School of Mathematics and Systems Engineering

Reports from MSI - Rapporter från MSI

Reconstruction of the Swedish Treebank Talbanken

Jens Nilsson
Johan Hall

Jun
2005

MSI
Växjö University
SE-351 95 VÄXJÖ

Report 05067
ISSN 1650-2647
ISRN VXU/MSI/DA/R/--05067/--SE

Reconstruction of the Swedish Treebank Talbanken

Jens Nilsson and Johan Hall

June 10, 2005

Contents

1 Introduction

Data-driven parsing techniques have a number of advantages over rule-based parsing techniques, such as fast development time, broad-coverage and robustness. Treebanks, collections of syntactically annotated sentences, are important resources for data-driven parsers. When developing a parser for Swedish one needs a treebank containing Swedish sentences, but currently there is a lack of Swedish treebanks of substantial size. This holds for the other Nordic languages too, with Danish as an exception. The absence of Swedish treebanks is remarkable considering that two corpora of Swedish text augmented with syntactic annotation have been created, one as early as 1974 named Talbanken (Einarsson [3], [4]), and another in the 80's named Syntag (Järborg [5]). Unfortunately, the annotation formats of these resources make them cumbersome to use for modern treebank tools and parsers. In a way, Sweden can be regarded as a pioneer in this area, but thereafter the work with creating new treebanks has decreased considerably.

The price to pay for the fast development time of data-driven parsers is that the creation of treebanks is time consuming and costly. On the other hand, treebanks have other areas of use than to provide data-driven parsers with data. They can be used as evaluation data for parsers in general, and for a number of other purposes (Nivre et. al. [11]). As long as there does not exist a large enough Swedish treebank, one can reuse the available resources, despite their limitations.

The aim of the project that this report presents is to recycle, or reconstruct, Talbanken to a more convenient format adapted to modern treebank tools and parsers. The motivation for this is not only to make a previously inaccessible linguistic resource available for research purposes. The second aim is to realize a possible encoding format of a theory-supporting treebank, in line with Nivre [10]. In other words, we strive to get some hands-on knowledge. The idea behind using a theory-supporting treebank (source) is that it shall be possible to extract several theory-specific treebanks (targets) from it. The source should contain as little redundancy as possible by providing information, that is needed by several targets, only once. Moreover, information that can be derived using a set of transformation rules, one set for each target, is left out. Such rules are then applied in order to create a theory-specific treebank by means of extraction and derivation of information from the source. With this in mind, we will in this project raise the question if it is possible to transform Talbanken's original annotation into an encoding format that can be used to extract both dependency trees and phrase structure trees, and if it is possible to leave out certain kinds of information that instead can be acquired by applying transformation rules (i.e. inference).

1.1 Talbanken

Talbanken is divided into four parts, which together comprise close to 320,000 words. The four parts are:

- "Professionell prosa" (Professional prose), having about 85,000 words.
- "Gymnasistsvenska" (Swedish by senior high school students), having about 85,000 words.

- "Samtal och debatt" (Conversation and debate), having about 75,000 words.
- "Boråsintervjuerna" (The Borås interviews), having about 75,000 words.

The first two are written language and the last two are spoken language, all syntactically annotated. "Professionell prosa" (P) is a collection of texts from e.g. brochures and newspapers, while "Gymnasistsvenska" (G) contains essays written by senior high school students on the topic "Familjen och äktenskapet än en gång" (Family and marriage once again). The spelling errors and other linguistic errors made by the students are not corrected. The first part of the spoken material, "Samtal och debatt" (SD), is transcriptions of conversations and debates on different topics, whereas "Boråsintervjuerna" (IB) comprises interviews with different persons on the topic of immigrants.

The primary goal of the project is to reconstruct P and make two versions of it available, one based on dependency and the other on constituency. The reconstruction program was created in order to make the quality of these versions of P as high as possible. The focus of this report will therefore be on the reconstruction of this part. The project was divided into three major steps, which is also reflected in the structure of the report. The first step (section 2) handles the reconstruction from the original annotation format, while having the notion of a theory-supporting treebank in mind. Step two (section 3) deals with the transformation of the output from step 1, into a phrase structure representation by applying a set of labeling rules. Step 3 (section 4) transforms the output of step 1 into a representation based on dependency grammar.

Since P was the major aim of the project, the reconstruction of the three other parts can be regarded as an additional bonus. Only a few adjustments were needed in the reconstruction program in order to take care of the peculiarities in G, SD and IB. Especially the two parts containing spoken language required a few minor changes in the program. These are described in Appendix C.

The idea throughout the whole project has been to stay as close to the original annotation as possible, as a kind of fidelity idea. If the annotation suggests one type of syntactic construction, then we stay with that construction throughout the stages in the process, i.e. we introduce changes and transformations only when necessary. A number of refinements are applied especially in the last phase of step 1, but besides that all kinds of unforced changes in the annotation are avoided in the reconstructions program. These might instead be applied as post-processing steps on the target treebanks. However, creating possible post-processing steps are not part of this project, but are among other things discussed in section 5.

Finally, we have chosen to convey the original encoding format to the encoding format Tiger-XML [9] in all steps throughout the whole transformation process, since there exist several convenient tools for treebanks encoded in Tiger-XML. These include searching, querying and visualization of trees. Tiger-XML is well suited for our needs to encode phrase structure, but is not directly applicable to dependency trees. However, it is possible to use Tiger-XML in order to capture dependency based structures given a predefined convention. The convention we comply with is defined in the document [6].

```
          1  1    1                        4       5           6
1   5 7   0  3    8                        2       0           2
-----------------------------------------------------------------
P10111025001      0000                     <<      GM          024
P10111025002      *DESSUTOM                ABOC    +A          024
P10111025003      HÖJS                     VVPS  PAFV          024
P10111025004      ÅLDERGRÄNSEN             NNDDSS  SS          024
P10111025005      TILL                     PR      OAPR        024
P10111025006      18                       RO      OADT        024
P10111025007      ÅR                       NN      OA          024
P10111025008      .                        IP      IP          024
```

Figure 1: "In addition, the age limit is raised to 18 years."

2 Reconstruction of Talbanken

The annotation in MAMBA consists of two layers. One is a lexical analysis layer, comprising part-of-speech information and morphological features, and the other is a syntactic analysis layer having grammatical functions. This section will start with a presentation of the important aspects of the MAMBA-format. After that, the difficult issue of sentence splitting in MAMBA will be addressed and our solution will be presented. Thereafter, we will deal with the identification of phrases. Moreover, the management of coordination in the MAMBA-format is not trivial and is dealt with separately in the transformation program. This is presented in section 2.5. Finally, this section ends with a discussion about the necessary refinements of the phrase trees.

2.1 The MAMBA-format

From a modern point of view, the MAMBA-format may seem a little obscure and odd. But it should be kept in mind that at the time of its creation, Talbanken was originally placed on punch cards having a limited upper line length. This does of course convey limitations on the encoding format. Talbanken was later transformed into a more convenient electronic form, but the limitations remained. A brief explanation of the format follows below. For a more extensive description, see Teleman [14].

The sentence in figure 1 is picked from Talbanken. The first two lines are not part of the annotation. They just mark important starting indices in the MAMBA-format.

TX: 1-4

The first four letters identify the text. The sentence in figure 1 is part of the text *P101*, where *P* stands for professional prose. A dummy word with the grammatical function TX in the first SC-column (see below) marks the beginning of a new text and the text-id is at the same time increased by one.

ST: 5-6

The following two digits represent the paragraph that the sentence below it belongs to. In figure 1, it is part of paragraph 11. It is reset to 1 for each new text, and each new paragraph begins with a dummy word marked *ST*.

MS: 7-9

The digits in the columns from 7 to 9 denote the index of the macro syntagm. It is also reset to 1 each time a text ends and a new one begins. A new macro syntagm starts with a dummy word which is marked *MS*.

WN: 10-12

Each word in the MAMBA-format has its own word-id. In fact, not only the words, but all lines have such an id. Note for instance that the first line in the example above is a dummy word with word-id 1. This id is unique within each graphical sentence (see column 62-64 below), and is therefore reset for each new graphical sentence.

SC: 13-17

The five *SC*-columns contain important information concerning the syntactic hierarchy. Since the space for syntactic information is rather limited (see the *SY*-columns below), a set of reserved dummy words can introduce additional hierarchy. This is encoded in the first four columns, which makes it possible to nest additional hierarchy as deep as four times. For instance, dummy words are placed in front of subordinate clauses, but they are used for other purposes as well. The fifth column keeps track of the number of grammatical functions that currently are added by dummy words, which can be an integer value from 0 to 9. It can also be the case, as in the example in figure 1, that the five columns are empty which means that the hierarchy depth is 0. In a way, the information in the fifth column is redundant, since all the necessary information about the syntactic hierarchy is kept in the first four columns. This information has therefore probably been included in order to do consistency checks. How the *SC*-columns are used to encode hierarchy is presented more closely in section 2.4.

WF: 18-41

This interval contains the word form. If it is a hierarchic dummy word, the first four letters contain the digits that occupy the SC-columns of the following words. If it is another kind of dummy word, the word form contains four zeros, i.e. *0000*, which is the case for the first line in figure 1. Since the interval has a fix maximum length of 24 characters, any word longer than that is truncated. So this is a clear example where the drawbacks of the annotation format, due to the limitation of the punch cards, becomes apparent. Also, only capital letters were written to the punch cards. Therefore, all small letters were mapped to its corresponding capital letter. All letters that really should be capital were preceded by an asterisk character. This is the case for the letter *D* in the word *Dessutom* in figure 1.

LX: 42-49

In these column, parts-of-speech and other morphological features are encoded. The eight columns are divided into four blocks with two letters each. The left-most block comprises the coarse-grained parts-of-speech (approx. 40 categories), whereas the other blocks contain more specific lexical and morphological information. An overview of the coarse-grained parts-of-speech is presented in Appendix A. For a more detailed description of these lexical and morphological categories, consult chapter 11 in Teleman [14].

