Chapter 2

Building and querying parallel treebanks

Martin Volk
University of Zurich, Institute of Computational Linguistics

Torsten Marek

Yvonne Samuelsson
Stockholm University, Department of Linguistics

This paper describes our work on building a trilingual parallel treebank. We have annotated constituent structure trees from three text genres (a philosophy novel, economy reports and a technical user manual). Our parallel treebank includes word and phrase alignments. The alignment information was manually checked using a graphical tool that allows the annotator to view a pair of trees from parallel sentences. This tool comes with a powerful search facility which supersedes the expressivity of previous popular treebank query engines.

1 Introduction

Recent years have seen a number of initiatives in building parallel treebanks (see Abeillé 2003; Nivre, De Smedt & Volk 2005). The current interest in treebanks is documented in international workshop series like "Linguistically Interpreted Corpora (LINC)" or "Treebanks and Linguistic Theories" (TLT).

We see a treebank as a particular kind of annotated corpus where each sentence is mapped to a special type of graph, a tree which represents its syntactic structure. Traditionally the graphs were constituent structure trees but recent years have also seen dependency treebanks. Constituent structure trees contain nodes and edges where each node holds a label for a group of words (as e.g. NP for noun phrase or VP for verb phrase). Dependency trees represent syntactic dependencies between words directly. We work with constituent structure trees that have labeled edges to denote functional relations which can easily

Martin Volk, Torsten Marek & Yvonne Samuelsson. 2017. Building and querying parallel treebanks. In Silvia Hansen-Schirra, Stella Neumann & Oliver Čulo (eds.), *Annotation, exploitation and evaluation of parallel corpora*, 9–35. Berlin: Language Science Press. DOI:10.5281/zenodo.283438

be mapped to dependencies. The concept of constituent structure trees in treebanking has been stretched beyond proper trees as defined in graph theory by accepting crossing edges and even secondary edges.

Parallel treebanks are treebanks over parallel corpora, i.e. the "same" text in two or more languages, where one text might be the source text and the other texts are translations thereof, or where all texts are translations of a text outside of the corpus. In addition to the syntactic annotation, a parallel treebank is aligned on the sub-sentential level, for example on the word level or the phrase level.

Parallel treebanks can be created automatically or manually. Automatic creation entails automatic parsing and automatic alignment, both of which will result in a certain amount of error at the current state of the technology. In this paper we focus on the manual creation of parallel treebanks.

Parallel treebanks can be used as training or evaluation corpora for word and phrase alignment, as input for example-based machine translation (EBMT), as training corpora for transfer rules, or for translation studies.

Parallel treebanks have evolved into a research field in the last decade. Cmejrek, Curin & Havelka (2003) at the Charles University in Prague have built a parallel treebank for the specific purpose of machine translation, the Czech-English Penn Treebank with tectogrammatical dependency trees. They have asked translators to translate part of the Penn Treebank into Czech with the clear directive to translate every English sentence with one in Czech and to stay as close as possible to the original.

Other parallel treebank projects include Croco (Hansen-Schirra, Neumann & Vela 2006) which is aimed at building an English-German treebank for translation studies, LinES an English-Swedish parallel treebank (Ahrenberg 2007), and the English-French HomeCentre treebank (Hearne & Way 2006), a hand-crafted parallel treebank consisting of 810 sentence pairs from a Xerox printer manual.

Our group has contributed to these efforts by building a tri-lingual parallel treebank called Smultron (Stockholm MULtilingal TReebank). Our parallel treebank consists of syntactically annotated sentences in three languages, taken from translated documents. Syntax trees of corresponding sentence pairs are aligned on a sub-sentential level. On the side we have also experimented with building parallel treebanks for the widely differing languages Quechua and Spanish (Rios, Göhring & Volk 2009).

In this paper we will first describe our parallel treebank and the difficulties in consistent annotation. We have developed a special alignment tool and present its functionality for alignment and search of parallel treebanks. To our know-

ledge this is the first dedicated tool that combines visualization, alignment and searching of parallel treebanks.

2 Building SMULTRON - The Stockholm MULtilingual TReebank

We have built a trilingual parallel treebank in English, German and Swedish. In its 2008 release SMULTRON consists of around 500 trees from the novel *Sophie's World* and 500 trees from economy texts (an annual report from a bank, a quarterly report from an international engineering company, and the banana certification program of the Rainforest Alliance) (Samuelsson & Volk 2006; 2007). The sentences in Sophie's World are relatively short (14.8 tokens on average in the English version), while the sentences in the economy texts are much longer (24.3 tokens on average; 5 sentences in the English version have more than 100 tokens).

Lately we have added 500 trees from another text genre: a user manual for a DVD player. This genre differs in that it contains a multitude of imperative constructions, many numerical expressions as well as many itemized and enumerated lists. SMULTRON version 2.0 consisting of 1500 trees from three text genres in three languages has been released in the beginning of 2010.[1]

2.1 Monolingual treebanking

For English and German, there are large monolingual treebanks that have resulted in standards for treebanking in these languages. We have followed these standards and (semi-automatically) annotated the German sentences of our treebank with Part-of-Speech tags and phrase structure trees (incl. edges labeled with functional information) according to the NEGRA guidelines (Brants et al. 1997).

For English, we have used the Penn Treebank guidelines which also prescribe phrase structure trees (with PoS tags, but only partially annotated with functional labels). However they differ from the German guidelines in many details. For example, the German trees use crossing edges for discontinuous units while the English trees introduce symbols for empty tokens plus secondary edges for the representation of such phenomena.

There has been an early history of treebanking in Sweden, dating back to the 1970s (cf. Nivre 2002. The old annotation schemes were difficult for automatic

[1] SMULTRON is freely available from http://kitt.cl.uzh.ch/kitt/smultron/

processing (in the case of Talbanken, Teleman 1974)[2] or too coarse-grained (in the case of Syntag, Järborg 1986). Therefore we have developed our own treebanking guidelines for Swedish inspired by the German guidelines.

We annotated the treebanks for all three languages separately, with the help of the treebank editor ANNOTATE[3]. ANNOTATE includes the TnT Part-of-Speech Tagger and Chunker for German. We added taggers and chunkers for Swedish and English. After finishing the monolingual treebanks, the trees were exported from the accompanying SQL database and converted into an XML format as input to our alignment tool, the TreeAligner.

Both the German trees and the Swedish trees are annotated with flat structures but subsequently automatically deepened to result in richer and linguistically more plausible tree structures.

