Novel Reordering Approaches in Phrase-Based Statistical Machine Translation

Stephan Kanthak, David Vilar, Evgeny Matusov, Richard Zens, and Hermann Ney
The authors are with the Lehrstuhl für Informatik VI,
Computer Science Department, RWTH Aachen University,
D-52056 Aachen, Germany.
E-mail: {kanthak,vilar,matusov,zens,ney}@informatik.rwth-aachen.de.

Abstract

This paper presents novel approaches to reordering in phrase-based statistical machine translation. We perform consistent reordering of source sentences in training and estimate a statistical translation model. Using this model, we follow a phrase-based monotonic machine translation approach, for which we develop an efficient and flexible reordering framework that allows to easily introduce different reordering constraints. In translation, we apply source sentence reordering on word level and use a reordering automaton as input. We show how to compute reordering automata on-demand using IBM or ITG constraints, and also introduce two new types of reordering constraints. We further add weights to the reordering automata. We present detailed experimental results and show that reordering significantly improves translation quality.

1 Introduction

Reordering is of crucial importance for machine translation. Already (Knight et al., 1998) use full unweighted permutations on the level of source words in their early weighted finite-state transducer approach which implemented single-word based translation using conditional probabilities. In a refinement with additional phrase-based models, (Kumar et al., 2003) define a probability distribution over all possible permutations of source sentence phrases and prune the resulting automaton to reduce complexity.

A second category of finite-state translation approaches uses joint instead of conditional probabilities. Many joint probability approaches originate in speech-to-speech translation as they are the natural choice in combination with speech recognition models. The automated transducer inference techniques OMEGA (Vilar, 2000) and GIATI (Casacuberta et al., 2004) work on phrase level, but ignore the reordering problem from the view of the model. Without reordering both in training and during search, sentences can only be translated properly into a language with similar word order. In (Bangalore et al., 2000) weighted reordering has been applied to target sentences since defining a permutation model on the source side is impractical in combination with speech recognition. In order to reduce the computational complexity, this approach considers only a set of plausible reorderings seen on training data.

Most other phrase-based statistical approaches like the Alignment Template system of Bender et al. (2004) rely on (local) reorderings which are implicitly memorized with each pair of source and target phrases in training. Additional reorderings on phrase level are fully integrated into the decoding process, which increases the complexity of the system and makes it hard to modify. Zens et al. (2003) reviewed two types of reordering constraints for this type of translation systems.

In our work we follow a phrase-based translation approach, applying source sentence reordering on word level. We compute a reordering graph on-demand and take it as input for monotonic translation. This approach is modular and allows easy introduction of different reordering constraints and probabilistic dependencies. We will show that it performs at least as well as the best statistical machine translation system at the IWSLT Evaluation.

Proceedings of the ACL Workshop on Building and Using Parallel Texts, pages 167–174,
Ann Arbor, June 2005. ©Association for Computational Linguistics, 2005

In the next section we briefly review the basic theory of our translation system based on weighted finite-state transducers (WFST). In Sec. 3 we introduce new methods for reordering and alignment monotonization in training. To compare different reordering constraints used in the translation search process we develop an on-demand computable framework for permutation models in Sec. 4. In the same section we also define and analyze unrestricted and restricted permutations with some of them being first published in this paper. We conclude the paper by presenting and discussing a rich set of experimental results.

2 Machine Translation using WFSTs

Let f_1^J and e_i^I be two sentences from a source and target language. Assume that we have word level alignments $\mathcal{A}$ of all sentence pairs from a bilingual training corpus. We denote with $\tilde{e}_1^J$ the segmentation of a target sentence e_1^I into J phrases such that f_1^J and $\tilde{e}_1^J$ can be aligned to form bilingual tuples $(f_j, \tilde{e}_j)$. If alignments are only *functions of target words* $A' : \{1, \ldots, I\} \rightarrow \{1, \ldots, J\}$, the bilingual tuples $(f_j, \tilde{e}_j)$ can be inferred with e. g. the GIATI method of (Casacuberta et al., 2004), or with our novel monotonization technique (see Sec. 3). Each source word will be mapped to a target phrase of one or more words or an "empty" phrase ε. In particular, the source words which will remain non-aligned due to the alignment functionality restriction are paired with the empty phrase.

