Word Reordering and a Dynamic Programming Beam Search Algorithm for Statistical Machine Translation

Christoph Tillmann*
IBM T. J. Watson Research Center

Hermann Ney†
RWTH Aachen

In this article, we describe an efficient beam search algorithm for statistical machine translation based on dynamic programming (DP). The search algorithm uses the translation model presented in Brown et al. (1993). Starting from a DP-based solution to the traveling-salesman problem, we present a novel technique to restrict the possible word reorderings between source and target language in order to achieve an efficient search algorithm. Word reordering restrictions especially useful for the translation direction German to English are presented. The restrictions are generalized, and a set of four parameters to control the word reordering is introduced, which then can easily be adopted to new translation directions. The beam search procedure has been successfully tested on the Verbmobil task (German to English, 8,000-word vocabulary) and on the Canadian Hansards task (French to English, 100,000-word vocabulary). For the medium-sized Verbmobil task, a sentence can be translated in a few seconds, only a small number of search errors occur, and there is no performance degradation as measured by the word error criterion used in this article.

1. Introduction

This article is about a search procedure for statistical machine translation (MT). The task of the search procedure is to find the most likely translation given a source sentence and a set of model parameters. Here, we will use a trigram language model and the translation model presented in Brown et al. (1993). Since the number of possible translations of a given source sentence is enormous, we must find the best output without actually generating the set of all possible translations; instead we would like to focus on the most likely translation hypotheses during the search process. For this purpose, we present a data-driven beam search algorithm similar to the one used in speech recognition search algorithms (Ney et al. 1992). The major difference between the search problem in speech recognition and statistical MT is that MT must take into account the different word order for the source and the target language, which does not enter into speech recognition. Tillmann, Vogel, Ney, and Zubiaga (1997) proposes a dynamic programming (DP)–based search algorithm for statistical MT that monotonically translates the input sentence from left to right. The word order difference is dealt with using a suitable preprocessing step. Although the resulting search procedure is very fast, the preprocessing is language specific and requires a lot of manual

* IBM T. J. Watson Research Center, Yorktown Heights, NY 10598. E-mail: ctill@us.ibm.com. The research reported here was carried out while the author was with Lehrstuhl für Informatik VI, Computer Science Department, RWTH Aachen.
† Lehrstuhl für Informatik VI, Computer Science Department, RWTH Aachen, D-52056 Aachen, Germany. E-mail: ney@informatik.rwth-aachen.de.

work. Currently, most search algorithms for statistical MT proposed in the literature are based on the A^* concept (Nilsson 1971). Here, the word reordering can be easily included in the search procedure, since the input sentence positions can be processed in any order. The work presented in Berger et al. (1996) that is based on the A^* concept, however, introduces word reordering restrictions in order to reduce the overall search space.

The search procedure presented in this article is based on a DP algorithm to solve the traveling-salesman problem (TSP). A data-driven beam search approach is presented on the basis of this DP-based algorithm. The cities in the TSP correspond to source positions of the input sentence. By imposing constraints on the possible word reorderings similar to that described in Berger et al. (1996), the DP-based approach becomes more effective: when the constraints are applied, the number of word reorderings is greatly reduced. The original reordering constraint in Berger et al. (1996) is shown to be a special case of a more general restriction scheme in which the word reordering constraints are expressed in terms of simple combinatorical restrictions on the processed sets of source sentence positions.[1] A set of four parameters is given to control the word reordering. Additionally, a set of four states is introduced to deal with grammatical reordering restrictions (e.g., for the translation direction German to English, the word order difference between the two languages is mainly due to the German verb group. In combination with the reordering restrictions, a data-driven beam search organization for the search procedure is proposed. A beam search pruning technique is conceived that jointly processes partial hypotheses according to *two* criteria: (1) The partial hypotheses cover the same set of source sentence positions, and (2) the partial hypotheses cover sets $\mathcal{C}$ of source sentence positions of equal cardinality. A partial hypothesis is said to **cover** a set of source sentence positions when exactly the positions in the set have already been processed in the search process. To verify the effectiveness of the proposed techniques, we report and analyze results for two translation tasks: the German to English Verbmobil task and French to English Canadian Hansards task.

The article is structured as follows. Section 2 gives a short introduction to the translation model used and reports on other approaches to the search problem in statistical MT. In Section 3, a DP-based search approach is presented, along with appropriate pruning techniques that yield an efficient beam search algorithm. Section 4 reports and analyzes translation results for the different translation directions. In Section 5, we conclude with a discussion of the achieved results.

2. Previous Work

2.1 IBM Translation Approach

In this article, we use the translation model presented in Brown et al. (1993), and the mathematical notation we use here is taken from that paper as well: a source string $f_1^J = f_1 \cdots f_j \cdots f_J$ is to be translated into a target string $e_1^I = e_1 \cdots e_i \cdots e_I$. Here, I is the length of the target string, and J is the length of the source string. Among all possible target strings, we will choose the string with the highest probability as given by Bayes'

1 The word reordering restriction used in the search procedure described in Berger et al. (1996) is not mentioned in Brown et al. (1993), although exactly the translation model described there is used. Equivalently, we use exactly the translation model described in Brown et al. (1993) but try different reordering restrictions for the DP-based search procedure.

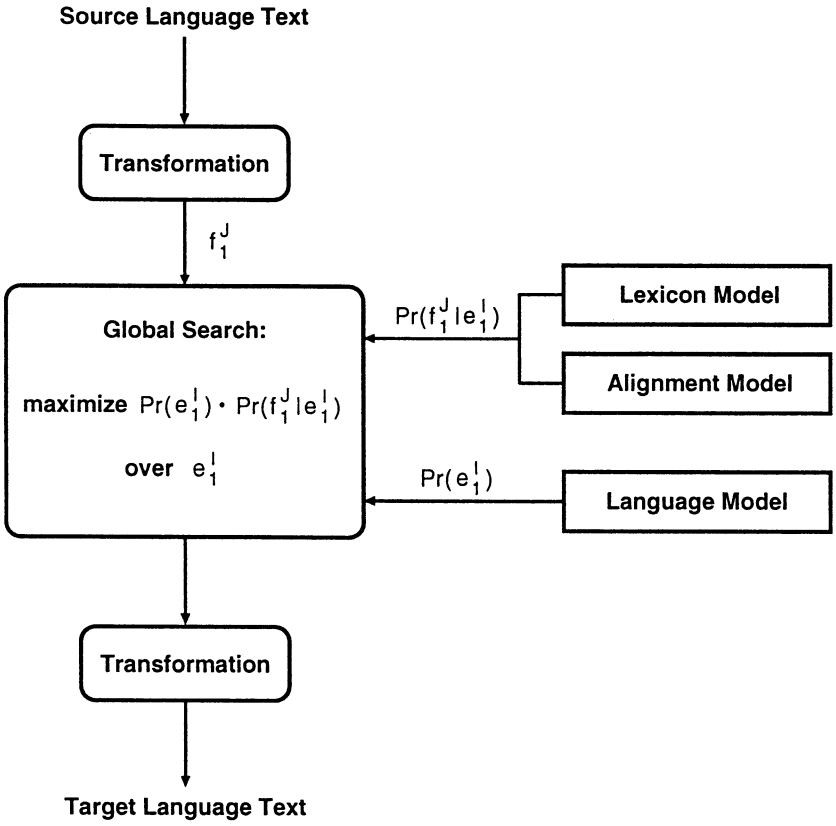


Figure 1
Architecture of the statistical translation approach based on Bayes' decision rule.

decision rule:

$$\begin{aligned} \hat{e}_1^I &= \arg\max_{e_1^I}\{Pr(e_1^I \mid f_1^J)\} \\ &= \arg\max_{e_1^I}\{Pr(e_1^I) \cdot Pr(f_1^J \mid e_1^I)\} \end{aligned} \quad (1)$$

$Pr(e_1^I)$ is the language model of the target language, whereas $Pr(f_1^J \mid e_1^I)$ is the string translation model. The language model probability is computed using a trigram language model. The string translation probability $Pr(f_1^J \mid e_1^I)$ is modeled using a series of five models of increasing complexity in training. Here, the model used for the translation experiments is the IBM-4 model. This model uses the same parameter set as the IBM-5 model, which in preliminary experiments did not yield better translation results. The actual implementation used during the experiments is described in Al-Onaizan et al. (1999) and in Och and Ney (2000). The argmax operation denotes the search problem (i.e., the generation of the output sentence in the target language). The overall architecture of the statistical translation approach is summarized in Figure 1. In general, as shown in this figure, there may be additional transformations to make the translation task simpler for the algorithm. The transformations may range from simple word categorization to more complex preprocessing steps that require some parsing of the source string. In this article, however, we will use only word catego-

rization as an explicit transformation step. In the search procedure both the language and the translation model are applied *after* the text transformation steps. The following "types" of parameters are used for the IBM-4 translation model:

Lexicon probabilities: We use the lexicon probability $p(f \mid e)$ for translating the single target word e as the single source word f. A source word f may be translated by the "null" word e_0 (i.e., it does not produce any target word e). A translation probability $p(f \mid e_0)$ is trained along with the regular translation probabilities.

Fertilities: A single target word e may be aligned to $n = 0, 1$ or more source words. This is explicitly modeled by the fertility parameter $\phi(n \mid e)$: the probability that the target word e is translated by n source words is $\phi(n \mid e)$. The fertility for the "null" word is treated specially (for details see Brown et al. [1993]). Berger et al. (1996) describes the extension of a partial hypothesis by a pair of target words (e', e), where e' is not connected to any source word f. In this case, the so-called spontaneous target word e' is accounted for with the fertility. Here, the translation probability $\phi(0 \mid e')$ and no-translation probability $p(f \mid e')$.

Class-based distortion probabilities: When covering a source sentence position j, we use distortion probabilities that depend on the previously covered source sentence positions (we say that a source sentence position j is covered for a partial hypothesis when it is taken account of in the translation process by generating a target word or the "null" word e_0). In Brown et al. (1993), two types of distortion probabilities are distinguished: (1) the leftmost word of a set of source words f aligned to the same target word e (which is called the "head") is placed, and (2) the remaining source words are placed. Two separate distributions are used for these two cases. For placing the "head" the center function $center(i)$ (Brown et al. [1993] uses the notation $\odot_i$) is used: the average position of the source words with which the target word e_{i-1} is aligned. The distortion probabilities are class-based: They depend on the word class $\mathcal{F}(f)$ of a covered source word f as well as on the word class $\mathcal{E}(e)$ of the previously generated target word e. The classes are automatically trained (Brown et al. 1992).

When the IBM-4 model parameters are used during search, an input sentence can be processed one source position at a time in a certain order primarily determined by the distortion probabilities. We will use the following simplified set of translation model parameters: lexicon probabilities $p(f \mid e)$ and distortion probabilities $p(j \mid j', J)$. Here, j is the currently covered input sentence position and j' is the previously covered input sentence position. The input sentence length J is included, since we would like to think of the distortion probability as normalized according to J. No fertility probabilities or "null" word probabilities are used; thus each source word f is translated as exactly one target word e and each target word e is translated as exactly one source word f. The simplified notation will help us to focus on the most relevant details of the DP-based search procedure. The simplified set of parameters leads to an unrealistic assumption about the length of the source and target sentence, namely, $I = J$. During the translation experiments we will, of course, not make this assumption. The implementation details for using the full set of IBM-4 model parameters are given in Section 3.9.2.

2.2 Search Algorithms for Statistical Machine Translation

In this section, we give a short overview of search procedures used in statistical MT: Brown et al. (1990) and Brown et al. (1993) describe a statistical MT system that is based on the same statistical principles as those used in most speech recognition systems (Jelinek 1976). Berger et al. (1994) describes the French-to-English Candide translation system, which uses the translation model proposed in Brown et al. (1993). A detailed description of the decoder used in that system is given in Berger et al. (1996) but has never been published in a paper: Throughout the search process, partial hypotheses are maintained in a set of priority queues. There is a single priority queue for each subset of covered positions in the source string. In practice, the priority queues are initialized only on demand; far fewer than the full number of queues possible are actually used. The priority queues are limited in size, and only the 1,000 hypotheses with the highest probability are maintained. Each priority queue is assigned a threshold to select the hypotheses that are going to be extended, and the process of assigning these thresholds is rather complicated. A restriction on the possible word reorderings, which is described in Section 3.6, is applied.

Wang and Waibel (1997) presents a search algorithm for the IBM-2 translation model based on the A^* concept and multiple stacks. An extension of this algorithm is demonstrated in Wang and Waibel (1998). Here, a *reshuffling* step on top of the original decoder is used to handle more complex translation models (e.g., the IBM-3 model is added). Translation approaches that use the IBM-2 model parameters but are based on DP are presented in García-Varea, Casacuberta, and Ney (1998) and Niessen et al. (1998). An approach based on the hidden Markov model alignments as used in speech recognition is presented in Tillmann, Vogel, Ney, and Zubiaga (1997) and Tillmann, Vogel, Ney, Zubiaga, and Sawaf (1997). This approach assumes that source and target language have the same word order, and word order differences are dealt with in a preprocessing stage. The work by Wu (1996) also uses the original IBM model parameters and obtains an efficient search algorithm by restricting the possible word reorderings using the so-called stochastic bracketing transduction grammar.

Three different decoders for the IBM-4 translation model are compared in Germann et al. (2001). The first is a reimplementation of the stack-based decoder described in Berger et al. (1996). The second is a greedy decoder that starts with an approximate solution and then iteratively improves this first rough solution. The third converts the decoding problem into an integer program (IP), and a standard software package for solving IP is used. Although the last approach is guaranteed to find the optimal solution, it is tested only for input sentences of length eight or shorter.

