METIS-II: The German to English MT System

Michael Carl
Institut für Angewandte Informationsforschung
66111 Saarbrücken, Germany
carl@iai.uni-sb.de

Abstract

Within the METIS-II project[1], we have implemented a machine translation system which uses transfer and expander rules to build an AND/OR graph of partial translation hypotheses and a statistical ranker to find the best path through the graph. The paper gives an overview of the architecture and an evaluation of the system for several languages.

1 Introduction

Recent machine translation techniques integrate rule-based knowledge and statistics: (Groves and Way, 2006) integrate rule-induced chunk translations with a statistical decoder; for (Richardson et al., 2001; Gamon et al., 2002), or (Ringger et al., 2004), linguistic rules describe what possible transformations a parse tree can undergo, but statistics decides under which conditions a particular rule is applied and (Quirk and Menezes, 2006) decide the combination of derivation trees by statistical means.

This paper outlines an MT architecture which uses rule-based devices to generate sets of partial translation hypotheses and a statistical *Ranker* to evaluate and retrieve the best hypotheses in their context.

The rule-based device generates an acyclic AND/OR graph which allows for compact representation of many different translations while the *Ranker* is a beam search algorithm which tries to find most likely paths in the AND/OR graph.

Unlike a usual statistical decoder (Germann et al., 2001; Koehn, 2004), our *Ranker* traverses the search graph to grade alternative paths and outputs a list of the n-best translations. The *Ranker* itself does not modify the graph. It does not permute chunks or items and it does not generate additional paths which are not already contained in the graph. The construction of the search graph and its evaluation are thus separated as two distinct tasks.

Starting from a SL sentence, the graph is incrementally constructed in three rule-based steps. The graph is then traversed and translations are ranked. Finally word tokens are generated for the n-best translations. This paper gives an overview on the steps 1 to 4.

1. The *Analyser* lemmatises and morphologically analyses the SL sentence. It produces a (flat) grammatical analysis of the sentence, detecting phrases and clauses and potential subject candidates. An outline of the *Analyser* is given in section 2.
2. During *Dictionary Lookup* analyzed SL sentence are matched on the transfer dictionary and TL equivalences are retrieved. We will give a review of the essential features in section 3.
3. The *Expander* inserts, deletes, moves and permutes items or chunks according to TL syntax. It is called *Expander* because it expands the search space through the word- and phrase translations retrieved from the lexicon. The *Expander* relies on a rule-based device. We give some examples in section 4
4. The *Ranker* relies on a beam search algorithm that iteratively traverses the graph and computes the most likely translations in a log-linear fashion (Och and Ney, 2002). The *Ranker* is explained in section 5.
5. A *Token Generator* generates surface wordforms from the lemmas and PoS tags. The Token Generator has been described in (Carl et al., 2005) and will be omitted here.

2 The Analyser

The Analyser reads the SL sentence and produces a flat sequence of feature bundles which contain chunking and topological information of the sentence (Müller, 2004). For instance, from the German SL sentence (1a) the representation (1b) would be generated.

Among other things, the analysis in (1b) comprises of a unique word number `wnrr`, the lemma `lu` and part-of-speech `c`, `sc` of the word, as well as morphological and syntactic information. It also contain chunking and topological information.

[1] `http://www.ilsp.gr/metis2/`

1 a Das Haus wurde von Hans gekauft (The house was purchased by Hans)
The house was from Hans purchased

```
     {lu=das,      wnrr=1,  c=w,sc=art,       phr=np;subjF,     cls=hs;vf}
    ,{lu=haus,     wnrr=2,  c=noun,           phr=np;subj,      cls=hs;vf}
    ,{lu=werden,   wnra=3,  c=verb,vt=fiv,    phr=vg_fiv,       cls=hs;lk}
1b  ,{lu=von,      wnrr=4,  c=w,sc=p,         phr=np;nosubjF,   cls=hs;mf}
    ,{lu=Hans,     wnrr=5,  c=noun,           phr=np;nosubj,    cls=hs;mf}
    ,{lu=kaufen,   wnra=6,  c=verb,vt=ptc2,   phr=vg_ptc,       cls=hs;rk}
    .
```

