly, deactivating the phrase reordering in the back-end translation system impacts the oracle performance only slightly. One solution might be to employ syntactic node labels from the source parse but not take its bracketing literally. The parse-forest interpretation (also called *labels chart*) of the source syntactic tree (Mylonakis and Sima'an 2011) is one possible way to do so. In this paper, we constrain the work to the single source parse accepting the

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

upper bound oracle performance is suboptimal and concentrate on exploring what source reordering does to the back-end translation system.

7 A Statistical model of syntactic tree transduction

We now present a model that learns from the source permuted parallel corpus (containing tuples $\langle s, a, \grave{s} \rangle$) a probabilistic optimization ($\arg\max_{\pi(s)} P_r(\pi(s) \mid s, \tau_s)$), where τ_s is the source parse and $\pi(s)$ is some eligible permutation of s. We view the permutations leading from s to $\grave{s}$ as a sequence of local tree transductions $\tau_{\grave{s}_0} \rightarrow \ldots \rightarrow \tau_{\grave{s}_n}$, where $\grave{s}_0 = s$ and $\grave{s}_n = \grave{s}$, and each transduction $\tau_{\grave{s}_{i-1}} \rightarrow \tau_{\grave{s}_i}$ is defined using any of the two kinds of local tree transduction operations (explained next, but see also Section 6) or alternatively no operation (NOP).

Theoretically speaking, we can define the probability of the sequence of local tree transductions $\tau_{\grave{s}_0} \rightarrow \ldots \rightarrow \tau_{\grave{s}_n}$ as

$$P(\tau_{\grave{s}_0} \rightarrow \ldots \rightarrow \tau_{\grave{s}_n}) = \prod_{i=1}^{n} P(\tau_{\grave{s}_i} \mid \tau_{\grave{s}_1} \ldots \tau_{\grave{s}_{i-1}}) \approx \prod_{i=1}^{n} P(\tau_{\grave{s}_i} \mid \tau_{\grave{s}_{i-1}}) \tag{2}$$

The latter Markov approximation seems reasonable because the preceding tree $\tau_{\grave{s}_{i-1}}$ is the result of all preceding transductions, thereby summarizing the changes but forgetting exact transductions leading to these changes. This definition deviates from the majority of earlier works, e.g., Tromble and Eisner (2009) and Visweswariah *et al.* (2010), which defines 0th-order Markov models where every transduction step is independent from preceding transductions. The work in Li *et al.* (2007) presents a model that conditions ITG-like transductions on local context, but that model conducts a recursive process over the parse tree that involves a single reordering and single translation step. Our model conducts full source reordering before actual translation starts.

The Markov conditioning makes the optimization $\arg\max_{\tau_{\grave{s}_x}} \prod_{i=1}^{x} P(\tau_{\grave{s}_i} \mid \tau_{\grave{s}_{i-1}})$, over an interesting space like that defined by ITG, intractable. Observe that the sequence of transductions can be represented compactly as a sequence of parse-forests, one parse-forest for each $i = 1 \ldots x$. However, the conditioning $P(\tau_{\grave{s}_i} \mid \tau_{\grave{s}_{i-1}})$ implies that every tree in forest i is conditioned on a tree in forest $i-1$. If the number of trees in every forest is guaranteed to be linear in input length n, the optimization will cost n^x. Reasonably (avoiding transduction cycles), the number of transductions x could be linear in n, leading to time-complexity $O(n^n)$. Earlier work (e.g., Tromble and Eisner 2009; Visweswariah *et al.* 2010) avoids the conditioning context altogether, effectively assuming that each intermediate permutation is independent from the preceding ones. Next we make a different approximation that retains the conditioning context of each local transduction but gives up on the global optimization over whole transduction derivations.

Algorithm. Initially all nodes in the source tree $\tau_{\grave{s}_0}$ are marked as *unvisited*. Our algorithm (see Figure 6) takes the next *unvisited node* with address μ in a *top-down, left-to-right (pre-order)* tree traversal of the current tree $\tau_{\grave{s}_{i-1}}$, marks this node as *visited* and then statistically selects the most likely of three transduction operations

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

Algorithm MERrd[+xlt]
Input: Source tree $\tau_{\grave{s}_0}$
Output: Permuted source string $\grave{s}$

BEGIN
  Initialization: All nodes in $\tau_{\grave{s}_0}$ are marked as *unvisited*.
  i = 0;
  For $\mu \in \tau_{\grave{s}_i}$ starting from root in top-down, left-right order
    If current node μ is unvisited
      Mark μ as visited.
      $\tau_{\grave{s}_{i+1}} := APPLY.OPERATION(\mu, \tau_{\grave{s}_i})$
      i:= i + 1
  Return $\grave{s} := FRINGE(\tau_{\grave{s}_i})$
END

Fig. 6. General pseudo-code for the described model (MERrd[+xlt]). The function *FRINGE* returns the ordered sequences of terminals (words) at the leaf nodes (the fringe) of the input tree. Function *APPLY.OPERATION* applies the statistical MaxEnt-trained conditional probability selector to select the most likely operation (**Perm**, **Del-Perm**, **NOP**) for input node μ in tree $\tau_{\grave{s}_i}$.

(defined next), and applies the selected operation to the current node. Note that our operations reord child nodes and could delete certain nodes, and hence the nodes marked *visited* remain as such (or are deleted) during the transduction sequence to avoid revisiting them.