This article will present a DP-based beam search decoder for the IBM-4 translation model. The decoder is designed to carry out an almost full search with a small number of search errors and with little performance degradation as measured by the word error criterion. A preliminary version of the work presented here was published in Tillmann and Ney (2000).

3. Beam Search in Statistical Machine Translation

3.1 Inverted Alignment Concept

To explicitly describe the word order difference between source and target language, Brown et al. (1993) introduced an alignment concept, in which a source position j is mapped to exactly one target position i:

$$\text{regular alignment:} \quad j \rightarrow i = a_j$$

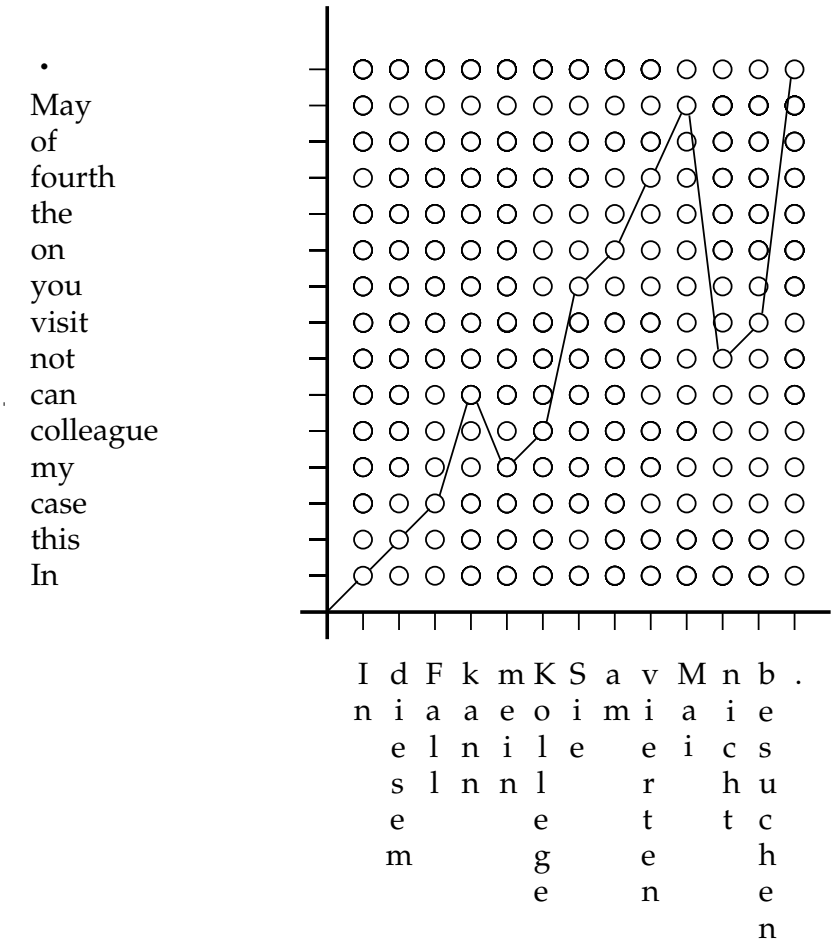


Figure 2
Regular alignment example for the translation direction German to English. For each German source word there is exactly one English target word on the alignment path.

An example for this kind of alignment is given in Figure 2, in which each German source position j is mapped to an English target position i. In Brown et al. (1993), this alignment concept is used for model IBM-1 through model IBM-5. For search purposes, we use the inverted alignment concept as introduced in Niessen et al. (1998) and Ney et al. (2000). An inverted alignment is defined as follows:

$$\text{inverted alignment:} \quad i \rightarrow j = b_i$$

Here, a target position i is mapped to a source position j. The *coverage* constraint for an inverted alignment is not expressed by the notation: Each source position j should be "hit" exactly once by the path of the inverted alignment $b_1^I = b_1 \cdots b_i \cdots b_I$. The advantage of the inverted alignment concept is that we can construct target sentence hypotheses from bottom to top along the positions of the target sentence. Using the inverted alignments in the maximum approximation, we rewrite equation (1) to obtain the following search criterion, in which we are looking for the most likely target

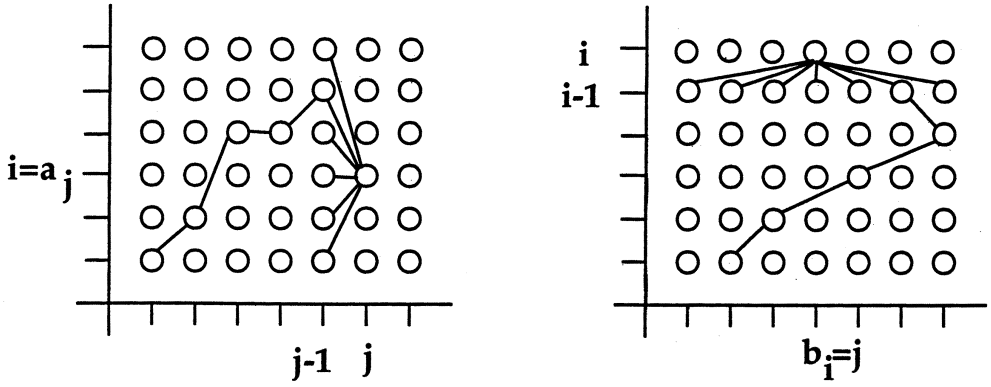


Figure 3
Illustration of the transitions in the regular and in the inverted alignment model. The regular alignment model (left figure) is used to generate the sentence from left to right; the inverted alignment model (right figure) is used to generate the sentence from bottom to top.

sentence e_1^I of length $I = J$ for an observed source sentence f_1^J of length J:

$$\max_I \left\{ p(J \mid I) \cdot \max_{e_1^I} \{ p(e_1^I) \cdot p(f_1^J \mid e_1^I) \} \right\} \tag{2}$$

$$\cong \max_I \left\{ p(J \mid I) \cdot \max_{e_1^I} \left\{ \prod_{i=1}^{I} p(e_i \mid e_{i-1}, e_{i-2}) \cdot \max_{b_1^I} \prod_{i=1}^{I} [p(b_i \mid b_{i-1}, J) \cdot p(f_{b_i} \mid e_i)] \right\} \right\}$$

$$= \max_I \left\{ p(J \mid I) \cdot \max_{e_1^I, b_1^I} \left\{ \prod_{i=1}^{I} [p(e_i \mid e_{i-1}, e_{i-2}) \cdot p(b_i \mid b_{i-1}, J) \cdot p(f_{b_i} \mid e_i)] \right\} \right\}$$

The following notation is used: e_{i-1}, e_{i-2} are the immediate predecessor target words, e_i is the word to be hypothesized, $p(e_i \mid e_{i-1}, e_{i-2})$ denotes the trigram language model probability, $p(f_{b_i} \mid e_i)$ denotes the lexicon probability for translating the target word e_i as source word f_{b_i}, and $p(b_i \mid b_{i-1}, J)$ is the distortion probability for covering source position b_i after source position b_{i-1}. Note that in equation (2) two products over i are merged into a single product over i. The translation probability $p(f_1^J \mid e_1^I)$ is computed in the maximum approximation using the distortion and the lexicon probabilities. Finally, $p(J \mid I)$ is the sentence length model, which will be dropped in the following (it is not used in the IBM-4 translation model). For each source sentence f_1^J to be translated, we are searching for the unknown mapping that optimizes equation (2):

$$i \rightarrow (b_i, e_i)$$

In Section 3.3, we will introduce an auxiliary quantity that can be evaluated recursively using DP to find this unknown mapping. We will explicitly take care of the coverage constraint by introducing a coverage set $\mathcal{C}$ of source sentence positions that have already been processed. Figure 3 illustrates the concept of the search algorithm using inverted alignments: Partial hypotheses are constructed from bottom to top along the positions of the target sentence. Partial hypotheses of length $i-1$ are extended to obtain partial hypotheses of the length i. Extending a partial hypothesis means covering a source sentence position j that has not yet been covered. For a given grid point in the

Table 1
DP-based algorithm for solving traveling-salesman problems due to Held and Karp. The outermost loop is over the cardinality of subsets of already visited cities.

input: cities $j = 1, \ldots, J$ with distance matrix $d_{jj'}$
initialization: $D(\{k\}, k) := d_{1k}$
for each path length $c = 2, \ldots, J$ do
  for each pair $(\mathcal{C}, j)$, where $\mathcal{C} \subseteq \{2, \ldots, J\}$ and $j \in \mathcal{C}$ and $|\mathcal{C}| = c$ do
    $D(\mathcal{C}, j) = \min_{j' \in \mathcal{C} \setminus \{j\}} \{d_{jj'} + D(\mathcal{C} \setminus \{j\}, j')\}$
traceback:
  - find shortest tour: $D^* = \min_{k \in \{2, \ldots, J\}} [D(\{2, \ldots, J\}, k) + d_{k1}]$
  - recover optimal sequence of cities

translation lattice, the unknown target word sequence can be obtained by tracing back the translation decisions to the partial hypothesis at stage $i = 1$. The grid points are defined in Section 3.3. In the left part of the figure the regular alignment concept is shown for comparison purposes.

3.2 Held and Karp Algorithm for Traveling-Salesman Problem

Held and Karp (1962) presents a DP approach to solve the TSP, an optimization problem that is defined as follows: Given are a set of cities $\{1, \ldots, J\}$ and for each pair of cities j, j' the cost $d_{jj'} > 0$ for traveling from city j to city j'. We are looking for the shortest tour, starting and ending in city 1, that visits all cities in the set of cities exactly once. We are using the notation $\mathcal{C}$ for the set of cities, since it corresponds to a coverage set of processed source positions in MT. A straightforward way to find the shortest tour is by trying all possible permutations of the J cities. The resulting algorithm has a complexity of $O(J!)$. DP can be used, however, to find the shortest tour in $O(J^2 \cdot 2^J)$, which is a much smaller complexity for larger values of J. The approach recursively evaluates the quantity $D(\mathcal{C}, j)$:

$$D(\mathcal{C}, j) := \text{costs of the partial tour starting in city 1, ending in city } j \text{, and visiting all cities in } \mathcal{C}$$

Subsets of cities $\mathcal{C}$ of increasing cardinality c are processed. The algorithm, shown in Table 1, works because not all permutations of cities have to be considered explicitly. During the computation, for a pair $(\mathcal{C}, j)$, the order in which the cities in $\mathcal{C}$ have been visited can be ignored (except j); only the costs for the best path reaching j has to be stored. For the initialization the costs for starting from city 1 are set: $D(\{k\}, k) = d_{1k}$ for each $k \in \{2, \ldots, |\mathcal{C}|\}$. Then, subsets $\mathcal{C}$ of increasing cardinality are processed. Finally, the cost for the optimal tour is obtained in the second-to-last line of the algorithm. The optimal tour itself can be found using a back-pointer array in which the optimal decision for each grid point $(\mathcal{C}, j)$ is stored.

Figure 4 illustrates the use of the algorithm by showing the "supergraph" that is searched in the Held and Karp algorithm for a TSP with $J = 5$ cities. When traversing the lattice from left to right following the different possibilities, a partial path to a node j corresponds to the subset $\mathcal{C}$ of all cities on that path together with the last visited

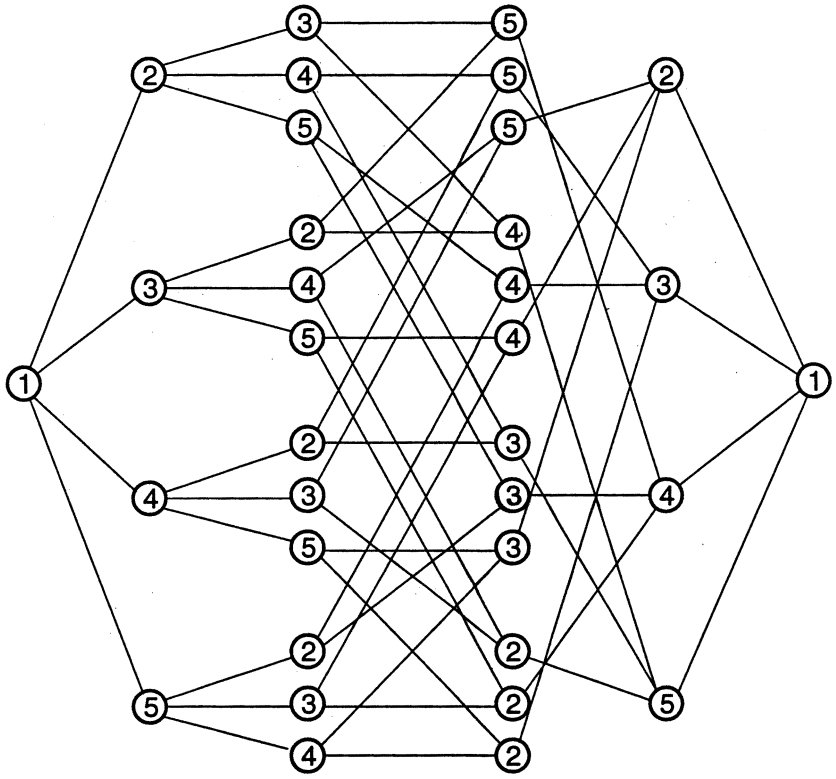


Figure 4
Illustration of the algorithm by Held and Karp for a traveling salesman problem with $J = 5$ cities. Not all permutations of cities have to be evaluated explicitly. For a given subset of cities the order in which the cities have been visited can be ignored.

city j. Of all the different paths merging into the node j, only the partial path with the smallest cost has to be retained for further computation.