A German sentence (1a) and the output of the *Analyser* (1b). Among others, the following features are used:

features for words:		features for phrases:		features for clauses/fields:	
lemma:	`lu=<lemma>`	noun phrase:	`phr=np`	main clause:	`cls=hs`
noun:	`c=noun`	finite verb:	`phr=vg_fiv`	Vorfeld:	`cls=vf`
verbs:	`c=verb,vt=ptc2|fiv`	participle:	`phr=vg_ptc`	Mittelfeld:	`cls=mf`
preposition:	`c=w,sc=p`	subject:	`phr=subjF`	linke Klammer:	`cls=lk`
article:	`c=w,sc=art`	not subject:	`phr=nosubjF`	rechte Klammer:	`cls=rk`

1c: Output of *dictionary lookup*:

```
 {lu=das,wnrr=1,c=w,sc=art, ... }
    @{c=art,n=146471}@{lu=the,c=AT0}.
    .
,{lu=Haus,wnrr=2,c=noun, ...}
    @{c=noun,n=268244}@{lu=company,c=NN1}.
    ,{c=noun,n=268246}@{lu=home,c=NN1}.
    ,{c=noun,n=268247}@{lu=house,c=NN1}.
    ,{c=noun,n=268249}@{lu=site,c=NN1}.
    .
,{lu=werden,wnrr=3,c=verb,vt=fiv, ...}
    @{c=verb,n=604071}@{lu=be,c=VBD} .
    ,{c=verb,n=604076}@{lu=will,c=VM0} .
    .
,{lu=von,wnrr=4,c=w,sc=p, ...}
    @{c=w,sc=p,n=587268}@{lu=by,c=PRP}.
    ,{c=w,sc=p,n=587269}@{lu=from,c=PRP}.
    ,{c=w,sc=p,n=587270}@{lu=of,c=PRF}.
    .
,{lu=Hans,wnrr=5,c=noun, ...}
    @{c=noun,n=265524}@{lu=hans,c=NP0}.
    .
,{lu=kaufen,wnrr=6,c=verb,vt=ptc2, ...}
    @{c=verb,n=307263}@{lu=buy,c=VVN}.
    ,{c=verb,n=307265}@{lu=purchase,c=VVN}.
.   .
```

The SL analysis of example 1b is transformed into an AND/OR graph enriched with retrieved TOs.

1d: Output of the *Expander*:

```
 {lu=das,wnrr=1,c=w,sc=art, ... }
    @{c=art,n=146471}@{lu=the,c=AT0}.
    .
,{lu=Haus,wnrr=2,c=noun, ...}
    @{c=noun,n=268244}@{lu=company,c=NN1}.
    ,{c=noun,n=268246}@{lu=home,c=NN1}.
    ,{c=noun,n=268247}@{lu=house,c=NN1}.
    ,{c=noun,n=268249}@{lu=site,c=NN1}.
    .
,{lu=werden,wnrr=3,c=verb,vt=fiv, ...}
    @{c=verb,n=604071}@{lu=be,c=VBD} .
    ,{c=verb,n=604076}@{lu=will,c=VM0} .
    .
,{lu=kaufen,wnrr=6,c=verb,vt=ptc2, ...}
    @{c=verb,n=307263}@{lu=buy,c=VVN}.
    ,{c=verb,n=307265}@{lu=purchase,c=VVN}.
    .
,{lu=von,wnrr=4,c=w,sc=p, ...}
    @{c=w,sc=p,n=587268}@{lu=by,c=PRP}.
    ,{c=w,sc=p,n=587269}@{lu=from,c=PRP}.
    ,{c=w,sc=p,n=587270}@{lu=of,c=PRF}.
    .
,{lu=Hans,wnrr=5,c=noun, ...}
    @{c=noun,n=265524}@{lu=hans,c=NP0}.
.   .
```

The *Expander* moves the participle (kaufen) behind the finite verb (werden).

```
<s id=1-0 lp="-9.227912">
the     AT0     146471
company NN1     268244
was     VBD     604071 PermFinVerb_hs
bought  VVN     307263 PermFinVerb_hs
by      PRP     587268 PermFinVerb_hs
hans    NP0     265524 PermFinVerb_hs
.       PUN     367491
</s>
```

```
<s id=1-1 lp="-9.682535">
the     AT0     146471
house   NN1     268247
was     VBD     604071 PermFinVerb_hs
purchased VVN   307265 PermFinVerb_hs
by      PRP     587268 PermFinVerb_hs
hans    NP0     265524 PermFinVerb_hs
.       PUN     367491
</s>
```

1e: The two best scored translations (`lp` feature) after *ranking* at the output of the Token Generator.

The parser produces a linguistically motivated, flat macro structure of German sentences, as coded by the `cls` feature.