We can then formulate the problem of finding the best translation $\hat{e}_1^I$ of a source sentence f_1^J:

$$\begin{aligned}
\hat{e}_1^I &= \operatorname*{argmax}_{e_1^I} Pr(f_1^J, e_1^I) \\
&= \operatorname*{argmax}_{\tilde{e}_1^J} \sum_{A \in \mathcal{A}} Pr(f_1^J, \tilde{e}_1^J, A) \\
&\cong \operatorname*{argmax}_{\tilde{e}_1^J} \max_{A \in \mathcal{A}} Pr(A) \cdot Pr(f_1^J, \tilde{e}_1^J | A) \\
&\cong \operatorname*{argmax}_{\tilde{e}_1^J} \max_{A \in \mathcal{A}} \prod_{f_j : j=1 \ldots J} Pr(f_j, \tilde{e}_j | f_1^{j-1}, \tilde{e}_1^{j-1}, A) \\
&= \operatorname*{argmax}_{\tilde{e}_1^J} \max_{A \in \mathcal{A}} \prod_{f_j : j=1 \ldots J} p(f_j, \tilde{e}_j | f_{j-m}^{j-1}, \tilde{e}_{j-m}^{j-1}, A)
\end{aligned}$$

In other words: if we assume a uniform distribution for $Pr(A)$, the translation problem can be mapped to the problem of estimating an m-gram language model over a learned set of bilingual tuples $(f_j, \tilde{e}_j)$. Mapping the bilingual language model to a WFST T is canonical and it has been shown in (Kanthak et al., 2004) that the search problem can then be rewritten using finite-state terminology:

$$\hat{e}_1^I = \texttt{project-output}(\texttt{best}(f_1^J \circ T)) \, .$$

This implementation of the problem as WFSTs may be used to efficiently solve the search problem in machine translation.

3 Reordering in Training

When the alignment function A' is not monotonic, target language phrases $\tilde{e}$ can become very long. For example in a completely non-monotonic alignment all target words are paired with the last aligned source word, whereas all other source words form tuples with the empty phrase. Therefore, for language pairs with big differences in word order, probability estimates may be poor.

This problem can be solved by reordering either source or target training sentences such that alignments become monotonic for all sentences. We suggest the following consistent source sentence reordering and alignment monotonization approach in which we compute optimal, minimum-cost alignments.

First, we estimate a cost matrix C for each sentence pair (f_1^J, e_1^I). The elements of this matrix c_{ij} are the local costs of aligning a source word f_j to a target word e_i. Following (Matusov et al., 2004), we compute these local costs by interpolating state occupation probabilities from the source-to-target and target-to-source training of the HMM and IBM-4 models as trained by the GIZA++ toolkit (Och et al., 2003). For a given alignment $A \subseteq I \times J$, we define the costs of this alignment $c(A)$ as the sum of the local costs of all aligned word pairs:

$$c(A) \;=\; \sum_{(i,j) \in A} c_{ij} \qquad (1)$$

The goal is to find an alignment with the minimum costs which fulfills certain constraints.

3.1 Source Sentence Reordering

To reorder a source sentence, we require the alignment to be a *function* of *source* words A_1: $\{1, \ldots, J\} \rightarrow \{1, \ldots, I\}$, easily computed from the cost matrix C as:

$$A_1(j) = \operatorname{argmin}_i c_{ij} \qquad (2)$$

We do not allow for non-aligned source words. A_1 naturally defines a new order of the source words f_1^J which we denote by $\breve{f}_1^J$. By computing this permutation for each pair of sentences in training and applying it to each source sentence, we create a corpus of reordered sentences.