3.3 DP-Based Algorithm for Statistical Machine Translation

In this section, the Held and Karp algorithm is applied to statistical MT. Using the concept of inverted alignments as introduced in Section 3.1, we explicitly take care of the coverage constraint by introducing a coverage set $\mathcal{C}$ of source sentence positions that have already been processed. Here, the correspondence is according to the fact that each source sentence position has to be covered exactly once, fulfilling the coverage constraint. The cities of the more complex translation TSP correspond roughly to triples (e', e, j), the notation for which is given below. The final path output by the translation algorithm will contain exactly one triple (e', e, j) for each source position j.

The algorithm processes subsets of partial hypotheses with coverage sets $\mathcal{C}$ of increasing cardinality c. For a trigram language model, the partial hypotheses are of the form $(e', e, \mathcal{C}, j)$, where e', e are the last two target words, $\mathcal{C}$ is a coverage set for the already covered source positions, and j is the last covered position. The target word sequence that ends in e', e is stored as a back pointer to the predecessor partial hypothesis (and recursively to its predecessor hypotheses) and is not shown in the notation. Each distance in the TSP now corresponds to the negative logarithm of the product of the translation, distortion, and language model probabilities. The following

Table 2
DP-based algorithm for statistical MT that consecutively processes subsets $\mathcal{C}$ of source sentence positions of increasing cardinality.

input: source language string $f_1 \cdots f_j \cdots f_J$
initialization
for each cardinality $c = 1, 2, \ldots, J$ do
  for each pair $(\mathcal{C}, j)$, where $\mathcal{C} \subseteq \{1, \ldots, J\}$ and $j \in \mathcal{C}$ and $|\mathcal{C}| = c$ do
    for each pair of target words $e', e \in E$

$$Q_{e'}(e, \mathcal{C}, j) = p(f_j \mid e) \max_{\substack{e'' \\ j' \in \mathcal{C} \setminus \{j\}}} \{p(j \mid j', J) \cdot p(e \mid e', e'') \cdot Q_{e''}(e', \mathcal{C} \setminus \{j\}, j')\}$$

traceback:
- find best end hypothesis: $\max_{e, e', j} \{p(\$ \mid e, e') \cdot Q_{e'}(e, \{1, \ldots, J\}, j)\}$
- recover optimal word sequence

auxiliary quantity is defined:

$$\begin{aligned} Q_{e'}(e, \mathcal{C}, j) \quad := \quad & \text{probability of the best partial hypothesis } (e_1^i, b_1^i), \text{ where} \\ & \mathcal{C} = \{b_k \mid k = 1, \ldots, i\}, b_i = j, e_i = e, \text{ and } e_{i-1} = e' \end{aligned}$$

The above auxiliary quantity satisfies the following recursive DP equation:

$$Q_{e'}(e, \mathcal{C}, j) = p(f_j \mid e) \cdot \max_{\substack{e'' \\ j' \in \mathcal{C} \setminus \{j\}}} \left\{ p(j \mid j', J) \cdot p(e \mid e', e'') \cdot Q_{e''}(e', \mathcal{C} \setminus \{j\}, j') \right\}$$

Here, j' is the previously covered source sentence position and e', e'' are the predecessor words. The DP equation is evaluated recursively for each hypothesis $(e', e, \mathcal{C}, j)$. The resulting algorithm is depicted in Table 2. Some details concerning the initialization and the finding of the best target language string are presented in Section 3.4. $p(\$ \mid e, e')$ is the trigram language probability for predicting the sentence boundary symbol \$. The complexity of the algorithm is $O(E^3 \cdot J^2 \cdot 2^J)$, where E is the size of the target language vocabulary.

3.4 Verb Group Reordering: German to English

The above search space is still too large to translate even a medium-length input sentence. On the other hand, only very restricted reorderings are necessary; for example, for the translation direction German to English, the word order difference is mostly restricted to the German verb group. The approach presented here assumes a mostly monotonic traversal of the source sentence positions from left to right.[2] A small number of positions may be processed sooner than they would be in that monotonic traversal. Each source position then generates a certain number of target words. The restrictions are fully formalized in Section 3.5.

A typical situation is shown in Figure 5. When translating the sentence monotonically from left to right, the translation of the German finite verb *kann*, which is the left verbal brace in this case, is skipped until the German noun phrase *mein Kollege*, which is the subject of the sentence, is translated. Then, the right verbal brace is translated:

2 Also, this assumption is necessary for the beam search pruning techniques to work efficiently.

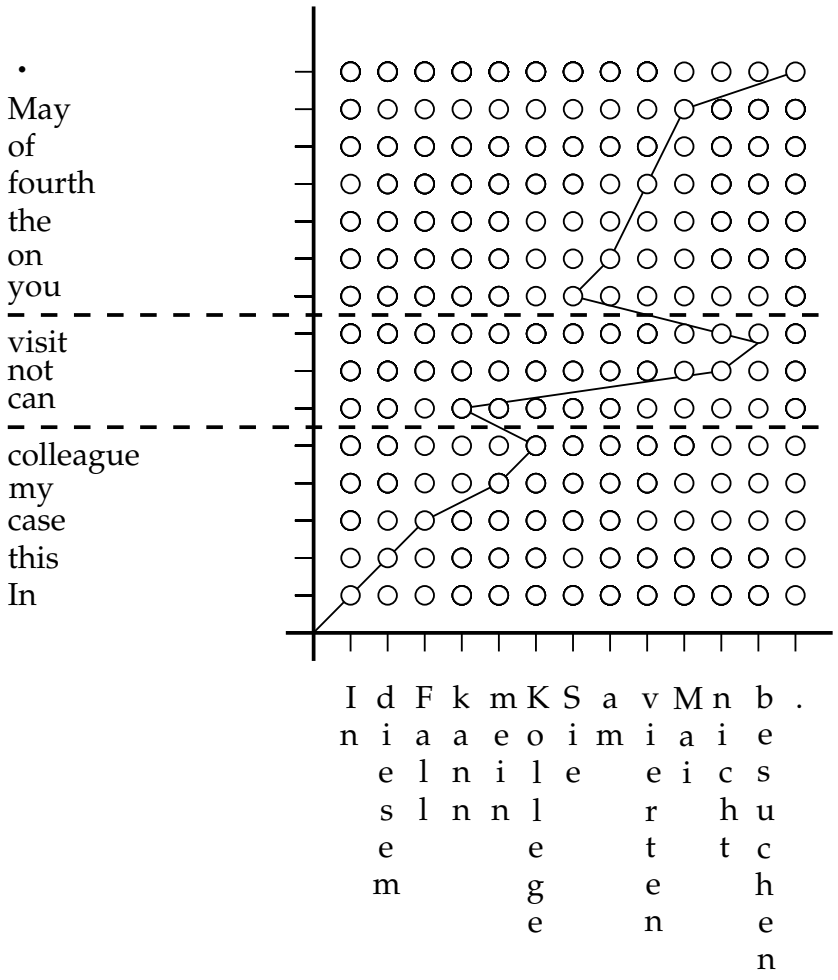


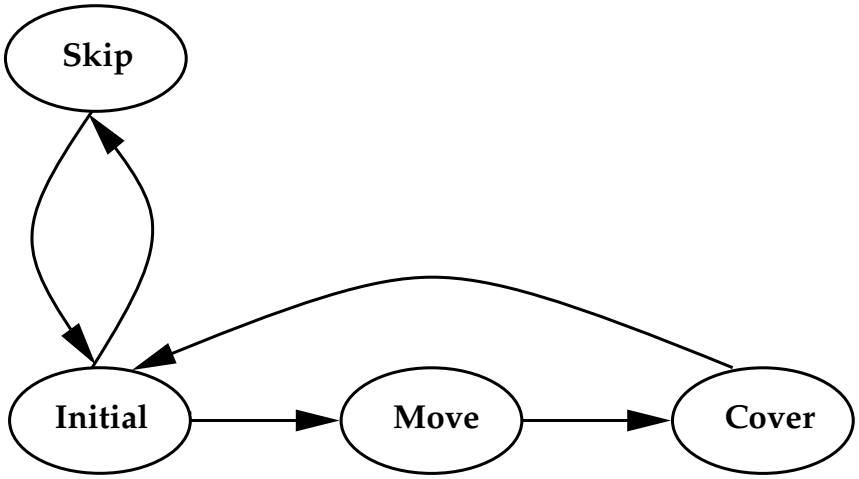
Figure 5
Word reordering for the translation direction German to English: The reordering is restricted to the German verb group.

The infinitive *besuchen* and the negation particle *nicht*. The following restrictions are used: One position in the source sentence may be skipped for a distance of up to $L = 4$ source positions, and up to two source positions may be moved for a distance of at most $R = 10$ source positions (the notation L and R shows the relation to the handling of the left and right verbal brace). To formalize the approach, we introduce four verb group states $\mathcal{S}$:

- *Initial*: A contiguous initial block of source positions is covered.
- *Skip*: One word may be skipped, leaving a "hole" in the monotonic traversal.
- *Move*: Up to two words may be "moved" from later in the sentence.
- *Cover*: The sentence is traversed monotonically until the state *Initial* is reached.

4. mein

5. Kollege



1. In	6. kann	7. nicht	9. Sie
2. diesem	12. Mai	8. besuchen	10. am
3. Fall	13. .		11. vierten

Figure 6
Order in which the German source positions are covered for the German-to-English reordering example given in Figure 5.

The states *Move* and *Skip* both allow a set of upcoming words to be processed sooner than would be the case in the monotonic traversal. The state *Initial* is entered whenever there are no uncovered positions to the left of the rightmost covered position. The sequence of states needed to carry out the word reordering example in Figure 5 is given in Figure 6. The 13 source sentence words are processed in the order shown. A formal specification of the state transitions is given in Section 3.5. Any number of consecutive German verb phrases in a sentence can be processed by the algorithm. The finite-state control presented here is obtained from a simple analysis of the German-to-English word reordering problem and is not estimated from the training data. It can be viewed as an extension of the IBM-4 model distortion probabilities.

Using the above states, we define partial hypothesis extensions of the following type:

$$(\mathcal{S}', \mathcal{C} \backslash \{j\}, j') \rightarrow (\mathcal{S}, \mathcal{C}, j)$$

Not only the coverage set $\mathcal{C}$ and the positions j, j', but also the verb group states $\mathcal{S}, \mathcal{S}'$, are taken into account. For the sake of brevity, we have omitted the target language words e, e' in the notation of the partial hypothesis extension. For each extension an uncovered position is added to the coverage set $\mathcal{C}$ of the partial hypothesis, and the verb group state $\mathcal{S}$ may change. A more detailed description of the partial hypothesis extension for a certain state $\mathcal{S}$ is given in the next section in a more general context. Covering the first uncovered position in the source sentence, we use the lan-

guage model probability $p(e \mid \$, \$)$. Here, \$ is the sentence boundary symbol, which is thought to be at position 0 in the target sentence. The search starts in the hypothesis $(\mathit{Initial}, \{\emptyset\}, 0)$. $\{\emptyset\}$ denotes the empty set, where no source sentence position is covered. The following recursive equation is evaluated:

$$Q_{e'}(e, \mathcal{S}, \mathcal{C}, j) \tag{3}$$
$$= p(f_j \mid e) \max_{\substack{e'', \mathcal{S}', j' \\ (\mathcal{S}', \mathcal{C}\setminus\{j\}, j') \rightarrow (\mathcal{S}, \mathcal{C}, j) \\ j' \in \mathcal{C}\setminus\{j\}}} \{p(j \mid j', J) \cdot p(e \mid e', e'') \cdot Q_{e''}(e', \mathcal{S}', \mathcal{C}\setminus\{j\}, j')\}$$

The search ends in the hypotheses $(\mathit{Initial}, \{1, \ldots, J\}, j)$; the last covered position may be in the range $j \in \{J-L, \ldots, J\}$, because some source positions may have been skipped at the end of the input sentence. $\{1, \ldots, J\}$ denotes a coverage set including all positions from position 1 to position J. The final translation probability Q_{F} is

$$Q_{\mathrm{F}} = \max_{\substack{e, e' \\ j \in \{J-L, \ldots J\}}} p(\$ \mid e, e') \cdot Q_{e'}(e, \mathit{Initial}, \{1, \ldots, J\}, j) \tag{4}$$

where $p(\$ \mid e, e')$ denotes the trigram language model, which predicts the sentence boundary \$ at the end of the target sentence. Q_{F} can be obtained using an algorithm very similar to the one given in Table 2. The complexity of the verb group reordering for the translation direction German to English is $\mathcal{O}(E^3 \cdot J \cdot (R^2 \cdot L \cdot R))$, as shown in Tillmann (2001).

3.5 Word Reordering: Generalization

For the translation direction English to German, the word reordering can be restricted in a similar way as for the translation direction German to English. Again, the word order difference between the two languages is mainly due to the German verb group. During the translation process, the English verb group is decomposed as shown in Figure 7. When the sentence is translated monotonically from left to right, the translation of the English finite verb *can* is moved, and it is translated as the German left verbal brace before the English noun phrase *my colleague*, which is the subject of the sentence. The translations of the infinitive *visit* and of the negation particle *not* are skipped until later in the translation process. For this translation direction, the translation of one source sentence position may be moved for a distance of up to $L = 4$ source positions, and the translation of up to two source positions may be skipped for a distance of up to $R = 10$ source positions (we take over the L and R notation from the previous section). Thus, the role of the skipping and the moving are simply reversed with respect to their roles in German-to-English translation. For the example translation in Figure 7, the order in which the source sentence positions are covered is given in Figure 8.