3 Dictionary Lookup

The input for *Dictionary Lookup* are annotated SL words as generated from the *Analyser* in example 1b. *Dictionary Lookup* retrieves target language equivalences with two functions:

- regroup words of a sentence into coherent meaning entities according to the entries the dictionary. The words may be distributed over several parts in the sentence according to principles as below.
- retrieve all possible groups in a sentence (perhaps overlapping and/or embedded) and retrieve all translation options for each group.

This process returns a structure as shown in examples (1c) and (2c). SL nodes as in (1b) are transformed into nodes of an acyclic AND/OR graph which consists of translation units (TUs). A TU is a set of words of the SL sentence for which one or more translation options (TOs) are retrieved from the lexicon. Each TO is — in turn — a flat tree with lexical information on the leaves.

2a: Dictionary entry with two TOs:
vor die Hunde gehen ↔ (go to the dogs | be buggered)

2b: SL sentence containing idiomatic expression:
Das **geht**, solange es Frauen gibt, nie **vor die Hunde**.

2c: Representation of discontinuous match:

```
,{lu=gehen|...|vor|der|hund,wnrr=2;10;11;12, ...}
  @{c=verb,n=13}@{lu=go,c=VVN}
                 ,{lu=to,c=TO0}
                 ,{lu=the,c=AT0},{lu=dog,c=NN2}.
  ,{c=verb,n=14}@{lu=be,c=VBZ},{lu=bugger,c=VVN}.
  .
```

Example (2c) shows a discontinuous TU with two TOs. This TU is generated when matching the dictionary entry (2a) on the sentence (2b). Notice that emphasized words in (2b) are matched even though their order has changed with respect to the lexicon entry (2a). The feature `wnrr` enumerates the consumed word positions of the match. The feature `n` gives the number of the dictionary entry.

3.1 Discontinuous matching

The complexity for matching discontinuous phrases is much higher than for matching continuous phrases. Matching a discontinuous phrase of length m on a sentence of length n may lead to a huge number of retrieved entries in the order of $O\begin{pmatrix} n \\ m \end{pmatrix}$, while for continuous phrases there is a maximum of $(n - m + 1)$ matches. Thus, there are more than 3000 possible ways to match a discontinuous phrase of 5 words on a 15-word sentence while a continuous phrase may lead to only 11 possible matches.

In our current implementation, we only allow discontinuous matches for verbal and nominal entries. All other types of dictionary entries, such as adjectives, adverbs, prepositions, idioms etc. are not eligible for discontinuous matching. In (Carl and Rascu, 2006) we have described various strategies to reject matched entries if they don't obey a predefined set of criteria.

For nominal entries, the head of the term e.g. *Ozonschicht* in (3) can be modified in the matched sentence, for instance by adjectives as in example (4). While we would like to validate the entry despite the intervening adjective *arktischen*, we want to reject the entry if the words co-occur 'by accident' in the same sentence and are actually unrelated. This would be the case if the words occurred in different noun phrases.

(3) Abbau der Ozonschicht ↔ ozone depletion
(4) **Abbau der** *arktischen* **Ozonschicht**

For verbal entries, various permutations of the words are possible, according to whether the entry occurs in a subordinate clause or in a main clause. These criteria are further developed in (Anastasiou and Čulo, 2007) making use of the German topological fields.

Verbal dictionary entries which consist of two or more parts usually have a nominal part and a verbal part. The nominal part (NP) may occur in the 'Mittelfeld' (`mf`) or in the 'Vorfeld' (`vf`) while the verbal part (V) is the left or right Klammer. It is, thus, not possible for such entries that the nominal part is distributed in the Vorfeld and Mittelfeld.

NP	V	example
`mf`	`lk`	Hans <u>schiebt</u> uns **den schwarzen Peter** <u>**zu**</u>.
`mf`	`rk`	Hans will uns **den schwarzen Peter** <u>zuschieben</u>.
`vf`	`lk`	**Den schwarzen Peter** <u>schiebt</u> uns Hans <u>zu</u>.
`vf`	`rk`	**Den schwarzen Peter** will uns Hans <u>zuschieben</u>.

In the above examples, the nominal part is marked in **bold** while the Verbal part is <u>underlined</u>

The same behavior is observed for separable prefixes and reflexive verbs as in examples (5) and (6).

5 Hans **lehnt** das Angebot **ab**.
≈ *Hans rejects the offer* `prefix`.

6 dass Hans **sich** immer **beeilt**.
≈ *that Hans himself always hurries up.*

3.2 Lexical Overgeneration

Similar to many statistical MT systems, to account for a maximum number of different contexts, the dictionary over-generates translation hypotheses which are then filtered and graded by the *Ranker* in the context of the generated sentence.