Let the current tree be $\tau_{\grave{s}_{i-1}}$ and let the current node has address x, syntactic label N_x, and directly dominate α_x (the ordered sequence of node labels under x). We optimize the conditional probability of the transduction operation it employs over the following kinds of operations, beside NOP:

Perm Permutes the children of x in $\tau_{\grave{s}_{i-1}}$ with probability

$$P(\pi(\alpha_x) \mid N_x \rightarrow \alpha_x, C_x) \tag{3}$$

where $\pi(\alpha_x)$ is a permutation of α_x (the ordered sequence of node labels under x) and C_x is a local tree context of node x in tree $\tau_{s'_{i-1}}$. Effectively, we employ a context-sensitive tree-rewrite rule: if C_x then $\overset{N_x}{\underset{\alpha_x}{\triangle}} \rightarrow \overset{N_x}{\underset{\pi(\alpha_x)}{\triangle}}$.

Del-Perm Select a child of x to delete, pull its children up directly under x, effectively changing α_x to some α_x^d, and then permute the children of the latter,

$$P(\alpha_x \rightsquigarrow \alpha_x^d, \pi(\alpha_x) \mid N_x \rightarrow \alpha_x, C_x) \tag{4}$$

where $\alpha_x \rightsquigarrow \alpha_x^d$ symbolizes the result of deleting a subtree under a child of x. This operation is also extended to subtrees of depth $n \in \{1, 2, 3, 5\}$ under x, i.e., a child is depth 1, a child with its children is depth 2, and so on. Effectively,

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

we employ a sequence of two context-sensitive tree-rewrite rules (1) if C_x then $\overset{N_x}{\underset{\alpha_x}{\triangle}} \rightarrow \overset{N_x}{\underset{\alpha_x^d}{\triangle}}$ followed by (2) if C_x then $\overset{N_x}{\underset{\alpha_x^d}{\triangle}} \rightarrow \overset{N_x}{\underset{\pi(\alpha_x^d)}{\triangle}}$.

Obviously, the number of possible permutations of α_x is factorial in the length of α_x. Fortunately, the source-permuted training data exhibits only a fraction of possible permutations even for longer α_x sequences. Furthermore, by conditioning probability on local context C_x, the number of permutations is limited to a handful set. By limiting the space of permutations in this fashion, however, our tree-based models are more vulnerable to the sparsity of local permutations and their contexts in the training data.

Now that we optimize the choice for each transduction on its own in a derivation, we cannot anymore guarantee that the full derivation will lead to good source reordering. Particularly, we risk making locally optimal choices but globally sub-optimal ones. We alleviate this by choosing individual transductions that lead to improved permutations both at the tree- and string-levels. We interpolate the terms in (3) and (4) with probability ratios of the idealized (desired) source-reordered language model P_{lm}:

$$P(TRANS_i|\ N_x \rightarrow \alpha_x, C_x) \times \frac{P_{lm}(\grave{s}_{i-1})}{P_{lm}(\grave{s}_i)}$$

where $TRANS_i$ is any of the two transduction operations or NOP, and P_{lm} is a language model trained on the $\grave{s}$ side of the corpus $\{\langle s, a, \grave{s}\rangle\}$ (trained over $\grave{s}$ strings obtained by resolving cross alignments). Optimizing this interpolation can be interpreted as optimizing the optimal permutation that can serve as input for a subsequent translation model among those that have good tree transductions. The language model ratio can be seen as one factor in the log-linear model[7], incorporating lexical features from windows of lexical context left and right of the subtree under the current node in the tree before the transduction is applied as well as in the tree resulting from the transduction.

Estimates and features. We estimate the string probabilities $P_{lm}(.)$ using 3-gram language models trained on the $\grave{s}$ side of the source-permuted parallel corpus $\{\langle s, a, \grave{s}\rangle\}$. We estimate the conditional probability $P_r(TRANS \mid N_x \rightarrow \alpha_x, C_x)$ using a Maximum-Entropy framework, where feature functions are defined to capture the permutation as a class, the node label N_x and its head POS tag, the child sequence α_x together with the corresponding sequence of head POS tags and other features corresponding to different contextual information[8].

[7] Although we could train this language model feature also with Maximum-Entropy, just like the other features, we found that such lexical features make training very slow. Therefore, we treat this language model feature separately from others.

[8] It is worthwhile highlighting the fact that the reordering transductions are learned from the word-aligned parallel corpus with source parse trees: the statistical learner will learn only those permutations actually encountered in the data for every conditioning context, i.e., production and contextual features. The number of possible child permutations is very small relative to the number of all possible permutations. The same applied to the other kinds of transductions.

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

We collect the following set of fifteen features by using the corresponding tree-based Oracle traversals (Section 6) of the alignments in the training data:

- *Local tree topology*. Subtree instances that include parent node and the ordered sequence of child node labels (1).
- *Dependency features*. Features that determine the POS tag of the head word of the current node (2), together with the sequence of POS tags of the head words of its child nodes (3), the POS tag of the head word of the parent (4), and grandparent nodes (5).
- *Syntactic features*. Apart from the whole path from the current node to the tree root node (6), we used three binary features from this class: (7) whether the parent node is a child of the node annotated with the same syntactic category, (8) whether the parent node is a descendant of the node annotated with the same syntactic category, and (9) if the current subtree is embedded into a '*S-SBAR*' subtree[9].
- *POS lexical features*. Bi- and tri-grams of POS tags of the left- and right-hand sides neighboring words (10–13).
- *Counters*. A number of words covered by a given constituent (14) and a number of children of the given node (15).