3.2 Alignment Monotonization

In order to create a "sentence" of bilingual tuples $(\breve{f}_1^J, \tilde{e}_1^J)$ we required alignments between reordered source and target words to be a *function* of *target* words $A_2 : \{1, \ldots, I\} \rightarrow \{1, \ldots, J\}$. This alignment can be computed in analogy to Eq. 2 as:

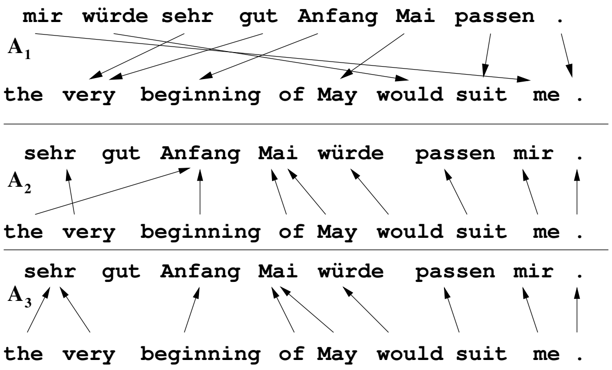
$$A_2(i) = \text{argmin}_j \, \breve{c}_{ij} \qquad (3)$$

where $\breve{c}_{ij}$ are the elements of the new cost matrix $\breve{C}$ which corresponds to the reordered source sentence. We can optionally re-estimate this matrix by repeating EM training of state occupation probabilities with GIZA++ using the reordered source corpus and the original target corpus. Alternatively, we can get the cost matrix $\breve{C}$ by reordering the columns of the cost matrix C according to the permutation given by alignment A_1.

In alignment A_2 some target words that were previously unaligned in A_1 (like "the" in Fig. 1) may now still violate the alignment monotonicity. The monotonicity of this alignment can not be guaranteed for *all* words if re-estimation of the cost matrices had been performed using GIZA++.

The general GIATI technique (Casacuberta et al., 2004) is applicable and can be used to monotonize the alignment A_2. However, in our experiments the following method performs better. We make use of the cost matrix representation and compute a monotonic minimum-cost alignment with a dynamic programming algorithm similar to the Levenshtein string edit distance algorithm. As costs of each "edit" operation we consider the local alignment costs. The resulting alignment A_3 represents a minimum-cost monotonic "path" through the cost matrix. To make A_3 a function of target words we do not consider the source words non-aligned in A_2 and also forbid "deletions" ("many-to-one" source word alignments) in the DP search.

An example of such consistent reordering and monotonization is given in Fig. 1. Here, we reorder the German source sentence based on the initial alignment A_1, then compute the function of target words A_2, and monotonize this alignment to A_3 with the dynamic programming algorithm. Fig. 1 also shows the resulting bilingual tuples $(\breve{f}_j, \tilde{e}_j)$.


mir würde sehr gut Anfang Mai passen .
$\mathbf{A_1}$
the very beginning of May would suit me .
sehr gut Anfang Mai würde passen mir .
$\mathbf{A_2}$
the very beginning of May would suit me .
sehr gut Anfang Mai würde passen mir .
$\mathbf{A_3}$
the very beginning of May would suit me .

```
sehr|the_very gut|$ Anfang|beginning
Mai|of_May würde|would passen|suit mir|me  .|.
```

Figure 1: Example of alignment, source sentence reordering, monotonization, and construction of bilingual tuples.

4 Reordering in Search

When searching the best translation $\tilde{e}_1^J$ for a given source sentence f_1^J, we permute the source sentence as described in (Knight et al., 1998):

$$\hat{e}_1^I = \texttt{project-output}(\texttt{best}(\texttt{permute}(f_1^J) \circ T))$$

Permuting an input sequence of J symbols results in $J!$ possible permutations and representing the permutations as a finite-state automaton requires at least 2^J states. Therefore, we opt for computing the permutation automaton on-demand while applying beam pruning in the search.

4.1 Lazy Permutation Automata

For on-demand computation of an automaton in the flavor described in (Kanthak et al., 2004) it is sufficient to specify a state description and an algorithm that calculates all outgoing arcs of a state from the state description. In our case, each state represents a permutation of a subset of the source words f_1^J, which are already translated.

This can be described by a bit vector b_1^J (Zens et al., 2002). Each bit of the state bit vector corresponds to an arc of the linear input automaton and is set to one if the arc has been used on any path from the initial to the current state. The bit vectors of two states connected by an arc differ only in a single bit. Note that bit vectors elegantly solve the problem of recombining paths in the automaton as states with the same bit vectors can be merged. As a result, a fully minimized permutation automaton has only a single initial and final state.