Overgeneration accounts for various phenomena such as lexical semantic ambiguities (example 7), negation and the insertion of the English 'do-support' example (8), semantically bound prepositions (9), and others.

7 Bank ↔ bank | bench
8 nicht ↔ do | not not
9 auf ↔ on | in | up | onto | upon | ...
10 stark ↔ strong | heavy | good | bad | high | ...

A specific problem of lexical ambiguities are 'intensifiers' or 'magnifiers'. For instance, the word "stark" (basically 'strong') can translate into many different adjectives depending on the context and the noun it modifies. Examples (11a-f) provide contexts in which **stark** is translated differently.

11a This is a **strong** man
11b It has been his **best** play
11c Paul has **high** temperature
11d The car was **badly** damaged
11e John is a **heavy** smoker
11f There was a **big** demand

The choice of the 'correct' TO is left for the *Ranker* to decide. Thus, whether 'not' or 'do not' are appropriate translation options for 'nicht' will be decided in the context of the generated target language sentence and maybe depends on whether an infinite verb follows 'not'.

4 The Expander

The *Expander* adds further translation hypotheses to the AND/OR graph. It is a rule-based device, which takes as its input the output of the *Dictionary Lookup*. The *Expander* essentially inserts, deletes and moves translation units in the graph. It also produces alternative partial translations.

For instance, parts of the German verbal group appear in the "linke Klammer" and other parts in the "rechte Klammer". In main clauses, the "Mittelfeld" intervenes between these two parts. For English these parts have to be re-joined. The *Expander* rule `ReorderFinVerb_hs` moves the translation of the participle "gekauft" in example 1c in a main clause (`cls=hs`) behind the finite verb "wurde".

The rule maps on a pattern of TUs in a main clause starting with the finite verb (`phr=vg_fiv`) and followed by an optional infinitive verb (`phr=vg_inf`) and a participle (`phr=vg_ptc`). Between the finite verb and an optional infinitive verb can be a number of 'non-verbs'. All nodes need to occur in the same main clause (`cls=hs`). The existential quantifier '`e`' requires that at least one reading of the TU must be compatible with the test, while the universal quantifier '`a`' requires all readings of the TU to be subsumed by the test. The finite verb, the infinite verb and the participle are marked by the marker `V`, `I` and `P` respectively. The action part of the rule — following the colon — moves the marked nodes into the desired word order, so that the verbs are grouped together in their right order.

```
ReorderFinVerb_hs =
    Ve{cls=hs}e{phr=vg_fiv},
    *e{cls=hs}a{phr~=vg_ptc;vg_inf},
   ^Ie{cls=hs}e{phr=vg_inf},
    Pe{cls=hs}e{phr=vg_ptc}
:   p(move=V->VIP).
```

While this operation deterministically moves the nodes in the graph, the formalism also allows operations to produce alternative permutations of sequences of TUs. The rule transforms the representation in (1c) into the graph in example (1d) which contains the (correct) word order "The house was purchased by Hans".

The same type of rule also applies for the adjustment of composed tenses and modal sentences in examples (12) and (13) respectively.

12a Hans **hat** das Haus **gekauft**.
≈ *Hans* **has** *the house* **purchased**.
12b Hans **has purchased** the house.

13a Hans **will** ein Haus **kaufen**.
≈ *Hans* **wants to** *a house* **purchase**.
13b Hans **wants to purchase** a house.

After reordering the verbal group, the *Expander* adjusts the subject. In contrast to English, German allows one phrasal element to precede the finite verb, which may or may not be the subject of the sentence. In some cases we know the subject from the German analysis. In these cases we can deterministically move the subject to its correct position as in example (14).

14a Gestern kam **Hans** in das Büro.
Yesterday came **Hans** *into the office.*
14b Yesterday **Hans** came into the office.

In other cases the German analysis provides several subject candidates. We generate a translation hypothesis for each possible permutation and let the *Ranker* decide which is the more likely subject.

15 a Die Katze trinkt die Milch.
The cat drinks the milk.
15b *(the milk drink the cat.*
| the cat drink the milk.)

