

METIS-II: Example-based machine translation using monolingual corpora - System description

Peter Dirix, Ineke Schuurman, and Vincent Vandeghinste

Centre for Computational Linguistics

Katholieke Universiteit Leuven

Maria Theresiastraat 21

B-3000 Leuven

Belgium

firstname.lastname@ccl.kuleuven.be

Abstract

The METIS-II project¹ is an example-based machine translation system, making use of minimal resources and tools for both source and target language, making use of a target-language (TL) corpus, but not of any parallel corpora.

In the current paper, we discuss the view of our team on the general philosophy and outline of the METIS-II system.

1 Introduction: Background of METIS-II

The METIS-II project is an example-based machine translation project, which in principle does not make use of parallel corpora. As most other known example-based machine translation (and statistical) systems make use of parallel corpora or bitexts, our system is a new approach towards the automated translation problem (Dologlou et al., 2003), although e.g. Grefenstette (1999) made use of the world wide web in combination with a bilingual lexicon to translate compounds from Spanish and German to English.

We devised our system to be used in those circumstances where other machine translation systems are not available or of insufficient quality, because of lack of sufficiently large parallel corpora, in general or for the given domain, or because of the unavailability of the desired language pair. This is often the case in the European context as there is a high number of smaller languages.

Building a rule-based system for language pairs involving smaller languages is too costly and time consuming. By building a hybrid system², which does not rely on parallel corpora and which does not

use an extensive rule set, the METIS-II consortium provides an alternative solution.

For a system like METIS it is therefore not necessary to invest scores of man years into developing a rule-based MT system or several man years into collecting and preparing bilingual corpora. METIS should work just using basic resources. The way the system is designed, however, should allow for the use of more advanced resources as well. It should for example allow the use of a source-language (SL) corpus plus the data that can be distilled from it. It should also allow for integration with a translation memory (TM). Once enough material has been translated and post-edited, such a TM is to be considered a very valuable part of the workflow. Therefore, such aspects should be taken into account when developing the framework. This (automated) TM is not going to be used the traditional way, but during translation itself to build up a parallel corpus containing all SL sentences and their translations (after approval by the user). This will be used as an extra bilingual set of preferred translations that can be selected by the METIS engine. This way the performance of METIS-II when dealing with phenomena like light verbs or prepositional objects may improve quite seriously. The real challenge however is to develop a system to start with for a given language pair or a given domain when little or no other resources are available but a bilingual dictionary and a TL corpus: it should be good enough that people are willing to use it because otherwise there will in the end be no ‘parallel’ corpus derived from TM to improve the quality of the translations! Therefore, within the current project we are concentrating on developing the main translation tool.

The rationale behind the METIS projects is that a monolingual corpus in the TL, together with a bilingual dictionary guiding the raw lemma-to-lemma translation, should in principle suffice to generate good translations using a combination of statistics

¹Project FP6-IST-003768 funded by the IST in the 6th Framework.

²EBMT systems are often hybrid, incorporating some rule-based and statistical methods (Somers, 2003). In this case, e.g. the chunker is rule-based.

and linguistic rules, i.e. a hybrid approach. This monolingual TL corpus is likely to contain (parts of) sentences with the target words in them. Finding and recombining these is in fact what METIS-II is about. Successful development of such a simple tool for a rather complex task could give NLP a real boost in circumstances in which little resources are available: tasks for which parallel corpora and other expensive resources were thought to be indispensable, are then proved to be feasible without them.

Although the languages involved in METIS-II (Dutch, German, Greek, and Spanish as SL, English as TL) do not really belong to the smaller languages referred to above, we refrain from using such resources that are usually only available for the larger languages. The system therefore needs to be designed in such a way that it can be used for other (Indo-European) languages by plugging in the appropriate language-dependent modules. Therefore, we make use of resources that either will already be available for most languages, smaller ones included, or can be developed rather easily and at low cost.

Next to the bilingual dictionary and a TL corpus we also make use of

1. a tokeniser,
2. a part-of-speech tagger,
3. a chunker,
4. a lemmatiser/morphological generator (Carl and Schütz, 2005).

In case the TL corpus is not yet tagged, chunked and lemmatised, this should be done as well, meaning that tools for doing so (1 - 4) should also be available for the TL. We are using the BNC as TL corpus, which is already tagged but not yet chunked and lemmatised. So we need a chunker and lemmatiser for English as well.