8 Translation and reordering experiments

We use exactly the same data and experimental setting used for the oracle experiments described in Section 6. For all systems and experiments in the sequel we employ a PBSMT back-end system that works both with lexicalized phrase reordering and the MSD reordering (see Section 2 for details).

The word alignments used for preparing the source-permuted strings for training our own models are obtained by *disambiguating* the alignments obtained from Giza++ followed by the grow-diag-final-and symmetrization heuristic. The disambiguation aims at resolving many-to-one or one-to-many alignments leaving only one-to-one alignments and unaligned words. The disambiguation selects among one-to-many and many-to-one alignment links for the same word only the single link with the highest score according to the lexical probability model obtained from training with Giza++.[10] As mentioned earlier, all translation systems, including the back-end models for source reordering, are trained on word alignments obtained by Giza++ with the grow-diag-final-and heuristic.

System names. *MERrd* configuration corresponds to the PBSMT system with the source side of the parallel corpus reordered using the present Maximum Entropy source reordering model, but the transduction operations are limited to permuting the children only. *MERrd+xlt* configuration refers to the set of systems which,

[9] The latter feature intends to model divergence in word order in relative clauses between Dutch and English which is illustrated in Figure 2.

[10] We also experimented with Giza++ followed by symmetrization using the intersection heuristic and found that the BLEU translation results did not change.

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

Table 3. *Experimental results*

System	EnNl			EnEs			EnZh		
	Dev	Test		Dev	Test		Dev	Test	
	BLEU	BLEU	NIST	BLEU	BLEU	NIST	BLEU	BLEU	NIST
Baselines									
PBSMT	23.88	24.04	6.29	32.31	31.70	7.48	18.71	22.21	5.28
PBSMT + MSD	24.07	24.04	6.28	32.45	31.85	7.47	18.99	21.18	5.30
Moses-chart	23.94	24.93	6.39	30.58	31.80	7.41	19.93	23.90	5.41
Oracle results (copied from Table 2)									
Oracle tree	24.70	24.80	6.32	32.76	32.21	7.51	20.23	23.44	5.35
Oracle string	26.28	27.02	6.50	34.09	33.52	7.60	23.01	26.08	5.52
Oracle mod + 2lt	25.05	25.04	6.36	32.24	32.18	7.51	20.64	23.75	5.37
Oracle mod + 3lt	25.11	25.27	6.37	32.22	32.34	7.52	20.71	23.59	5.37
Oracle mod + 5lt	25.07	25.23	6.37	32.51	32.37	7.55	20.93	23.93	5.39
Reordering systems									
MERrd	24.64	**24.72**	6.33	31.97	32.19	7.52	19.82	**23.17**	5.33
MERrd + 2lt	24.61	**24.99**	6.35	31.70	32.11	7.50	20.02	23.01	5.33
MERrd + 3lt	24.82	**24.98**	6.34	31.65	32.25	7.52	20.21	**23.14**	5.34
MERrd + 5lt	24.78	**25.12**	6.37	31.99	**32.38**	7.52	20.29	**23.17**	5.35

beside child permutation, includes a deletion operation with the maximum number of tree layers that can be deleted set to x. All reordered systems include an MSD model as a supporting reordering mechanism.

Translation results. Table 3 shows the results of automatic evaluation using BLEU and NIST (Doddington 2002) metrics. BLEU scores measured on the test data, which are statistically significant from the best PBSMT results, are marked with bold. The statistical significance calculations[11] have been done for a 95% confidence interval and 1000 re-samples, following the guidelines in Koehn (2004).

Discussion. Our results show that source-reordering is beneficial for language pairs with high mutual word order disparity. In contrast to English–Dutch and English–Chinese translation tasks, the statistical significance test reveals that all but the *MERrd+5lt* English–Spanish PBSMT system with rearranged input are not different from the translation quality delivered by Moses. This disappointing result for the English-to-Spanish translation task may be explained by the fact that many reordering differences are resolved by standard reordering models (distance-based and MSD).

A gap between the maximum reachable performance shown by tree transduction systems and the translation quality delivered by our model is 0.05–0.29 BLEU points for English–Dutch, 0.01–0.09 for English–Spanish, and 0.27–0.76 for

[11] Recent work (Dyer *et al.* 2011) argues that reporting averages and standard deviations over multiple MERT runs per system could lead to a better comparison between system performance.

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

English–Chinese. These numbers demonstrate that there are some potentially usable regularities not captured by our current conditional tree-transduction model.

9 Analysis of translation after source reordering

So far, our oracle and system experiments warrant two conclusions:

- *Oracle string* source-reordered systems offer a large improvement potential over standard *PBSMT*. *Oracle tree*-constrained systems offer only a small part of this potential.
- Our own tree-based source reordering PBSMT model variations do improve over standard *PBSMT*, albeit the improvement does not exploit the full potential improvement offered by tree-based source reordering shown by the oracles.

In this section we analyze the quality of the reordering mechanism, and concentrate on its impact on the robustness and adequacy of the translation model. We study three possible explanations for the gap between tree-based systems, *Oracle string*, and standard *PBSMT*:

H1 A source reordering approach may or may not lead to improved word order prior to translation. Our hypothesis here is that tree-based source reordering does improve source word order when measured against the output of *Oracle string* as gold standard, and that the ranking in improved source reordering is predictive of final translation quality.