5 The ranker

The *Ranker* works similar to a decoder as used in statistical machine translation. Och and Ney (2002) extend the noisy channel model of Brown et al. (1993) by adding weighing coefficients with feature functions and combining them in a log linear fashion. As a statistical decoder, the *Ranker* is a search procedure which seeks to find the target sentence $\hat{e}$ with the highest probability:

$$\hat{e} = \texttt{argmax} \sum_{m}^{M} w_m h_m(\cdot)$$

where h_m is a feature function and w_m is a weighing coefficient. The feature functions h_m can be independent and trained on separate data while the weighing coefficients w_m are used to tune the system.

The *Ranker* is a beam-search algorithm which traverses the AND/OR graph in a breadth first manner. At each step the nodes are weighted by the feature functions and all expanded sentence prefixes are stored in the beam until its maximum width (currently 1000) is reached. From there on only the highest weighted sentence are further expanded. We have experimented with various feature functions to weigh the nodes. We describe their settings in this section. An evaluation is given in section 6.

Output of the *Ranker* are the n-best graded translation paths through the graph. For the example in example (1d) the two best translations (word forms) are shown in (1e). The output also indicates the resources used to generate the translations, among other things, the number of the translation entries and the *Expander* rules.

5.1 Language model

We have tested various language models, all of them making use of the BNC[2] and all are generated using the CMU language modelling toolkit[3]. The BNC is a tagged collection of texts making use of the CLAWS5 tag set which comprises roughly 70 different tags.

[2]The British National Corpus (BNC) consists of more than 100 million words in more than 6 million sentences `http://www.natcorp.ox.ac.uk/`

[3]which can be downloaded from `http://www.speech.cs.cmu.edu/SLM_info.html`

The CMU language modelling toolkit generates n-gram language models (LMs) from tokenised texts. These LMs are then used as a feature function of the *Ranker*.

The CMU toolkit generates a vocabulary of up to 65535 words which occur most frequently in the training material. It supports open LMs which account for unknown words and closed LMs which assume all tokens to be known in the training material. A LM made up of CLAWS5 tags would be a closed language model since there are less than 70 different tags in this tag set and all tags are likely to occur in the training material.

The closed LMs assume that only items in the training data will occur in the test data, while open LMs save some of the probability mass for (unknown) words in the test data which did not occur in the training set. These words will be mapped on the item UNK.

To find suited LMs for our application, we have experimented with the following parameters:

- number of sentences: 100K, 1M and 6M
- different ways of preprocessing the BNC:
  - open token-based LM
  - closed mixed lemma-tag LM
  - closed mixed token-tag LM
  - orthogonal lemma-tag LM
- 3 and 4-gram token LMs and 4 to 7-gram PoS-tag LMs

5.2 Open token-based LM

The open token-based LM assumes (lower-cased) surface word-forms as the input to the *Ranker*. This requires token generation to take place on the output of the *Expander* previous to the *Ranker*.

5.3 Closed mixed token-tag model

The vocabulary of the closed mixed token-tag model consists of word tokens (thus the un-lemmatised BNC) but unknown words will be mapped on their CLAWS5 tag. Assume the reference set contains the sentence "John likes strawberries" but "strawberries" does not occur in the vocabulary of the 60000 most frequent tokens. Instead of letting the CMU toolkit map "strawberries" on the tag UNK, we would replace it by the CLAWS5 tag <NN2>. In this way we generate a closed model with a finite number of different tokens (the 69 CLAWS5 tags plus the 60000 most frequent tokens in the reference set). Analogically at runtime, previous to the *Ranker*, we would generate tokens from the lemmas. The *Ranker* would consult the LM's vocabulary and map any unknown word on the CLAWS5 tag. (Manning and Schütze, 1999) suggest to map unknown words on two tags: one for numbers and all other unknown words on one other tag. With our strategy unknown tokens are mapped on many more tags. In this way we can make sure that any sentence contains only known tokens.

5.4 Closed mixed lemma-tag model

The closed mixed lemma-tag model works essentially similar to the Closed mixed token-tag model but makes use of the 60000 most frequent lemmas. Thus the above reference sentence would be lemmatised into "John like strawberry", and - given "strawberry" is not among the 60000 most frequent lemmas in the training corpus - it would be preprocessed into "John like <NN2>". At runtime, lemmas would be transformed into word tokens on the output of the *Ranker*.