The approach described below differs from the one adopted in METIS-I in that

- sentences are cut up in smaller chunks;
- linguistic information is also used outside the mapping rules;
- the TL corpus is indexed in different ways in order to increase the time efficiency;
- a general-purpose working prototype is built.

In a first stage, the consortium partners conduct separate experiments on different ways of chunking (no chunking, grammatical chunking, n-grams),

indexing, and creating a search engine. Other approaches can be found in (Markantonatou et al., 2005) and (Badia et al., 2005).

METIS-II (like METIS-I) targets the construction of free text translations making use of pattern-matching techniques and target-language retrieval from a large monolingual TL corpus. The system's performance and adaptability is enhanced by:

- breaking sentence-internal barriers: the system retrieves pieces of sentences (chunks) and recombines them to produce a final translation;
- extending the resources and integrating new languages;
- using post-editing facilities;
- adopting semi-automated techniques for adapting the system to different translation needs;
- taking into account real user needs, especially as far as the post-editing facilities mentioned before are concerned.

2 Global description of the METIS-II system

When translating a word by means of the bilingual dictionary, translations one gets are often inaccurate, as it is often the case that one and the same lemma, even when the tag is taken into account as well, may be translated in several ways. In such a case the right choice often depends on its context: the choice of an adjective may depend on the noun it is combined with, and the same holds for the relation between the verb and its object noun or the presence of a determiner before a noun, e.g.:

- (1) Ik beschouw Churchill als een groot
I consider Churchill as a tall/great
politician.
politician.
I consider Churchill to be a great politician.

In a first step the sentence to be translated is tokenised, tagged, lemmatised and chunked. When all lemmas in the SL sentence have got one or more translations in the TL, one may try to find this 'sentence' as such in the target language. The order of words in the TL often differs from that in the SL. Therefore all translated lemmas are offered chunk by chunk in a bag, i.e. unordered. It is clear that finding the literal translation of the SL sentence in the TL corpus is not very likely to succeed, except

for fixed expressions and the like. Therefore, our procedure is implemented in a bottom-up way. First the lowest-level chunks are handed to the search engine to find a match in the SL corpus. One of the tasks in searching the TL corpus is finding the right translation of words (rather: lemmas) on basis of the context, next to the correct order. That is why co-occurrence in NPs is so important. In order to translate clauses and whole sentences, the same procedure is applied to combinations of verbs and heads of NPs and PPs (always using the bag-of-lemmas approach), until every level of the shallow parse tree has been checked with the TL corpus.

To translate expressions, they have to be chunked as such in the SL analysis. The expression needs to be the lowest level of the shallow parse tree and is translated immediately using the expressions section of the bilingual dictionary.

When these are found, the various translations are assigned probability scores, and it depends on these scores which translation is favoured. These scores also determine how the translated string is presented to the end user for treatment during post-editing; unreliable or doubtful translations are marked as such. Before post-editing takes place, postprocessing has been taken care of by the system itself (automatic ‘adjustment’ of agreement, morphological generation of terms and the like).

In the next sections, we will describe of which modules METIS-II consists, and the requirements that are already clear (as we are still experimenting [cf. (Vandeghinste et al., 2005)], several things are still unclear).

3 General concepts

Before the various modules are described, some more general concepts should be described as these play an important role in our system.

3.1 Universal data format

The idea is to have one universal data format for all the data that go through the system. It is an XML format that can be read and produced by all the modules and tools involved. Each single module picks the parts it is interested in and adds further information when needed. The representation can be piped through the different processes and visualised in the GUI of the user environment. The proposed format is not definitive yet, since the research on the search engine might force us to add additional features.

The representation needs to

1. represent all information added and needed by the different processing modules and tools, e.g.
 - reading in the (tagged) source sentence
 - morphological analysis and lemmatisation
 - chunking
 - dictionary lookup
 - add synonyms from other sources, e.g. WordNet³
 - apply mapping rules
 - perform syntactical and morphological generation
 - output target-language sentence
2. allow and deal with ambiguities on several levels
 - ambiguity of tags (more possible tags attached to one token)
 - ambiguity of lemmas (more possible lemmas for one token)
 - ambiguity of translations (more possible translations)
 - ambiguity introduced by the tag-mapping rules (the rules have more than one right-hand side)
 - ambiguity of chunks/bags (more possibilities because of tag, lemma, translation and tag-mapping ambiguities)