We generalize the two approaches for the different translation directions as follows: In both approaches, we assume that the source sentence is mainly processed monotonically. A small number of upcoming source sentence positions may be processed earlier than they would be in the monotonic traversal: The states *Skip* and *Move* are used as explained in the preceding section. The positions to be processed outside the monotonic traversal are restricted as follows:

- The number of positions dealt with in the states *Move* and *Skip* is restricted.
- There are distance restrictions on the source positions processed in those states.

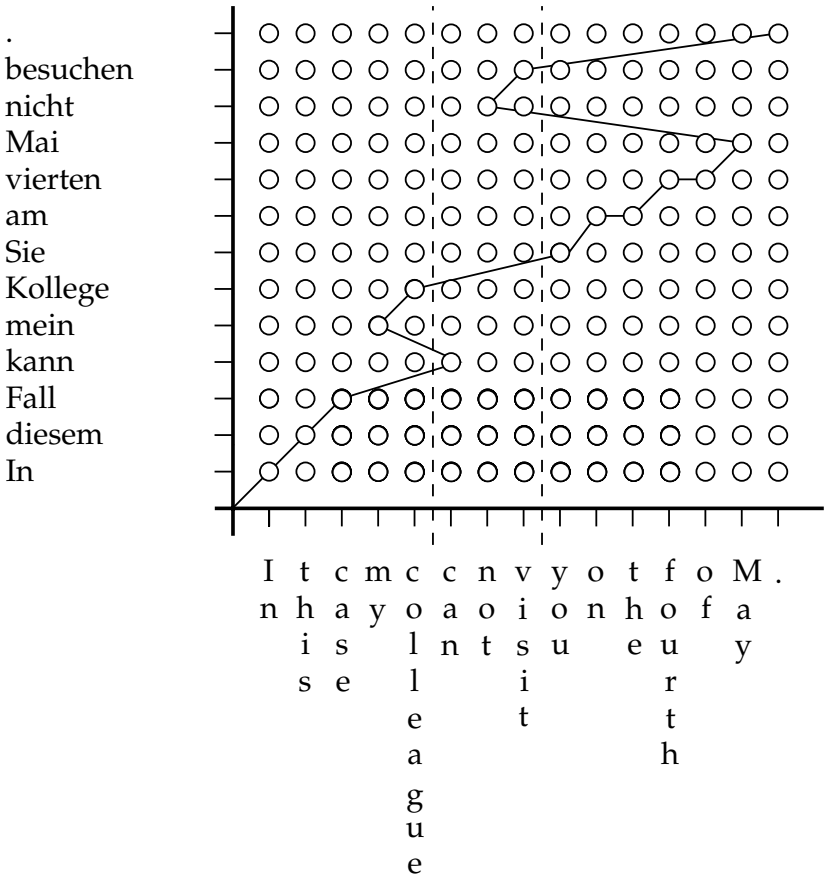


Figure 7
Word reordering for the translation direction English to German: The reordering is restricted to the English verb group.

These restrictions will be fully formalized later in this section. In the state *Move*, some source sentence positions are "moved" from later in the sentence to earlier. After source sentence positions are moved, they are marked, and the translation of the sentence is continued monotonically, keeping track of the positions already covered. To formalize the approach, we introduce four reordering states $\mathcal{S}$:

- *Initial*: A contiguous initial block of source positions is covered.
- *Skip*: A restricted number of source positions may be skipped, leaving "holes" in the monotonic traversal.
- *Move*: A restricted number of words may be "moved" from later in the sentence.
- *Cover*: The sentence is traversed monotonically until the state *Initial* is reached.

To formalize the approach, the following notation is introduced:

$$r_{\max}(\mathcal{C}) \quad = \quad \max_{c \in \mathcal{C}} c$$

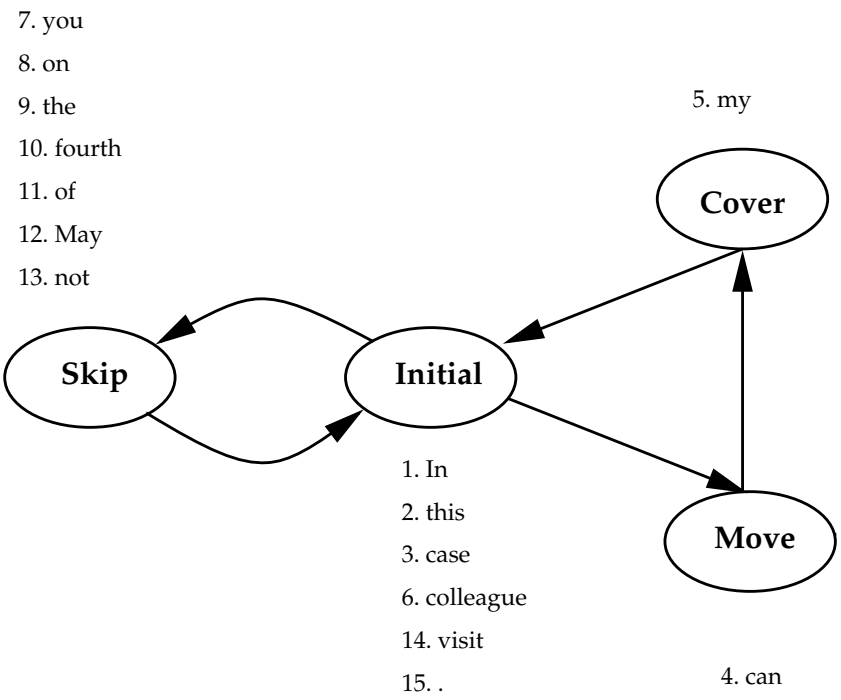


Figure 8
Order in which the English source positions are covered for the English-to-German reordering example given in Figure 7.

$$
\begin{aligned}
l_{\min}(\mathcal{C}) &= \min_{c \notin \mathcal{C}} c \\
u(\mathcal{C}) &= card(\{c \mid c \notin \mathcal{C} \text{ and } c < r_{\max}(\mathcal{C})\}) \\
m(\mathcal{C}) &= card(\{c \mid c \in \mathcal{C} \text{ and } c > l_{\min}(\mathcal{C})\}) \\
w(\mathcal{C}) &= r_{\max}(\mathcal{C}) - l_{\min}(\mathcal{C})
\end{aligned}
$$

$r_{\max}(\mathcal{C})$ is the rightmost covered and $l_{\min}(\mathcal{C})$ is the leftmost uncovered source position. $u(\mathcal{C})$ is the number of "skipped" positions, and $m(\mathcal{C})$ is the number of "moved" positions. The function $card(\cdot)$ returns the cardinality of a set of source positions. The function $w(\mathcal{C})$ describes the "window" size in which the word reordering takes place. A procedural description for the computation of the set of successor hypotheses for a given partial hypothesis $(\mathcal{S}, \mathcal{C}, j)$ is given in Table 3. There are restrictions on the possible successor states: A partial hypothesis in state *Skip cannot* be expanded into a partial hypothesis in state *Move* and vice versa. If the coverage set for the newly generated hypothesis covers a contiguous initial block of source positions, the state *Initial* is entered. No other state $\mathcal{S}$ is considered as a successor state in this case (hence the use of the continue statement in the procedural description). The set of successor hypotheses *Succ* by which to extend the partial hypothesis $(\mathcal{S}, \mathcal{C}, j)$ is computed using the constraints defined by the values for *numskip*, *widthskip*, *nummove*, and *widthmove*, as explained in the Appendix. In particular, a source position k is discarded for extension if the "window" restrictions are violated. Within the restrictions all possible successors are computed. It can be observed that the set of successors, as computed in Table 3, is never empty.

Table 3
Procedural description to compute the set *Succ* of successor hypotheses by which to extend a partial hypothesis $(\mathcal{S}, \mathcal{C}, j)$.

input: partial hypothesis $(\mathcal{S}, \mathcal{C}, j)$
$Succ := \{\emptyset\}$
for each $k \notin \mathcal{C}$ do
    Set $\mathcal{C}' = \mathcal{C} \cup \{k\}$
    if $u(\mathcal{C}') = 0$
        $Succ := Succ \cup (Initial, \mathcal{C}', k)$
        continue
    if ($\mathcal{S} = Initial$) or ($\mathcal{S} = Skip$)
        if $w(\mathcal{C}') \leq widthskip$ **and** $u(\mathcal{C}') \leq numskip$
            $Succ := Succ \cup (Skip, \mathcal{C}', k)$
    if ($\mathcal{S} = Initial$) or ($\mathcal{S} = Move$)
        if $k \neq l_{\min}(\mathcal{C}')$ **and** $w(\mathcal{C}') \leq widthmove$ **and** $m(\mathcal{C}') \leq nummove$
            $Succ := Succ \cup (Move, \mathcal{C}', k)$
    if ($\mathcal{S} = Move$) or ($\mathcal{S} = Cover$)
        if $(l_{\min}(\mathcal{C}') = k)$
            $Succ := Succ \cup (Cover, \mathcal{C}', k)$
output: set *Succ* of successor hypotheses

There is an asymmetry between the two reordering states *Move* and *Skip*: While in state *Move*, the algorithm is not allowed to cover the position $l_{\min}(\mathcal{C})$. It must first enter the state *Cover* to do so. In contrast, for the state *Skip*, the newly generated hypothesis always remains in the state *Skip* (until the state *Initial* is entered.) This is motivated by the word reordering for the German verb group. After the right verbal brace has been processed, no source words may be moved into the verbal brace from later in the sentence. There is a redundancy in the reorderings: The same reordering might be carried out using either the state *Skip* or *Move*, especially if *widthskip* and *widthmove* are about the same. The additional computational burden is alleviated somewhat by the fact that the pruning, as introduced in Section 3.8, does not distinguish hypotheses according to the states. A complexity analysis for different reordering constraints is given in Tillmann (2001).

3.6 Word Reordering: IBM-Style Restrictions

We now compare the new word reordering approach with the approach used in Berger et al. (1996). In the approach presented in this article, source sentence words are aligned with hypothesized target sentence words.[3] When a source sentence word is aligned, we say its position is covered. During the search process, a partial hypothesis is extended by choosing an uncovered source sentence position, and this choice is restricted. Only one of the first n uncovered positions in a coverage set may be chosen, where n is set to 4. This choice is illustrated in Figure 9. In the figure, covered positions are marked by a filled circle, and uncovered positions are marked by an unfilled circle. Positions that may be covered next are marked by an unfilled square. The restrictions for a coverage set C can be expressed in terms of the expression $u(\mathcal{C})$ defined in the previous section: The number of uncovered source sentence positions to the left of the rightmost covered position. Demanding $u(\mathcal{C}) \leq 3$, we obtain the S3 restriction

3 In Berger et al. (1996), a morphological analysis is carried out and word morphemes are processed during the search. Here, we process only full-form words.

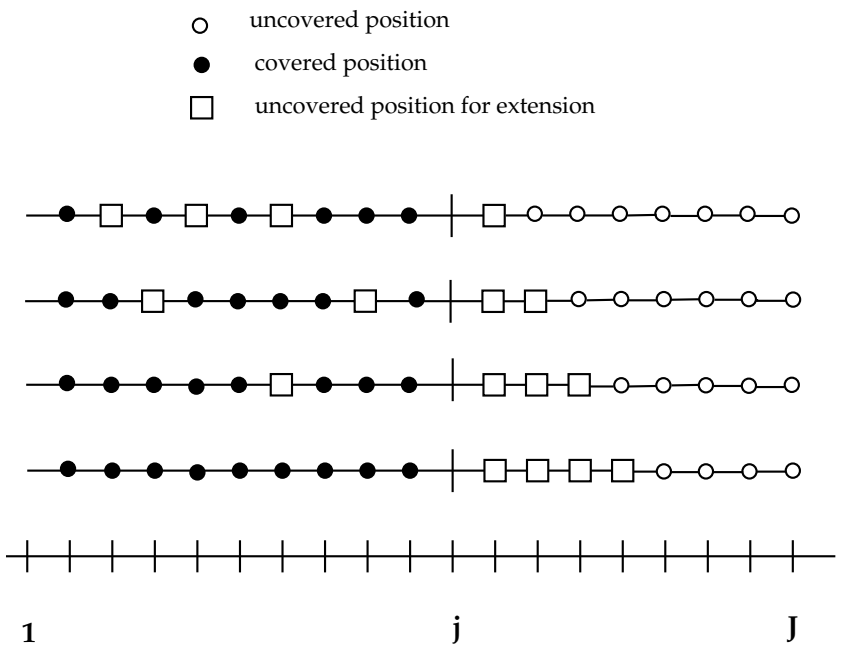


Figure 9
Illustration of the IBM-style reordering constraint.

introduced in the Appendix. An upper bound of $\mathcal{O}(E^3 \cdot J^4)$ for the word reordering complexity is given in Tillmann (2001).

3.7 Empirical Complexity Calculations

In order to demonstrate the complexity of the proposed reordering constraints, we have modified our translation algorithm to show, for the different reordering constraints, the overall number of successor states generated by the algorithm given in Table 3. The number of successors shown in Figure 10 is counted for a pseudotranslation task in which a pseudo–source word x is translated into the identically pseudo–target word x. No actual optimization is carried out; the total number of successors is simply counted as the algorithm proceeds through subsets of increasing cardinality. The complexity differences for the different reordering constraints result from the different number of coverage subsets $\mathcal{C}$ and corresponding reordering states $\mathcal{S}$ allowed. For the different reordering constraints we obtain the following results (the abbreviations MON, GE, EG, and S3 are taken from the Appendix):

- MON: For this reordering restriction, a partial hypothesis is always extended by the position $l_{\min}(\mathcal{C})$, hence the number of processed arcs is J.
- GE, EG: These two reordering constraints are very similar in terms of complexity: The number of word reorderings is heavily restricted in each. Actually, since the distance restrictions (expressed by the variables $widthskip$ and $widthmove$) apply, the complexity is linear in the length of the input sentence J.
- S3: The S3 reordering constraint has a complexity close to J^4. Since no distance restrictions for the skipped positions apply, the overall search space is significantly larger than for the GE or EG restriction.