Even with on-demand computation, complexity using full permutations is unmanagable for long sentences. We further reduce complexity by additionally constraining permutations. Refer to Figure 2 for visualizations of the permutation constraints which we describe in the following.

4.2 IBM Constraints

The IBM reordering constraints are well-known in the field of machine translation and were first described in (Berger et al., 1996). The idea behind these constraints is to deviate from monotonic translation by postponing translations of a limited number of words. More specifically, at each state we can translate any of the *first* l yet uncovered word positions. The implementation using a bit vector is straightforward. For consistency, we associate window size with the parameter l for all constraints presented here.

4.3 Inverse IBM Constraints

The original IBM constraints are useful for a large number of language pairs where the ability to skip some words reflects the differences in word order between the two languages. For some other pairs, it is beneficial to translate some words at the end of the sentence first and to translate the rest of the sentence nearly monotonically. Following this idea we can define the *inverse IBM constraints*. Let j be the first uncovered position. We can choose any position for translation, unless $l-1$ words on positions $j' > j$ have been translated. If this is the case we must translate the word in position j. The inverse IBM constraints can also be expressed by

$$\texttt{invIBM}(x) = \texttt{transpose}(\texttt{IBM}(\texttt{transpose}(x)))\,.$$

As the `transpose` operation can not be computed on-demand, our specialized implementation uses bit vectors b_1^J similar to the IBM constraints.

4.4 Local Constraints

For some language pairs, e.g. Italian – English, words are moved only a few words to the left or right. The IBM constraints provide too many alternative permutations to chose from as each word can be moved to the end of the sentence. A solution that allows only for local permutations and therefore has very low complexity is given by the following permutation rule: the next word for translation comes from the window of l positions[1] counting from the first yet uncovered position. Note, that the local constraints define a true subset of the permutations defined by the IBM constraints.

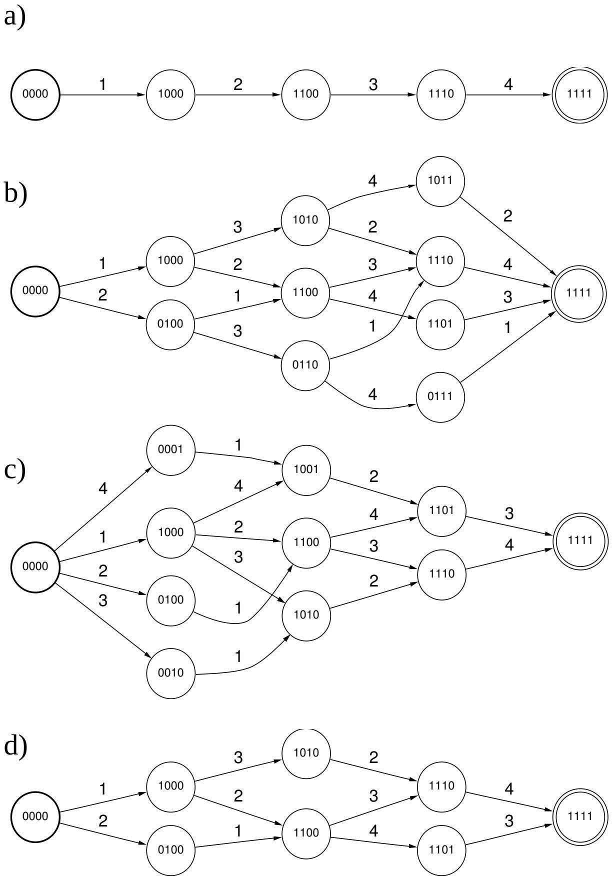


Figure 2: Permutations of a) positions $j = 1, 2, 3, 4$ of a source sentence $f_1 f_2 f_3 f_4$ using a window size of 2 for b) IBM constraints, c) inverse IBM constraints and d) local constraints.