Each step in the overall process adds or changes a section delimited by XML tags. We use three types of representations, the $\langle s \rangle$ tags (sequences, i.e. ordered sets of tokens or bags, thus chunks, clauses, sentences), $\langle b \rangle$ tags (bags, i.e. unordered sets of tokens or chunks) and $\langle t \rangle$ (tokens). The lowest level of the representations (leaves in the tree) is called a ‘token’. The sequences and bags are roots of (embedded) graphs. The type of root tells how the nodes are connected in the subgraphs (ordered or unordered sets). We do not allow cyclic graphs. Tokens do only occur as leaves of the tree and or the lowest-level representation.

³Languages that have not got their own implementation of WordNet, could use bilingual dictionaries to English and the English WordNet to find synonyms and other relations.

3.2 Dictionary format

Every tab-separated dictionary is easily converted to the XML dictionary format by a simple script. We need at least four columns: source-language lemma, source-language PoS, target-language lemma and target-language PoS. The source-language lemma and PoS are represented by `<sll>` and `<slt>` tags. The translations are represented by lemma-tag pairs (`<tll>` and `<tlt>` tags). Adding additional tags allow for discontinuous units to be represented.

The tags in the dictionary are those of the lemma (i.e. abstracting away from plural etc), unless some tokens are to show up in a particular form (i.e. in fixed expressions). Note that we cannot do with only one column of PoS tags ‘because a noun in the SL will become a noun in the TL as well’. Note that the situation is not always that straightforward, for example when one word in the SL is to be translated in several in the TL. But especially when the tag sets of SL and TL are designed in different ways (i.e. form-oriented and function-oriented, resp.) there are many inconsistencies.

The Dutch-English dictionary was compiled from the free Ergane⁴ dictionary and the Dutch part of EuroWordNet⁵ (Dirix, 2002a). The entries and PoS tags are checked manually. It contains about 110 000 lemma-to-lemma translations.

Example:

- (2) zijn oog laten vallen op
one’s eye let fall on
have one’s eye on

```
<lx>
  <sll>
    <u i="1"><abstr level="token"></u>
    <u i="2">oog</u>
    <u i="3">laten</u>
    <u i="4">vallen</u>
    <u i="5">op</u>
  </sll>
  <slt>
    <u i="1">VNW</u>
    <u i="2">N</u>
    <u i="3">WW</u>
    <u i="4">WW</u>
    <u i="5">VZ</u>
  </slt>
  <tll>
    <u i="1">have</u>
```

```
    <u i="2"><abstr level="token"></u>
    <u i="3">eye</u>
    <u i="4">on</u>
  </tll>
  <tlt>
    <u i="1">VV?</u>
    <u i="2">DTS</u>
    <u i="3">NN1</u>
    <u i="4">PRP</u>
  </tlt>
</lx>
```

The `<u>` tags represent continuous units. In this case, all Dutch words can permute and have to be in separate units. We use `<abstr>` in order to abstract the possessive pronoun in Dutch and in the English, since the expression can be used for all persons and both numbers. Abstraction cannot only be done on the token level, but also on the phrase or clause level.

3.3 Weights

Every step in the translation process which leads to ambiguities takes all the alternatives into account and applies a weight to each of these solutions. Introducing weights allows for disambiguation and choosing the most likely translation.

When tagging is performed, TnT provides us with probabilities of alternative tags, which we will use as weights. To get the weights of the different shallow parse trees, we multiply the weights of the tagged tokens for that shallow parse tree, and assign this product to the shallow parse tree. The weights of the tagged tokens are then set to 1.

Lemmatization occasionally leads to different alternatives. When this is the case, the same type of weight assignment is applied as above.

When a lemma is looked up in the bilingual dictionary, this can result in several alternative translations. For now, we assign an equal weight to these alternatives, but in a later stage we might apply weights based on the frequency of the alternative. Experiments will have to show if this improves the average translation quality.

When matching a bag with a corpus entry, we use the frequency of that corpus entry divided by the total frequency of all the matching corpus entries.

The total weight of a translation alternative will be the product of all the above mentioned weights. The user will also be able to tune the weights of different PoS and sub-PoS categories, e.g. assign a lower penalty for not translating articles or light verbs. The end user can also set the weight assigned to using phrases stored in the TM.