SY: 50-61

These six blocks with two letters each are used to assign grammatical functions to the words. All words having the same syntactic function on a specific level are part of the same phrase. For example, the word *åldersgränsen* (the age limit) is the only word in figure 1 with the category *SS* (for subject) in the the left-most *SY*-column, and will thus form a phrase of its own. Also, the words *till*, *18* and *år* have the same syntactic category in the first column, i.e. *OA* (adverbial), so they form a phrase. Within this phrase, *18* is the only word that lacks additional information in the second column. This is the way that heads in phrases are marked. The two other words have the categories *PR* and *DT*, and will form two distinct phrases within the phrase. An extensive presentation of the grammatical functions is found in Teleman [14], and a list of the ones occurring in *P* is also shown in table 6 (appendix B).

GM: 62-64

The last three columns comprise three digits, identifying the graphical sentence that the word is a part of. New graphical sentences begin with a dummy word marked *GM* or *GX*. It is important to note that graphical sentences are not the same as macro syntagms. The division into graphical sentences is in principle independent of the division into macro syntagms. Somewhat simplified, MAMBA adopts the strategy that a graphical sentence is a sequence of words ending with a full stop, question mark or explanation mark. Furthermore, it is often the case that lists in a text annotated in the MAMBA-format are encoded as a single *GM* or *GX*, even in the case where the list items are or can be regarded as sentences themselves. So a graphical sentence can span over several macro syntagms, and a macro syntagm can span over several graphical sentences.

2.2 From MAMBA to bare phrase structure

The above subsection gave a short description of the MAMBA-protocol and hopefully provides enough information to understand the rest of this section. The remaining subsections will explain the process of creating a version of the data that we call bare phrase structure. The MAMBA-protocol does not contain information about phrase labels. Our definition of a bare phrase structure is a syntactic annotation based on constituency with unspecified phrase labels. Here we will outline the course of action in the reconstruction program. In principle, it consists of five phases.

First, the input module takes a file containing texts encoded in the MAMBA-format, and extracts the necessary information described above, line by line. It returns a list of MAMBA-lines. Secondly, another module takes such a list and categorizes each item as either (1) a non-grammatical dummy word, (2) a hierarchic dummy word, or (3) an actual word. The following categorization conditions are used:

- Non-grammatical dummy words are recognized by the fact that they have the grammatical function *TX*, *ST*, *GX*, *GM*, or *MS*, which mark the beginning of a new text, paragraph, graphical sentence containing other graphical sentences, graphical sentence, or macro syntagm, respectively. These are discarded due to the fact that they do not contribute to the syntactic analysis. The indices of *GX*, *GM* and *MS* are in some cases used to split sentences (see next subsection), but the dummy words are not part of the actual sentence. Also, one of the peculiarities in the MAMBA-format is that a dummy word occupies the position of the relative pronoun *som* in case it is absent. That is, if *that* in the sentence *The girl [that] the boy met* is absent, then a dummy word with the word form == marks its position, i.e. *The girl == the boy met.* These dummies are also removed.

- All hierarchic dummy words are identified when looking at the first position in the LX-column. If a line in the MAMBA-format has one of the dummy categories listed in table 5 in Appendix A at this position, then it is a hierarchic dummy word. These words are not part of the sentence, but will influence the syntactic hierarchy of the following actual words.

- All other items in the list of MAMBA-lines are treated as actual words. For these items, all letters of the word form are mapped to their corresponding lower case, unless a letter is preceded by an asterisk. In that case the asterisk is removed and the capital letter is preserved.

The third module of the reconstruction program tries to identify phrases in the syntactic hierarchy captured by the grammatical functions, leaving the node labels of the phrases unspecified. In the fourth phase, we reduce the lexical categories. The complete list of distinct lexical categories is too large for our purposes. This phase can been seen as optional, but the lexical categories we map to affect the way the labeling of non-terminals is done (section 3). A complete listing of the tags is shown in figure 4, and a more extensive description is presented in Teleman [14]. Finally, we refine the output from phase four, since it is not well-suited for later transformations.

2.3 Sentence splitting

Sentence splitting in MAMBA is not a simple matter. There are at least two candidates: the indices in the *GM*-columns and *MS*-columns. However, none of them alone is an optimal solution. If only GM is used, then one will not split graphical sentences containing several sentences divided by e.g. semicolon, or a list (marked as a single graphical sentence) containing several sentences. If, on the other hand, one uses MS as the sentence splitter, the text will in many cases be too fragmented. MS-dummies are for example inserted for coordinated sentences such as *John sings and Sue plays.*. In this example, *John sings* is one

```
P10216058001     0000                    GM      OOGM        054
P10216058002     *TOALETTARTIKLAR        NN  SS  OO          054
P10216058003     (                       IR      OOIR        054
P10216059004     0000                    <<MX    MS          054
P10216059005     KAN                     QVPS    FV          054
P10216059006     ÄVEN                    ABOC    +A          054
P10216059007     KÖPAS                   VVIV  PAIV          054
P10216059008     I                       PR      RAPR        054
P10216059009     MARKETENTERIET          NNDD    RA          054
P10216058010     )                       IR      OOJR        054
P10216058011     -                       IT      OOIT        054
P10216058012     1000                    AF      OOAN        054
P1021605801310002RAKHYVEL                NN  SS  SP          054
P102160580141000220CH                    ++OC    ++          054
P1021605801510002RAKBORSTE               NN  SS  SP          054
P1021605801610002-                       IT      IT          054
...
```

Figure 2: "Toilet commodities (can also be bought in the shop) - razor and razor brush ..."

macro syntagm and *and Sue plays.* another, which we believe is an incorrect sentence split.

Even worse, it is possible to insert a macro syntagm within another macro syntagm, as in the example in figure 2. The graphical sentence beginning with macro syntagm 58 (column 7-9) is interrupted by another macro syntagm with index 59. The first one then continues when macro syntagm 59 ends. In this example, it would be more natural to keep the graphical sentence as one unit. This feels like a better solution compared to inserting a sentence split on each side of the inner macro syntagm, or to lift the inner one out and merge the two parts with identical macro syntagms indices.

A slightly more complex solution is therefore applied, with a combination of both candidates. The first rule says that dummies marked *GM* or *GX* always function as sentence splitter. In addition, one would like to divide long graphical sentences composed of several syntactically distinct macro syntagms separate by punctuation marks like colon and semi colon (and in some cases even full stops, exclamation marks and question marks). These syntactic delimiters have macro syntagm index 0 in the MAMBA-format. So sentences are split when these two conditions, punctuation mark and index 0, occur together.

This is not a perfect solution, but at least an adequate one. Some sentence splits will still look strange, which is sometimes due to inadequate splitting conditions in our implementation and sometimes to a questionable encoding in the MAMBA-format. More work could surely be put into this matter in order to improve the sentence splits. Another solution could of course be to revise the data.

Before concluding this subsection, some important conversion decisions need to be sorted out. Given the above strategy to split sentences, it may happen that a sentence contains more than one MS-unit. These units at least resemble

```
P10835069001     0000                          <<       GM          074
P10835069002     *DEN                          PODPHH   SS          074
P10835069003     1000                          RC       SSET        074
P1083506900410002SOM                           PORPHH   SS          074
P1083506900510002VÄNTAR                        VVPS     FV          074
P108350690061000 2MED                          PR       OAPR        074
P10835069007100021100                          IF       OA          074
P1083506900811003ATT                           IM       IM          074
P108350690091100 3TA                           VVIV     IV          074
P10835069010110 03UT                           ABZA     PL          074
P1083506901111003ÅLDERSPENSIONEN               NNDDSS   OO          074
P1083506901210002TILL                          PR       TAPR        074
P1083506901310002EFTER                         PR       TATAPR      074
P10835069014100026 7-ÅRSMÅNADEN                NNDDSS   TATA        074
P10835069015     FÅR                           FVPS     FV          074
P10835069016     HÖGRE                         AJKP     OOAT        074
P10835069017     PENSION                       NN       OO          074
P10835069018     .                             IP       IP          074
```

Figure 3: "The one who waits to take out his age pension until after the month he turns 67 years will get higher pension."

sentences of their own, and should be treated accordingly in the syntactic hierarchy. Thus, whenever the MAMBA-annotation mark the start of a new macro syntagm (as in line 4, figure 2), all grammatical functions for the words within that macro syntagm (line 5 to 9) will be preceded by a unique pseudo function *MSx* (where x is numeric value). This works similar to the expansion of dummy words introducing hierarchy. Both of them are described more closely in the next section. In the sentence *John sings and Sue plays*, the first two words would be preceded by *MS1*, and the following three by *MS2*. Furthermore, for consistency reasons, all words will be part of exactly one *MSx*-unit. This means that those words that are not part of an internal macro syntagm are instead part of the default *MSx*-unit, usually *MS1*. The example in the following section will make this more clear.

2.4 Phrase identification

The identification of phrases is the core of the reconstruction program. We will begin by discussing how the MAMBA-format uses the dummy words introducing hierarchy.

The expansion of dummy words

In the example in figure 3, there are two such dummy words, positioned at the third (RC) and seventh (IF) line.

The word form of the first dummy word has the value *1000*. This value is important for the syntactic structure of the following words. Especially, all non-zero digits are significant, which in this case applies only to the first digit, that

```
 1  MS1 SS              Den
 2  MS1 SS ET SS        som
 3  MS1 SS ET FV        väntar
 4  MS1 SS ET OA PR     med
 5  MS1 SS ET OA IM     att
 6  MS1 SS ET OA IV     ta
 7  MS1 SS ET OA PL     ut
 8  MS1 SS ET OA OO     ålderspensionen
 9  MS1 SS ET TA PR     till
10  MS1 SS ET TA TA PR  efter
11  MS1 SS ET TA TA     67-årsmånaden
12  MS1 FV              får
13  MS1 OO AT           högre
14  MS1 OO              pension
15  MS1 IP              .
```

Figure 4: The expanded grammatical functions for the example in figure 3

is *1*. This shall be interpreted as follows: all words in the rest of the graphical sentence, having the digit *1* as the first digit in the *SC*-columns (i.e. column 13), are influenced by the grammatical function(s) of the dummy word. This holds for all words from line 4 to 14 in the example above, and the grammatical functions of the dummy word (*SS ET*) are inserted in front of the grammatical functions for these lines. For instance, the word *som*, having the grammatical function *SS*, is preceded by *SS ET*, which entails the sequence *SS ET SS*. In other words, the dummy word introduces a relative clause (denoted *ET*) spanning all the way down to line 14. It modifies the subject (denoted *SS*) at line 2 as a kind of attribute. So these are the reasons why the dummy word has the grammatical functions *SS ET*.