2.1.1 Automatic treebank deepening

The German NEGRA annotation guidelines (Brants et al. 1997) result in rather flat phrase structure trees. This means, for instance, no unary nodes, no "unnecessary" NPs (noun phrases) within prepositional phrases and no finite verb phrases. Using a flat tree structure for manual treebank annotation has two big advantages for the human annotator: 1) the annotator needs to make fewer decisions, and 2) the annotator has a better overview of the trees. This comes at the cost of the trees not being complete from a linguistic point of view. One could ask why an NP that consists of only one daughter is not marked, or why an NP that is part of a PP is not marked, while the same NP outside a PP is explicitly annotated. These restrictions also have practical consequences: If certain phrases (e.g. NPs within PPs) are not explicitly marked, then they can only indirectly be searched in corpus linguistics studies.

In addition to the linguistic drawbacks of the flat syntax trees, they are also problematic for phrase alignment in a parallel treebank. Our goal is to align subsentential units (such as phrases and clauses) to get fine-grained correspondences between languages. The alignment focuses on meaning, rather than sentence structure. For example, sentences can have alignment on a higher level of the tree (for instance, if the sentence carries the same meaning in both languages), without necessarily having alignment on all lower levels (for instance, if the sentence contains an NP without direct correspondence in the other language). We

[2] Talbanken has recently been cleaned and converted to a dependency treebank by Joakim Nivre and his group. See http://w3.msi.vxu.se/ nivre/research/talbanken.html

[3] Annotate is a treebank editor developed at the University of Saarbrücken. See http://www.coli.uni-sb.de/sfb378/negra-corpus/annotate.html

prefer to have "deep trees" to be able to draw the alignment between the German sentences and the parallel Swedish sentences on as many levels as possible; in fact, the more detailed the sentence structure is, the more expressive is our alignment.

We deepened the flat phrase structure trees automatically with a script, which automatically inserts nodes to create the deeper structure. However, these insertions must be totally unambiguous, so that no errors are introduced. The input for this program is a tree description in TIGER-XML (König & Lezius 2002), an interface format which can be created and used by the treebank tool TIGER-Search[4]. The output is a deepened TIGER-XML tree. We have measured that the automatic node insertion resulted in an increase of almost 60% additional nodes.

2.1.2 Completeness and consistency checks over treebanks

Completeness and consistency are important characteristics of corpus annotation. Tree completeness means that each token and each node is part of the tree.[5] This can easily be checked and should ideally be part of the annotation tool.

Consistency checking is more complicated. Consistent annotation means that the same token sequence (or part-of-speech sequence or phrase sequence) is annotated in the same way across the treebank. Annotation error detection has been explored for part-of-speech annotation (Dickinson & Detmar Meurers 2003; Loftsson 2009) and syntactic annotation (Ule & Simov 2004; Dickinson & Meurers 2005).

The variation *n*-gram approach for syntactic annotation (Dickinson & Meurers 2003; 2005) is a method for detecting strings which occur multiple times in the corpus with varying annotation. The approach can detect bracketing and labeling errors in constituency annotation.

2.2 Aligning trees

Establishing translation correspondences is a difficult task. This task is traditionally called alignment and is usually performed on the paragraph level, sentence level and word level. Alignment answers the question: Which part of a text in language L1 corresponds in meaning to which part of a text in language L2 (under the assumption that the two texts represent the same meaning in different languages)?

[4] See also http://www.ims.uni-stuttgart.de/projekte/TIGER.

[5] Different treebanks take different positions on whether special tokens like punctuation symbols should be part of the tree. For example, the Penn Treebank guidelines require punctuation marks to be part of the tree, whereas the German TIGER guidelines leave them unattached.

There is considerable interest in automating the alignment process. Automatic sentence alignment of legacy translations helps to fill translation memories. Automatic word alignment is a crucial step in training statistical machine translation systems. Both sentence and word alignment have to deal with 1-to-many alignments, e.g. sometimes a sentence in one language is translated as two or three sentences in the other language.

In other respects sentence alignment and word alignment are fundamentally different. It is relatively safe to assume the same sentence order in both languages when computing sentence alignment. But such a monotonicity assumption is not possible for word alignment which needs to allow for word order differences and thus for crossing alignments. While basic algorithms for sentence alignment can rely on unsophisticated measures like sentence length in characters and still produce good results, word alignment algorithms use cross-language cooccurrence frequencies as a key feature.

Our work focuses on word alignment and on an intermediate alignment level which we call **phrase alignment**. Phrase alignment encompasses the alignment from simple noun phrases and prepositional phrases all the way to complex clauses. For example, on the word alignment level we want to establish the correspondence of the German "verb form plus separated prefix" *fing an* with the English verb form *began.* In phrase alignment, we mark the correspondence of the verb phrases *ihn in den Briefkasten gesteckt* and *dropped it in the mail box*. For the alignment we have developed a specific tool called TreeAligner (Lundborg et al. 2007), which displays two trees and allows the user to draw alignment lines by clicking on phrases and words.

We regard phrase alignment as alignment between linguistically motivated phrases, in contrast to work in statistical machine translation where phrase alignment is defined as the alignment between arbitrary consecutive word sequences. Our phrase alignment is alignment between nodes in constituent structure trees. See Figure 1 for an example of a tree pair with word and phrase alignment. Green lines indicate exact alignments and red lines represent fuzzy alignments (cf.§2.2.2).

It is our belief that linguistically motivated phrase alignment provides useful phrase pairs for example-based machine translation, and provides interesting insights for translation science and cross-language comparisons. Phrase alignments are particularly useful for annotating correspondences of idiomatic or metaphoric language use.

2.2.1 Related research

Our research on word and phrase alignment is related to previous work on word alignment as e.g. in the Blinker project (Melamed 1998) or in the UPLUG project (Lars, Merkel & Petterstedt 2003). Alignment work on parallel treebanks is rare. Most notably there is the Prague Czech-English treebank (Kruijff-Korbayová, Chvátalová & Postolache 2006) and the Linköping Swedish-English treebank (Ahrenberg 2007). There has not been much work on the alignment of linguistically motivated phrases. Tinsley et al. (2007) and Groves, Hearne & Way (2004) report on semi-automatic phrase alignment as part of their research on example-based machine translation.

The most comprehensive study is probably the recent PhD thesis by Zhechev (2009). The author describes his system for automatic phrase alignment over parallel trees which is based on word alignment probabilities provided by GIZA. He evaluates his system against the manually aligned HomeCentre treebank and reports on about 78% recall for 80% precision. These results are comparable to Ambati & Lavie (2008). These approaches are unsupervised in the sense that human-aligned trees are used only for evaluation.

Tiedemann & Kotzé (2009) present a supervised approach which automatically learns phrase alignment features from our parallel treebank. By training on 400 aligned trees and testing on the remaining 100, they report on 80% precision and 76% recall.

Considering the fact that the alignment task is essentially a semantic annotation task, we may also compare our work to other tasks in semantic corpus annotation, for example, the frame-semantic annotation in the German SALSA project (cf. Burchardt et al. 2006).

2.2.2 Our alignment guidelines

We have compiled alignment guidelines for word and phrase alignment between annotated syntax trees. The guidelines consist of general principles, concrete rules and guiding principles. The most important general principles are:

1. Align items that can be re-used as units in a machine translation system.
2. Align as many items (i.e. words and phrases) as possible.
3. Align as close as possible to the tokens.