H2 The back-end translation system employs a phrase-table extracted from the source-reordered strings to target sentences: the hypothesis is that the phrase-table *quality* improves due to (tree-based) source reordering. We need effective measures of quality that may explain differences of performance between various kinds of source reordering systems defined in this paper.

H3 Source reordering improves the lexical choice in the output of the full system.

We concentrate on the English-to-Dutch and English-to-Spanish translation tasks and the systems developed for them. All oracles as well as tree-based transduction systems work with a back-end phrase-based translation system enhanced with lexicalized and MSD reordering. Our study is structured in three parts corresponding to the input (H1), phrase table (H2), and output (H3) of the translation model applied after source reordering.

9.1 Input: assessing permutation quality before translation

To measure the quality of reordering performance in the output of the reordering module (input of the translation system), we use the LRscore metric (Birch and Osborne 2010), which scores the difference in word order between two datasets balanced by a lexical component[12]. The LRscore averages over all pairs of a

[12] Note that the input strings under comparison are lexically identical.

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

Table 4. *LRscore as a measure of reordering success for two language pairs*

Input of system	LRscore (system to Oracle string)	
	English–Dutch	English–Spanish
Oracle string	100.00 (gold standard)	100.00 (gold standard)
PBSMT	72.64 (no source reordering)	93.20 (no source reordering)
Oracle tree	77.03	94.10
Oracle mod + 2lt	77.15	94.16
Oracle mod + 3lt	79.84	94.23
Oracle mod + 5lt	81.14	94.25
MERrd	72.97	93.93
MERrd + 2lt	73.83	94.07
MERrd + 3lt	74.12	94.09
MERrd + 5lt	75.98	94.09

candidate translation t and its reference translation r in a test set $\mathscr{T}$ of length N (number of segments):

$$\text{LRscore} = \frac{1}{N} \times \sum_{\langle r,t\rangle \in \mathscr{T}} \alpha * R(r,t) + (1-\alpha) BLEU(r,t) \tag{5}$$

where $R(r,t) = d(r,t) * BP(r,t)$ is the reordering metric dependent on the Hamming distance $d(r,t)$ and the brevity penalty $BP(r,t)$ defined as

$$BP(r,t) = \begin{cases} 1 & \text{if } |\text{t}| > |\text{r}| \\ e^{1-\frac{|r|}{|t|}} & \text{if } |\text{t}| \leq |\text{r}| \end{cases}$$

with $|x|$ denoting the length of every string x in words.

As a word order gold standard we used the *Oracle string* output, which constitutes the source permutation that contains the smallest number of crossing alignments with the target sentence.

The results of reordering quality evaluation for two language pairs can be found in Table 4. For both language pairs, the source without pre-processing (input to *PBSMT*) has the lowest LRscore, whereas *Oracle mod + 5lt* has the highest possible LRscore (closest to *Oracle string*). Clearly, the potential space for gain for source reordering over PBSMT for English–Spanish (with 93.20% for *PBSMT*) is about one-fourth of that for English–Dutch (with 72.64% for *PBSMT*). The difference between *Oracle mod +5lt* and *PBSMT* is about 10 points for English–Dutch, with the *Oracle mod +5lt* closing approximately *only one-third* of the gap between the input of *PBSMT* and that of *Oracle string*. This difference is far smaller for English–Spanish, with *Oracle mod +5lt* closing *less than 10%* of the gap between the input of *PBSMT* and that of *Oracle string*.

Regarding hypothesis H1, we conclude that tree-based source reordering does have the potential to improve source order but cannot bridge the lion's share with the gold standard *Oracle string*. Hence, limited source reordering is one reason that *Oracle tree* systems offer limited performance gain.

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

Table 5. *Size of phrase tables for English-to-Dutch system (millions of instances)*

System	Phrase table size	
	English–Dutch	English–Spanish
PBSMT	29.80	56.89
Oracle string	48.43	62.02
Oracle tree	33.00	58.12
Oracle mod + 2lt	33.21	58.11
Oracle mod + 3lt	34.40	58.12
Oracle mod + 5lt	34.15	58.20
MERrd	29.91	58.19
MERrd + 2lt	30.02	58.15
MERrd + 3lt	30.07	58.20
MERrd + 5lt	29.82	58.22

9.2 Phrase table quality measures

Now we study the impact of source reordering on the phrase table (hypothesis H2). We concentrate on two indicators: phrase table size and phrase table coverage on held out sets. The first measures increase in diversity of phrase pairs, whereas the second measures whether the increase is useful in covering unseen input of the same kind.

Phrase table size and coverage. Table 5 contrasts the number of phrases extracted from the corpus with reordered and monotone source parts. Clearly, all source-reordered systems have larger phrase tables, with increase in size approximately predicting improved translation performance (see Table 3). The phrase pair extraction method underlying phrase-based translation has the property that more monotone the word alignment in the training parallel data is, more phrase pairs can be extracted. Yet, beside monotonization, phrase tables size may increase due to inconsistent alignments or reorderings (noise). Hence, next we directly measure the coverage of the phrase table on held-out data. We observe the percentage of phrases from held-out datasets that are covered by the primary[13] phrase table. Our assumption was that the source reordering will create more monotone alignments in training data allowing more and longer phrases, and hence possibly better coverage of phrases in unseen data.