Dummy words introducing hierarchy can be nested, which is the case at line 7. This one has the grammatical function *OA* and the word form *1100*. Again, all non-zero digits are of importance. All words in the graphical sentence having the digits *11* in the beginning of the *SC*-columns are preceded by *OA*. This holds for line 8 to 11. However, since this dummy word is positioned "inside" the dummy word at line 3, the dummy word at line 7 attaches its grammatical function(s) before the former dummy word. Thus, the grammatical functions for a word such as *att* at line 8 will after the insertion be *SS ET OA IM*. Note also that line 12 to 14 are affected by the dummy word at line 3 but not by the dummy word at line 7, since their *SC*-digits (*1000*) equal the digits for the first dummy word. Only *SS ET* will therefore be attached to these words.

To conclude, the complete expansion of the grammatical functions for the example in figure 3 looks as in figure 4. As mentioned in the previous section, one pseudo function, identifying which macro syntagm it belongs to, is also added. This is done after the ordinary insertion presented above and is the reason why all lines in figure 4 begin with *MS1*.

The identification of phrases

After the insertion described above, the module for identifying phrases takes over. As long as there are no coordination structures in a sentence, the task for this module is quite straightforward. This subsection deals with the identification of phrases when there is no coordination involved. Coordination will instead be presented in the next subsection.

The core of this module is in principle a recursive method that takes two parameters: a list of MAMBA-words with the expanded grammatical functions and the current analysis depth. When the words are analyzed and the inner hierarchy is identified, the function returns a non-terminal node. The node has a right-hand side of non-terminals and terminals comprising the identified hierarchy of expanded MAMBA-words. The current analysis depth is the same as the index in the list of grammatical functions.

This function is called once for each sentence, and all words are fed to it together with depth 1 (it starts by looking at all the words' left-most grammatical functions). The main principle that is used to identify phrases within a list of words is to make everything with the same grammatical function at the current analysis depth part of the same phrase. When a list of words has been partitioned with respect to the grammatical function, the function evokes itself with the words in each such partition, and adds one to the current analysis depth. The returned non-terminal is then inserted in the right-hand side. All words lacking a grammatical function at the current analysis depth are treated as terminals and added to the right-hand side.

To make things more concrete, have a look at the sentence in figure 4. The analysis starts by sending the whole sentence to the recursive function with depth 1. All words at this depth have the default macro syntagm *MS1*, so the function evokes itself by providing it with all the words and adding one to the current depth (i.e. 2). When this function is completed, it returns a non-terminal with the syntactic structure for all the words with macro syntagm *MS1* at depth 1. It is then added to the right-hand side of the evoking function.

For all the words at depth 2 having *MS1* at depth 1, the recursive function identifies four partitions in the second column: *SS* (1-11), *FV* (12), *OO* (13-14) and *IP* (15). These four are therefore identified as phrases, and the function calls itself once for each of them and the depth is increased to 3. Within *SS*, the word *Den* lacks additional grammatical functions, which entails that it is treated as a terminal. Besides this terminal, *SS* contains only one partition (*ET*), and the recursive call is repeated. So the phrase identified by the grammatical function *SS* starts with one terminal followed by one non-terminal. Within *ET* four phrases are found (*SS*, *FV*, *OA*, *TA*), which in turn are treated as phrases and the function calls itself again four times. The identification of terminals and nonterminals continues in this fashion, and the complete bare phrase structure tree representing the sentence is depicted in figure 5.

Moreover, all phrase labels are given the unspecified value *??*, and their edge labels (to its parent node) are assigned the grammatical function that identified the phrase. When the function identifies a terminal, the word form and the first category in the *LX*-columns are retrieved. Also, since terminals are identified by the lack of a grammatical function, all terminals' edge labels are left unspecified. They are also denoted using two question marks, *??*.

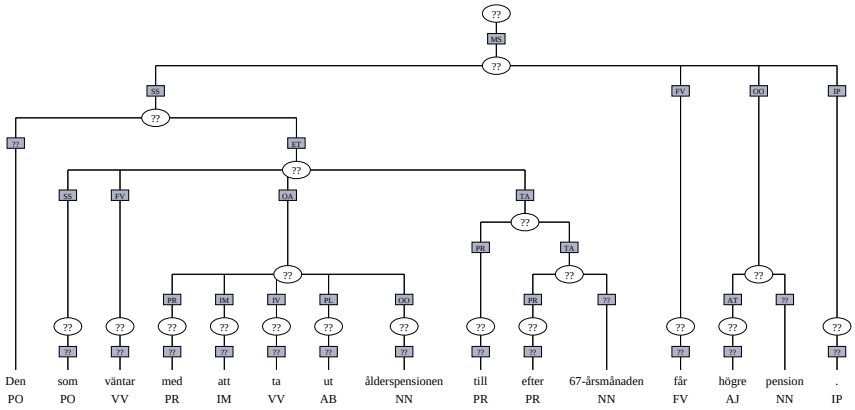

Figure 5: The bare phrase structure tree of the sentence in figure 3

```
P20622065001    0000                <<  BT  GM          064
P20622065002    *DET                PODP    SSDT        064
P20622065003    SKYDDET             NNDD    SS          064
P20622065004    BORDE               VVPT    FV          064
P20622065005    HA                  HVIV    IV          064
P20622065006    KOMMIT              QVSN    IX          064
P20622065007    FÖR                 PR      TAPR        064
P20622065008    LÄNGE               ABZA    TA          064
P20622065009    SEDAN               ID      TAPR        064
P20622065010    .                   IP      IP          064
```

Figure 6: Discontinuous construction: "That protection should have come a long time ago."

Discontinuous phrases and alphabetical displacement

The above presentation of phrase identification is not restricted to unbroken sequences of grammatical functions. It is fully acceptable that a grammatical function within a phrase is scattered, having other grammatical functions (and/or words without such information) in between. They will still together constitute a phrase in the MAMBA-format.

For instance, have a look at the temporal adverbial at line 7 to 9 in figure 6. It contains two *PR*, *för* and *sedan*, which together form a discontinuous multiword preposition. It is divided by the word *länge* which lacks a grammatical function at depth 2. It will thus not be a part of the phrase identified by *PR*, but will instead constitute a terminal node by itself. The bare phrase structure tree of the sentence is depicted in figure 7.

Sometimes it happens that a phrase (or clause) contains two or more distinct components that have the same grammatical function. For example, it may have two space adverbials. To let both of them be assigned the same grammatical function is not possible, not even if the two space adverbials are separated by

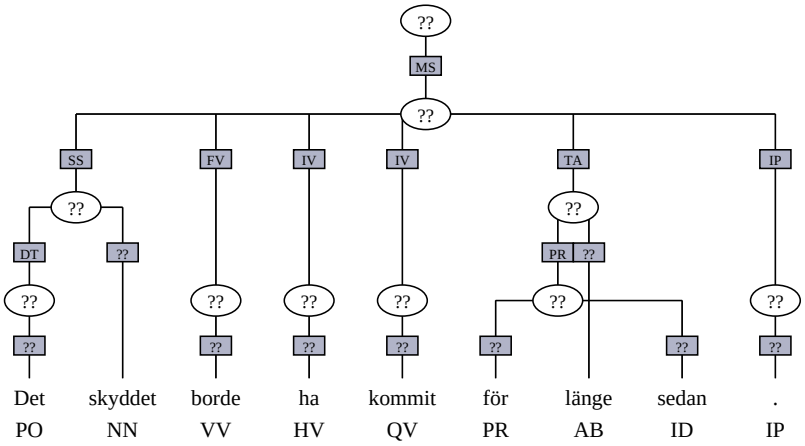

Figure 7: The bare phrase structure tree of the sentence in figure 6

other grammatical functions. They would then not form distinct phrases, because of the above described treatment of discontinuity. The MAMBA-format applies a cunning solution to this problem, which we call alphabetical displacement of grammatical functions. The idea is that new components of the same kind as some other component within the same constituent change their grammatical functions. The first letter of its grammatical function remains the same, whereas the second letter is substituted by the next letter in the alphabet.

Such a situation occurs in the sentence in figure 8. Two time adverbials modify the finite verb in the main clause, *sista gången* (*for the last time*) and *i samband med 1971 års taxering* (*during the taxation of 1971*). The first one is assigned the grammatical function *TA*. In order to distinguish the second time adverbial from the first one, the latter has to be altered. Thus, the grammatical function's second letter of the second time adverbial, e.g. *A*, is replaced by *B*, and the new grammatical function will then be *TB*. Moreover, if the sentence had had one additional time adverbial, it would have been alphabetically displaced once more and therefore been assigned *TC*, and so forth.

When the function for identifying phrases has distinguished the two time adverbials in figure 8, the alphabetical displacement of the second one is no longer needed. An alphabetically displaced grammatical function is mapped to its corresponding non-displaced grammatical function. The others are mapped to themselves. Thus, *TB* is mapped to *TA*, and *TA* to *TA*. This can be seen in the bare phrase structure tree in figure 9 where both the time adverbials have the same grammatical function *TA*.

The sentence in figure 6 also contains an alphabetical displacement. The two non-finite verbs *ha* (*have*) and *kommit* (*come*) both have the same grammatical function, *IV*. Since they are analyzed as being part of two distinct grammatical components in the annotation, the latter word was instead assigned *IX*. The complete list of all the mappings is shown in table 6, appendix B.