The first principle is central to our work. The focal point is whether a phrase pair is general enough to be re-used as translation unit in a machine translation

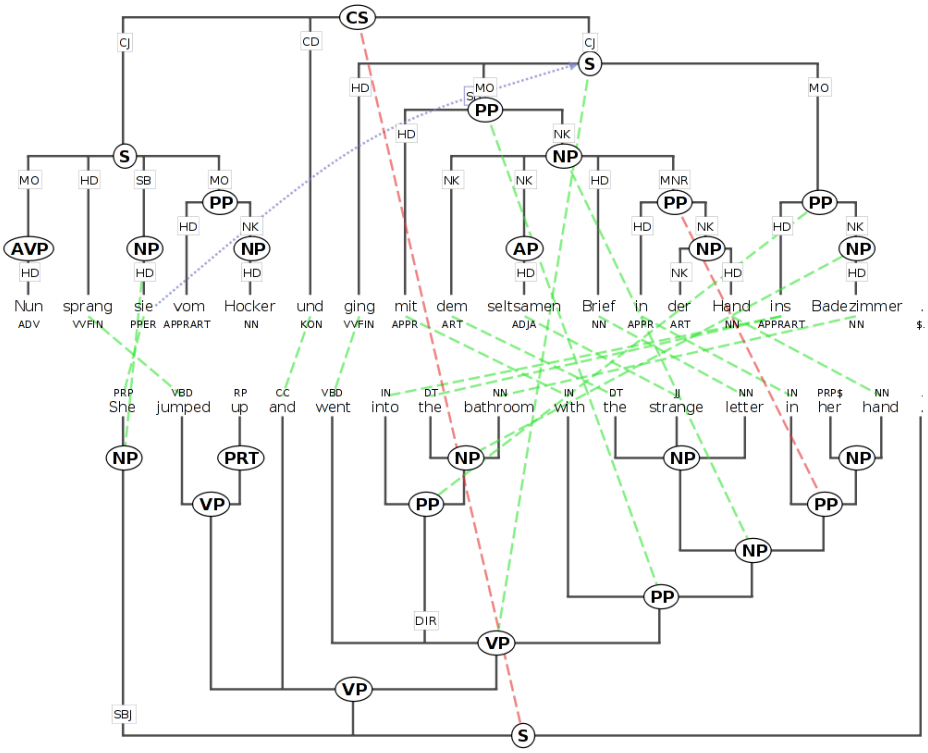

Figure 1: Three pair German-English with word and phrase alignments.

system. For example, in our Sophie's World treebank we have decided not to align *die Verwunderung über das Leben* with *their astonishment at the world* although these two phrases were certainly triggered by the same phrase in the Norwegian original, and both have a similar function in the two corresponding sentences. These two phrases in isolation are too far apart in meaning to license their re-use. We are looking for correspondences like *was für eine seltsame Welt* and *what an extraordinary world* which would make for a good translation in many other contexts.

Some special rules follow from this principle. For example, we have decided that a pronoun in one language shall never be aligned with a full noun in the other, since such a pair is not directly useful in a machine translation system.

Principles 2 and 3 are more technical. Principle 2 tells our annotators that alignment should be comprehensive. We want to re-use as much as possible from the treebank, so we have to look for as many alignments as possible. Principle 3 says that in case of doubt the alignment should go to the node that is closest to the terminals. For example, our German treebank guidelines require a multi-word proper noun to first be grouped in a PN phrase which is a single daughter node of a noun phrase [[Sofie Amundsen]PN]NP. When we align the name, principle 3 tells us to draw the alignment line from the German PN node since it is closer to the tokens than the German NP node.

Often we are confronted with phrases that are not exact translation correspondences but approximate translation correspondences. Consider the phrases *mehr als eine Maschine* and *more than a piece of hardware*. This pair does not represent the closest possible translation, but it represents a possible translation in many contexts. In a way we could classify this pair as the "second-best" translation. To allow for such distinctions we provide our annotators with a choice between exact translation correspondences and approximate correspondences. We also use the term **fuzzy correspondence** to refer to and give an intuitive picture of these approximate correspondences. The option to distinguish between different alignment strengths sounded very attractive at the start. But where and how can we draw the line between exact and fuzzy translation correspondences? We have formulated some clear-cut rules:

- If an acronym is to be aligned with a spelled-out term, it is always an approximate alignment. For example, in our economy reports the English acronym *PT* stands for *Power Technology* and is aligned to the German *Energietechnik* as a fuzzy correspondence.
- Proper names shall be aligned as exact alignments (even if they are spelled differently across languages; e.g. *Sofie* vs. *Sophie*).

But many open questions persist. Is *einer der ersten Tage im Mai* an exact or rather a fuzzy translation correspondence of *early May*? We decided that it is not an exact correspondence. How shall we handle *zu dieser Jahreszeit* vs. *at this time of the year* where a literal translation would be *in this season*? We decided that the former is still an exact correspondence. These examples illustrate the difficulties in distinguishing between exact and approximate translation correspondence. Automatically ensuring the overall consistency of the alignment decisions is a difficult task. We have built a tool to ensure the consistency within the exact and approximate alignment classes. The tool computes the token span

for each alignment and checks if the same token span pairs have always received the same alignment type. For example, if the phrase pair *mit einer blitzschnellen Bewegung* and *with a lightning movement* is once annotated as exact alignment, then it should always be annotated as exact alignment. Figure 1 shows approximate alignments between the PPs *in der Hand* and *in her hand*. It was classified as approximate rather than exact alignment since the German PP lacks the possessive determiner.

Currently our alignment guidelines are more than 15 pages long with examples for English-German and English-Swedish alignments. The challenge was to compile precise and comprehensive guidelines to ensure smooth and consistent alignment decisions. In Samuelsson & Volk (2006) we have reported on experiments to evaluate inter-annotator agreement from our alignment tasks. Here we summarize an experiment described in detail in Volk, Marek & Samuelsson (2008) in which we evaluated our alignment guidelines.

2.2.3 Inter-annotator agreement experiments

In order to evaluate the inter-annotator agreement for the alignment task we performed the following experiment. We gave 20 tree pairs in German and English to 12 advanced undergraduate students. Half of the tree pairs were taken from our Sophie's World treebank and the other half from our Economy treebank. We made sure that there was one 1-to-2 sentence alignment in the sample. The students did not have access to the gold standard alignment.

In class we demonstrated the alignment tool to the students, and we introduced the general alignment principles to them. Then the students were given a copy of the alignment guidelines. We asked them to do the alignments independently of each other and to the best of their knowledge according to the guidelines.