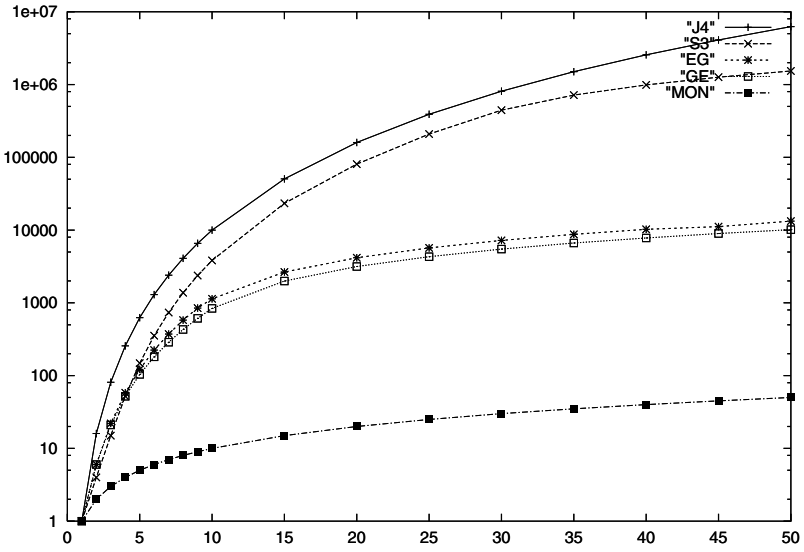


Figure 10
Number of processed arcs for the pseudotranslation task as a function of the input sentence length J (y-axis is given in log scale). The complexity for the four different reordering constraints MON, GE, EG, and S3 is given. The complexity of the S3 constraint is close to J^4.

3.8 Beam Search Pruning Techniques

To speed up the search, a beam search strategy is used. There is a direct analogy to the data-driven search organization used in continuous-speech recognition (Ney et al. 1992). The full DP search algorithm proceeds cardinality-synchronously over subsets of source sentence positions of increasing cardinality. Using the beam search concept, the search can be focused on the most likely hypotheses. The hypotheses $Q_{e'}(e, \mathcal{C}, j)$ are distinguished according to the coverage set $\mathcal{C}$, with two kinds of pruning based on this coverage set:

1. The *coverage* pruning is carried out separately for each coverage set $\mathcal{C}$.
2. The *cardinality* pruning is carried out jointly for all coverage sets $\mathcal{C}$ with the same cardinality $c = c(\mathcal{C})$.

After the pruning is carried out, we retain for further consideration only hypotheses with a probability close to the maximum probability. The number of surviving hypotheses is controlled by *four* kinds of thresholds:

- the coverage pruning threshold $t_{\mathcal{C}}$
- the coverage histogram threshold $n_{\mathcal{C}}$
- the cardinality pruning threshold t_c
- the cardinality histogram threshold n_c

For the coverage and the cardinality pruning, the probability $Q_{e'}(e, \mathcal{C}, j)$ is adjusted to take into account the uncovered source sentence positions $\bar{\mathcal{C}} = \{1, \ldots, J\} \backslash \mathcal{C}$. To make

this adjustment, for a source word f at an uncovered source position, we precompute an upper bound $\bar{p}(f)$ for the product of language model and lexicon probability:

$$\bar{p}(f) = \max_{e'',e',e}\{p(e \mid e', e'') \cdot p(f \mid e)\}$$

The above optimization is carried out only over the word trigrams (e, e', e'') that have actually been seen in the training data. Additionally, the observation pruning described below is applied to the possible translations e of a source word f. The upper bound is used in the beam search concept to increase the comparability between hypotheses covering different coverage sets. Even more benefit from the upper bound $\bar{p}(f)$ can be expected if the distortion and the fertility probabilities are taken into account (Tillmann 2001). Using the definition of $\bar{p}(f)$, the following modified probability $\bar{Q}_{e'}(e, \mathcal{C}, j)$ is used to replace the original probability $Q_{e'}(e, \mathcal{C}, j)$, and all pruning is applied to the new probability:

$$\bar{Q}_{e'}(e, \mathcal{C}, j) = Q_{e'}(e, \mathcal{C}, j) \cdot \prod_{j \in \bar{\mathcal{C}}} \bar{p}(f_j)$$

For the translation experiments, equation (3) is recursively evaluated over subsets of source positions of equal cardinality. For reasons of brevity, we omit the state description $\mathcal{S}$ in equation (3), since no separate pruning according to the states $\mathcal{S}$ is carried out. The set of surviving hypotheses for each cardinality c is referred to as the beam. The size of the beam for cardinality c depends on the ambiguity of the translation task for that cardinality. To fully exploit the speedup of the DP beam search, the search space is dynamically constructed as described in Tillmann, Vogel, Ney, Zubiaga, and Sawaf (1997), rather than using a static search space. To carry out the pruning, the maximum probabilities with respect to each coverage set $\mathcal{C}$ and cardinality c are computed:

- *Coverage pruning:* Hypotheses are distinguished according to the subset of covered positions $\mathcal{C}$. The probability $\hat{Q}(\mathcal{C})$ is defined:

$$\hat{Q}(\mathcal{C}) = \max_{e,e',j} \bar{Q}_{e'}(e, \mathcal{C}, j)$$

- *Cardinality pruning:* Hypotheses are distinguished according to the cardinality $c(\mathcal{C})$ of subsets $\mathcal{C}$ of covered positions. The probability $\hat{Q}(c)$ is defined for all hypotheses with $c(C) = c$:

$$\hat{Q}(c) = \max_{\substack{\mathcal{C} \\ c(\mathcal{C})=c}} \hat{Q}(\mathcal{C})$$

The coverage pruning threshold $t_{\mathcal{C}}$ and the cardinality pruning threshold t_c are used to prune active hypotheses. We call this pruning **translation pruning**. Hypotheses are pruned according to their translation probability:

$$\begin{aligned} \bar{Q}_{e'}(e, \mathcal{C}, j) &< t_{\mathcal{C}} \cdot \hat{Q}(\mathcal{C}) \\ \bar{Q}_{e'}(e, \mathcal{C}, j) &< t_c \cdot \hat{Q}(c) \end{aligned}$$

For the translation experiments presented in Section 4, the negative logarithms of the actual pruning thresholds t_c and $t_{\mathcal{C}}$ are reported. A hypothesis $(e', e, \mathcal{C}, j)$ is discarded if its probability is below the corresponding threshold. For the current experiments, the

coverage and the cardinality threshold are constant for different coverage sets $\mathcal{C}$ and cardinalities c. Together with the translation pruning, **histogram pruning** is carried out: The overall number $N(\mathcal{C})$ of active hypotheses for the coverage set $\mathcal{C}$ and the overall number $N(c)$ of active hypotheses for all subsets of a given cardinality may not exceed a given number; again, different numbers are used for coverage and cardinality pruning. The coverage histogram pruning is denoted by $n_{\mathcal{C}}$, and the cardinality histogram pruning is denoted by n_c:

$$\begin{aligned} N(\mathcal{C}) &> n_{\mathcal{C}} \\ N(c) &> n_c \end{aligned}$$

If the numbers of active hypotheses for each coverage set $\mathcal{C}$ and cardinality c, $N(\mathcal{C})$ and $N(c)$, exceed the above thresholds, only the partial hypotheses with the highest translation probabilities are retained (e.g., we may use $n_{\mathcal{C}} = 1{,}000$ for the coverage histogram pruning).

The third type of pruning conducted **observation pruning**: The number of words that may be produced by a source word f is limited. For each source language word f the list of its possible translations e is sorted according to

$$p(f \mid e) \cdot p_{\text{uni}}(e)$$

where $p_{\text{uni}}(e)$ is the unigram probability of the target language word e. Only the best n_o target words e are hypothesized during the search process (e.g., during the experiments to hypothesize, the best $n_o = 50$ words was sufficient.

3.9 Beam Search Implementation

In this section, we describe the implementation of the beam search algorithm presented in the previous sections and show how it is applied to the full set of IBM-4 model parameters.

3.9.1 Baseline DP Implementation. The implementation described here is similar to that used in beam search speech recognition systems, as presented in Ney et al. (1992). The similarities are given mainly in the following:

- The implementation is data driven. Both its time and memory requirements are strictly linear in the number of path hypotheses (disregarding the sorting steps explained in this section).
- The search procedure is developed to work most efficiently when the input sentences are processed mainly monotonically from left to right. The algorithm works cardinality-synchronously, meaning that all the hypotheses that are processed cover subsets of source sentence positions of equal cardinality c.
- Since full search is prohibitive, we use a beam search concept, as in speech recognition. We use appropriate pruning techniques in connection with our cardinality-synchronous search procedure.

Table 4 shows a two-list implementation of the search algorithm given in Table 2 in which the beam pruning is included. The two lists are referred to as $\mathcal{S}$ and $\mathcal{S}_{\text{new}}$: $\mathcal{S}$ is the list of hypotheses that are currently expanded, and $\mathcal{S}_{\text{new}}$ is the list of newly

Table 4
Two-list implementation of a DP-based search algorithm for statistical MT.

input: source string $f_1 \cdots f_j \cdots f_J$
initial hypothesis lists: $\mathcal{S} = \{(\$, \$, \{\emptyset\}, 0)\}$
for each cardinality $c = 1, 2, \ldots, J$ do
  $\mathcal{S}_{\text{new}} = \{\emptyset\}$
  for each hypothesis $(e', e, \mathcal{C}, j') \in \mathcal{S}$, where $j' \in \mathcal{C}$ and $|\mathcal{C}| = c$ do
    Expand $(e', e, \mathcal{C}, j')$ using probabilities $p(f_j \mid e) \cdot p(j \mid j', J) \cdot p(e \mid e', e'')$
    Look up and add or update expanded hypothesis in $\mathcal{S}_{\text{new}}$
  Sort hypotheses in $\mathcal{S}_{\text{new}}$ according to translation score
  Carry out cardinality pruning
  Sort hypotheses in $\mathcal{S}_{\text{new}}$ according to coverage set $\mathcal{C}$ and translation score
  Carry out coverage pruning
  Bookkeeping of surviving hypotheses in $\mathcal{S}_{\text{new}}$
  $\mathcal{S} := \mathcal{S}_{\text{new}}$
output: get best target word sequence e_1^I from bookkeeping array

generated hypotheses. The search procedure processes subsets of covered source sentence positions of increasing cardinality. The search starts with $\mathcal{S} = \{(\$, \$, \{\emptyset\}, 0)\}$, where \$ denotes the sentence start symbol for the immediate two predecessor words and $\{\emptyset\}$ denotes the empty coverage set, in which no source position is covered yet. For the initial search state, the position last covered is set to 0. A set $\mathcal{S}$ of active hypotheses is expanded for each cardinality c using lexicon model, language model, and distortion model probabilities. The newly generated hypotheses are added to the hypothesis set $\mathcal{S}_{\text{new}}$; for hypotheses that are not distinguished according to our DP approach, only the best partial hypothesis is retained for further consideration. This so-called recombination is implemented as a set of simple lookup and update operations on the set $\mathcal{S}_{\text{new}}$ of partial hypotheses. During the partial hypothesis extensions, an anticipated pruning is carried out: Hypotheses are discarded before they are considered for recombination and are never added to S_{new}. (The anticipated pruning is not shown in Table 4. It is based on the pruning thresholds described in Section 3.8.) After the extension of all partial hypotheses in $\mathcal{S}$, a pruning step is carried out for the hypotheses in the newly generated set $\mathcal{S}_{\text{new}}$. The pruning is based on two simple sorting steps on the list of partial hypotheses $\mathcal{S}_{\text{new}}$. (Instead of sorting the partial hypotheses, we might have used hashing.) First, the partial hypotheses are sorted according to their translation scores (within the implementation, all probabilities are converted into translation scores by taking the negative logarithm $-\log()$). Cardinality pruning can then be carried out simply by running down the list of hypotheses, starting with the maximum-probability hypothesis, and applying the cardinality thresholds. Then, the partial hypotheses are sorted a second time according to their coverage set $\mathcal{C}$ *and* their translation score. After this sorting step, all partial hypotheses that cover the same subset of source sentence positions are located in consecutive fragments in the overall list of partial hypotheses. Coverage pruning is carried out in a single run over the list of partial hypotheses: For each fragment corresponding to the same coverage set $\mathcal{C}$, the coverage pruning threshold is applied. The partial hypotheses that survive the two pruning stages are then written into the so-called bookkeeping array (Ney et al. 1992). For the next expansion step, the set $\mathcal{S}$ is set to the newly generated list of hypotheses. Finally, the target translation is constructed from the bookkeeping array.

3.9.2 Details for IBM-4 Model. In this section, we outline how the DP-based beam search approach can be carried out using the full set of IBM-4 parameters. (More details can be found in Tillmann [2001] or in the cited papers.) First, the full set of IBM-4 parameters does not make the simplifying assumption given in Section 3.1, namely, that source and target sentences are of equal length: Either a target word e may be aligned with several source words (its fertility is greater than one) or a single source word may produce zero, one, or two target words, as described in Berger et al. (1996), or both. Zero target words are generated if f is aligned to the "null" word e_0. Generating a single target word e is the regular case. Two target words (e', e'') may be generated. The costs for generating the target word e' are given by its fertility $\phi(0 \mid e')$ and the language model probability; no lexicon probability is used. During the experiments, we restrict ourselves to triples of target words (e, e', e'') actually seen in the training data. This approach is used for the French-to-English translation experiments presented in this article.