It is important to remember when talking about alphabetic displacement that the second component in a coordination is not alphabetically displaced.

P10106012001	0000	<<	GM	011
P10106012002	*FRIVILLIG	AJ	SSAT	011
P10106012003	SÄRBESKATTNING	VN	SS	011
P10106012004	TILLÄMPAS	VVPSSMPAFV		011
P10106012005	SISTA	POSU	TADT	011
P10106012006	GÅNGEN	NNDD	TA	011
P10106012007	I	PR	TBPR	011
P10106012008	SAMBAND	ID	TBPR	011
P10106012009	MED	ID	TBPR	011
P10106012010	1971	RO	TBDTDT	011
P10106012011	ÅRS	NN	GGTBDT	011
P10106012012	TAXERING	VN	TB	011
P10106012013	.	IP	IP	011

Figure 8: ”Voluntary individual taxation will be applied for the last time during the taxation of 1971.”

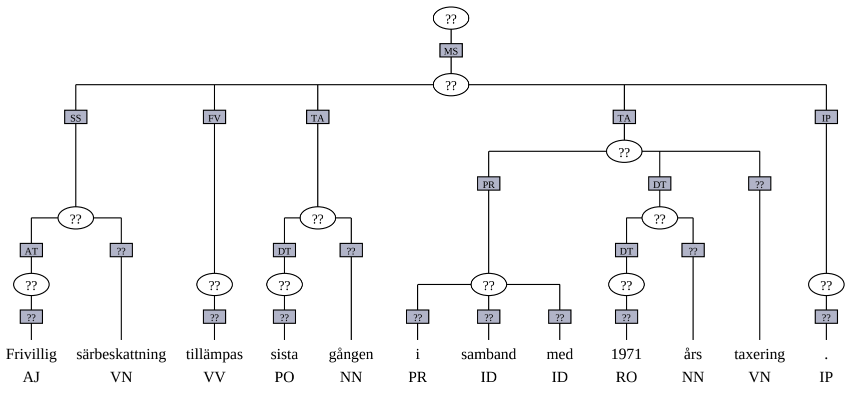

Figure 9: The bare phrase structure tree of the sentence in figure 8

Coordination in the MAMBA-format will be the main focus in the next subsection.

2.5 Management of coordination

Coordination in the MAMBA-format is quite cumbersome. The shallow constituent analysis in the MAMBA-format creates problems, especially for the coordination of components that are not complete phrases or clauses in the MAMBA-analysis, such as *all [red cars] and [blue bikes] in Sweden* or *[John saw] and [Sue heard] Mary*. Since coordination is tricky, the program uses a different approach to handle it, compared to the general identification of phrases.

Finding conjunctions

The program starts by identifying all conjunctions within each identified phrase in a preprocessing step. Two types of words in the MAMBA-format function as conjunctions. The criteria for identifying a word as a conjunction within a phrase at a specific depth are:

- An *SC*-column has the grammatical function *++*.
- An *SC*-column has the grammatical function *IK* and the first two columns in the *LX*-data contain the lexical categories *IK* and *++*.

The first type is either a real coordinating word, or a coordinating dummy word which is used to mark that two components are coordinated, but where the intermediate conjunction is absent. The word *och* at line 3 in figure 10 is a case of the former, which coordinates two nouns within the phrase identified by the grammatical function *SS*. Line 13 in the same figure is an example of a dummy conjunction. These are distinguished from real conjunctions by the fact that they have two plus characters (*++*) as the word form. Dummy words are treated no different than ordinary conjunctions. However, since they are not actually part of the original text, they are removed in the next step.

By the second criterion, the comma at line 11 is also identified as a conjunction. So within the adverbial at line 8 to 14 there are two conjunctions. Moreover, the annotation makes a difference between punctuation marks which function as conjunction and those which do not. For example, if the comma at line 11 did not have a coordinating task, it would not have *++* in the second *LX*-column. The line would instead look like this:

```
P10614022011      ,                          IK       RAIK         021
```

The second criterion would then no longer be applicable and the comma would thus be handled as an ordinary word by the phrase identification module.

Properties of conjuncts

Coordination in the MAMBA-format can be divided into two main groups, true coordination and pseudo-coordination. The first group is divided into four subgroups:

- Coordination of macro syntagms

```
P10614022001    0000                    <<      GM          021
P10614022002    *BÅDE                   ABZA    SSVA        021
P10614022003    PAKET                   NN      SS          021
P10614022004    OCH                     ++OC    SS++        021
P10614022005    ADRESSKORT              NN  SS  SS          021
P10614022006    KÖRS                    VVPS  PAFV          021
P10614022007    UT                      ABZA    PL          021
P10614022008    TILL                    PR      RAPR        021
P10614022009    ADRESSATENS             NNDDHHGGRADT        021
P10614022010    BOSTAD                  NN      RA          021
P10614022011    ,                       IK++    RAIK        021
P10614022012    KONTOR                  NN      RA          021
P10614022013    ++                      ++      RA++        021
P10614022014    ETC                     ABOC    RA          021
P10614022015    SENAST                  AJSU    TAPR        021
...
```

Figure 10: "Both the package and the address card are delivered to the home address, office etc. before ..."

- Coordination of separate phrases of the same kind
- Coordination of components that contain more than one phrase
- Coordination within noun phrases

The handling of coordinated macro syntagms is no different from the general treatment of macro syntagms. So if there are two (or more) coordinated macro syntagms with a conjunction in between within the same sentence, they will be captured in one non-terminal node each, all of them with the edge label *MS*. The conjunction will thus be part of the macro syntagm that is given by the annotation, which usually is the second conjunct. Furthermore, the third subgroup is no different from the first in the sense it is handled by the ordinary phrase identifier. The sentence in figure 11 contains an instance of the third subgroup. The conjunction at line 12 coordinates *acceptera kamraterna* (*IV* and *OO*) with *visa hänsyn mot dem* (*IV*, *OO* and *OA*), where neither can be considered to be full phrases or clauses. The solution adopted in MAMBA is to treat the three grammatical functions of the second conjunct as a so called coordination phrase, *+F*. An additional layer is inserted in the annotation, which belongs to the phrase of the first coordinated component. The conjunction will be a part of the coordination phrase. The bare phrase structure tree created given the sentence is shown in figure 12.

In fact, the program treats all phrases beginning with a conjunctions as instances of the third case, and consequently does not regard them as coordination at all. This implies that a conjunction in the beginning of a sentence, as in *Och den förblir mindre ...* (*And it remains smaller ...*), will receive the phrase structure tree in figure 13.

Moreover, all kinds of pseudo-coordination mentioned in the beginning of this section do not cause the program any additional work besides the normal

```
P10208033001      0000                 !!      GM          029
P10208033002      *FÖRSÖK              VVIP    FV          029
P10208033003      REDAN                ABZA    CA          029
P10208033004      FRÅN                 PR      TAPR        029
P10208033005      FÖRSTA               POSU    TADT        029
P10208033006      DAGEN                NNDD    TA          029
P10208033007      1000                 IF      OO          029
P1020803300810001ATT                   IM      IM          029
P1020803300910001ACCEPTERA             VVIV    IV          029
P1020803301010001KAMRATERNA            NNDDHH  OO          029
P10208033011100011100                  +F      +F          029
P10208033012110010CH                   ++OC    ++          029
P1020803301311001VISA                  VVIV    IV          029
P1020803301411001HÄNSYN                NN      OO          029
P1020803301511001MOT                   PR      OAPR        029
P1020803301611001DEM                   PODPHHAAOA          029
P10208033017      .                    IP      IP          029
```

Figure 11: "Try already from day one to accept your friends and show consideration to them."

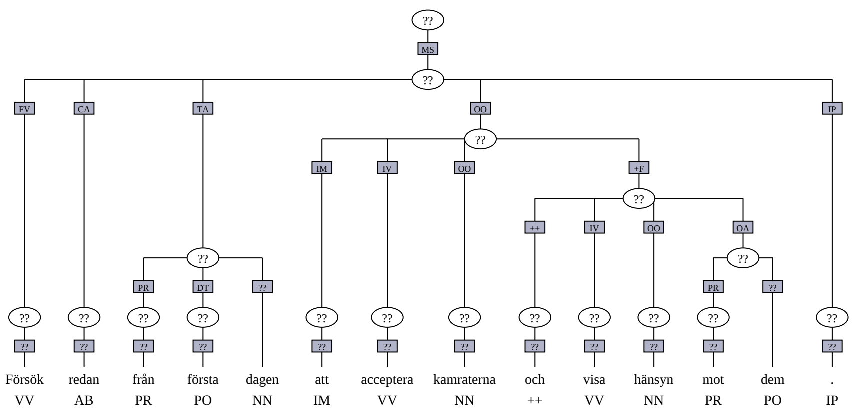

Figure 12: The bare phrase structure tree of the sentence in figure 11.

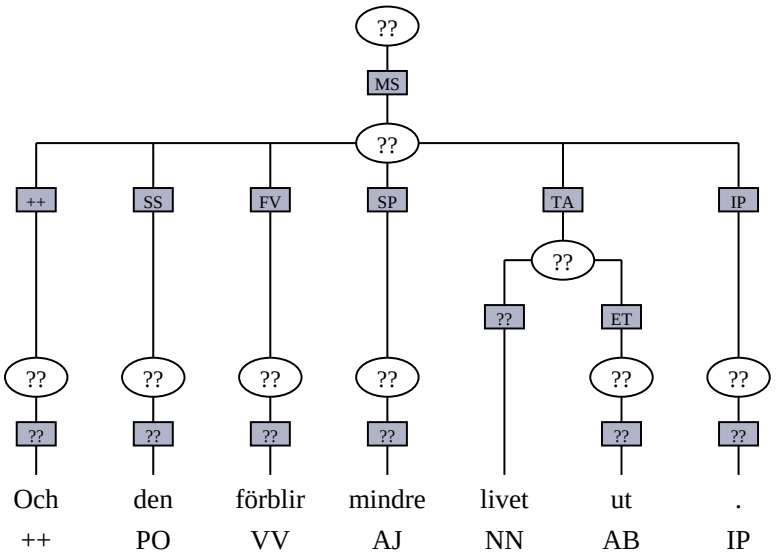

Figure 13: Example of a conjunction in the beginning of a sentence.

phrase identification. See Teleman [14] for a description of the heterogeneous group of pseudo-coordination.