Table 1: Alignment Frequencies in the Gold Standard

	Alignment Type	**exact**	**fuzzy**	**total**
Sophie part	word alignment	75	3	78
	phrase alignment	46	12	58
Economy part	word alignment	159	19	178
	phrase alignment	62	9	71

Our own annotation of the 20 tree pairs (the gold standard alignment) contains the alignments shown in Table 1. In the Sophie part of the experiment treebank

we have 78 word-to-word alignments and 58 phrase-to-phrase alignments. Note that some phrases consist only of one word and thus the same alignment information is represented twice. We have deliberately kept this redundancy.

The alignments in the Sophie part consist of 125 times 1-to-1 alignments, 4 times 1-to-2 alignments and one 1-to-3 alignment (*wäre* vs. *would have been*) when viewed from the German side. There are 3 times 1-to-2 alignments (e.g. *introducing* vs. *stellte vor*) and no other 1:many alignment when viewed from the English side. In the Economy part the picture is similar.

The student alignments showed a huge variety in terms of numbers of alignments. In the Sophie part they ranged from 125 alignments to bare 47 alignments (exact alignments and fuzzy alignments taken together). In the Economy part, the variation was between 259 and 62 alignments. On closer inspection we found that the student with the lowest numbers works as a translator and chose to use a very strict criterion of translation equivalence rather than translation correspondence. Three other students at the end of the list were not native speakers of either German or English. We therefore decided to exclude these 4 students from the following comparison.

The student alignments allow for the investigation of a number of interesting questions:

- How did the students' alignments differ from the gold standard?
- Which were the alignments done by all students?
- Which were the alignments done by single students only?
- Which alignments varied most between exact and fuzzy alignment?

2.2.4 Inter-annotator agreement results

The remaining 8 students reached between 81% and 48% overlap with our gold standard on the Sophie part, and between 89% and 66% overlap with our gold standard on the Economy texts. This can be regarded as their recall values if we assume that the gold standard represents the correct alignments. These students additionally had between 2 and 22 own alignments in the Sophie part and between 12 and 55 own alignments in the Economy part.

So the interesting question is: What kind of alignments have they missed, and which were the additional own alignments that they suggested (alignments that are not in the gold standard)? We first checked the students with the highest numbers of own alignments. We found that some of these alignments were due

to the fact that students had ignored the rule to align as close to the tokens as possible (principle 3 above).

Another reason was that students sometimes aligned a word (or some words) with a node. For example, one student had aligned the word *natürlich* to the phrase *of course* instead of to the word sequence *of course*. Our alignment tool allows that, but the alignment guidelines discourage such alignments. There might be exceptional cases where a word-to-phrase alignment is necessary in order to keep valuable information, but in general we try to stick to word-to-word and phrase-to-phrase alignments.

Another discrepancy occurred when the students aligned a German verb group with a single verb form in English (e.g. *ist zurückzuführen* vs. *reflecting*). We have decided to only align the full verb to the full verb (independent of the inflection). This means that we align only *zurückzuführen* to *reflecting* in this example.

The uncertainties on how to deal with different grammatical forms led to the most discrepancies. Shall we align the definite NP *die Umsätze* with the indefinite NP *revenues* since it is much more common to drop the article in an English plural NP than in German? Shall we align a German genitive NP with an of-PP in English (*der beiden Divisionen* vs. *of the two divisions*)? We have decided to give priority to form over function and thus to align the NP *der beiden Divisionen* with the NP *the two divisions*. But of course this choice is debatable.

When we compute the **intersection** of the alignments done by all students (ignoring the difference between exact and fuzzy alignments), we find that about 50% of the alignments done by the student with the smallest number of alignments is shared by all other students. All of the alignments in the intersection are in our gold standard file. This indicates that there is a core of alignments that are obvious and uncontroversial. Most of them are word alignments.

When we compute the **union** of the alignments done by all students (again ignoring the difference between exact and fuzzy alignments), we find that the number of alignments in the union is 40% to 50% higher than the number of alignments done by the student with the highest number of alignments. It is also about 40% to 50% higher than the number of alignments in the gold standard. This means that there is considerable deviation from the gold standard.

Other discrepancies concern cases of differing grammatical forms, e.g. a German definite singular noun phrase (*die Hand*) that was aligned to an English plural noun phrase (*hands*) in the gold standard but missed by all students. Finally there are a few cases where obvious noun phrase correspondences were simply overlooked by all students (*sich - herself*) although the tokens themselves were aligned. Such cases should be handled by an automated process in the align-

ment tool that projects from aligned tokens to their mother nodes (in particular in cases of single token phrases).

2.2.5 Working with the TreeAligner

The tree alignments in SMULTRON and in the experiments above were done with a tool called TreeAligner. Let us look at the alignment process in more detail.

When our monolingual treebanks were finished, the trees were exported from the editor system and converted into TIGER-XML, an XML format for encoding syntax graphs with crossing dominance branches and secondary edges. TIGER-XML has been defined as input format for TIGERSearch, a query tool for monolingual treebanks (see §3.1). We use TIGER-XML also as input format for the TreeAligner (Volk et al. 2006).

The TreeAligner program is a graphical user interface to specify (or correct) word and phrase alignments between pairs of syntax trees. [6] The TreeAligner is roughly similar to alignment tools such as I*Link (Ahrenberg, Merkel & Andersson 2002) or Cairo Smith & Jahr it is especially tailored to visualize and align full syntax trees. The TreeAligner is unique in that it allows the alignments of linguistically motivated phrases via node alignments in parallel constituent structure trees (cf. Samuelsson & Volk 2007).

The TreeAligner operates on an alignment file in an XML format developed by us. This file describes the alignments between two TIGER-XML treebanks (specified in the alignment file) holding the trees from language one and language two respectively. For example the alignment between two nodes is represented as:

(1)
```
<align type="good">
  <node treebank_id="de" node_id="s153_11"/>
  <node treebank_id="en" node_id="s144_10"/>
</align>
```

This says that node 11 in sentence 153 of the German treebank (de) is aligned with node 10 in sentence 144 of the English treebank (en). The node identifiers refer to the IDs in the TIGER-XML treebanks. The alignment is given the label "good" or "fuzzy" depending on the degree of meaning correspondence.

[6] The TreeAligner was implemented in Python by Joakim Lundborg and Torsten Marek. It is freely available at http://www.cl.uzh.ch/treealigner.html

The alignment file might initially be empty when we start manual alignment from scratch, or it might contain automatically computed alignments for correction. The TreeAligner displays tree pairs with the trees in mirror orientation (one top-up and one top-down) exemplified in Figure 1. The trees are displayed with node labels, edge labels and part-of-speech tags.

Each alignment is displayed as a dotted line between two nodes (or words) across two trees. Clicking on a node (or a word) in one tree and dragging the mouse pointer to a node (or a word) in the other tree inserts an alignment line. The type of the alignments is represented by its color. Our experiments indicate that eventually more alignment types than just the two used in SMULTRON will be needed to precisely represent fine-grained translation differences. In its most recent version, the TreeAligner supports arbitrarily many alignment types, which can describe many different levels or modes of alignment. These distinctions could prove useful when exploiting the aligned treebanks for Machine Translation and other applications.