4.5 ITG Constraints

Another type of reordering can be obtained using Inversion Transduction Grammars (ITG) (Wu, 1997). These constraints are inspired by bilingual bracketing. They proved to be quite useful for machine translation, e.g. see (Bender et al., 2004). Here, we interpret the input sentence as a sequence of segments. In the beginning, each word is a segment of its own. Longer segments are constructed by recursively combining two adjacent segments. At each

[1] both covered and uncovered

		Chinese	English	Japanese	English	Italian	English
train	sentences	20 000		20 000		66107	
	words	182 904	160 523	209 012	160 427	410 275	427 402
	singletons	3 525	2 948	4 108	2 956	6 386	3 974
	vocabulary	7 643	6 982	9 277	6 932	15 983	10 971
dev	sentences	506		506		500	
	words	3 515	3 595	4 374	3 595	3 155	3 253
	sentence length (avg/max)	6.95 / 24	7.01 / 29	8.64 / 30	7.01 / 29	5.79 / 24	6.51 / 25
test	sentences	500		500		506	
	words	3 794	–	4 370	–	2 931	3 595
	sentence length (avg/max)	7.59 / 62	7.16 / 71	8.74 / 75	7.16 / 71	6.31 / 27	6.84 / 28

Table 1: Statistics of the Basic Travel Expression (BTEC) corpora.

combination step, we either keep the two segments in monotonic order or invert the order. This process continues until only one segment for the whole sentence remains. The on-demand computation is implemented in spirit of Earley parsing.

We can modify the original ITG constraints to further limit the number of reorderings by forbidding segment inversions which violate IBM constraints with a certain window size. Thus, the resulting reordering graph contains the intersection of the reorderings with IBM and the original ITG constraints.

4.6 Weighted Permutations

So far, we have discussed how to generate the permutation graphs under different constraints, but permutations were equally probable. Especially for the case of nearly monotonic translation it is make sense to restrict the degree of non-monotonicity that we allow when translating a sentence. We propose a simple approach which gives a higher probability to the monotone transitions and penalizes the non-monotonic ones.

A state description b_1^J, for which the following condition holds:

$$Mon(j) :\ b_{j'} = \delta(j' \leq j)\ \forall\, 1 \leq j' \leq J$$

represents the monotonic path up to the word f_j. At each state we assign the probability α to that outgoing arc where the target state description fullfills $Mon(j+1)$ and distribute the remaining probability mass $1 - \alpha$ uniformly among the remaining arcs. In case there is no such arc, all outgoing arcs get the same uniform probability. This weighting scheme clearly depends on the state description and the outgoing arcs only and can be computed on-demand.

5 Experimental Results

5.1 Corpus Statistics

The translation experiments were carried out on the *Basic Travel Expression Corpus* (BTEC), a multilingual speech corpus which contains tourism-related sentences usually found in travel phrase books. We tested our system on the so called Chinese-to-English (CE) and Japanese-to-English (JE) Supplied Tasks, the corpora which were provided during the International Workshop on Spoken Language Translation (IWSLT 2004) (Akiba et al., 2004). In addition, we performed experiments on the Italian-to-English (IE) task, for which a larger corpus was kindly provided to us by ITC/IRST. The corpus statistics for the three BTEC corpora are given in Tab. 1. The development corpus for the Italian-to-English translation had only one reference translation of each Italian sentence. A set of 506 source sentences and 16 reference translations is used as a development corpus for Chinese-to-English and Japanese-to-English and as a test corpus for Italian-to-English tasks. The 500 sentence Chinese and Japanese test sets of the IWSLT 2004 evaluation campaign were translated and automatically scored against 16 reference translations after the end of the campaign using the IWSLT evaluation server.

5.2 Evaluation Criteria

For the automatic evaluation, we used the criteria from the IWSLT evaluation campaign (Akiba et al., 2004), namely word error rate (WER), position-independent word error rate (PER), and the BLEU and NIST scores (Papineni et al., 2002; Doddington, 2002). The two scores measure accuracy, i. e. larger scores are better. The error rates and scores were computed with respect to *m*ultiple reference transla-