Concerning the second and fourth subgroup, the program makes no difference between them. They are treated in the same way, with a few adjustments needed to take care of some of the peculiarities in the annotation of coordination within noun phrases. Consider the second conjunction in figure 14 at line 7. This is a case of the second subgroup where two *SS* are coordinated. Note first that the second *SS* is not alphabetically displaced. It does not have the grammatical function *ST*. All components that are coordinated in the MAMBA-format keep their grammatical function unchanged. The conjunction at line 16 is also a case of the second subgroup, where two components with *IO* are coordinated.

The conjunction at line 5 is on the other hand an instance of subgroup four. Here two nouns, *tvekan* (*hesitation*) and *dröjsmål* (*delay*) are coordinated, both encoded as the head words in the subject (*SS*), since they are lacking grammatical functions. This example illustrates the fact that the program has to be able to treat words, not having grammatical functions, as special cases when they occur in coordinations. In coordination, the program handles all conjuncts that lack grammatical functions as if they had grammatical functions. Two underscores, i.e. _ _ , are used as the grammatical function for all coordinated words lacking a grammatical function in the original MAMBA-annotation [1]. Since this is not a real grammatical function, all of them are taken care of in step 2. They will either be removed or replaced by the grammatical function *CJ* (see section 3).

[1] The reason why we use _ _ as a label here is because an empty label or two spaces as label is not permitted in the TIGER-format. One of these would otherwise comply better to the fidelity idea.

```
P10207028001    0000                    <<      GM          024
P10207028002    *EN                     EN      SSDT        024
P10207028003    SEKUNDS                 NN    GGSSDU        024
P10207028004    TVEKAN                  VN      SS          024
P10207028005    ELLER                   ++EL    SS++        024
P10207028006    DRÖJSMÅL                NN      SS          024
P10207028007    ELLER                   ++EL    ++          024
P10207028008    ETT                     EN      SSDT        024
P10207028009    TILL                    ABZA    SSATAA      024
P10207028010    SYNES                   ID      SSATAA      024
P10207028011    OSKYLDIGT               AJ      SSAT        024
P10207028012    SLARV                   NN      SS          024
P10207028013    KAN                     QVPS    FV          024
P10207028014    KOSTA                   VVIV    IV          024
P10207028015    DIG                     POPPHHAAIO          024
P10207028016    OCH                     ++OC    ++          024
P10207028017    ÄVEN                    ABOC    IO+A        024
P10207028018    KAMRATERNA              NNDDHH  IO          024
P10207028019    LIVET                   NNDD    OO          024
P10207028020    .                       IP      IP          024
```

Figure 14: "A second of hesitation or delay or seemingly innocent negligence can cause the death of you and also the friends."

For the second and fourth subgroup, when the program has identified two conjuncts (described in the next subsection) and the intermediate conjunction, they need to be given a phrase structure. We have chosen the strategy to group each conjunct in a new phrase, which is not part of the original MAMBA-format. Its edge label is set to the grammatical function of the identified coordinated component, which, for instance, is _ _ for the phrases above both *tvekan* and *dröjsmål*, and *IO* for phrase above *dig* on line 15. In the refining phase, we will move the label(s) of the conjuncts over the *CONJP*-phrase, but for now we stay as true as possible to the original annotation, according to the fidelity idea. The phrase label is set to *INS* and the phrase label of the surrounding phrase of the whole coordination is *CONJP*. The conjuncts themselves are given the same phrase structure as they would have if they had been ordinary phrases identified by the phrase identification module. The bare phrase structure tree of the sentence is shown in figure 15.

This additional insertion of nodes can be regarded as a kind of tree structure deepening not motivated by the annotation. One could argue that by introducing new phrases outside the normal phrase identification module, we violate the fidelity idea. On the other hand, it has become apparent during the work of resolving coordination in the MAMBA-format that the structure is too flat to cope with several different kinds of coordinating constructions. Some tricky examples are presented later on.

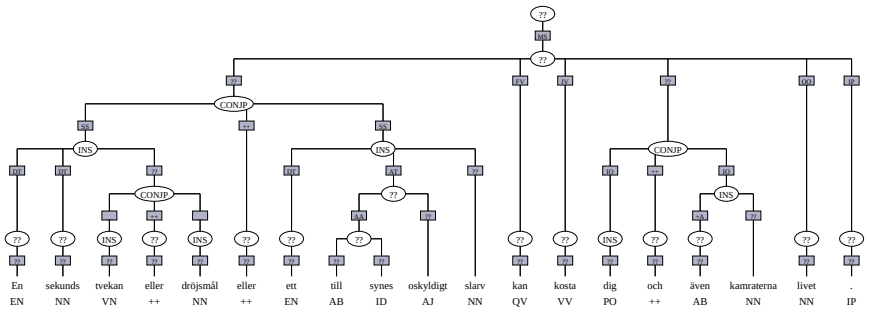

Figure 15: The bare phrase structure tree of the sentence in figure 14.

Finding conjuncts

We briefly mentioned that the conjunctions are identified in a preprocessing step for each identified phrase. In the general case, a phrase can contain more than one conjunction and two conjuncts. Nothing prevents a sentence to have for example three coordinated components with two intermediate conjunctions, which is the case for the conjunctions at line 11 and 13 in figure 10. A phrase with one or more conjunctions can also contain other words that are not part of the coordination. This is the situation for the words *till* and *adressatens* in the same figure. Only the nouns in the phrase with edge label RA are part of the coordination. This forces us to implement the preprocessing in a more complex way than we could have done if all coordinations had been simple and uniform.

Another peculiarity in the MAMBA-annotation is that it tries to make a special kind of semantic distinction in coordination within noun phrases. When the coordinated components do not have at the same referent, an extra layer is added in the annotation, by inserting two spaces. For example, if the nouns in the noun phrase *blue and red cars* (in subject position), refer to cars that are either blue or red, then they do not have the same referents. However, if the noun phrase refers to cars that are both blue and red, they have the same referents. In the former case, the annotation is:

```
blue      SSAT
and       SS++
red       SSAT
cars      SS
```

whereas the analysis in the latter case is:

```
blue      SS  AT
and       SS++
red       SS  AT
cars      SS
```

This semantic distinction might be of interest for those who are interested in a semantically annotated treebank. However, since the MAMBA-format in most other aspects is focused on making a syntactic annotation, the semantic distinction feels a little odd. And since we only have the intention of reconstructing

```
...
P12120043012     PÅ                    PR       AAPR          036
P12120043013     GRUND                 ID       AAPR          036
P12120043014     AV                    ID       AAPR          036
P12120043015     REDAN                 ABZA     AA  ATCA      036
P12120043016     FATTADE               TP     PAAA  AT        036
P12120043017     BESLUT                NN       AA            036
P12120043018     ELLER                 ++EL     AA++          036
P12120043019     FÄRDIGA               AJ       AA  AT        036
P12120043020     PLANER                NN       AA            036
...
```

Figure 16: "... because of already taken decisions or completed plans ..."

the syntactic annotation, we chose to disregard this distinction in step 2. However, in the first step the program follows the fidelity idea and will treat them as the grammatical function having two underscores (__), which is the same as the grammatical function given to words lacking grammatical function, described above. There are also numerous occurrences of inconsistency in the data concerning this semantic distinction. Moreover, the MAMBA-format uses these two spaces for another purpose as well. Look at the adverbial in figure 16. Here, *redan fattade* (*already taken*) is encoded to modify *beslut* (*decisions*), and *färdiga* (*completed*) to modify (*plans*).

The overview of the algorithm for dealing with coordination is as follows. When the conjunctions have been found in the preprocessing step for a phrase, the algorithm loops though all conjunctions. Within the loop it remembers, in principle, which grammatical function is located to the left of the conjunction and then searches backwards as long as the grammatical functions of the words remain the same. This sequence of words is identified as a conjunct. Then it loops forwards, starting to the right of the conjunction, as long as the grammatical functions of the words are the same as the grammatical function of the first word. It then stops and creates a phrase for the identified coordinated component. If it encounters a conjunction on its way to the right in the phrase, which it does at line 13 in figure 10, the described search to the right is repeated and the coordinated structure will therefore comprise three coordinated component and two conjunctions. If it on the other hand encounters a word that is not a conjunction and has a different grammatical function, the outermost loop is restarted if there are more conjunctions in the phrase. This would happen at line 4 in this example:

```
John    SS
and     ++
Mary    SS
saw     FV
Sue     OO
and     ++
Bill    OO
```

A new search to the left is then required from the second conjunction in order

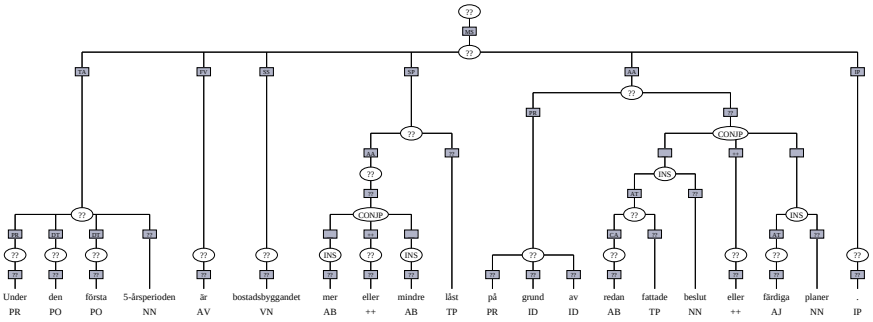

Figure 17: The bare phrase structure tree of the sentence in figure 16.

to identify *Sue* as coordinated component. If there are no more conjunctions, the preprocessing is completed.