Often one tree needs to be aligned to two (or more) trees in the other language. The TreeAligner therefore provides the option to browse the trees independently.

The TreeAligner is designed as a stand-alone tool (i.e. it is not prepared for collaborative annotation). It stores every alignment in an XML file (in the format described above) as soon as the user moves to a new tree pair.

Lately, we have included an interactive module that suggests word and phrase alignments. It follows an alignment memory strategy in analogy to translation memories. This means that the module stores each alignment made by the human annotator. If a new tree pair is to be aligned, the module checks whether any token sequence in the current trees has been previously aligned. If so, it suggests the stored alignment to the annotator.

2.2.6 Consistency checks over alignments

Based on the lessons learned in the inter-annotator agreement experiments, we have improved our alignment guidelines. The question is how we can ensure that the guidelines are followed. We would like to determine whether the alignments are complete and consistent, in similarity to quality checks over treebanks.

For consistency checking of the alignments, we checked for all aligned single tokens and all aligned token sequences whether they are aligned in the same way (i.e. with the predicate 'exact' or 'fuzzy') to the same corresponding tokens. We also checked whether the aligned token sequences differ in length (calculated as number of characters). Large length differences point to possibly erroneous alignments.

Additionally, we examined the cases where different types of nodes are aligned across the languages (e.g., when an adjective phrase in one language is aligned with a prepositional phrase in the other). These consistency checks were initially done manually over an extracted table of the aligned token sequences (with their node labels). This allowed us to sort the token sequences according to different criteria and to abstract away from the dense forest of syntactic information and alignment lines in the TreeAligner.

In order to provide faster feedback about internal alignment link consistency, recent versions of the TreeAligner contain a module for consistency checks that are computed during annotation. We distinguish between two different methods, general structural constraints and association probability. Structural constraints are applied regardless of language or corpus, as they express certain invalid subgraphs. One structural constraint that has proven useful to the annotators is branch link locality, which demands that if two phrases p_1, p_2 are aligned, any transitive successor of p_1 may only be aligned to a successor of p_2. While there are some systematic problems with this constraint, it is very effective in exposing inconsistencies among the monolingual annotations and spotting simple mistakes.

The other approach relies on measuring association strength between collocates. In our case, we define an alignment link to be our collocate and check if, given the totality of all alignment links in the current corpus, we can reject it as an improbable hypothesis. For this, we use contingency tables and a χ^2 statistic for non-parametric data.

Another (forthcoming) method for consistency checking of alignment draws on the variation *n*-gram approach for syntactic annotation (Dickinson & Meurers 2003; 2005). It considers alignment as a string-to-string mapping and, treating the target string as a label, examines each source string and their labels, to find inconsistencies in the alignment. Several heuristics are used to filter the set of variations, based on source language context and based on the nature of alignments in aligned corpora. One additional, complementary, method predicts what phrasal node (if any) a constituent should be aligned to, based on the word alignment.

3 Searching parallel treebanks

Since the inception of treebanks, many languages and tools for querying syntactically annotated corpora have been developed. Most of the tools and query languages have been designed for a specific corpus and a specific annotation format.

Our survey focuses on TGrep and TIGERSearch since they were most influential for our own work. We are well aware of related approaches on searching parallel treebanks such as Nygaard & Johannesen (2004) and Petersen (2006).

3.1 Setting the standard: TGrep and TIGERSearch

TGrep2[7] (Rohde 2005) is a tool for querying structured syntax trees in traditional Penn Treebank "bracketed notation". It supports a wide range of structural operators apart from normal dominance or precedence checks and aims for maximal succinctness of corpus queries. Corpora can be queried using a command line interface, either in interactive or batch mode.

TIGERSearch is a powerful treebank query tool developed at the University of Stuttgart by Wolfgang Lezius (cf. König & Lezius 2002; Lezius 2002a). The TIGER query language is similar in expressiveness to TGrep2, but comes with a graphical user interface and highlighting of the syntax trees, frequency tables for objects identified in the query, and support for exporting query result sets. TIGERSearch has been implemented in Java and is freely available for research purposes. Because of its clearly defined input format and its powerful query language, it has become the corpus query system of choice for many linguists.

The TIGER query language is based on feature-value descriptions of all linguistic objects (tokens and constituents), dominance, precedence and sibling relations in the tree, node predicates (e.g. with respect to token arity and continuity), variables for referencing objects, regular expressions over values for varying the query precision, and queries over secondary edges (which constitute a secondary graph level).

A complex query might look like the following example with > denoting direct dominance, >* denoting general dominance, the dot denoting immediate precedence, and the # symbol introducing variables. This query is meant to find sequences of a noun phrase followed by two prepositional phrases where both PPs are attached to the noun in the NP:

(2)
```
#np:[cat="NP"] >* #n1:[pos="NN"]&
#np  >  #pp1:[cat="PP"]&
#n1  .  #pp1&
#pp1 >* #n2:[pos="NN"]&
#np  >  #pp2:[cat="PP"]&
#n2  .  #pp2
```

[7] TGrep can be found at http://tedlab.mit.edu/~dr/TGrep2/

This query says: Search for an NP (call it #np) that dominates a noun #n1 (line 1) and two PPs (lines 2 and 5). #pp1 must follow immediately after the noun #n1 (line 3), and #pp2 must follow immediately after the noun within the #pp1 (lines 4 and 6). This query finds, for instance, the German noun phrase "*Die Anhörung vor dem Konkursgericht zur Offenbarungserklärung*" (English "a hearing on the Disclosure Statement before the Bankruptcy Court") where both PPs are attached to the noun "*Anhörung*" in our SMULTRON economy treebank. Like TGrep2, TIGER is a language for querying monolingual treebanks and thus needed to be extended for our goal of querying parallel treebanks. More generally, the design of the input format influences the design of the query language to a large degree, since it defines what can be queried. For instance, the TIGER object model supports crossing branches, leading to non-terminal nodes whose terminal successors are not a proper substring of the sentence. The TIGER query language thus has special functions for dealing with discontinuous nodes. In contrast, the Penn Treebank formalism does not support crossing branches, and thus TGrep2 has no means for this notion.

3.2 The TreeAligner search module

Merz & Volk (2005) listed the requirements for a parallel treebank search tool. Based on these we have re-implemented TIGERSearch for parallel treebanks and integrated it into the TreeAligner.