⁴<http://www.travlang.com/Ergane/>

⁵(Vossen et al., 1999)

3.4 Mapping rules

Mapping rules are used to perform changes between SL and TL tokens and strings, or to relate such tokens and strings. An example of the latter are the tag-mapping rules. Other mapping rules may insert, delete, modify or permute tokens and strings. An example of insertion is *do-support*, which as a consequence also modifies the appearance of other tokens (him, see , John, ? → do, him, see, John, ?). The tag sets used in SL and TL are likely to be different. As we are to know which tag in the SL tag set corresponds to which tag in the TL tag set, we are to draw a table in which equivalent tags are related (one-to-one, many-to-one or one-to-many). When translating between Dutch and English, using the CGN tag set (Van Eynde, 2004) for Dutch and the CLAWS5 tag set⁶, over 300 CGN tags are to be related to some 70 CLAWS5 tags. This means that in quite a number of cases several CGN tags are to be mapped onto one and the same CLAWS5 tag, although there are also a number of cases in which it is the other way around. This is also because of the fact that the CGN tag set is form-based and the BNC tag set is function-based.

The attributes of the Dutch tag for possessive pronouns (VNW(*bez*)) cover distinctions in person, number and form reduction (63 different combinations in total), while CLAWS5 make no subdivisions and has only one tag (DPS). On the other hand, a simplex tag for singular present tense (1st person) WW (*pv, tgw, ev*) is related to a series of tags via the tag-mapping rule

[V-1] WW(*pv, tgw, ev*) → VBB, VDB, VHB, VVB, ... since the BNC makes a distinction between auxiliary verbs (*to be, to have, and to do*), modal auxiliary verbs and other verbs.

Which of these CLAWS5 tags turns out to be the correct one for a given verb, is deduced from the lexicon. For *ben*, for example, the correct tag to be related to the Dutch one will be VBB, as the lemma of *am* has a VB?⁷ tag associated with it in the lexicon.

4 Global translation flow

A sentence to be translated is to go through the following modules (cf. Figure 1):

⁶Cf. <http://www.comp.lancs.ac.uk/ucrel/claws5tags.html>.

⁷We use question marks for generalisation of BNC tags. VB? is a generalisation of all combinations starting with VB: VBB, VBD, VBG, VBI, VBN, and VBZ.

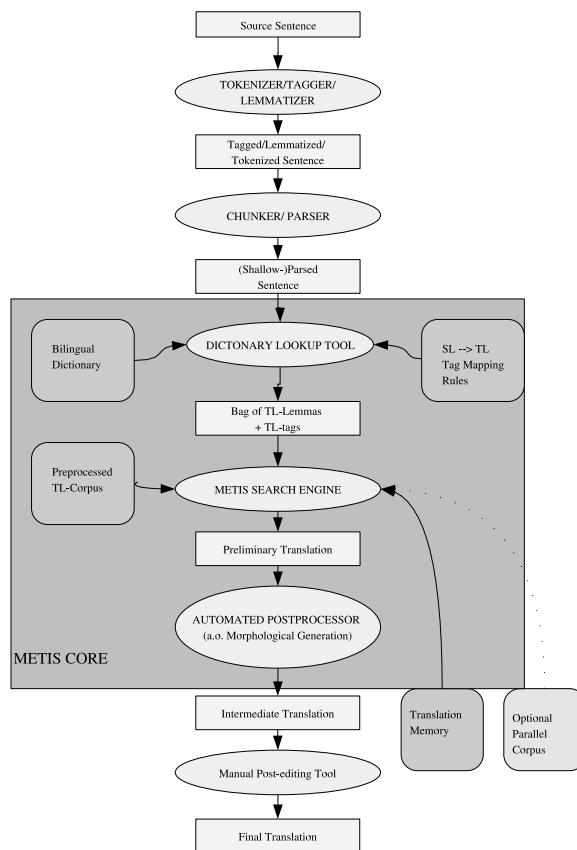


Figure 1: Data flow

4.1 SL analysis

4.1.1 Tokeniser

All language resources (source and target) should be in UTF-8. If a language is using non-Unicode compliant tools or resources (tokenizer, tagger/lemmatiser) the METIS main engine will pipe the tool input and output text through a converter before and after the process, so that no information is lost. Input information should be in text format, if desired with XML-compliant markup. Database servers storing the lexicon and corpus tables should also have UTF-8 as their default character encoding.