5.5 Orthogonal lemma-tag model

In the orthogonal lemma-tag model we compute two LMs: a CLAWS5 tag n-gram model (LM_tag) and a lemma m-gram model (LM_lem). Following (Manning and Schütze, 1999)[p.202-203] we compute in addition a cooccurrence weight of the lemmas and their tag according to Laplace's law with the following equation:

$$w(lem, tag) = \frac{NL}{NL + C(lem)} * (C(lem, tag) + 1)$$

Where NL is half the number of different tags (i.e. 69/2), $C(lem)$ is the number of occurrences of the token in the BNC and $C(lem, tag)$ is the number of cooccurrences of a lemma and a tag. For instance the lemma "tape-recorder" has 103 occurrences in the BNC. The weights for "tape-recorder" given their tag are shown in the table (16), where <*> accounts for the possibility that a lemma/tag occurs in the test translations but did not occur in the training set.

lemma	tag	#	w(lem, tag)
tape-recorder	AJ0	3	1.00
tape-recorder	NN1	87	22.08
tape-recorder	NN2	13	3.51
tape-recorder	<*>	0	0.25

Table 16: Example for cooccurrence weights

The orthogonal lemma-tag model consists thus of three feature functions which are computed for each node in the beam.

6 Evaluation

The quality of the translations depends on modules of the system and their parameters:

1. Precision and coverage of the German SL analysis
2. Contents of dictionary and the matching performance
3. Quality of the expander rules
4. Feature functions and weights used in the ranker
5. Precision token generation

In a concise evaluation setting, to have a clear picture of the overall performance of the system, each module should be tested on its own and in relation to the other modules. However, within the framework of this project, only parts of the modules and their interaction could be evaluated.

Thus, precision and coverage of the German SL analysis results mainly from commercial applications[4] and has been omitted here. Token generation is essentially deterministic and has been evaluated in previous work (Carl et al., 2005). In a large testing scenario token generation achieves more than 99% precision.

An evaluation of the dictionary lookup strategies is given in (Carl and Rascu, 2006) and a further enhancement of the method is described in (Anastasiou and Čulo, 2007). On a set of roughly 60 sentences, it is found that matching of continuous and discontinuous verbal entries reaches precision and recall of almost 100% and 90% respectively.

Until now we did not develop a method for an independent evaluation of the *Expander* and the *Ranker*. Such methods would be highly desirable and reveal to what extent these modules generate an optimum output with a given input. Lacking such methods, we have evaluated the system as a whole from source language to target language, using BLEU and NIST measures.

6.1 First experiment

In a first experiment we have compared the four LMs of section 5. All else being equal, the orthogonal lemma-tag model as in section 5.5 consistently showed the best results when varying weights for the feature functions so that we gave up further experiments with the open token and the closed token-tag and lemma-tag models of sections 5.2, 5.3 and 5.4. All other evaluation experiments are thus based on the the orthogonal model as in section 5.5.

We have tested the system on four languages (Dutch, German, Greek and Spanish) into English based on 50 sentences for each of the languages. A representation similar to the dictionary-lookup output (i.e. as in example (1c)) was provided by our Dutch, Greek and Spanish partners for this experiment, together with three reference translations. A separate set of *Expander* rules was developed for each source language, consisting of five rules for

[4] http://www.iai-sb.de

Greek up to approx. 20 rules for German.

Language	BLEU	NIST
Dutch	0.4034	6.4489
Spanish	0.3701	5.7304
Greek	0.2138	5.1220
German	0.1671	3.9197

Table 17: Results of first Experiment

The ranker used the orthogonal lemma-tag model with the 6M-n3[5] lemma- and the 1M-n4 tag models. The results are given in the table (17).

The differences in BLEU and NIST scores for the four languages is — besides their similarity to English and the length of the test sentences — also due to the quality, coverage and ambiguity of the lexicon, and thus its ambiguity. The table (18) shows that conditions are worst for German: the German test set has on average the longest sentences (13.2 words) and the highest lexical ambiguity. There are on average 3.6 TOs per word. Note that this is, compared to a statistical MT system, very little where a word can have up to 100 or more translations. However, dictionaries for Greek, Dutch and Spanish produce on average less than 2 TOs per word. With more than 1.3 tokens per TO, the German lexicon has also the longest translations.

language	Tok/TO	TO/TU	length
Greek	1.0208	1.9959	9.5
Dutch	1.1258	1.9796	10.8
Spanish	1.1930	1.9506	7.8
German	1.3282	3.6352	13.2

Table 18 : Properties of dictionary output and sentence length for the first experiment

6.2 Second experiment

Another set of evaluations was conducted one a German test set of 200 sentences. As in the first test, there were three reference translations for each test sentence. The sentences were selected (and partially constructed) so that they cover a range of known translation problems including:

- lexical translation problems:
  separable prefixes, fixed verb constructions, degree of adjectives and adverbs, lexical ambiguities, and others
- syntactic translation problems:
  nominalisation, determination, word order, different complementation, relative clauses, tense/aspect, head switching, prepositions, category change, and others

[5] a 3-gram model trained on all 6 Million sentences of the BNC

In a first test suit we took the expander rules from the first experiment and varied feature weights between 0.01 and 10, using lemma LMs with 3 and 4-grams and the tag LMs with 4, 5, 6, and 7-grams. With the best combination of language models and weights we obtained BLEU value of 0.1861.[6]

test	NIST	BLEU	token model	tag model
1	5.4801	0.1861	6M-n3	6M-n3
2	5.3193	0.2231	5M-n3	5M-n7
B	6.3644	0.3133	—	—

Table 19: Results of 200 test translations

In a second test suit, we further developed and refined some expander rules for handling adverbs and negation particles, such as ‘never’, ‘usually’ extraposition of prenominal adjectives (e.g. “der vom Baum gefallene Apfel” would become “The apple fallen from the tree”) and “um . . . zu” constructions. We used the 50 sentences from the first experiment as a development set and tested on a set of 200 sentences. The BLEU score increased to 0.2231. As can be seen in the table (19), NIST values decreased slightly.

The public version of Systran (Babelfish), however, largely outperforms our efforts. Their results on the same test set can be seen in the last line in table (19).

7 Related work and outlook

We have described a machine translation system within the METIS-II project which joins a transfer dictionary with simple reordering rules and a statistical ranker. A general overview of the METIS-II project is given in (Dirix et al., 2005) and in (Vandeghinste et al., 2006). More detailed descriptions of the various realisations of METIS-II are in (Badia et al., 2005; Markantonatou et al., 2006; Vandeghinste et al., 2007).

An approach related to METIS-II has been suggested by (Carbonell et al., 2006). Like METIS-II their so-called “Context-based machine translation” also makes use of a transfer dictionary and a target language corpus. The dictionary provides basic word (and phrase) translations which are used to retrieve chunks from the target language corpus. From the best sequence of overlapping chunks the translation is generated.

As in so-called “generation-heavy” translation (Habash, 2004), our expander rules tackle some of

[6] On a 1GB/2.8GHz, single core Linux machine it takes less than 4 minutes to translate the 200 sentences. Most of the time is spent for SL *Analyser* and loading LMs into the *Ranker*. *Expander*, *Dictionary Lookup* and *Token Generation* only needs a small fraction of the total time.

the translation divergences thereby producing numerous partial translation hypotheses. This "symbolic overgeneration" is then constrained by a statistical ranker making use of several statistical feature functions. A similar idea for generation was suggested by (Langkilde and Knight, 1998) who use 2-gram language models to find the best path in a word lattice. Recently, the LOGON-project (Oepen et al., 2007) use statistical feature functions to select best rule-induced structures at various stages during processing. Basically, the common idea of these approaches would be to use statistics for closing "knowledge gaps" during processing.

The core idea of our work is also similar to (Brown and Frederking, 1995) who use a statistical English Language Model (ELM) to select between alternate partial translations produced by three symbolic MT system from the PANGLOSS Mark III system. In contrast to their approach we build a search graph with flat reordering rules.

In the future we plan to further enhance the various modules of our METIS implementation. In particular revise and add more *Expander* rules so as to caputre as yet unprocessed translation divergences. We also plan to add further feature functions, to take into account lexical weights (which may be trained on parallel texts). We also plan to evaluate various traces from the rules (i.e. traces from *Dictionary Lookup* and *Expander*) that fired during the construction of the AND/OR graph. The traces would constitute separate feature functions which can be taken into account by the *Ranker*.

References

Dimitra Anastasiou and Oliver Čulo. 2007. Using Topological Information for Detecting Idiomatic Verb Phrases in German. In *presented at PALC Conference*, Lodz, Poland.

Toni Badia, Gemma Boleda, Maite Melero, and Antonio Oliver. 2005. An n-gram approach to exploiting monolingual corpus for MT. In *Proceedings of the Example-Based Machine Translation Workshop held in conjunction with the 10th Machine Translation Summit*, pages 1–8, Phuket, Tailand.

Ralf Brown and Robert Frederking. 1995. Applying statistical English language modelling to symbolic machine translation. In *Proceedings of the 6th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI)*, pages 221–239, Leuven, Belgium.

Jaime Carbonell, Steve Klein, David Miller, Michael Steinbaum, Tomer Grassiany, and Jochen Frei. 2006. Context-based machine translation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas*, pages 19–28, Cambridge, Massachusetts.