We allow the power of TIGERSearch queries on both treebanks plus additional alignment constraints. For example, a typical query could ask for a sentence S dominating a prepositional phrase PP in treebank one. This query can be combined with the constraint that the S in treebank one is aligned to a verb phrase VP in treebank two which also dominates a PP. Such a query would be expressed in 3 lines as:

(3)
```
German treebank  #t1:[cat="S"]  > [cat="PP"]
English treebank #t2:[cat="VP"] > [cat="PP"]
Alignment        #t1--#t2
```

These three lines are entered into three separate input fields in the user interface (cf. the three input fields in the bottom left in Figure 2). Lines 1 and 2 contain the queries over the two monolingual treebanks. Line 3 contains the alignment constraint. Note that the treebank queries 1 and 2 closely follow the TIGERSearch syntax. In particular they allow the binding of variables (marked with #) to specific linguistic objects in the query. These variables are used in

the alignment constraint in line 3. The reuse of the variables is the crucial idea which enabled a clear design of the TreeAligner Search Module by keeping the alignment constraints separate from the queries over the two treebanks.

The above query will find the tree pair in Figure 2 because it matches the alignment between the English VP *closed the front door behind her* and the elliptical German sentence *schloß hinter sich die Tür* (which lacks the subject, but is still annotated as S).

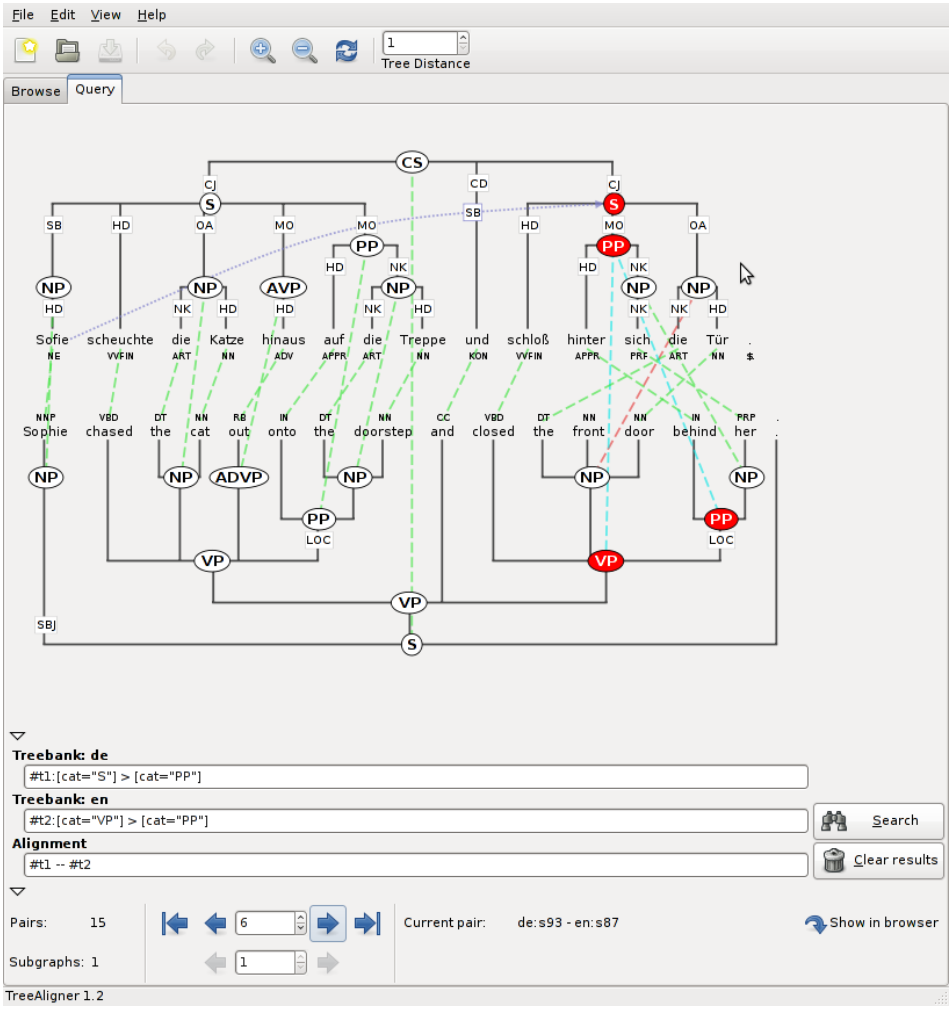

Figure 2: Screenshot of the TreeAligner with the Search Module

The Search Module in the TreeAligner is intended for any parallel treebank where the monolingual treebanks can be converted into TIGER-XML and where the alignment information can be converted to the Smultron XML alignment format. The separation of these parts makes it possible to query each treebank separately as well. The system is divided into a monolingual query facility and an alignment query facility that makes use of the former to perform its job. This design choice made it necessary to (re)implement TIGERSearch, the alignment query facility, and the integration into the TreeAligner.

We chose to reimplement TIGERSearch in Python which influenced the feature set. Even though the implementation of TIGERSearch is well documented (in Lezius 2002a among others) and the Java source codes are available under an Open Source license, the reimplementation is not a trivial task.

The query language for the alignment constraints is kept simple as well. The user can specify that two linguistic objects must be aligned (with exact alignment or approximate alignment). And such constraints can be combined with *AND* statements into more complex constraints. We cannot foresee all options on how a parallel treebank will be queried. We have therefore focused on a clear design of the Search Module rather than overloading it with features. This will facilitate the integration of more features as they are requested by users.

3.2.1 Limitations of the TIGER query language

While certain limitations of query languages are due to the original design and could only be approximated, other valid queries may simply be missing from the query language. Lai & Bird (2004) give a list of seven sample queries that each query formalism should support, regardless of the annotation formalism.

Here we deal with queries that contain universal quantification, i.e. selecting a tree by stating constraints over sets of nodes rather than individual nodes. The sample queries contain two examples where this is needed (Lai & Bird 2004):

Q2. Find sentences that do not include the word *saw*.

Q5. Find the first common ancestor of sequences of a noun phrase followed by a verb phrase.

With the TIGER query language and its implementation TIGERSearch (Lezius 2002a), these queries can only be approximated. The result set generated for the approximated queries will likely contain errors.

Because of the technical nature of the discussion in this section we speak of syntax *graphs* rather than trees. These graphs are directed, acyclic and do not

contain structure sharing (i.e. each node has exactly one direct ancestor). However, due to crossing branches, TIGER trees cannot be stored as nested lists or XML DOM trees directly, which is the usual understanding of trees.

Node descriptions are boolean expressions of feature constraints of the form "(feature=value)". They are the basis for finding nodes (assignments) in the corpus which are then used for the constraint resolution in TIGER queries.

In the TIGER query language, every node variable is implicitly existentially quantified, i.e. the query

(4) `#s:[cat="S"] !>* #w:[word="saw"]`

returns all combinations of two nodes #s, #w in all graphs, such that #s does not dominate #w (the exclamation mark is the negation operator). From the graphs that were requested in Q2, it will only contain the graphs that do contain the word *saw* outside of an S node. All graphs that do not contain any *saw* will not show up in the result set. Another attempt to formulate Q2 is the query

(5) `#s:[cat="S"] >* #w:[word!="saw"]`

which returns all combinations of all words except *saw* that are dominated by an S node.