Words lacking a grammatical function and words having two spaces as grammatical function are mapped to the same grammatical function, _ _ . A reason for this is to be to handle coordination involving two spaces better. For example, when searching to the left from the conjunction in figure 16, it will identify the three preceding words as one coordinated component. It will also treat the two following words as one conjunct. In constructions like these, it is the correct solution to let words lacking grammatical functions be equal to the words with two spaces as their grammatical function (see figure 17). In other cases it might not be the right strategy, partly because of the inconsistencies in the annotation. Overall, we believe that the number of correct analysis we identify in this way is greater than it would be if we had adopted the strategy to treat them as distinct grammatical functions.

2.6 Refining the bare phrase structure

The output of the conversion so far contains a lot of peculiarity and unnecessary constructions inherited from the MAMBA-format. Therefore, we want to refine the output into a more appropriate treebank format for later transformations. There are some decisions that must be made. For example, should unnecessary non-terminals which do not contain any information be removed and what should we do with the word level dummy conjunctions marked with *++* in the original MAMBA-annotation? As far as possible, the refinement phase should not result in important loss of information.

The resulting bare phrase structure can in a later process be transformed into phrase structure by putting labels on the non-terminals (section 3). It can also be transformed into a dependency based structure by removing the non-terminals. This will be described in more detail in section 4.

Removing dummy conjunctions

As mention in section 2.2, we removed the dummy words representing absent relative pronouns, since they are not a part of the original text and are not needed during subsequent transformation steps. On the other hand, dummy

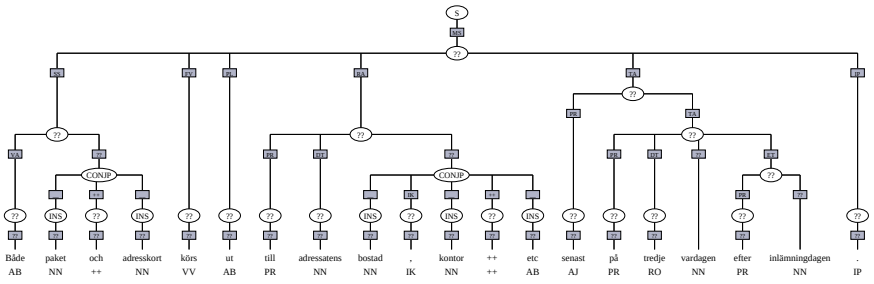

Figure 18: Before removing the dummy conjunction word form *++* between *kontor* and *etc*.

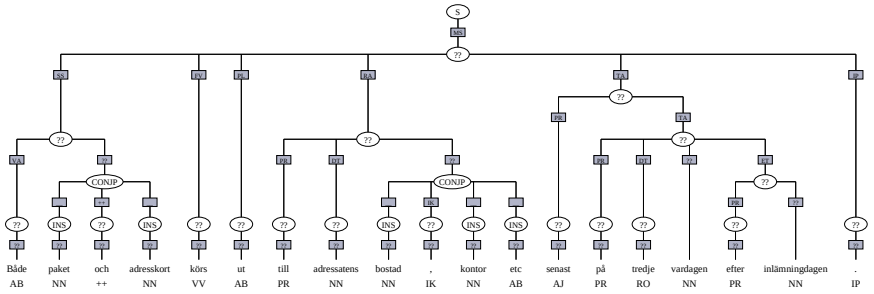

Figure 19: After removing the dummy conjunction word form *++* between *kontor* and *etc*.

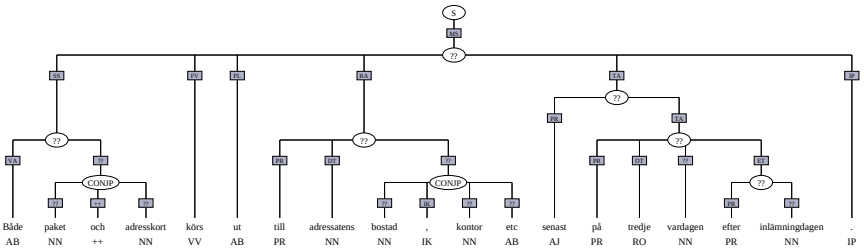

Figure 20: After removing the unary non-terminals.

conjunctions *++* are needed during the identification of coordinations, see section 2.5. They are only annotated in the MAMBA-format to mark that two components are coordinated, but where the conjunction is absent. These dummy conjunctions are now removed with their corresponding non-terminals. In the figures 18 and 19 we can see that the dummy conjunction *++* is removed between the words *kontor* and *etc*. This removal will break the rule of not losing any information, but keeping the word form *++* is not a good decision either because it is not part of the original text.

Removing unary non-terminals

All the non-terminals with one child which does not contain any important information is removed. This decision can be debated, since in some syntactic theories it is argued that these unary non-terminals should be preserved. If someone later on wants these unary non-terminals they can easily be reintroduced. There are four cases to handle:

1. If a unary non-terminal has either the edge label *??* or *_ _* both to its parent and to its child, we can simply remove this non-terminal. A number of examples of this case, before and after removing unary non-terminals can be seen in figure 19 and 20, for instance over the words *paket* and *adresskort*.

2. If a unary non-terminal has one of the edge labels *??* or *_ _* either to its parent or to its child and an informative edge label either to its parent or to its child, we preserve the informative edge label and remove the non-terminal. This case can also be seen before and after deletion in figure 19 and 20, for example the non-terminal located over the word *till* is deleted.

3. If a unary non-terminal has informative edge labels both to its parent and to its child, we do not remove the non-terminal but label the non-terminal with the special category *NAC* (*Not A Constituent*) to preserve the functional information. We can see a case of this rule in figure 21, which preserves the edge labels *AN* and *SP*.

4. If a unary non-terminal has a child which is a coordination and an informative edge label to its parent, we do nothing because we need this unary non-terminal in a later processing phase. The reason for this is described

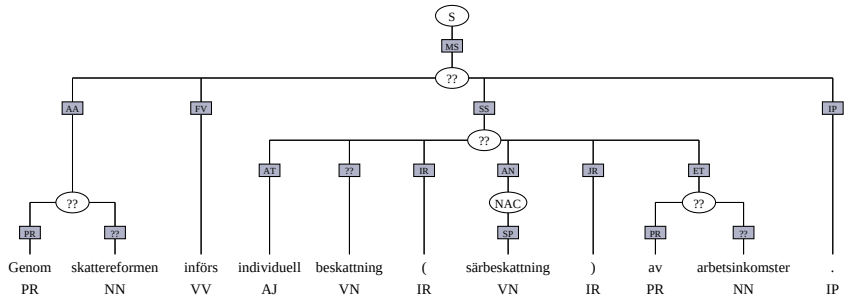

Figure 21: Preserve the unary non-terminal located over the word form *särbeskattning* by inserting a *NAC* node.

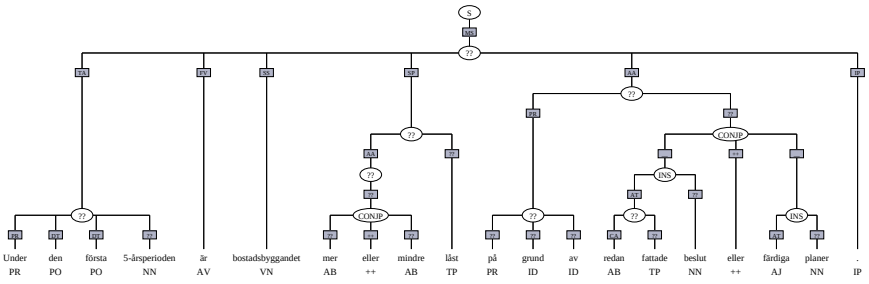

Figure 22: Preserve the unary non-terminal over the conjunction.

later in this section. In figure 22 there is a unary non-terminal over the coordination phrase with the words *mer eller mindre*.

Finding heads and modifiers

By definition in the MAMBA encoding the head word of a phrase does not have any grammatical function. In step one, the conversion uses the dummy edge label *??* to encode the head for the relation between a non-terminal and a terminal. To do this transformation we traverse every terminal and if there is a dummy edge label *??* we replace it with the edge label *HD*.

Also the dummy edge label _ _ can be considered as marking a head. If there exists a head in the subtree for a non-terminal with an edge label _ _ we replace the dummy edge label with the edge label *HD*, if not we replace it with a modifier labeled *MD*. An example of finding heads and modifiers is shown in figure 23, where the phrase *arbetsfördelningen eller hierarkin* is the head of the phrase and the phrase *Den strikta* modifies the head.

Identifying the edge label over coordination phrases

The goal of this transformation is to identify the edge label which will reside over each coordination and replace the edge labels for the conjuncts with a new edge

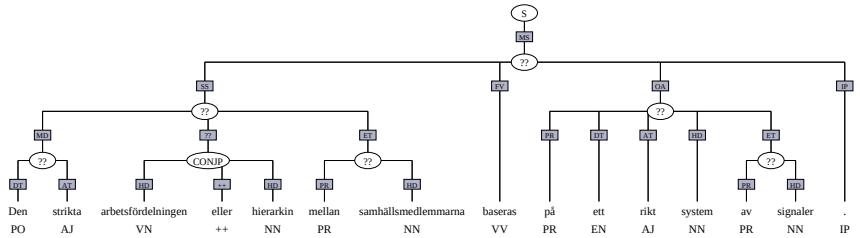

Figure 23: Naming of heads and modifier as the edge labels *HD* and *MD*.