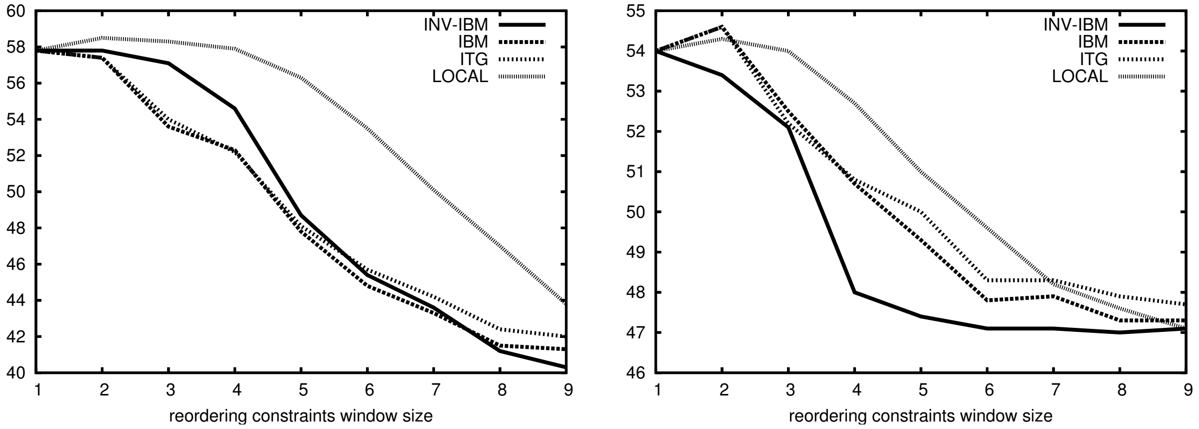


Figure 3: Word error rate [%] as a function of the reordering window size for different reordering constraints: Japanese-to-English (left) and Chinese-to-English (right) translation.

tions, when they were available. To indicate this, we will label the error rate acronyms with an m. Both training and evaluation were performed using corpora and references in lowercase and without punctuation marks.

5.3 Experiments

We used reordering and alignment monotonization in training as described in Sec. 3. To estimate the matrices of local alignment costs for the sentence pairs in the training corpus we used the state occupation probabilities of GIZA++ IBM-4 model training and interpolated the probabilities of source-to-target and target-to-source training directions. After that we estimated a smoothed 4-gram language model on the level of bilingual tuples $f_j, \tilde{e}_j$ and represented it as a finite-state transducer.

When translating, we applied moderate beam pruning to the search automaton only when using reordering constraints with window sizes larger than 3. For very large window sizes we also varied the pruning thresholds depending on the length of the input sentence. Pruning allowed for fast translations and reasonable memory consumption without a significant negative impact on performance.

In our first experiments, we tested the four reordering constraints with various window sizes. We aimed at improving the translation results on the development corpora and compared the results with two baselines: reordering only the source training sentences and translation of the unreordered test sentences; and the GIATI technique for creating bilingual tuples $(f_j, \tilde{e}_j)$ without reordering of the source sentences, neither in training nor during translation.

5.3.1 Highly Non-Monotonic Translation (JE)

Fig. 3 (left) shows word error rate on the Japanese-to-English task as a function of the window size for different reordering constraints. For each of the constraints, good results are achieved using a window size of 9 and larger. This can be attributed to the Japanese word order which is very different from English and often follows a subject-object-verb structure. For small window sizes, ITG or IBM constraints are better suited for this task, for larger window sizes, inverse IBM constraints perform best. The local constraints perform worst and require very large window sizes to capture the main word order differences between Japanese and English. However, their computational complexity is low; for instance, a system with local constraints and window size of 9 is as fast (25 words per second) as the same system with IBM constraints and window size of 5. Using window sizes larger than 10 is computationally expensive and does not significantly improve the translation quality under any of the constraints.

Tab. 2 presents the overall improvements in translation quality when using the best setting: inverse IBM constraints, window size 9. The baseline without reordering in training and testing failed completely for this task, producing empty translations for 37 % of the sentences[2]. Most of the original alignments in training were non-monotonic which resulted in mapping of almost all Japanese words to ε when using only the GIATI monotonization technique. Thus, the proposed reordering methods are of crucial importance for this task.

[2] Hence a NIST score of 0 due to the brevity penalty.