For all systems, phrase coverage was measured by averaging over the same ten-fold, independent random partitioning of the training corpus into a held-out and an extraction set as follows. For each $i \in [1..10]$ the following three steps were conducted: (1) A 1K-line held-out partition Φ_i is randomly extracted and the remaining part is called Φ_i^C; (2) a pair of multi-sets of bilingual phrases is extracted, one (called H_i) from the held-out partition Φ_i and another (E_i) from Φ_i^C; and (3) two coverage scores were calculated: the coverage of phrase-types (called *Cov. uniq*) is calculated as the ratio $\frac{|H_i^s \cap E_i^s|}{|E_i^s|}$, where H^s and E^s are the sets underlying the

[13] Primary phrase table is extracted from the whole training corpus minus the held-out set.

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

Table 6. *Coverage of phrase types (Cov. uniq.) and tokens (weighted by frequency – Cov.) averaged over ten-fold held-out sets*

	English to Dutch		English to Spanish	
System	Cov. uniq. Phrase types (%)	Cov. Phrase tokens (%)	Cov. uniq. Phrase types (%)	Cov. Phrase tokens (%)
PBSMT	43.72	58.24	63.74	69.21
Oracle string	64.08	72.62	73.10	81.20
Oracle tree	40.65	50.87	60.97	64.02
Oracle mod + 2lt	44.19	55.21	62.02	64.91
Oracle mod + 3lt	48.23	55.86	67.12	68.70
Oracle mod + 5lt	47.14	59.12	63.16	63.44
MERrd	41.23	52.10	61.81	61.01
MERrd + 2lt	44.02	53.02	63.79	63.01
MERrd + 3lt	47.75	50.84	60.04	64.98
MERrd + 5lt	47.21	51.73	64.12	65.17

corresponding multi-sets, and the frequency-weighted coverage of phrases (called *Cov.*) given by $\frac{|H_i^s \cap E_i|}{|E_i|}$.[14] Table 6 reports both coverage scores averaged over ten partitions for English–Dutch and English–Spanish.

Clearly, *Oracle string* improves the coverage of the phrase table (types and frequency-weighted). The disappointing coverage (both phrase type and frequency-weighted) of *Oracle tree* for both language pairs is illuminating: this system acquires many more phrase types (Table 5) but has lower coverage than *PBSMT*. The *Oracle mod* systems have improved *phrase-type* coverage (Cov. uniq) even relative to *PBSMT*, but cannot bridge more than a small fraction of difference between *Oracle string* and *PBSMT*. It is interesting to note that the *Cov.* (frequency-weighted coverage) is lower than the *PBSMT* coverage almost for all tree-based systems. This suggests that phrase-type coverage is improved mainly for phrase-types expected to occur infrequently but might have decreased for phrase-types expected to occur frequently, with the likely situation that the added phrases are noise.

We conclude here that tree-based systems lead to larger phrase tables that hardly have any improved coverage which supports the hypothesis that these systems' phrase-based translation model suffers somewhat from the tree-based source reordering. For the phrase-based model, tree-based source reordering creates more noisy alignments in the training data. This is, however, counter balanced by resolving long-distance reorderings, which explains the final translation improvement over *PBSMT*.

9.3 Output: effect of source reordering on final lexical choice

We now focus on the output of translation systems. We would like to know how lexical choice varies as the source reordering varies. It is conceivable that changing

[14] Here the intersection of a multi-set E_i with a set H_i^s produces a multi-set containing all instances in the intersection, each weighted with its frequency in E_i. The size of a multi-set is given by the number of tokens it contains.

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

Table 7. *Lexical precision/recall and F-measure analysis of MT output*

	English–Dutch			English–Spanish		
System	Precision	Recall	F-measure	Precision	Recall	F-measure
PBSMT	57.92	58.07	57.99	64.70	65.25	64.97
Oracle string	58.73	58.65	58.69	65.48	66.07	65.77
Oracle tree	58.16	58.11	58.13	65.02	65.56	65.29
Oracle mod + 2lt	58.27	58.17	58.22	65.08	65.56	65.32
Oracle mod + 3lt	58.54	58.23	58.38	65.10	65.59	65.34
Oracle mod + 5lt	58.59	58.62	58.60	65.12	65.61	65.36
MERrd	58.05	52.18	58.11	64.78	65.37	65.07
MERrd + 2lt	58.12	58.21	58.16	64.84	65.39	65.11
MERrd + 3lt	58.14	58.12	58.13	64.84	65.38	65.11
MERrd + 5lt	58.22	58.23	58.22	65.86	65.39	65.62

the source order lead to different phrase pairs and hence offer the target language model other options to choose from. While the lexical choice is certainly a part of BLEU scores, here we look at other complementary measures that we can interpret by neutralizing the word order differences.

Instead of BLEU1 (bag of words BLEU), we calculate the precision, recall, and F1 of the output quality in terms of lexical choice. We consider both the output sentences of a given system as well as the corresponding reference translations as bags of word tokens and consider the intersection between these two bags of word tokens for defining

$$Precision = \frac{C_{ov}}{C_{transl}} \qquad Recall = \frac{C_{ov}}{C_{ref}} \tag{6}$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{7}$$

where C_{ov} is the total number of word tokens across the test corpus that are in the intersection of the MT system output and the human reference translations (as bags of words), C_{transl} is the total number of word tokens in the translation system output, and C_{ref} is the total number of words in the reference translations.

The results can be found in Table 7. The results of all systems are largely comparable, supporting the conclusion that source reordering only marginally impacts the system's lexical choice, i.e., its impact concentrates on improved reordering of the output.