A source-language corpus may be preprocessed in order to get additional linguistic information about the source language, e.g. about frequencies of collocations.

The tokenizer takes a SL sentence as input. One of its tasks is the separation of words and punctuation. The tokenizer adds tags marking words and sentences.

Another task is the identification of continuous multiword units (MWUs). These may be compound prepositions (such as the English *in line with*), conjunctions (*as far as*), adverbs (*time and again*), determiners (*a lot of*), named entities (*Lernout & Hauspie*), or expressions in a foreign language (*a priori*). Furthermore, for Continental Germanic languages, the tokeniser must be able to recognise the constituting full words in cases like *in-en uitvoer*, short for *invoer en uitvoer* (import and export), *treinbegeleiders en -bestuurders* short for *treinbegeleiders en treinbestuurders* (train guards and train drivers), *LehrerInnen*, short for *Lehrer und/oder Lehrerinnen* (male and/or female teachers).

4.1.2 PoS tagger

Any tagger can be used, the same holds for the tag set. In case the tagger used provides probabilities concerning the tags to be selected, these can be used to adjust the weights. The tagger we use, is the TnT tagger (Brants, 2001), trained on the Corpus Spoken Dutch.

4.1.3 Lemmatiser

Tokens are related to lemmas in order to facilitate searching in the bilingual dictionary.

One of the tasks of the lemmatiser is to relate discontinuous parts of tokens. In Dutch and German there are verbs like *openmaken*, *aufmachen* (to open), i.e. verbs with separable particles, which may be realised with other words intervening:

- (3) a. Hij *maakt* zijn cadeautje *open*.
He makes his present open
He opens his present.
b. Er *macht* sein Geschenk *auf*.

The lemmatiser may come up with more than one lemma for a given token, thus implying that it is ambiguous. Sometimes such a string may be ambiguous from a human perspective as well, sometimes only for the machine (because of lacking world knowledge, for example). An example of the latter:

- (4) a. Hij stond te *bedelen*.
b. He was begging/endiowing.

bedelen 8566 WW(inf,vrij,zonder)
bedelen 8557 WW(inf,vrij,zonder)

These are two homonyms (also with different pronunciation), whose inflected forms also differ.

Their LemmaID can be used in order to distinguish them, e.g. in the past tense.

bedelde *bedelen* 8566
bedeelde *bedelen* 8557

4.1.4 Chunker / Shallow parser

In this component the separate parts are identified which will be searched for in the TL corpus. There are in principle two ways to identify such parts:

1. making use of grammatical units (NPs, clauses, ...)
2. making use of statistical units (n-grams)

We are experimenting with grammatical units and use the in-house developed ShaRPa chunker⁸.

There are also several ways to enrich these bare chunks: adding information with respect to its head, identify the subject NP, ... Fairly trivial tools can account for this, stating for example for Dutch that the head of an NP is always the last element; or that the subject is always the leftmost NP except when agreement tells otherwise, although we are aware of the fact that this is not always true. The basic idea is that even with such trivial tools the result will be better as without them.

4.2 SL to TL mapping

METIS makes use of flat bilingual dictionaries, i.e. dictionaries consisting of at least a pair of lemmas and their part-of-speech tag. The tokens in the source and/or target language may be complex, for example when a verb comes with a fixed preposition. Even more complex expressions (like complete phrases) may be contained in it, or they may be stored in a separate dictionary, depending on existing resources. The tags can be used as they are, i.e. with the original PoS tags. These are mapped onto the one used in the target corpus, i.e. the BNC. This is done making use of the mapping rules.

zijn WW → *be* VB
zijn VNW → *his* DP

In case of homonyms with different LemmaIDs one has to add this additional information in order to link the proper lemmas.

⁸Described in (Vandeghinste, 2005). An evaluation for Dutch is available in (Vandeghinste and Tjong Kim Sang, 2004).

portier N → 79808 *door* NN
portier N → 79809 *doorkeeper* NN

One could also make use of some of the information stored in the tags to distinguish such lemmas. In this case we could have used

portier N(onz) → *door* NN
portier N(zijd) → *doorkeeper* NN

In many cases one token in the SL can be translated in several ways in the TL. All these translations are to be taken into account. Furthermore, sometimes a series of tokens get a special translation as an idiomatic expression. One of the problems in this respect is that these tokens can be realised in a discontinuous way, and some of them are subject to variation (for example when a reflexive pronoun is involved).