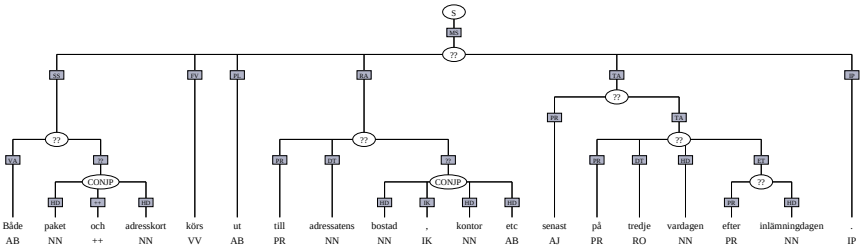

Figure 24: Before identifying the edge label over the non-terminals *CONJP* and renaming the conjuncts with edge label *CJ*.

label *CJ*. This way of annotating the coordinations is inspired by the NEGRA treebank annotation schema [13, 1].

In figure 24 we can see that the edge label over the coordinations is the dummy edge label *??*. If the conjuncts' edge labels are equal, we take this edge label to name the edge label over the coordination and rename the conjuncts with the new edge label *CJ*. After this transformation the previous example is as in figure 25. In some cases the conjuncts' edge labels differ and the strategy is then to first find a head amongst the conjuncts, and then we set the label over the coordination to *HD*. If this cannot be found, we take the leftmost conjunct's edge label to be the edge label over the coordination. We then use a *NAC* non-terminal to preserve the edge labels which are not moved up over the

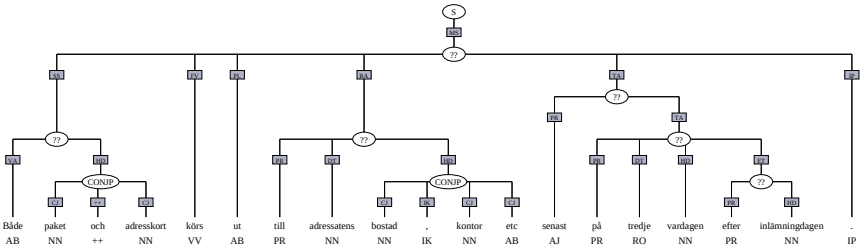

Figure 25: After identifying the edge label over the non-terminals *CONJP* and renaming the conjuncts with edge label *CJ*.

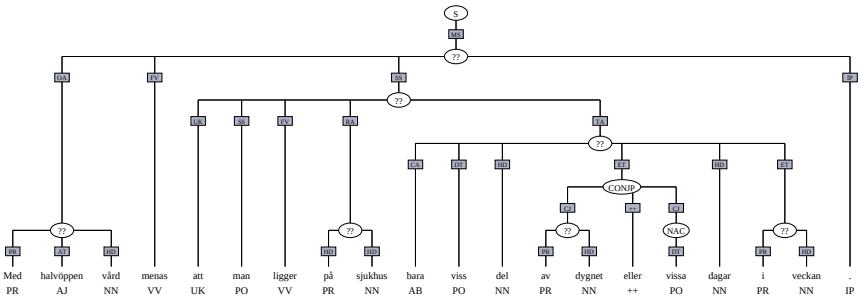

Figure 26: Preserving the information by inserting a NAC node.

coordination phrase. In figure 26 shows this case, where the edge label is taken from the leftmost conjunct and a NAC non-terminal is inserted to preserve the edge label DT.

3 Labeling bare phrase structure trees

The goal of this step is to transform the bare phrase structure trees into phrase structure trees with suitable phrase labels. The previous step can be viewed as a non-trivial conversion and reconstruction from one encoding format (MAMBA) to another format (Tiger-XML). In this step we use the bare phrase structure encoded in Tiger-XML and put labels on the non-terminals.

As a result of the refinement phase in step 1, we get a more suitable bare phrase structure, which is more easy to work with. It is of course possible in the future to do another labeling of non-terminals, for example with different phrase labels or with a deeper structure. Here we have chosen to use a small set of phrase labels. The labeling process is preformed by traversing the trees recursively in a bottom-up fashion. For each non-terminal we collect information about the parts-of speech of the terminals, the edge labels to its children and the edge label to its parent.

A labeling rule is a quadruple (C, P, L, N), where:

1. C and P are lists of grammatical functions,
2. L is a list of lexical categories,
3. N is a non-terminal node label.

A labeling rule (C, P, L, N) assigns the label N to a node n if the following conditions are satisfied:

1. n has a child with a grammatical function $g \in C$, or $C = *$,
2. n has a grammatical function $g \in P$, or $P = *$,
3. n has a child with a lexical category $l \in L$, or $L = *$.

In table 1 we show the set of rules we use to label the phrases, ordered by decreasing priority. After looking up the appropriate phrase label, the non-terminal is labeled with this category except when the non-terminal already

#	C	P	L	N
1	MS	*	*	ROOT
2	PR	*	*	PP
3	SS, FV	*	*	S
4	*	+F	*	S
5	IV, IM	*	*	VP
6	*	VS, VO	*	VP
7	DT, AT, ET	*	*	NP
8	HD	DT	PN, MN, AN, VN, NN, PO, RO	NP
9	HD	DT, PR	*	XP
10	HD	*	PN, MN, AN, VN, NN, PO, RO	NP
11	HD	SS, OO	*	NP
12	HD	*	AJ, TP, SP	AP
13	HD	*	AB	AVP
14	*	SS, OO	*	NP
15	*	*	*	XP

Table 1: A list of labeling rules. The C-column (Child) enumerates the edge labels from the non-terminal to its children and the P-column (Parent) enumerates the edge labels from the non-terminal to its parent. The L-column (Lexical categories) contains the parts-of-speech for the children to the non-terminal, and the N-column (Non-terminal) contains the resulting phrase label.

has the phrase label *NAC* or the non-terminal is a coordination node, which is handled as a special case described later on.

In figure 27, the phrase node *S* gets its label because there is a finite verb labeled *FV* amongst the children, as well as a subject *SS* (rule 3). Moreover, the modifier *AT*, *individuell*, modifies the head word *beskattning* and this is only found in a noun phrase *NP* (rule 7). The edge label *PR* as a child in a phrase is a strong indicator that it is a prepositional phrase *PP*, and can be found in two places in the example: *Genom skattereformen* and *av arbetsinkomster* (rule 2). We can also see that the non-terminal *NAC* is left untouched.

The infinitive marker *att* labeled *IM* and the non-finite *acceptera* labeled *IV* are evidences of a verb phrase *VP* (rule 5). Such an example is shown in figure 28. In the same example, we can also see a clause fragment with the parent edge label *+F* which indicates that some component is missing, but it is nevertheless labeled *S* according to the present set of labels (rule 4).

In figure 29, there is an example of a multiword unit *i samband med*. In some annotation schemes such cases are given their own phrase label, but we have chosen not to do so. Instead we introduce a phrase labels *XP* (rule 9), which can be viewed as a phrase label containing all phrases we do not want to discern. Another example where we used the dummy phrase *XP* is what one can call determiner phrases.

Special case for coordinations

Finally, there is a special case for labeling the coordination phrases. A special list of labeling rules is used to determine the phrase label for coordination. A

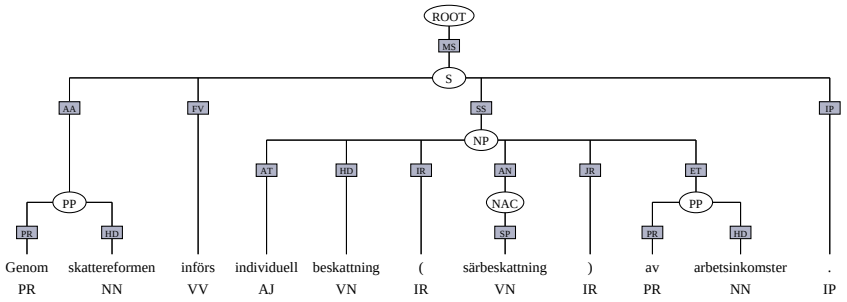

Figure 27: After the labeling of the phrases.

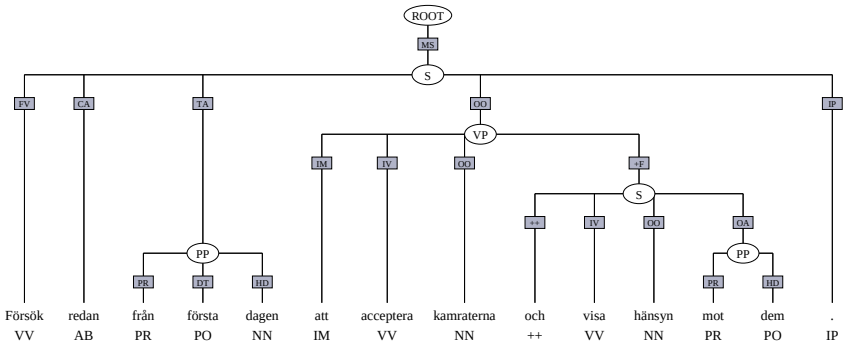

Figure 28: After the labeling of the phrases.

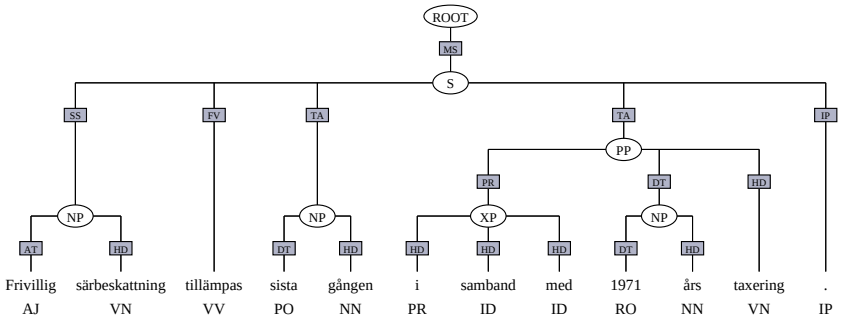

Figure 29: After the labeling of the phrases.