9.4 Discussion

In the background of our analysis we had the following conclusion from Section 6: Source reordering has large potential for performance improvement if source permutation is unlimited, but when limited by the simple tree-based reordering only a fraction of the potential remains. The empirical analysis so far sheds light on why this is the case. Following is the summary:

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

- Tree-based source reordering manages to arrive at a closer word order to the gold standard *Oracle string* (virtually target order) than the original source sentence. However, measured using the LRscore metric (Birch and Osborne 2010), the best tree-based reordering only covers one-third of the LRscore gap between original sentence and oracle string reordered sentence. This is one explanation for the limited potential of the tree-based oracles relative to the *Oracle string*.
- Source-reordering leads to larger phrase tables. For *Oracle string* this increase leads to substantial increase in coverage of unseen phrase pairs in held-out input (about 25% for English–Dutch and 12% for English–Spanish). For the tree-based oracle, the phrase tables are larger but there is hardly any coverage improvement over the PBSMT table, suggesting that the size increase is mainly due to somewhat arbitrary source reordering (looking at that from the string, phrase-based level).
- Source reordering hardly impacts lexical selection in the translation system output.

Together these findings refine the common wisdom that tree-based source reordering constrains the space of source permutations in effective and thus harmless ways. Indeed, tree-based source reordering can deal with long-range reorderings that the baseline PBSMT system cannot deal with. But because tree-based reordering follows the constituency structure and adheres to the limitations of reordering the sister branches, it could conduct selective/noisy source reordering at the string level that sometimes may cancel the benefits of source reordering. The latter does not necessarily imply that tree-based reordering is problematic per se, but that its combination with a PBSMT back-end system is fraught with certain difficulties.

The lack of improvement in lexical choice in translation output may look expected. We think that this hints at the fact that in source reordering (as a pre-processing component to a phrase-based MT model) the lexical choice is rigidly conditioned on the choice of source reordering. This is because a single permutation is input to the translation model. It might be better to input a distribution of possible source permutations to the translation model.

Along the way we also suggested that the LRscore metric (Birch and Osborne 2010), when combined with phrase table coverage, is a reasonably good predictor of the system's reordering capabilities and its final translation performance (in BLEU). Figures 7(a) and (b) show plots of the BLEU score against LRscore and expected phrase Coverage. A reasonable interpretation of these plots is that the relative BLEU score change (increase/decrease) can be reasonably predicted from a suitable combination of LRscore and Coverage. This combination of performance predictors could save system development time because it avoids system optimization time (Minimum Error-Rate Training) and actual translation. How to conduct such a combination most effectively is left as a topic for future research.

Finally, it is important to stress the particular limitations of this study when interpreting the empirical findings. Our experiments are based on automatically generated word alignments and parse trees, which are far from ideal. Our oracle

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

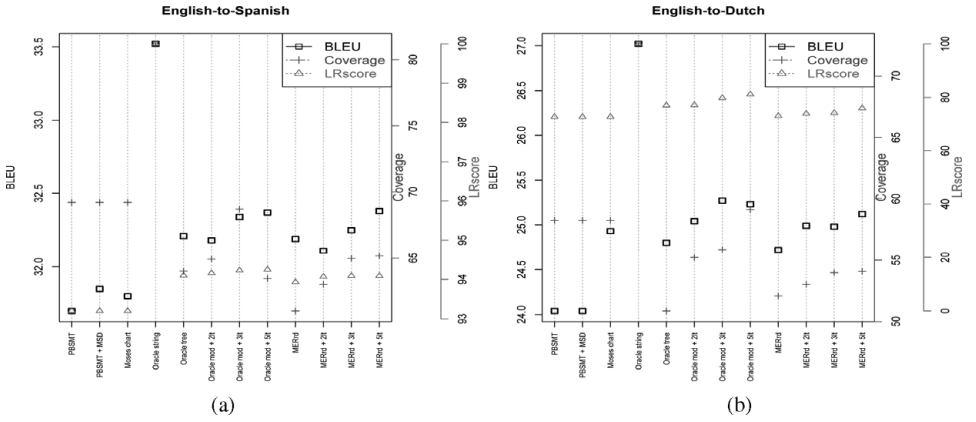


Fig. 7. Plots of final BLEU scores against the corresponding LRscores and expected phrase coverage. (a) English–Spanish systems; (b) English–Dutch systems.

results suggest that the quality of word alignments is sufficient to allow for performance improvement over the baseline PBSMT system. The quality of parser output and the potential mismatch between monolingual constituency boundaries and phrase pair boundaries could be a more serious limitation, however. It can be interesting to compare the results shown by translation and reordering systems based on automatically generated alignments with the ones using manual word alignments in the same spirit as it is done in DeNero and Uszkoreit (2011).

The greedy tree-based (oracle) systems reflect a dominant section of current practice in studying source reordering. We suspect that the limitations of these greedy tree-based source reordering systems will also hold for more optimally trained tree-based systems, i.e., those that work with global search for reordering sequences over the source parse. Our findings should be seen to highlight the need for a more robust syntax-sensitive source reordering approach than an approach based on local transduction operations over a constituency tree output by a source language parser.

10 Conclusions and future work

In this paper we studied source reordering with/without syntax as a pre-processing step to a translation model. We started out with gaging the upperbound on performance an certain idealized source reordering with/without tree-based constraints. Source reordering has large potential to improve translation output, albeit this potential is limited when the reordering follows a single tree structure.