Another approach for mapping a single source language word to several target language words involves preprocessing by the word-joining algorithm given in Tillmann (2001), which is similar to the approach presented in Och, Tillmann, and Ney (1999). Target words are joined during a training phase, and several joined target language words are dealt with as a new lexicon entry. This approach is used for the German-to-English translation experiments presented in this article.

In order to deal with the IBM-4 fertility parameters within the DP-based concept, we adopt the distinction between **open** and **closed hypotheses** given in Berger et al. (1996). A hypothesis is said to be open if it is to be aligned with more source positions than it currently is (i.e., at least two). Otherwise it is called closed. The difference between open and closed is used to process the input sentence one position a time (for details see Tillmann 2001). The word reordering restrictions and the beam search pruning techniques are directly carried over to the full set of IBM-4 parameters, since they are based on restrictions on the coverage vectors $\mathcal{C}$ only.

To ensure its correctness, the implementation was tested by carrying out forced alignments on 500 German-to-English training sentence pairs. In a forced alignment, the source sentence f_1^J and the target sentence e_1^I are kept fixed, and a full search without re-ordering restrictions is carried out only over the unknown alignment a_1^J. The language model probability is divided out, and the resulting probability is compared to the Viterbi probability as obtained by the training procedure. For 499 training sentences the Viterbi alignment probability as obtained by the forced-alignment search was exactly the same as the one produced by the training procedure. In one case the forced-alignment search did obtain a better Viterbi probability than the training procedure.

4. Experimental Results

Translation experiments are carried out for the translation directions German to English and English to German (Verbmobil task) and for the translation directions French to English and English to French (Canadian Hansards task). Section 4.1 reports on the performance measures used. Section 4.2 shows translation results for the Verbmobil task. Sections 4.2.1 and 4.2.2 describe that task and the preprocessing steps applied. In Sections 4.2.3 through 4.2.5, the efficiency of the beam search pruning techniques is shown for German-to-English translation, as the most detailed experiments are conducted for that direction. Section 4.2.6 gives translation results for the translation direction English to German. In Section 4.3, translation results for the Canadian Hansards task are reported.

4.1 Performance Measures for Translation Experiments

To measure the performance of the translation methods, we use three types of automatic and easy-to-use measures of the translation errors. Additionally, a subjective evaluation involving human judges is carried out (Niessen et al. 2000). The following evaluation criteria are employed:

- *WER (word error rate):* The WER is computed as the minimum number of substitution, insertion, and deletion operations that have to be performed to convert the generated string into the reference target string. This performance criterion is widely used in speech recognition. The minimum is computed using a DP algorithm and is typically referred to as *edit* or *Levenshtein* distance.
- *mWER (multireference WER):* We use the Levenshtein distance between the automatic translation and several reference translations as a measure of the translation errors. For example, on the Verbmobil TEST-331 test set, an average of six reference translations per automatic translation are available. The Levenshtein distance between the automatic translation and each of the reference translations is computed, and the minimum Levenshtein distance is taken. The resulting measure, the mWER, is more robust than the WER, which takes into account only a single reference translation.
- *PER (position-independent word error rate):* In the case in which only a single reference translation per sentence is available, we introduce as an additional measure the position-independent word error rate (PER). This measure compares the words in the two sentences *without* taking the word order into account. Words in the reference translation that have no counterpart in the translated sentence are counted as substitution errors. Depending on whether the translated sentence is longer or shorter than the reference translation, the remaining words result in either insertion (if the translated sentence is longer) or deletion (if the translated sentence is shorter) errors. The PER is guaranteed to be less than or equal to the WER. The PER is more robust than the WER since it ignores translation errors due to different word order in the translated and reference sentences.
- *SSER (subjective sentence error rate):* For a more fine-grained evaluation of the translation results and to check the validity of the automatic evaluation measures subjective judgments by test persons are carried out (Niessen et al. 2000). The following scale for the error count per sentence is used in these subjective evaluations:

0.0	:	semantically correct and syntactically correct
...	:	...
0.5	:	semantically correct and syntactically wrong
...	:	...
1.0	:	semantically wrong (independent of syntax)

  Each translated sentence is judged by a human examiner according to the above error scale; several human judges may be involved in judging the same translated sentence. Subjective evaluation is carried out only for the Verbmobil TEST-147 test set.

Table 5
Training and test conditions for the German-to-English Verbmobil corpus (*number of words without punctuation).

		German	English
Training:	Sentences	58,073	
	Words	519,523	549,921
	Words*	418,979	453,632
Vocabulary:	Size	7,911	4,648
	Singletons	3,453	1,699
TEST-331:	Sentences	331	
	Words	5,591	6,279
	Bigram/Trigram Perplexity	84.0/68.2	49.3/38.3
TEST-147:	Sentences	147	
	Words	1,968	2,173
	Bigram/Trigram Perplexity	—	34.6/28.1

4.2 Verbmobil Translation Experiments

4.2.1 The Task and the Corpus. The translation system is tested on the Verbmobil task (Wahlster 2000). In that task, the goal is the translation of spontaneous speech in face-to-face situations for an appointment scheduling domain. We carry out experiments for both translation directions: German to English and English to German. Although the Verbmobil task is still a limited-domain task, it is rather difficult in terms of vocabulary size, namely, about 5,000 words or more for each of the two languages; second, the syntactic structures of the sentences are rather unrestricted. Although the ultimate goal of the Verbmobil project is the translation of *spoken* language, the input used for the translation experiments reported on in this article is mainly the (more or less) correct orthographic transcription of the spoken sentences. Thus, the effects of spontaneous speech are present in the corpus; the effect of speech recognition errors, however, is not covered. The corpus consists of 58,073 training pairs; its characteristics are given in Table 5. For the translation experiments, a trigram language model with a perplexity of 28.1 is used. The following two test corpora are used for the translation experiments:

TEST-331: This test set consists of 331 test sentences. Only automatic evaluation is carried out on this test corpus: The WER and the mWER are computed. For each test sentence in the source language there is a range of acceptable reference translations (six on average) provided by a human translator, who is asked to produce word-to-word translations wherever it is possible. Part of the reference sentences are obtained by correcting automatic translations of the test sentences that are produced using the approach presented in this article with different reordering constraints. The other part is produced from the source sentences without looking at any of their translations. The TEST-331 test set is used as held-out data for parameter optimization (for the language mode scaling factor and for the distortion model scaling factor). Furthermore, the beam search experiments in which the effect of the different pruning thresholds is demonstrated are carried out on the TEST-331 test set.

TEST-147: The second, separate test set consists of 147 test sentences. Translation results are given in terms of mWER and SSER. No parameter optimization

is carried out on the TEST-147 test set; the parameter values as obtained from the experiments on the TEST-331 test set are used.

4.2.2 Preprocessing Steps. To improve the translation performance the following preprocessing steps are carried out:

Categorization: We use some categorization, which consists of replacing a single word by a category. The only words that are replaced by a category label are proper nouns denoting German cities. Using the new labeled corpus, all probability models are trained anew. To produce translations in the "normal" language, the categories are translated by rule and are inserted into the target sentence.

Word joining: Target language words are joined using a method similar to the one described in Och, Tillmann, and Ney (1999). Words are joined to handle cases like the German compound noun "Zahnarzttermin" for the English "dentist's appointment," because a single word has to be mapped to two or more target words. The word joining is applied only to the target language words; the source language sentences remain unchanged. During the search process several joined target language words may be generated by a single source language word.

Manual lexicon: To account for unseen words in the test sentences and to obtain a greater number of focused translation probabilities $p(f \mid e)$, we use a bilingual German-English dictionary. For each word e in the target vocabulary, we create a list of source translations f according to this dictionary. The translation probability $p_{\text{dic}}(f \mid e)$ for the dictionary entry (f, e) is defined as

$$p_{\text{dic}}(f \mid e) = \begin{cases} \dfrac{1}{N_e} & \text{if } (f,e) \text{ is in dictionary} \\ 0 & \text{otherwise} \end{cases}$$

where N_e is the number of source words listed as translations of the target word e. The dictionary probability $p_{\text{dic}}(f \mid e)$ is linearly combined with the automatically trained translation probabilities $p_{\text{aut}}(f \mid e)$ to obtain smoothed probabilities $p(f \mid e)$:

$$p(f \mid e) = (1 - \lambda) \cdot p_{\text{dic}}(f \mid e) + \lambda \cdot p_{\text{aut}}(f \mid e)$$

For the translation experiments, the value of the interpolation parameter is fixed at $\lambda = 0.5$.

4.2.3 Effect of the Scaling Factors. In speech recognition, in which Bayes' decision rule is applied, a language model scaling factor α_{LM} is used; a typical value is $\alpha_{\text{LM}} \approx 15$. This scaling factor is employed because the language model probabilities are more reliably estimated than the acoustic probabilities. Following this use of a language model scaling factor in speech recognition, such a factor is introduced into statistical MT, too. The optimization criterion in equation (1) is modified as follows:

$$\hat{e}_1^I = \arg\max_{e_1^I} \{p(e_1^I)^{\alpha_{\text{LM}}} \cdot p(f_1^J \mid e_1^I)\}$$

where $p(e_1^I)$ is the language model probability of the target language sentence. In the experiments presented here, a trigram language model is used to compute $p(e_1^I)$. The

Table 6
Computing time, mWER, and SSER for three different reordering constraints on the TEST-147 test set. During the translation experiments, reordered words are not allowed to cross punctuation marks.

Reordering constraint	CPU time [sec]	mWER [%]	SSER [%]
MON	0.2	40.6	28.6
GE	5.2	33.3	21.0
S3	13.7	34.4	19.9

effect of the language model scaling factor α_{LM} is studied on the TEST-331 test set. A minimum mWER is obtained for $\alpha_{\mathrm{LM}} = 0.8$, as reported in Tillmann (2001). Unlike in speech recognition, the translation model probabilities seem to be estimated as reliably as the language model probabilities in statistical MT.

A second scaling factor α_{D} is introduced for the distortion model probabilities $p(j \mid j', J)$. A minimum mWER is obtained for $\alpha_{\mathrm{D}} = 0.4$, as reported in Tillmann (2001). The WER and mWER on the TEST-331 test set increase significantly, if no distortion probability is used, for the case $\alpha_{\mathrm{D}} = 0.0$. The benefit of a distortion probability scaling factor of $\alpha_{\mathrm{D}} = 0.4$ comes from the fact that otherwise, a low distortion probability might suppress long-distant word reordering that is important for German-to-English verb group reordering. The setting $\alpha_{\mathrm{LM}} = 0.8$ and $\alpha_{\mathrm{D}} = 0.4$ is used for all subsequent translation results (including the translation direction English to German).

4.2.4 Effect of the Word Reordering Constraints. Table 6 shows the computing time, mWER, and SSER on the TEST-147 test set as a function of three reordering constraints: MON, GE, and S3 (as discussed in the Appendix). The computing time is given in terms of central processing unit (CPU) time per sentence (on a 450 MHz Pentium III personal computer). For the SSER, it turns out that restricting the word reordering such that it may not cross punctuation marks improves translation performance significantly. The average length of the sentence fragments that are separated by punctuation marks is rather small: 4.5 words per fragment. A coverage pruning threshold of $t_{\mathcal{C}} = 5.0$ and an observation pruning of $n_o = 50$ are applied during the experiments.[4] No other type of pruning is used.[5]

The MON constraint performs worst in terms of both mWER and SSER. The computing time is small, since no reordering is carried out. Constraints GE and S3 perform nearly identically in terms of both mWER and SSER. The GE constraint, however, works about three times as fast as the S3 constraint.

Table 7 shows example translations obtained under the three different reordering constraints. Again, the MON reordering constraint performs worst. In the second and third translation examples, the S3 word reordering constraint performs worse than the GE reordering constraint, since it cannot take the word reordering due to the German verb group properly into account. The German finite verbs *bin* (second example) and *könnten* (third example) are too far away from the personal pronouns *ich* and *Sie* (six

4 For the translation experiments, the negative logarithm of the actual pruning thresholds t_c and $t_{\mathcal{C}}$ is reported; for simplicity reasons we do not change the notation.
5 In a speech-to-speech demo system, we would use the GE reordering restriction and a slightly sharper pruning in order to achieve translation times of about one second per sentence.

Table 7
Example translations for the translation direction German to English using three different reordering constraints: MON, GE, and S3.

Input:	Ja, wunderbar. Können wir machen.
MON:	Yes, wonderful. Can we do.
GE:	Yes, wonderful. We can do that.
S3:	Yes, wonderful. We can do that.
Input:	Das ist zu knapp , weil ich ab dem dritten in Kaiserslautern bin. Genaugenommen nur am dritten. Wie wäre es denn am ähm Samstag, dem zehnten Februar?
MON:	That is too tight , because I from the third in Kaiserslautern. In fact only on the third. How about ähm Saturday , the tenth of February?
GE:	That is too tight, because I am from the third in Kaiserslautern. In fact only on the third. Ähm how about Saturday, February the tenth?
S3:	That is too tight, from the third because I will be in Kaiserslautern. In fact only on the third. Ähm how about Saturday, February the tenth?
Input:	Wenn Sie dann noch den siebzehnten könnten, wäre das toll, ja.
MON:	If you then also the seventeenth could, would be the great, yes.
GE:	If you could then also the seventeenth, that would be great, yes.
S3:	Then if you could even take seventeenth, that would be great, yes.
Input:	Ja, das kommt mir sehr gelegen. Machen wir es dann am besten so.
MON:	Yes, that suits me perfectly. Do we should best like that.
GE:	Yes, that suits me fine. We do it like that then best.
S3:	Yes, that suits me fine. We should best do it like that.

and five source sentence positions, respectively) to be reordered properly. In the last example, the less restrictive S3 reordering constraint leads to a better translation; the GE translation is still acceptable, though.