Michael Carl and Ecaterina Rascu. 2006. A dictionary lookup strategy for translating discontinuous phrases. In *Proceedings of the 11th EAMT Conference*, pages 49–58, Oslo, Norway.

Michael Carl, Paul Schmidt, and Jörg Schütz. 2005. Reversible Template-based Shake & Bake Generation. In *In Proceedings of the Example-Based Machine Translation Workshop held in conjunction with the 10th Machine Translation Summit*, pages 17–26, Phuket, Tailand.

Peter Dirix, Ineke Schuurman, and Vincent Vandeghinste. 2005. Metis II: Example-based machine translation using monolingual corpora - System description. In *Proceedings of the Example-Based Machine Translation Workshop held in conjunction with the 10th Machine Translation Summit*, pages 43–50, Phuket, Tailand.

Michael Gamon, Eric Ringger, Simon Corston-Oliver, and Robert Moore. 2002. Machine-learned contexts for linguistic operations in German sentence realization. In *Proceedings of the 40th annual ACL Conference*, pages 25–32, Philadelphia.

Ulrich Germann, Michael Jahr, Kevin Knight, Daniel Marcu, and Kenji Yamada. 2001. Fast Decoding and Optimal Decoding for Machine Translation. In *Proceedings of the 39th ACL and 10th Conference of the European Chapter*, pages 228–235, Toulouse, France.

Declan Groves and Andy Way. 2006. Hybrid Data Driven Models of MT. *Machine Translation*, 19.

Nizar Habash. 2004. The use of a structural n-gram language model in generation-heavy hybrid machine translation. In *Proceeding 3rd International Conference on Natural Language Generation (INLG '04), volume 3123 of LNAI, Springer*, pages 61–69.

Philipp Koehn. 2004. Pharaoh: A beam search decoder for phrase-based statistical machine translation models. In *AMTA*, pages 115–124.

Irene Langkilde and Kevin Knight. 1998. The Practical Value of n-grams in generation. In *Proceedings of the 9th International Natural Language Workshop (INLG '98)*, Niagara-on-the-Lake, Ontario.

Christopher Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. MIT, Cambridge, Massachusetts.

Stella Markantonatou, S Sofianopoulos, V. Spilioti, G. Tambouratzis, M. Vassiliou, and O. Yannoutsou. 2006. Using Patterns for Machine Translation. In *Proceedings of the 11th EAMT conference*, pages 239–246, Oslo, Norway.

Frank Henrik Müller, 2004. *Stylebook for the Tübingen Partially Parsed Corpus of Written German (TüPP-D/Z)*. http://www.sfb441.uni-tuebingen.de/a1/pub.html.

Franz J. Och and Hermann Ney. 2002. Discriminative Training and Maximum Entropy Models for Statistical Machine Translation. In *Proceedings of the 40th annual ACL Conference*, pages 295–302, Philadelphia, PA.

Stephan Oepen, Erik Velldal, Jan Tore Lonning, Paul Meurer, Victoria Rosen, and Dan Flickinger. 2007. Towards Hybrid Quality-Oriented Machine Translation – On Linguistics and Probabilities in MT. In *Proceedings of the 11th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI)*, Skövde, Sweden.

Christopher Quirk and Arul Menezes. 2006. Dependency Treelet Translation: The convergence of statistical and example-based machine translation? *Machine Translation*, 20(1).

Stephen Richardson, William Dolan, Arul Menezes, and Jessie Pinkham. 2001. Achieving commercial-quality translation with example-based methods. In *Proceedings of Machine Translation Summit VIII*, pages 293–298, Santiago de Compostela, Spain.

Eric Ringger, Michael Gamon, Robert C. Moore, David Rojas, Martine Smets, and Simon Corston-Oliver. 2004. Lexically Informed Statistical Models of Constituent Structure for Ordering in Sentence Realization. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING)*, pages 673–680, Geneva.

Vincent Vandeghinste, Ineke Schuurman, Michael Carl, Stella Markantonatou, and Toni Badia. 2006. METIS-II: Machine Translation for Low Resource Languages. In *Proceedings of the 5th international conference on Language Resources and Evaluation (LREC)*, pages 24–26, Genoa, Italy.

Vincent Vandeghinste, Peter Dirix, and Ineke Schuurman. 2007. The effect of a few rules on a data-driven MT system. In *Proceedings of METIS workshop*, Leuven, Belgium.