The number of possible translations is reflected in the probability scores. These can be determined, for example by the number of translations in the bilingual dictionary, a score in the dictionary or with respect to their use in the TL corpus.

At this point, tag-mapping rules could apply. In the METIS-I project⁹, we developed a set of tag-mapping rules to transform PoS categories and verbal tenses from Dutch CGN format to English BNC format (Dirix, 2002b).

4.3 TL generation

4.3.1 Preprocessing of the TL corpus

Thus far the developer could make use of his own in-house resources (tagger, chunker, bilingual dictionary etc.). From here on he has to make use of the tools developed by the METIS-II consortium, in order to arrive at the proper translations, the most important tool of these being the METIS Search Engine. First we will say something about the way the TL corpus is to be preprocessed.

The TL corpus should be tokenised, tagged, lemmatised and chunked. The corpus should be preprocessed at the same level as the input sentence. One could use the same tools as for the SL. Of course the tools should be adapted to deal with the TL if necessary. When the corpus was already prepared in all these respects, one has to verify whether the results are compatible with what has been done for the SL. If not, one may have to write some wrappers, mapping rules or the like. In order to be able to perform

⁹Predecessor of METIS-II, IST-2001-32775.

a fast search in the TL corpus, it has to be preprocessed in other ways as well, using indexing and drawing frequency tables out of the corpus. Many statistics can be made based on the TL corpus. They can be used at several points during the translation. The same preprocessing steps are to be executed for the TM and the parallel corpus, if available and used in the system.

The consortium will make use of collocation statistics in order to find out which tokens frequently come together. Tables with often co-occurring lemmas are being derived. These may help to weed out the most unlikely translations of tokens in a sentence (or rather: to give them the most appropriate weights) when several translations are possible. This way one may reduce the number of possible translations offered to the search engine

A fast way of searching in the TL corpus is necessary. One way to do so is to convert the preprocessed corpus into a database. This can be done in several ways: all NPs are indexed on the head noun, all sentences are indexed on the main verb, and so on.

In order to determine the order in which the chunks found are to be combined to derive a correct sentence, one could make use of templates. These are derived from the TL corpus, for example by replacing all NP chunks by the label NP (and possibly some information about the missing NP, e.g. its lexical head), and the same for other types of chunks.

4.3.2 Search engine

The METIS-II translation engine as such is to be a very modular one. The kernel and the language-specific modules can be written in any programming language (Java, Perl, C, ...), as long as input and output conform to the universal data format. The METIS Search engine should be able to take a bag of TL lemmas and TL tags as its input, and look it up in the preprocessed TL corpus.

As we also want METIS-II to cope with several kinds of texts, we have to anticipate several switches and slots: for the domain-specific translations, specialised term databases have to be connected, grammar checkers (source and target), etc. Thus the system needs to be open to a certain extent. What the language-specific modules will exactly look like, depends on the outcome of the current experiments.

After SL analysis, the resulting shallow parse tree is processed depth-first.

In order to translate the first node in the shallow parse tree, each of the daughters of this node

is translated first. When this concerns a lemma, this lemma is looked up in the bilingual dictionary.

All daughters of a node are put in a bag and this bag is matched with the TL corpus which is preprocessed up to the same level as the original SL node (for instance up to the basic NP level).

When all daughters of the shallow parse tree have been translated, all these translations are put in a bag. We try to find a match with the TL corpus, where we use the heads of each node to find the best match.

Syntactic generation could be considered a subpart of the translation engine, i.e. the part in which all the parts and pieces found by the search engine will be combined in order to yield correct translations (serialisation).

At this stage, we have an engine for NP translations (cf. Vandeghinste et al., 2005). It uses an indexed database of NPs drawn from the BNC, and tries to match the bags of translated words with the database. The same procedure will be used on a more abstract level (lemmas substituted with tags or generalised expressions) to find whole clauses and sentences.

4.3.3 Automated postprocessing

Although the parts of the translation are already in a correct order, some other phenomena still need to be taken care of: the combinations of lemma and tag are to be realised as tokens (Carl and Schütz, 2005), agreement (adjusting number, person, ...). Which phenomena exactly are dealt with here depends on the TL.