4.2.5 Effect of the Beam Search Pruning Thresholds. In this section, the effect of the beam search pruning is demonstrated. Translation results on the TEST-331 test set are presented to evaluate the effectiveness of the pruning techniques.[6] The quality of the search algorithm with respect to the GE and S3 reordering constraints is evaluated using two criteria:

1. The number of search errors for a certain combination of pruning thresholds is counted. A search error occurs for a test sentence if the final translation probability Q_F for a candidate translation e_1^I as given in equation (4) is smaller than a reference probability for that test sentence. We will compute reference probabilities two ways, as explained below.
2. The mWER performance measure is computed as a function of the pruning thresholds used. Generally, decreasing the pruning threshold

6 The CPU times on the TEST-331 set are higher, since the average fragment length is greater than for the TEST-147 set.

Table 8
Effect of the coverage pruning threshold t_C on the number of search errors and mWER on the TEST-331 test set (no cardinality pruning carried out: $t_c = \infty$). A cardinality histogram pruning of 200,000 is applied to restrict the maximum overall size of the search space. The negative logarithm of t_C is reported.

Reordering constraint	t_C	CPU time [sec]	Search errors $Q_{ref} > Q_F$	Search errors $Q_{F^*} > Q_F$	mWER [%]
GE	0.01	0.21	318	323	73.5
	0.1	0.43	231	301	53.1
	1.0	1.43	10	226	30.3
	2.5	4.75	5	142	25.8
	5.0	29.6	—	35	24.6
	7.5	156	—	2	24.9
	10.0	630	—	—	24.9
	12.5	1300	—	—	24.9
S3	0.01	5.48	314	324	70.0
	0.1	9.21	225	303	50.9
	1.0	46.2	4	223	31.6
	2.5	190	—	129	28.4
	5.0	830	—	—	28.3

leads to a higher word error rate, since the optimal path through the translation lattice is missed, resulting in translation errors.

Two automatically generated reference probabilities are used. These probabilities are computed *separately* for the reordering constraints GE and S3 (the difference is not shown in the notation, but will be clear from the context):

Q_{ref}**:** A forced alignment is carried out between each of the test sentences and its corresponding reference translation; only a single reference translation for each test sentence is used. The probability obtained for the reference translation is denoted by Q_{ref}.

Q_{F^*}**:** A translation is carried out with conservatively large pruning thresholds, yielding a translation close to the one with the maximum translation probability. The translation probability for that translation is denoted by Q_{F^*}.

First, in a series of experiments we study the effect of the coverage and cardinality pruning for the reordering constraints GE and S3. (When we report on the different pruning thresholds, we will show the negative logarithm of those pruning thresholds.) The experiments are carried out on two different pruning "dimensions":

1. In Table 8, only coverage pruning using threshold t_C is carried out; no cardinality pruning is applied: $t_c = \infty$.
2. In Table 9, only cardinality pruning using threshold t_c is carried out; no coverage pruning is applied: $t_C = \infty$.

Both tables use an observation pruning of $n_o = 50$. The effect of the coverage pruning threshold t_C is demonstrated in Table 8. For the translation experiments reported in this table, the cardinality pruning threshold is set to $t_c = \infty$; thus, no comparison between partial hypotheses that do not cover the same set C of source sentence

Table 9
Effect of the cardinality pruning threshold t_c on the number of search errors and mWER on the TEST-331 test set (no coverage pruning is carried out: $t_C = \infty$). A coverage histogram pruning of 1,000 is applied to restrict the overall size of the search space. The negative logarithm of t_c is shown.

Reordering constraint	t_c	CPU time [sec]	Search errors $Q_{\text{ref}} > Q_{\text{F}}$	Search errors $Q_{\text{F}^*} > Q_{\text{F}}$	mWER [%]
GE	1.0	0.03	45	287	48.5
	2.0	0.06	20	277	41.9
	3.0	0.13	16	266	37.7
	4.0	0.30	6	239	34.1
	5.0	0.55	2	212	30.5
	7.5	3.2	—	106	26.6
	10.0	14.2	—	32	25.1
	12.5	42.2	—	5	24.9
	15.0	93.9	—	—	24.9
	17.5	176.7	—	—	24.9
S3	1.0	0.02	10	331	51.4
	2.0	0.05	1	283	46.2
	3.0	0.10	1	274	43.3
	4.0	0.22	—	251	40.2
	5.0	0.50	—	227	37.5
	7.5	4.3	—	171	32.9
	10.0	26.8	—	99	30.8
	12.5	123.3	—	49	28.9
	15.0	430	—	—	28.2

positions is carried out. To restrict the overall size of the search space in terms of CPU time and memory requirements, a cardinality pruning of $n_c = 200{,}000$ is applied. As can be seen from Table 8, mWER and the number of search errors decrease significantly as the coverage pruning threshold t_C increases. For the GE reordering constraint, mWER decreases from 73.5% to 24.9%. For a coverage pruning threshold $t_C \geq 5.0$, mWER remains nearly constant at 25.0%, although search errors still occur. For the S3 reordering constraint, mWER decreases from 70.0% to 28.3%. The largest coverage threshold tested for the S3 constraint is $t_C = 5.0$, since for larger threshold values t_C, the search procedure cannot be carried out because of memory and time restrictions. The number of search errors is reduced as the coverage pruning threshold is increased. It turns out to be difficult to verify search errors by looking at the reference translation probabilities Q_{ref} alone. The translation with the maximum translation probability seems to be quite narrowly defined. The coverage pruning is more effective for the GE constraint than for the S3 constraint, since the overall search space for the GE reordering is smaller.

Table 9 shows the effect of the cardinality pruning threshold t_c on mWER when no coverage pruning is carried out (a histogram coverage pruning of 1,000 is applied to restrict the overall size of the search space). The cardinality threshold t_c has a strong effect on mWER, which decreases significantly as the cardinality threshold t_c increases. For the GE reordering constraint, mWER decreases from 48.5% to 24.9%; for the S3 reordering constraint, mWER decreases from 51.4% to 28.2%. For the coverage threshold $t = 15.0$, the GE constraint works about four times as fast as the S3 constraint, since the overall search space for the S3 constraint is much larger. Although the overall search space is much larger for the S3 constraint, for smaller values of the coverage

Table 10
Effect of observation pruning on the number of search errors and mWER on the TEST-331 test set (parameter setting: $t_c = \infty$, $t_{\mathcal{C}} = 10.0$). No histogram pruning is applied. The results are reported for the GE constraint.

Observation pruning n_o	CPU time [sec]	Search errors $Q_{\text{ref}} > Q_{\text{F}}$	Search errors $Q_{\text{F}*} > Q_{\text{F}}$	mWER [%]
1	2.0	13	284	29.3
2	5.9	6	239	26.9
3	10.8	2	196	25.7
5	23.6	2	140	25.3
10	62.9	—	99	24.8
25	238	—	44	24.5
50	630	—	—	24.9

threshold $t_{\mathcal{C}} \leq 5.0$, the S3 constraint works as fast as the GE constraint or even faster, because only a very small portion of the overall search space is searched for small values of the cardinality pruning threshold t_c. There is some computational overhead in expanding a partial hypothesis for the GE constraint because the finite-state control has to be handled. No results are obtained for the S3 constraint and the coverage threshold $t_c = 17.5$ because of memory restrictions. The number of search errors is reduced as the cardinality pruning threshold is increased. Again, it is difficult to verify search errors by looking at the reference translation probabilities alone.

Both coverage and cardinality pruning are more efficient for the GE reordering constraint than for the S3 reordering constraint. For the S3 constraint, no translation results are obtained for a coverage threshold $t_c > 5.0$ without cardinality pruning applied because of memory and computing time restrictions. For the GE constraint virtually a full search can be carried out where only observation pruning is applied: Identical target translations and translation probabilities are produced for the hypothesis files for the two cases (1) $t_{\mathcal{C}} = 10.0$, $t_c = \infty$, and (2) $t_{\mathcal{C}} = \infty$, $t_c = 15.0$. (Actually, for one test sentence in the TEST-331 test set, the translations are different, although the translation probabilities are exactly the same.) Since the pruning is carried out independently on two different pruning dimensions, no search errors will occur if the thresholds are further increased.

Table 10 shows the effect of the observation pruning parameter n_o on mWER for the reordering constraint GE. mWER is significantly reduced by hypothesizing up to the best 50 target words e for a source language word f. mWER increases from 24.9% to 29.3% when the number of hypothesized words is decreased to only a single word.

Table 11 demonstrates the effect of the combination of the coverage pruning threshold $t_{\mathcal{C}} = 5.0$ and the cardinality pruning threshold $t_c = 12.5$, where the actual values are found in informal experiments: In a typical setting of the two parameters t_c should be at least twice as big as $t_{\mathcal{C}}$. For the GE reordering constraint, the average computing time is about seven seconds per sentence without any loss in translation performance as measured in terms of mWER. For the S3 reordering constraint, the average computing time per sentence is 27 seconds. Again, the combination of coverage and cardinality pruning works more efficiently for the GE constraint. The memory requirement for the algorithm is about 100 MB.

4.2.6 English-to-German Translation Experiments. A series of translation experiments for the translation direction English to German are also carried out. The results, given

Table 11
Demonstration of the combination of the two pruning thresholds $t_C = 5.0$ and $t_c = 12.5$ to speed up the search process for the two reordering constraints GE and S3 ($n_o = 50$). The translation performance is shown in terms of mWER on the TEST-331 test set.

Reordering constraint	t_C	t_c	CPU time [sec]	Search errors $Q_{ref} > Q_F$	Search errors $Q_{F*} > Q_F$	mWER [%]
GE	5.0	12.5	6.9	0	38	24.7
S3	5.0	12.5	26.9	0	65	29.2

Table 12
Translation results for the translation direction English to German on the TEST-331 test set. The results are given in terms of computing time, WER, and PER for three different reordering constraints: MON, EG, and S3.

Reordering constraint	CPU time [sec]	WER [%]	PER [%]
MON	0.5	70.6	57.0
EG	10.1	70.1	55.9
S3	53.2	70.1	55.8

in terms of WER and PER, are shown in Table 12. For the English-to-German translation direction, a single reference translation for each test sentence is used to carry out the automatic evaluation. The translation task for the translation direction English to German is more difficult than for the translation direction German to English; the trigram language model perplexity increases from 38.3 to 68.2 on the TEST-331 test set, as can be seen in Table 5. No parameter optimization is carried out for this translation direction; the parameter settings are carried over from the results obtained in Table 11.

The word error rates for the translation direction English to German are significantly higher than those for the translation direction German to English. There are several reasons for this: German vocabulary and perplexity are significantly larger than those for English, and only a single reference translation per test sentence is available for English-to-German translation. There is only a very small difference in terms of word error rates for the reordering constraints EG and S3; in particular, WER is 70.1% for both. The reordering constraint MON performs slightly worse: WER increases to 70.6%, and PER increases to 57.0%. Table 13 shows translation examples for the translation direction English to German. The MON constraint performs worst; there is no significant difference in quality of translations produced under the EG and the S3 constraints.

4.3 Canadian Hansards Translation Experiments

4.3.1 The Task and the Corpus. The second corpus on which we perform translation experiments is the Hansard corpus. By law, the proceedings of the Canadian parliament are recorded in both French and English. (For historical reasons, these proceedings are called "Hansards.") The remarks of the parliament members are written down in whichever of the two languages they use. They are then translated into the other language to produce complete sets of the proceedings, one in French and the other in English. The resulting bilingual data have been sentence-aligned using statistical methods (Brown et al. 1990). Originally, about three million sentences were selected. Here, we use a subset of the original training data; the details regarding this subset

Table 13
Example translations for the translation direction English to German using three different reordering constraints: MON, EG, and S3.

Input:	Yeah , that wouldn't be bad. Do you have any ideas where I could stay?
MON:	Ja, das wäre schade. Haben Sie irgendwelche Ideen wo ich könnten übernachten?
EG:	Ja, das wäre nicht schlecht. Haben Sie irgendwelche Ideen wo wir wohnen könnten?
S3:	Ja, das wäre nicht schlecht. Haben Sie irgendwelche Ideen wo wir wohnen könnten?
Input:	Oh, that sounds great . Could you arrange a suite for me?
MON:	Oh, das klingt gut. Könnten Sie unbedingt ein Suite bei mir?
EG:	Oh, das klingt gut. Könnten Sie einen Suite ausmachen für mich?
S3:	Oh, das klingt gut. Könnten Sie mir einen Suite ausmachen?
Input:	Well, I still need your signature here and then I will check with your company.
MON:	Also, ich konnte Arbeitskraft Unterschrift hier und ich werde nachsehen mit Ihrer Firma.
EG:	Also, ich bräuchte noch Ihre Unterschrift und dann gucke ich hier mit Ihrer Firma.
S3:	Also, ich brauche hier noch Ihre Unterschrift und dann werde ich veranlassen mit Ihrer Firma.

Table 14
Training and test conditions for the Hansards task (*number of words without punctuation).

		French	English
Train:	Sentences	1,470,473	
	Words	24,338,195	22,163,092
	Words*	22,175,069	20,063,378
Vocabulary:	Size	100,269	78,332
	Singletons	40,199	31,319
Test:	Sentences	5,432	
	Words	97,646	80,559
	Bigr./Tri. Perplexity	196.9/121.8	269.9/179.8

are given in Table 14. The Hansards corpus presents by far a more difficult task than the Verbmobil corpus in terms of vocabulary size and number of training sentences. The training and test sentences are less restrictive than for the Verbmobil task. For the translation experiments on the Hansards corpus, no word joining is carried out. Two target words can be produced by a single source word, as described in Section 3.9.2.