We presented a novel model of tree-based source reordering and showed that it does improve translation performance significantly, but we also found that this improvement is only a small part of the potential improvement that oracle source reordering offers. We set out to analyze the reasons for the limited improvement (or lack of it) by looking at the translation model that follows source reordering from three angles: its input, its phrase table, and its output. Our main conclusion is that the impact of source reordering on the phrase table can be large: increased table size

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

may imply increased phrase coverage in unseen data, but this is not always the case for tree-based source reordering. Essentially we find that, as well as improvement in source reordering, the phrase table size, and coverage of phrases in heldout data can be predictive of system output improvement. These conclusions are dependent on the quality of parser output and automatic word alignments.

Our future work will aim at balancing the wish for efficient phrase-based translation and effective tree-based source reordering. We might abandon the pre-processing architecture toward an architecture where the translation model input is a distribution over source reorderings given the original input. This will demand major changes to the total architecture of our models leading to learning latent structure for source reordering inspired by, for example, Dyer and Resnik (2010), DeNero and Uszkoreit (2011), and Mylonakis and Sima'an (2011). We also intend to study tree-based source-reordering with hierarchical phrase-based translation at the back-end. In both cases it seems worthwhile to consider source syntax for labeling a representation dedicated to the hierarchical translation equivalence underlying a reordering mechanism, thereby putting less attention on monolingual constituency structure.

Acknowledgments

We would like to thank Gideon Maillette de Buy Wenniger and Markos Mylonakis for assistance with data preparation. We also thank three JNLE reviewers who provided us with insightful comments on a previous version. Both authors are supported by VIDI grant (nr. 639.022.604) from The Netherlands Organization for Scientific Research (NWO). We also wish to thank Andrey Ziyatdinov (Bioengineering, Department of Automatic Control (ESAII), Barcelona Tech, Spain) for his valuable comments on R code.

References

Birch, A., and Osborne, M. 2010. LRscore for evaluating lexical and reordering quality in MT. In *Proceedings of the Joint Workshop on Statistical Machine Translation and MetricsMATR*, Uppsala, Sweden, July 15–16, pp. 327–32. Stroudsburg, PA: Association for Computational Linguistics.

Brown, P., Della Pietra, V., Della Pietra, S., and Mercer, R. 1993. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics* **19**(2): 263–311.

Chiang, D. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, Ann Arbor, MI, USA, pp. 263–70. Stroudsburg, PA: Association for Computational Linguistics.

Chiang, D. 2007. Hierarchical phrase-based translation. *Computational Linguistics* **2**(33): 201–28.

Collins, M., Koehn, P., and Kučerová, I. 2005. Clause restructuring for statistical machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, Ann Arbor, MI, USA, pp. 531–40. Stroudsburg, PA: Association for Computational Linguistics.

Costa-jussà, M. R., and Fonollosa, J. A. R. 2006. Statistical machine reordering. In *Proceedings of the Joint Conference on Human Language Technology and the Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP)*, New York, NY, USA, pp. 70–6.

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

DeNeefe, S., Knight, K., Wang, W., and Marcu, D. 2007. What can syntax-based MT learn from phrase-based MT? In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, Prague, Czech Republic, pp. 755–63.

DeNero, J., and Uszkoreit, J. 2011. Inducing sentence structure from parallel corpora for reordering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 193–203. Edinburgh, Scotland, UK: Association for Computational Linguistics.

Doddington, G. 2002. Automatic evaluation of machine translation quality using n-grams co-occurrence statistics. In *Proceedings of the Conference on Human Language Technology (HLT)*, San Diego, CA, USA, pp. 128–32.

Dyer, C., Clark, J. H., Lavie, A., and Smith, N. A. 2011. Unsupervised word alignment with arbitrary features. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, Portland, OR, pp. 409–19.

Dyer, C., and Resnik, P. 2010. Context-free reordering, finite-state translation. In *Proceedings of Human Language Technology and North American Chapter of the ACL (HLT-NAACL)*, Los Angeles, CA, USA, pp. 858–66.

Galley, M., Hopkins, M., Knight, K., and Marcu, D. 2004. What's in a translation rule? In *Proceedings of Human Language Technology and North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*, Boston, MA, USA, pp. 273–80.

Genzel, D. 2010. Automatically learning source-side reordering rules for large-scale machine translation. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, Beijing, China, pp. 376–84.

Huang, L., Zhang, H., Gildea, D., and Knight, K. 2009. Binarization of synchronous context-free grammars. *Computational Linguistics* **35**(4): 559–95.

Isozaki, H., Sudoh, K., Tsukada, H., and Duh, K. 2010. Head finalization: a simple reordering rule for Sov languages. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, Uppsala, Sweden, pp. 244–51. Stroudsburg, PA: Association for Computational Linguistics.

Katz-Brown, J., Petrov, S., McDonald, R. T., Och, F. J., Talbot, D., Ichikawa, H., Seno, M., and Kazawa, H. 2011. Training a parser for machine translation reordering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), a Meeting of SIGDAT, a Special Interest Group of the ACL*, Edinburgh, Scotland, UK, pp. 183–92.

Khalilov, M. 2009. *New statistical and Syntactic Models for Machine Translation*. Ph.D. thesis, Universitat Politècnica de Catalunya, Barcelona, Spain.

Khalilov, M., and Sima'an, K. 2010. A discriminative syntactic model for source permutation via tree transduction. In *Proceedings of the Fourth Workshop on Syntax and Structure in Statistical Translation (SSST-4) at the International Conference on Computational Linguistics (COLING)*, Beijing, China, pp. 92–100.