Reordering:	mWER [%]	mPER [%]	BLEU [%]	NIST
BTEC Japanese-to-English (JE) dev				
none	59.7	58.8	13.0	0.00
in training	57.8	39.4	14.7	3.27
+ 9-inv-ibm	40.3	32.1	45.1	8.59
+ rescoring*	39.1	30.9	53.2	9.93
BTEC Chinese-to-English (CE) dev				
none	55.2	52.1	24.9	1.34
in training	54.0	42.3	23.0	4.18
+ 7-inv-ibm	47.1	39.4	34.5	6.53
+ rescoring*	48.3	40.7	39.1	8.11

Table 2: Translation results with optimal reordering constraints and window sizes for the BTEC Japanese-to-English and Chinese-to-English development corpora. *Optimized for the NIST score.

	mWER [%]	mPER [%]	BLEU [%]	NIST
BTEC Japanese-to-English (JE) test				
AT	41.9	33.8	45.3	9.49
WFST	42.1	35.6	47.3	9.50
BTEC Chinese-to-English (CE) test				
AT	45.6	39.0	40.9	8.55
WFST	46.4	38.8	40.8	8.73

Table 3: Comparison of the IWSLT-2004 automatic evaluation results for the described system (WFST) with those of the best submitted system (AT).

Reordering:	mWER [%]	mPER [%]	BLEU [%]	NIST
none	25.6	22.0	62.1	10.46
in training	28.0	22.3	58.1	10.32
+ 4-local	26.3	20.3	62.2	10.81
+ weights	25.3	20.3	62.6	10.79
+ 3-ibm	27.2	20.5	61.4	10.76
+ weights	25.2	20.3	62.9	10.80
+ rescoring*	22.2	19.0	69.2	10.47

Table 4: Translation results with optimal reordering constraints and window sizes for the test corpus of the BTEC IE task. *Optimized for WER.

Further improvements were obtained with a rescoring procedure. For rescoring, we produced a k-best list of translation hypotheses and used the word penalty and deletion model features, the IBM Model 1 lexicon score, and target language n-gram models of the order up to 9. The scaling factors for all features were optimized on the development corpus for the NIST score, as described in (Bender et al., 2004).

5.3.2 Moderately Non-Mon. Translation (CE)

Word order in Chinese and English is usually similar. However, a few word reorderings over quite large distances may be necessary. This is especially true in case of questions, in which question words like “where” and “when” are placed at the end of a sentence in Chinese. The BTEC corpora contain many sentences with questions.

The inverse IBM constraints are designed to perform this type of reordering (see Sec. 4.3). As shown in Fig. 3, the system performs well under these constraints already with relatively small window sizes. Increasing the window size beyond 4 for these constraints only marginally improves the translation error measures for both short (under 8 words) and long sentences. Thus, a suitable language-pair-specific choice of reordering constraints can avoid the huge computational complexity required for permutations of long sentences.

Tab. 2 includes error measures for the best setup with inverse IBM constraints with window size of 7, as well as additional improvements obtained by a k-best list rescoring.

The best settings for reordering constraints and model scaling factors on the development corpora were then used to produce translations of the IWSLT Japanese and Chinese test corpora. These translations were evaluated against multiple references which were unknown to the authors. Our system (denoted with WFST, see Tab. 3) produced results competitive with the results of the best system at the evaluation campaign (denoted with AT (Bender et al., 2004)) and, according to some of the error measures, even outperformed this system.

5.3.3 Almost Monotonic Translation (IE)

The word order in the Italian language does not differ much from the English. Therefore, the absolute translation error rates are quite low and translating without reordering in training and search already results in a relatively good performance. This is reflected in Tab. 4. However, even for this language pair it is possible to improve translation quality by performing reordering both in training and during translation. The best performance on the development corpus is obtained when we constrain the reodering with relatively small window sizes of 3 to 4 and use either IBM or local reordering constraints.

On the test corpus, as shown in Tab. 4, all error measures can be improved with these settings.

Especially for languages with similar word order it is important to use *weighted* reorderings (Sec. 4.6) in order to prefer the original word order. Introduction of reordering weights for this task results in notable improvement of most error measures using either the IBM or local constraints. The optimal probability α for the unreordered path was determined on the development corpus as 0.5 for both of these constraints. The results on the test corpus using this setting are also given in Tab. 4.