Lezius (2002b) already acknowledges this restriction and proposes to extend the TIGER query formalism with a universal quantifier and the implication operator. While this is natural given the unification-based evaluation of queries in TIGERSearch, an implementation comes at great computational cost. For each universal quantifier in a query, all nodes in the graph have to be iterated to find out if they satisfy the implication.

3.2.2 Extensions of the query language in the TreeAligner

The solution suggested by Lezius (2002b) builds upon the query calculus that is at the core of TIGERSearch's query evaluation engine. In contrast, the query engine in the TreeAligner is based on node sets, and combinations of nodes from the different sets to satisfy the constraints given in a query. We summarize our approach in the following. More details can be found in Marek, Lundborg & Volk (2008).

In the previous analysis of Q2, we showed that it is possible to rephrase the query using logical equivalents. Therefore, the query "get all S nodes that do not

contain the word *saw* " can be rephrased into "get all graphs where all instances of *saw*, if any, are not dominated by a specific S node". We already demonstrated that it is not possible to express this query within the old formalism, because one of the operands ("all instances of *saw*, if any") is a *set* of nodes rather than a single node. In order to get correct results, we introduce a new type into the query language: the node set.

3.2.3 Node Sets

Traditional node descriptions are still bound by an existential quantifier. A node set, in contrast, is bound by a variable that starts with a percentage symbol:

(6) `#s:[cat="S"] !>* %w:[word="saw"]`

If one operand in a constraint is a node set instead of a node, the semantics of the constraint are changed. In this case, only those assignments to #s are returned where the constraint holds for each node in the node set %w. In the example at hand, only those S nodes are returned that do not dominate any word *saw* in a graph.

The semantics of the node predicates that are defined in the TIGER query language do not change, they still operate at the node level. In the query

(7) `%np:[cat="NP"] & tokenarity(%np, 2)`

the node set %np will contain all NPs whose token arity is 2. In other words, the query matches all NPs that consist of two tokens (e.g. "*Cash flow*" or "*this increase*").

If each variable is bound by an existential quantifier, evaluation of a query (or rather, one term in a query in Disjunctive Normal Form) can terminate as soon as one node description does not yield any results. Graphs that do not contain matching nodes for any of the descriptions will also be disregarded. In the presence of node sets, this behavior is wrong. But graphs without any occurrence of *saw* are valid results for the query. Because of that, the semantics of node descriptions bound to node sets are changed. In contrast to nodes, which may not be undefined, they can be the empty set. If this is the case, a constraint is trivially true.

With this change in place, TIGER is in Cantor's paradise, and no one shall expel it from there. With the basic semantics of set types defined, new set predicates

can be introduced to refine queries. As an example, consider the query "Return all NPs that do not contain any prepositional phrase PP, but only if the graph contains PPs". With empty node sets allowed, the query would have to be written as

(8) `[cat="NP"] !>* %pp:[cat="PP"] & [cat="PP"]`

to ensure that at least one PP exists. As a side effect, the result set contains one entry for each combination of NP and PP in a matching graph, which is slightly more than what the query was supposed to yield. If a node set must not be empty, set algebra operations like cardinality, element containment, union and intersection could be added to TIGER.

Instead of adding support for set operations, we introduced two new predicates that operate exclusively on node sets: *empty* and *nonempty*. The semantics of the predicates can be inferred from the names, and the previous query can be written in a straightforward manner:

(9) `[cat="NP"] !>* %pp:[cat="PP"] & nonempty(%pp)`

This makes it possible to search for graphs that do not contain a specific kind of nodes by using the predicate *empty*. The query

(10) `%w:[pos="DT"] & empty(%w)`

returns all graphs that do not contain any determiner. For example, in our SMULTRON economy treebank we find determinerless English headlines such as "*Group orders grew 8 percent, revenues 10 percent*".

4 Conclusions

We have shown that building parallel treebanks is a complex process. For our SMULTRON treebank we have used separate tools for creating the monolingual treebanks and the alignment. We have improved the process by automatic treebank deepening, interactive visualisation tools, automatic alignment suggestions and consistency checking over trees and alignments.

Still, the process remains burdensome in particular since the alignments constitute semantic annotations. We have shown that good alignment guidelines are

important. Our experiments have helped us to realize that the guidelines need to contain a host of fine-grained alignment rules and illustrative examples to clarify critical cases.

Our alignment work would have been impossible without the TreeAligner, our tool for interactive alignment and searching of parallel treebanks. The alignment module provides for quick drag-and-click alignments and supports various views on the aligned trees. The search module allows powerful treebank searches combining constraints over trees and alignments. We have implemented a query language that was inspired by TIGERSearch but which supersedes TIGERSearch with support for universal quantification.

Future research may go in various directions. We would like to move from a split development of monolingual treebanks and subsequent alignment to a more integrated development process. This should include annotation projection and cross-language consistency checks in every phase of the development process. Moreover recent work on automatic word and phrase alignment should be better integrated into the TreeAligner.

Annotating a parallel treebank is labor-intensive, but it provides such a wealth of cross-language observations that make it worthwhile and rewarding.

Acknowledgments

We gratefully acknowledge financial support for the Smultron project by Granholms stiftelse, Rausings stiftelse and the University of Zurich.

References

Abeillé, Anne. 2003. Building and using parsed corpora. In Anne Abeillé (ed.), *Text, speech and language technology*. Dordrecht: Kluwer Academic.

Ahrenberg, Lars. 2007. LinES: An english-swedish parallel Treebank. In *Proceedings of Nodalida 2007*.

Ahrenberg, Lars, Magnus Merkel & Mikael Andersson. 2002. A system for incremental and interactive word linking. In *Proceedings of LREC-2002*, 485–490.

Ambati, Vamshi & Alon Lavie. 2008. Improving syntax driven translation models by re-structuring divergent and non-isomorphic parse tree structures. In *Proceedings of the Student Research Workshop at the Eighth Conference of the Association for Machine Translation in the Americas*.

Brants, Thorsten, Roland Hendriks, Sabine Kramp, Brigitte Krenn, Cordula Preis, Wojciech Skut & Hans Uszkoreit. 1997. *Das NEGRA-Annotationsschema*. Tech. rep. Saarbrücken: Universität des Saarlandes. http://www.coli.uni-sb.de/sfb378/negra-corpus/negra-corpus.html.

Burchardt, Aljoscha, Katrin Erk, Anette Frank, Andrea Kowalski, Sebastian Padó & Manfred Pinkal. 2006. The SALSA corpus: A German corpus resource for lexical semantics. In *Proceedings of LREC 2006*, 969–974.