Khalilov, M., and Sima'an, K. 2011. Context-sensitive syntactic source-reordering by statistical transduction. In *Proceedings of the International Joint Conference on Natural Language Processing (IJCNLP)*, Chiang Mai, Thailand, pp. 38–46.

Klein, D., and Manning, C. 2003. Accurate unlexicalized parsing. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, Sapporo, Japan, pp. 423–30.

Koehn, P. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the Conference on Empirical Methods on Natural Language Processing (EMNLP)*, Barcelona, Spain, pp. 388–95.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. 2007. Moses: open-source toolkit for statistical machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, Prague, Czech Republic, pp. 177–80.

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

Koehn, P., Och, F., and Marcu, D. 2003. Statistical phrase-based machine translation. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (HLT-NAACL)*, Edmonton, Canada, pp. 48–54.

Li, L. 1998. A comparison of word order in English and Chinese. *Poznań Studies in Contemporary Linguistics (Formerly: Papers and Studies in Contrastive Linguistics)* **34**: 153–61.

Li, C., Minghui, L., Zhang, D., Li, M., Zhou, M., and Guan, Y. 2007. A probabilistic approach to syntax-based reordering for statistical machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, Prague, Czech Republic, pp. 720–7.

Marcus, M. P., Santorini, B., and Marcinkiewicz, M. A. 1993. Building a large annotated corpus of English: the Penn treebank. *Computational Linguistics* **19**(2): 313–30.

Mylonakis, M., and Sima'an, K. 2011. Learning hierarchical translation structure with linguistic annotations. *Proceedings of the Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, Portland, OR, USA.

Och, F. 1999. An efficient method for determining bilingual word classes. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, Maryland, USA, pp. 71–6.

Och, F. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, Sapporo, Japan, pp. 160–7.

Och, F., and Ney, H. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, Philadelphia, PA, USA, pp. 295–302.

Och, F., and Ney, H. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics* **29**(1): 19–51.

Och, F., and Ney, H. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics* **30**(4): 417–49.

Papineni, K., Roukos, S., Ward, T., and Zhu, W. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, Philadelphia, PA, USA, pp. 311–18.

Popovic', M., and Ney, H. 2006. POS-based word reorderings for statistical machine translation. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*, Genoa, Italy, pp. 1278–83.

PVS, A. 2010. A data mining approach to learn reorder rules for SMT. In *Proceedings of Human Language Technologies: the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL/HLT)*, Los Angeles, CA, USA, pp. 52–7.

Ramanathan, A., Bhattacharyya, P., Hegde, J., Shah, R., and Sasikumar, M. 2008. Simple syntactic and morphological processing can help English–Hindi statistical machine translation. In *Proceedings of the International Joint Conference on Natural Language Processing (IJCNLP)*, Hyderabad, India, pp. 513–20.

Stolcke, A. 2002. SRILM: an extensible language modeling toolkit. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP)*, Denver, CO, USA, pp. 901–4.

Tillman, C. 2004. A unigram orientation model for statistical machine translation. In *Proceedings of Human Language Technologies: the Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*, Boston, MA, USA, pp. 101–4.

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press

Tromble, R., and Eisner, J. 2009. Learning linear ordering problems for better translation. In *Proceedings of the Conference on Empirical Methods on Natural Language Processing (EMNLP)*, Singapore, pp. 1007–16.

Visweswariah, K., Navratil, J., Sorensen, J., Chenthamarakshan, V., and Kambhatla, N. 2010. Syntax-based reordering with automatically derived rules for improved statistical machine translation. In *Proceeding of the International Conference on Computational Linguistics (COLING)*, Beijing, China, pp. 1119–27.

Visweswariah, K., Rajkumar, R., Gandhe, A., Ramanathan, A., and Navratil, J. 2011. A word reordering model for improved machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Edinburgh, Scotland, UK, pp. 486–96.

Wang, C., Collins, M., and Koehn, P. 2007. Chinese syntactic reordering for statistical machine translation. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing (EMNLP) and Conference on Computational Natural Language Learning (CoNLL)*, Prague, Czech Republic, pp. 737–45.

Wang, W., May, J., Knight, K., and Marcu, D. June 2010. Re-structuring, re-labeling, and re-aligning for syntax-based machine translation. *Computational Linguistics* **36**: 247–77.

Wu, D. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics* **3**(23): 377–403.

Wu, D., and Wong, H. 1998. Machine translation with a stochastic grammatical channel. In *Proceedings of the Joint Conference of the Annual Meeting of the Association for Computational Linguistics (ACL) and the International Conference on Computational Linguistics (COLING)*, Columbus, OH, USA, pp. 1408–15.

Xia, F., and McCord, M. 2004. Improving a statistical MT system with automatically learned rewrite patterns. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, Geneva, Switzerland, pp. 508–14.

Zens, R., and Ney, H. 2003. A comparative study on reordering constraints in statistical machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, Sapporo, Japan, pp. 144–51.

Zens, R., Och, F., and Ney, H. 2002. Phrase-based statistical machine translation. In *Proceedings of KI: advances in Artificial Intelligence*, pp. 18–32.

Zollmann, A., and Venugopal, A. 2006. Syntax-augmented machine translation via chart parsing. In *Proceedings of the North American Association for Computational Linguistics Conference (NAACL)*, pp. 138–41.

Zwarts, S., and Dras, M. 2007. Syntax-based word reordering in phrase-based statistical machine translation: why does it work? *Proceedings of the MT Summit XI*, Copenhagen, Denmark.

https://doi.org/10.1017/S1351324912000162 Published online by Cambridge University Press