6 Conclusion

In this paper, we described a reordering framework which performs source sentence reordering on word level. We suggested to use optimal alignment functions for monotonization and improvement of translation model training. This allowed us to translate monotonically taking a reordering graph as input. We then described known and novel reordering constraints and their efficient finite-state implementations in which the reordering graph is computed on-demand. We also utilized weighted permutations. We showed that our monotonic phrase-based translation approach effectively makes use of the reordering framework to produce quality translations even from languages with significantly different word order. On the Japanese-to-English and Chinese-to-English IWSLT tasks, our system performed at least as well as the best machine translation system.

Acknowledgement

This work was partially funded by the Deutsche Forschungsgemeinschaft (DFG) under the project "Statistische Textübersetzung" (Ne572/5) and by the European Union under the integrated project TC-STAR – Technology and Corpora for Speech to Speech Translation (IST-2002-FP6-506738).

References

Y. Akiba, M. Federico, N. Kando, H. Nakaiwa, M. Paul, and J. Tsujii. 2004. *Overview of the IWSLT04 Evaluation Campaign*. Proc. Int. Workshop on Spoken Language Translation, pp. 1–12, Kyoto, Japan.

S. Bangalore and G. Riccardi. 2000. *Stochastic Finite-State Models for Spoken Language Machine Translation*. Proc. Workshop on Embedded Machine Translation Systems, pp. 52–59.

O. Bender, R. Zens, E. Matusov, and H. Ney. 2004. *Alignment Templates: the RWTH SMT System*. Proc. Int. Workshop on Spoken Language Translation, pp. 79–84, Kyoto, Japan.

A. L. Berger, P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, J. R. Gillett, A. S. Kehler, and R. L. Mercer. 1996. *Language Translation Apparatus and Method of Using Context-based Translation Models*. United States Patent 5510981.

F. Casacuberta and E. Vidal. 2004. Machine Translation with Inferred Stochastic Finite-State Transducers. Computational Linguistics, vol. 30(2):205-225.

G. Doddington. 2002. *Automatic Evaluation of Machine Translation Quality Using n-gram Co-Occurrence Statistics*. Proc. Human Language Technology Conf., San Diego, CA.

S. Kanthak and H. Ney. 2004. *FSA: an Efficient and Flexible C++ Toolkit for Finite State Automata using On-demand Computation*. Proc. 42nd Annual Meeting of the Association for Computational Linguistics, pp. 510–517, Barcelona, Spain.

K. Knight and Y. Al-Onaizan. 1998. *Translation with Finite-State Devices*. Lecture Notes in Artificial Intelligence, Springer-Verlag, vol. 1529, pp. 421–437.

S. Kumar and W. Byrne. 2003. *A Weighted Finite State Transducer Implementation of the Alignment Template Model for Statistical Machine Translation*. Proc. Human Language Technology Conf. NAACL, pp. 142–149, Edmonton, Canada.

E. Matusov, R. Zens, and H. Ney. 2004. *Symmetric Word Alignments for Statistical Machine Translation*. Proc. 20th Int. Conf. on Computational Linguistics, pp. 219–225, Geneva, Switzerland.

F. J. Och and H. Ney. 2003. *A Systematic Comparison of Various Statistical Alignment Models*. Computational Linguistics, vol. 29, number 1, pp. 19–51.

K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. 2002. *BLEU: a Method for Automatic Evaluation of Machine Translation*. Proc. 40th Annual Meeting of the Association for Computational Linguistics, Philadelphia, PA, pp. 311–318.

J. M. Vilar, 2000. *Improve the Learning of Subsequential Transducers by Using Alignments and Dictionaries*. Lecture Notes in Artificial Intelligence, Springer-Verlag, vol. 1891, pp. 298–312.

D. Wu. 1997. *Stochastic Inversion Transduction Grammars and Bilingual Parsing of Parallel Corpora*. Computational Linguistics, 23(3):377–403.

R. Zens, F. J. Och and H. Ney. 2002. *Phrase-Based Statistical Machine Translation*. In: M. Jarke, J. Koehler, G. Lakemeyer (Eds.): KI - Conference on AI, KI 2002, Vol. LNAI 2479, pp. 18-32, Springer Verlag.

R. Zens and H. Ney. 2003. *A Comparative Study on Reordering Constraints in Statistical Machine Translation*. Proc. Annual Meeting of the Association for Computational Linguistics, pp. 144–151, Sapporo, Japan.