4.3.2 Translation Results. As can be seen in Table 15 for the translation direction French to English and in Table 16 for the translation direction English to French, the word error rates are rather high compared to those for the Verbmobil task. The reason for the higher error rates is that, as noted in the previous section, the Hansards task is by far less restrictive than the Verbmobil task, and the vocabulary size is much

Table 15
Computing time, WER, and PER for the translation direction French to English using the two reordering constraints MON and S3. An almost "full" search is carried out.

Reordering constraint	CPU time [sec]	WER [%]	PER [%]
MON	2.5	65.5	53.0
S3	580.0	64.9	51.4

Table 16
Computing time, WER, and PER for the translation direction English to French using the two reordering constraints MON and S3. An almost "full" search is carried out.

Reordering constraint	CPU time [sec]	WER [%]	PER [%]
MON	2.2	66.6	56.3
S3	189.1	66.0	54.4

larger. There is only a slight difference in performance between the MON and the S3 reordering constraints on the Hansards task. The computation time is also rather high compared to the Verbmobil task: For the S3 constraint, the average translation time is about 3 minutes per sentence for the translation direction English to French and about 10 minutes per sentence for the translation direction French to English. The following parameter setting is used for the experiment conducted here: $t_{\mathcal{C}} = 5.0$, $t_c = 10.0$, $n_{\mathcal{C}} = 250$, and $t_o = 12$. (The actual parameters are chosen in informal experiments to obtain reasonable CPU times while permitting only a small number of search errors.) No cardinality histogram pruning is carried out. As for the German-to-English translation experiments, word reordering is restricted so that it may not cross punctuation boundaries. The resulting fragment lengths are much larger for the translation direction English to French, and still larger for the translation direction French to English, when compared to the fragment lengths for the translation direction German to English, hence the high CPU times. In an additional experiment for the translation direction French to English and the reordering constraint S3, we find we can speed up the translation time to about 18 seconds per sentence by using the following parameter setting: $t_{\mathcal{C}} = 3.0$, $t_c = 7.5$, $n_{\mathcal{C}} = 20$, $n_c = 400$, and $n_o = 5$. For the resulting hypotheses file, PER increases only slightly, from 51.4% to 51.6%.

Translation examples for the translation direction French to English under the S3 reordering constraint are given in Table 17. The French input sentences show some preprocessing that is carried out beforehand to simplify the translation task (e.g., *des* is transformed into *de les* and *l'est* is transformed into *le est*). The translations produced are rather approximative in some cases, although the general meaning is often preserved.

5. Conclusions

We have presented a DP-based beam search algorithm for the IBM-4 translation model. The approach is based on a DP solution to the TSP, and it gains efficiency by imposing constraints on the allowed word reorderings between source and target language. A data-driven search organization in conjunction with appropriate pruning techniques

Table 17
Example translations for the translation direction French to English using the S3 reordering constraint.

Input	Je crois que cela donne une bonne idée de les principes à retenir et de ce que devraient être nos responsabilités.
S3	I think it is a good idea of the principles and to what should be our responsibility.
Input	Je pense que, indépendamment de notre parti, nous trouvons tous cela inacceptable.
S3	I think, regardless of our party, we find that unacceptable.
Input	Je ai le intention de parler surtout aujourd' hui de les nombreuses améliorations apportées à les programmes de pensions de tous les Canadiens.
S3	I have the intention of speaking today about the many improvements in pensions for all Canadians especially those programs.
Input	Chacun en lui - même est très complexe et le lien entre les deux le est encore davantage de sorte que pour beaucoup la situation présente est confuse.
S3	Each in itself is very complex and the relationship between the two is more so much for the present situation is confused.

is proposed. For the medium-sized Verbmobil task, a sentence can be translated in a few seconds on average, with a small number of search errors and no performance degradation as measured by the word error criterion used.

Word reordering is parameterized using a set of four parameters, in such a way that it can easily be adopted to new translation directions. A finite-state control is added, and its usefulness is demonstrated for the translation direction German to English, in which the word order difference between the two languages is mainly due to the German verb group. Future work might aim at a tighter integration of the IBM-4 model distortion probabilities and the finite-state control; the finite-state control itself may be learned from training data.

The applicability of the algorithm applied in the experiments in this article is not restricted to the IBM translation models or to the simplified translation model used in the description of the algorithm in Section 3. Since the efficiency of the beam search approach is based on restrictions on the allowed coverage vectors $\mathcal{C}$ alone, the approach may be used for different types of translation models as well (e.g., for the multiword-based translation model proposed in Och, Tillmann, and Ney [1999]). On the other hand, since the decoding problem for the IBM-4 translation model is provably NP-complete, as shown in Knight (1999) and Germann et al. (2001), word reordering restrictions as introduced in this article are essential for obtaining an efficient search algorithm that guarantees that a solution close to the optimal one will be found.

Appendix: Quantification of Reordering Restrictions

To quantify the reordering restrictions in Section 3.5, the four non-negative numbers $numskip$, $widthskip$, $nummove$, and $widthmove$ are used ($widthskip$ corresponds to L, $widthmove$ corresponds to R in Section 3.4; here, we use a more intuitive notation). Within the implementation of the DP search, the restrictions are provided to the

algorithm as an input parameter of the following type:

$$S\_numskip\_widthskip\_M\_nummove\_widthmove$$

The meaning of the reordering string is as follows: The two numbers following S that are separated by an underscore describe the way words may be skipped; the two numbers following M that are separated by an underscore describe the way words may be moved during word reordering. The first number after S and M denotes the number of positions that may be skipped or moved, respectively (e.g., for the translation direction German to English [GE in the chart below], one position may be skipped and two positions may be moved). The second number after S and M restricts the distance a word may be skipped or moved, respectively. These "width" parameters restrict the word reordering to take place within a "window" of a certain size, established by the distance between the positions $l_{\min}(\mathcal{C})$ and $r_{\max}(\mathcal{C})$ as defined in Section 3.5. In the notation, either the substring headed by S or that headed by M (or both) may be omitted altogether to indicate that the corresponding reordering is not allowed. Any numerical value in the string may be set to INF, denoting that an arbitrary number of positions may be skipped/moved or that the moving or skipping distance may be arbitrarily large. The following reordering strings are used in this article:

Word reordering string	Description
ϵ (short: **MON**)	The empty string denotes the reordering restriction in which no reordering is allowed.
S_01_04_M_02_10 (short: **GE**)	This string describes the German-to-English word reordering. Up to one word may be skipped for at most 4 positions, and up to two words may be moved up to 10 positions.
S_02_10_M_01_04 (short: **EG**)	This string describes the English-to-German word reordering. Up to two words may be skipped for at most 10 positions and up to one word may be moved for up to 4 positions.
S_03_INF (short: **S3**)	This string describes the IBM-style word reordering given in Section 3.6. Up to three words may be skipped for an unrestricted number of positions. No words may be moved.
S_INF_INF or M_INF_INF (short: **NO**)	These strings denote the word re-ordering without restrictions.

The word reordering strings can be directly used as input parameters to the DP-based search procedure to test different reordering restrictions within a single implementation.

Acknowledgments

This work has been supported as part of the Verbmobil project (contract number 01 IV 601 A) by the German Federal Ministry of Education, Science, Research and Technology and as part of the Eutrans

project (ESPRIT project number 30268) by the European Community. Some of the experiments on the Canadian Hansards task have been carried out by Nicola Ueffing using the existing implementation of the search algorithm (Och, Ueffing, and Ney [2001]). We would like to thank the anonymous reviewers for their detailed comments on an earlier version of this article. Also, we would like to thank Niyu Ge, Scott McCarley, Salim Roukos, Nicola Ueffing, and Todd Ward for their valuable remarks.

References

Al-Onaizan, Yaser, Jan Curin, Michael Jahr, Kevin Knight, John Lafferty, Dan Melamed, Franz-Josef Och, David Purdy, Noah Smith, and David Yarowsky. 1999. Statistical machine translation. Final Report, Johns Hopkins University Summer Workshop (WS 99) on Language Engineering, Center for Language and Speech Processing, Baltimore.

Berger, Adam L., Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, John R. Gillett, John D. Lafferty, Robert L. Mercer, Harry Printz, and Lubos Ures. 1994. The Candide system for machine translation. In *Proceedings of the ARPA Human Language Technology Workshop*, pages 152–157, San Mateo, California, March.

Berger, Adam L., Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, Andrew S. Kehler, and Robert L. Mercer. 1996. Language translation apparatus and method of using context-based translation models. U.S. Patent 5510981.

Brown, Peter F., John Cocke, Vincent J. Della Pietra, Stephen A. Della Pietra, Fred Jelinek, John Lafferty, Robert L. Mercer, and Paul S. Roosin. 1990. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85.

Brown, Peter F., Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Brown, Peter F., Peter V. deSouza, Vincent J. Della Pietra, and Robert L. Mercer. 1992. Class-based *n*-gram models of natural language. *Computational Linguistics*, 18(4):467–479.

García-Varea, Ismael, Francisco Casacuberta, and Hermann Ney. 1998. An iterative DP-based search algorithm for statistical machine translation. In *Proceedings of the Fifth International Conference on Spoken Language Processing (ICSLP 98)*, pages 1135–1139, Sydney, Australia, November.

Germann, Ulrich, Michael Jahr, Kevin Knight, Daniel Marcu, and Kenji Yamada. 2001. Fast decoding and optimal decoding for machine translation. In *Proceedings of the 38th Annual Conference of the Association for Computational Linguistics (ACL 2001)*, pages 228–235, Toulouse, France, July.

Held, Michael and Richard M. Karp. 1962. A dynamic programming approach to sequencing problems. *SIAM*, 10(1):196–210.

Jelinek, Fred. 1976. Speech recognition by statistical methods. *Proceedings of the IEEE*, 64:532–556.

Knight, Kevin. 1999. Decoding complexity in word-replacement translation models. *Computational Linguistics*, 25(4):607–615.

Ney, Hermann, Dieter Mergel, Andreas Noll, and Annedore Paeseler. 1992. Data driven search organization for continuous speech recognition in the SPICOS system. *IEEE Transactions on Signal Processing*, 40(2):272–281.

Ney, Hermann, Sonja Niessen, Franz-Josef Och, Hassan Sawaf, Christoph Tillmann, and Stefan Vogel. 2000. Algorithms for statistical translation of spoken language. *IEEE Transactions on Speech and Audio Processing*, 8(1):24–36.

Niessen, Sonja, Franz-Josef Och, Gregor Leusch, and Hermann Ney. 2000. An evaluation tool for machine translation: Fast evaluation for MT research. In *Proceedings of the Second International Conference on Language Resources and Evaluation*, pages 39–45, Athens, Greece, May.

Niessen, Sonja, Stefan Vogel, Hermann Ney, and Christoph Tillmann. 1998. A DP-based search algorithm for statistical machine translation. In *Proceedings of the 36th Annual Conference of the Association for Computational Linguistics and the 17th International Conference on Computational Linguistics (ACL/COLING 98)*, pages 960–967, Montreal, Canada, August.

Nilsson, Nils J. 1971. *Problem Solving Methods in Artificial Intelligence*. McGraw Hill, New York.

Och, Franz-Josef and Hermann Ney. 2000. A comparison of alignment models for statistical machine translation. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING 2000)*, pages 1086–1090, Saarbrücken, Germany, July–August.

Och, Franz-Josef, Christoph Tillmann, and Hermann Ney. 1999. Improved alignment models for statistical machine translation. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC 99)*, pages 20–28, College Park, Maryland, June.

Och, Franz-Josef, Nicola Ueffing, and Hermann Ney. 2001. An efficient (A)* search algorithm for statistical machine translation. In *Proceedings of the Data-Driven Machine Translation Workshop, 39th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 55–62, Toulouse, France, July.

Tillmann, Christoph. 2001. *Word Re-ordering and Dynamic Programming Based Search Algorithm for Statistical Machine Translation*. Ph.D. thesis, University of Technology, Aachen, Germany.

Tillmann, Christoph and Hermann Ney. 2000. Word re-ordering and DP-based search in statistical machine translation. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING 2000)*, pages 850–856, Saarbrücken, Germany, July–August.

Tillmann, Christoph, Stefan Vogel, Hermann Ney, and Alex Zubiaga. 1997. A DP-based search using monotone alignments in statistical translation. In *Proceedings of the 35th Annual Conference of the Association for Computational Linguistics (ACL 97)*, pages 289–296, Madrid, July.

Tillmann, Christoph, Stefan Vogel, Hermann Ney, Alex Zubiaga, and Hassan Sawaf. 1997. Accelerated DP-based search for statistical translation. In *Proceedings of the Fifth European Conference on Speech Communication and Technology (Eurospeech 97)*, pages 2667–2670, Rhodos, Greece, September.

Wahlster, Wolfgang. 2000. *Verbmobil: Foundations of Speech-to-Speech Translation*. Springer Verlag, Berlin.

Wang, Ye-Yi and Alex Waibel. 1997. Decoding algorithm in statistical translation. In *Proceedings of the 35th Annual Conference of the Association for Computational Linguistics (ACL 97)*, pages 366–372, Madrid, July.

Wang, Ye-Yi and Alex Waibel. 1998. Fast decoding for statistical machine translation. In *Proceedings of the Fifth International Conference on Spoken Language Processing (ICSLP 98)*, pages 2775–2778, Sydney, Australia, December.

Wu, Dekai. 1996. A polynomial-time algorithm for statistical machine translation. In *Proceedings of the 34th Annual Conference of the Association for Computational Linguistics (ACL 96)*, pages 152–158, Santa Cruz, California, June.