Up to here, we have looked up the bags of lemmas in the target language corpus, and retrieved the sentence/clause/chunk structures. This means we need morphological generation. Based on the lemmas (coming from the target side of the bilingual dictionary) and the tags (coming from the target side of the tag-mapping rules, or the target side of the dictionary), we can generate the target token. In order to be sure that a lemma-tag combination leads to a unique token, we added some features to the CLAWS5 tagset, where we noticed that several tokens could be generated from one lemma-tag combination (see for example the CLAWS5 tag for the past tense forms of the verb *to be*, which is VBD for both singular past and plural past). As a result of all these steps, we now have reached the stage of ‘intermediate translation’. This should be of a pretty good quality. After postprocessing, the end user is supposed to do some post-editing in order to get a

final, proper translation. These translations are to be fed to the translation memory.

5 Conclusions

A first evaluation (see Vandeghinste et al., 2005) concerning the translation of NPs along the lines described in this paper, shows that we are on the right track. In this experiment, a set of 685 NPs (2/3 fiction, 1/3 newspaper) were translated. In almost 58% the translation ranked by the system was a correct one, in another 14% the correct translation was among the other translation alternatives. In quite a number of cases no translation or a wrong one were given due to the coverage of the lexicon (over 37 000 lemmas and over 110 000 entries, which is still too small, and it turns out that many Belgian Dutch words are still lacking). Extending the lexicon is therefore likely to improve our results. In our approach, in which we are essentially trying to build up the translation of a sentence out of the combination of translated chunks (based on the BNC), the translation of smaller units, like NPs is one of the building blocks.

We are aware of the fact that when translating sentences for example the verb may also influence which translation of an NP (cf. section 2) is to be considered the best one: searching the TL corpus is to guide this.

Breaking up the sentence, one of the major differences with METIS-I, seems to be a successful approach.

6 References

- T. Badia, G. Boleda, M. Melero, and A. Oliver. 2005. An *n*-gram approach to exploiting a monolingual corpus for Machine Translation. In *Proceedings of the EBMT Workshop 2005*, Phuket (this volume).
- T. Brants. 2001. *TnT - A Statistical Part-of-Speech Tagger*. Published online at <http://www.coli.uni-sb.de/thorsten/tnt>.
- M. Carl and J. Schütz. 2005. A Reversible Lemmatizer/Token-generator for English. In *Proceedings of the EMBT Workshop 2005*, this volume.
- P. Dirix. 2002a. *The METIS Project: Lexical Resources*. Internship Report, KULeuven.
- P. Dirix. 2002b. *The METIS Project: Tag-mapping rules*. Paper, KULeuven.

- Y. Dologlou, S. Markantonatou, G. Tambouratzis, O. Yannoutsou, A. Fourla, and N. Ioannou, 2003. Using Monolingual Corpora for Statistical Machine Translation: The METIS System. In *Proceedings of EAMT - CLAW 2003*, Dublin, pp. 61-68.
- G. Grefenstette. 1999. The World Wide Web as a Resource for Example-Based Machine Translation Tasks. *ASLIB, Translating and the Computer 21*. London.
- S. Markantonatou, S. Sofianopoulos, V. Spilioti, Y. Tambouratzis, M. Vassiliou, O. Yannoutsou, and N. Ioannou, 2005. Monolingual Corpus-based MT using Chunks. In *Proceedings of the EBMT Workshop 2005*, Phuket (this volume).
- H. Somers. 2003. An overview of EBMT. In M. Carl and A. Way (ed.), *Recent advances in example-based machine translation*, Kluwer Academic Publishers, Dordrecht.
- V. Vandeghinste. 2005. *Manual for ShaRPa 2.0*. Internal Document. Centre for Computational Linguistics, Leuven.
- V. Vandeghinste, P. Dirix, and I. Schuurman. 2005. Example-based Translation without Parallel Corpora: First experiments on a prototype. In *Proceedings of the EBMT Workshop 2005*, Phuket (this volume).
- V. Vandeghinste and E. Tjong Kim Sang. 2004. Using a Parallel Transcript/Subtitle Corpus for Sentence Compression. In *Proceedings of LREC2004*. ELRA. Paris.
- F. Van Eynde. 2004. *Tagging and Lemmatisation for the Spoken Dutch Corpus*. Internal report.
- P. Vossen, L. Bloksma, and P. Boersma. 1999. *The Dutch WordNet*. University of Amsterdam.