Cmejrek, Martin, Jan Curin & Jiří Havelka. 2003. Treebanks in machine translation. In *Proceedings of the 2nd Workshop on Treebanks and Linguistic Theories*, 209–212.

Dickinson, Markus & Walt Detmar Meurers. 2003. Detecting errors in part-of-speech annotation. In *Proceedings of EACL-03*, 107–114.

Dickinson, Markus & Walt D. Meurers. 2003. Detecting inconsistencies in treebanks. In *Proceedings of TLT-03*, 45–56.

Dickinson, Markus & Walt D. Meurers. 2005. Detecting errors in discontinuous structural annotation. In *Proceedings of ACL-05*, 322–329.

Groves, Declan, Mary Hearne & Andy Way. 2004. Robust sub-sentential alignment of phrase-structure trees. In *Proceedings of Coling 2004*, 1072–1078.

Hansen-Schirra, Silvia, Stella Neumann & Michaela Vela. 2006. Multi-dimensional annotation and alignment in an English-German translation corpus. In *Proceedings of the 5th Workshop on NLP and XML (NLPXML-2006): Multi-Dimensional Markup in Natural Language Processing*, 35–42. Trento: ACL.

Hearne, Mary & Andy Way. 2006. Disambiguation strategies for data-oriented translation. In *Proceedings of the 11th Conference of the European Association for Machine Translation 2006 (EAMT 2006)*, 59–68.

Järborg, Jerker. 1986. *SynTag Dokumentation: Manual för SynTaggning*. Tech. rep. Department of Swedish, Göteborg University.

Kruijff-Korbayová, Ivana, Klára Chvátalová & Oana Postolache. 2006. Annotation guidelines for the Czech-English word alignment. In *Proceedings of LREC 2006*.

König, Ekkehard & Wolfgang Lezius. 2002. *The TIGER language – a description language for syntax graphs. Part 1: User's guidelines*. Tech. rep.

Lai, Catherine & Steven Bird. 2004. Querying and updating treebanks: A critical survey and requirements analysis. In *Proceedings of the Australasian Language Technology Workshop 2004*.

Lars, Ahrenberg, Magnus Merkel & Michael Petterstedt. 2003. Interactive word alignment for language engineering. In *Proceedings of EACL-2003*.

Lezius, Wolfgang. 2002a. *Ein Suchwerkzeug für syntaktisch annotierte Textkorpora*. IMS, University of Stuttgart PhD thesis.

Lezius, Wolfgang. 2002b. TIGERSearch – ein Suchwerkzeug für Baumbanken. In Stephan Busemann (ed.), *Proceedings der 6. Konferenz zur Verarbeitung natürlicher Sprache (KONVENS 2002)*, 107–114.

Loftsson, Hrafn. 2009. Correcting a POS-tagged corpus using three complementary methods. In *Proceedings of EACL-09*, 523–531. Athens.

Lundborg, Joakim, Torsten Marek, Maël Mettler & Martin Volk. 2007. Using the stockholm treealigner. In *Proceedings of the 6th Workshop on Treebanks and Linguistic Theories 2007*.

Marek, Torsten, Joakim Lundborg & Martin Volk. 2008. Extending the TIGER query language with universal quantification. In *Proceedings of KONVENS*, 3–14. Athens.

Melamed, Dan. 1998. *Manual annotation of translational equivalence: The Blinker project*. Tech. rep. 98-06, IRCS. Philadelphia PA.

Merz, Charlotte & Martin Volk. 2005. Requirements for a parallel treebank search tool. In *Proceedings of GLDV-Conference, Sprache, Sprechen und Computer / Computer Studies in Language and Speech 2005*. Peter Lang Verlag.

Nivre, Joakim. 2002. What kinds of trees grow in Swedish soil? A comparison of four annotation schemes for Swedish. In *Proceedings of First Workshop on Treebanks and Linguistic Theory*. Sozopol, Bulgaria.

Nivre, Joakim, Koenraad De Smedt & Martin Volk. 2005. Treebanking in Northern Europe: A white paper. In Henrik Holmboe (ed.), *Nordisk Sprogteknologi. Nordic language technology. Årbog for Nordisk Sprogteknologisk Forskningsprogram 2000-2004*, 97–112. Copenhagen: Museum Tusculanums Forlag.

Nygaard, Lynne & Janne B. Johannesen. 2004. Searchtree: A user-friendly treebank search interface. In *Proceedings of 3rd Workshop on Treebanks and Linguistic Theories*, 183–189.

Petersen, Ulrik. 2006. Querying both parallel and treebank corpora: Evaluation of a corpus query system. In *Proceedings of LREC 2006*.

Rios, Annette, Anne Göhring & Martin Volk. 2009. A Quechua-Spanish parallel treebank. In *Proceedings of the 7th Workshop on Treebanks and Linguistic Theories*.

Rohde, Doug L. T. 2005. *TGrep2 User Manual*. http://tedlab.mit.edu/~dr/Tgrep2/.

Samuelsson, Yvonne & Martin Volk. 2006. Phrase alignment in parallel treebanks. In Jan Hajic & Joakim Nivre (eds.), *Proceedings of the Fifth Workshop on Treebanks and Linguistic Theories*, 91–102.

Samuelsson, Yvonne & Martin Volk. 2007. Alignment tools for parallel treebanks. In *Proceedings of GLDV Frühjahrstagung 2007*.

Smith, Noah A. & Michael E. Jahr. 2000. Cairo: An alignment visualization tool. In *Proceedings of LREC-2000*.

Teleman, Ulf. 1974. *Manual För Grammatisk Beskrivning Av Talad Och Skriven Svenska*. Lund: Studentlitt.

Tiedemann, Jörg & Gideon Kotzé. 2009. Building a large machine-aligned parallel treebank. In *Proceedings of the 8th International Workshop on Treebanks and Linguistic Theories*, 197–208.

Tinsley, John, Ventsislav Zhechev, Mary Hearne & Andy Way. 2007. Robust language pair-independent sub-tree alignment. In *Machine Translation Summit XI Proceedings*.

Ule, Tylman & Kiril Simov. 2004. Unexpected productions may well be errors. In *Proceedings of LREC-04*. Lisbon, Portugal.

Volk, Martin, Torsten Marek & Yvonne Samuelsson. 2008. Human judgements in parallel treebank alignment. In *Proceedings of the COLING Workshop on Human Judgements in Computational Linguistics*. Manchester, UK.

Volk, Martin, Sofia Gustafson-Capková, Joakim Lundborg, Torsten Marek, Yvonne Samuelsson & Frida Tidström. 2006. Xml-based phrase alignment in parallel treebanks. In *Proceedings of EACL Workshop on Multi-dimensional Markup in Natural Language Processing 2006*.

Zhechev, Ventsislav. 2009. *Automatic generation of parallel treebanks: An efficient unsupervised system*. School of Computing at Dublin City University PhD thesis.