L ∪ P	N
PN, MN, AN, VN, NN, PO, NP, CNP	CNP
XP, PR, TP, SP, RO, CXP	CXP
AJ, AP, CAP	CAP
PP, CPP	CPP
AB, AVP, CAVP	CAVP
S, CS	CS
AV, BV, HV, WV, QV, MV, KV, SV, GV, FV, VV, VP, CVP	CVP

Table 2: A list of labeling rules to label the coordination phrases. The column L ∪ P enumerates the parts-of-speech and phrase labels of the conjuncts, and the N-column (Non-terminal) contains the resulting phrase label.

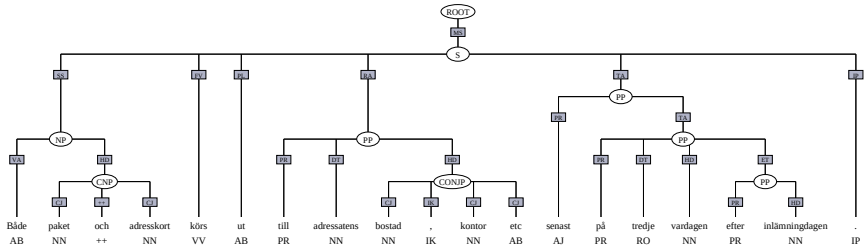

Figure 30: After the labeling of the phrases.

labeling rule for coordination is a triple (L, P, N), where:

1. L is a list of lexical categories,
2. P is a list of non-terminal labels,
3. N is a non-terminal node label.

A labeling rule (L, P, N) assigns the label N to a node n if the conjuncts have the lexical category $l \in L$ or the non-terminal label $p \in P$. In table 2 we show the set of rules we use to label the coordination phrases. If there exist different conjuncts which satisfy two or more labeling rules the coordination phrase will be labeled with the default coordination label *CONJP*.

In the example shown in figure 30 the phrase label *CNP* is used because the conjuncts' lexical category is *NN*, and the phrase label *CONJP* is used to indicate that there are two suitable phrase labels, *CNP* and *CAVP*. In this particular example, the latter phrase should perhaps be labeled *CNP* as well, but the algorithm assigns the label *CONJP*.

4 Extracting dependency trees

In the transformation from bare phrase structure to phrase structure, the reconstruction program was forced to augment nodes with node labels. The transformation to dependency grammar is on the other hand not a matter of adding

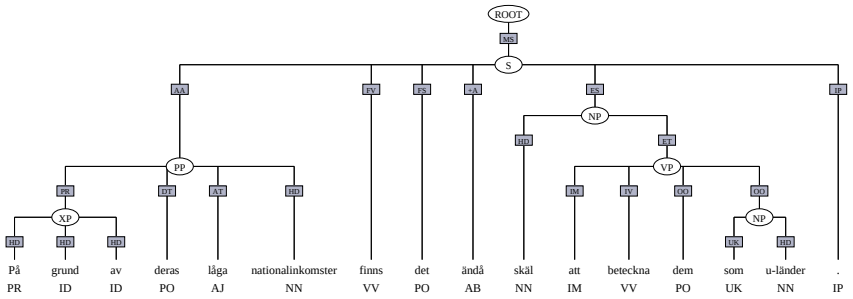

Figure 31: Multiple heads: "Because of their low national income, there are still reasons to regard them as development countries"

information. Instead, it performs a selection among the available and perhaps ambiguous information.

As input to this step, we use the output from the first step described in the section 2, i.e. bare phrase structure trees. One might consider using the labeled phrase structure as input to this step, since the non-terminal labels, such as *NP* and *VP*, can be useful in the construction of the dependency structure. We have decided to use only the original information instead, since the non-terminal labels are extracted from the same information source.

The transformation going from the source into dependency annotation relies on a number of premises, and much of the ideas are inspired by the technique of head-finding rules (Collins [2], Magerman [7]). The idea is that each phrase contains one head word, and consequently, all other words in that phrase will be non-head words. Collins' goal was also to extract dependency trees from the Penn Treebank by using head-finding rules, and the problem with the Penn Treebank (and treebanks based on constituency in general) is that it does not specify which word in each phrase's subtree is the head. Collins uses a list of priority rules by searching among the children of each phrase to find an appropriate head. The selected head depends on the phrase label of the parent and the phrase label and parts-of-speech of the children.

We apply an algorithm that resembles the one described above in the sense that we are trying to find one head for each phrase. However, the information in the output step 2 is different from the information in Penn, since the information source in this step is the edge labels. In the original MAMBA-annotation, phrase heads were identified by leaving the grammatical function unspecified. These correspond to the words in the phrase structure having the edge label *HD*. In the case where a phrase contains exactly one such head word, the task of choosing the head of the phrase is trivial. Unfortunately, not all phrases have this nice property. It can be the case that a phrase lacks such a word, which is most notable at the main clause level. For example, this is the case for all trees in section 3 (no edge label is marked *HD* at the highest level). Moreover, a phrase can also have more than one head, which is the case for the phrase *XP*, containing the multi-word preposition *på grund av* (*because of*) in figure 31. The problem here is to choose a tenable solution for the two non-trivial cases.

First, the algorithm traverses each tree bottom-up and with the current

Edge label
Head (HD)
Finite verb (FV)
Non-finite verb (IV)
Predicative complement (SP)
Subjects and objects (SS, ES, FS, VS, OO, EO, FO, IO, VO)
Clause adverbials (AA, KA, RA, OA, TA)
Phrase adverbials (+A, CA, MA, NA, VA, XA)
Nominal pre-modifier (AT)
Nominal post-modifier (ET)
Other noun dependents (DT, XT)
Unclassifiable dependent (XX)
Other functions not involved in coordination (e.g. +F, EF, XF, IM, AN)
Conjunct (CJ)
Coordinator, conjunction (++)
Punctuation (I?, IC, IG, IK, IP, IQ, IR, IS, IT, IU, JC, JG, JR, ST)

Table 3: The priority list of head-finding rules

design only takes the children's edge labels for each phrase into consideration. Compared to Collins, the transformation makes use of head-finding rules, but it uses only the edge label of the children and disregards the information given by the parent node, such as phrase label and edge label. Table 3 lists the head-finding rules. As mentioned above, the reason why we have decided not to use the phrase labels is that these are inferred by the other information.

The list of head-finding rules is also a priority list, where the first rule of the list has the highest priority and the last rule the lowest. In principle, for each phrase, the algorithm looks at the children's edge labels. Then one of two things happens:

- If it finds exactly one edge label with higher priority than all the other, it makes the child with that edge label the head of the phrase.
- If it finds two or more edge labels with the same priority and with higher priority than all the other, then it lets the left-most of these children be the head of the phrase.

When a head has been identified, the second task for the algorithm is to make all other words dependents to the head. In the simple case, when the head is an individual word (terminal node), that word will attach to all its children. In the other case, when the head in a phrase (non-terminal node), the word that was identified as the head of that phrase will attach to all the children.

The corresponding procedure holds for the children too. If a child is an individual word, then that word is the dependent of the head. For example, this is the case for the dependent *Genom* and the head *skattereformen* in figure 27. Also, the dependency relation between the head word and the dependent is set to the edge label of the child, which here will be *PR*. If the child on the other hand is a phrase, the head word of that phrase is the dependent to the head. This is the kind of relation between the words *skattereformen* and *införs* in the same figure. The dependency relation between them is *AA*, since that is the edge label

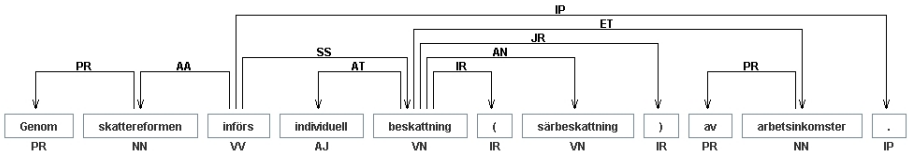

Figure 32: The dependency tree of the sentence in figure 27

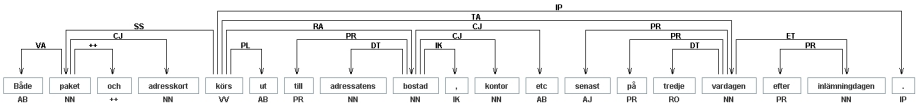

Figure 33: The bare phrase structure tree of the sentence in figure 30

of the phrase that is dependent on *införs*. Figure 32 depicts the corresponding dependency trees. Note also that it is always the highest positioned edge label that is used to label the dependency relation, which becomes apparent in the relation between *beskattning* and the apposition *särbeskattning*. Here the edge label AN is picked, not the lower positioned SP.

The first edge label in the list of head-finding is the edge label HD, which for instance marks the head noun in noun and prepositional phrases. The following six lines contain edge labels most often found in clauses and are places where there should not be any HD-labels present. First of all, if there exists a finite verb FV in a phrase, then the rules prefer this one as the head. If no finite verb is found, we let the non-finite verb IV be the head. The following three rules are applied rarely, but are necessary to cope with e.g. elliptic constructions in clauses. If no verbs are present, we let the predicative complement have precedence over the others. Then we make the left-most occurrence of subject, or object, the head, dominating all kinds of adverbials, which are also divided into two groups. Adverbials, such as time adverbials TA, have precedence over "smaller" adverbials like negations NA.

After the adverbials come the edge labels most often occurring in noun phrases, such as determiner DT, and other attributes placed before (AT) and after (ET) the head noun. If the head noun in a noun phrase is missing (due to e.g. ellipsis), ET will be a dependent of AT, and ET also has precedence over all other attributes, i.e. DT and XT.

Among the following list of edge labels, it is important to note that CJ has precedence over $++$, and a consequence of the algorithm described above is that the left-most child with the edge label CJ will be the head of all other children in all conjunction phrases, similar to Mel'cuk [8]. Since conjunctions in dependency grammar are cumbersome (as well as for other theories), this solution can be regarded as good as any other solution. We will therefore stick to this for now. And an example of what coordinated constructions look like in a dependency tree given this solution is shown in figure 33, having figure 30 as input.

All other priority rules should hardly be used at all, since these edge labels are used for terminals and non-terminals that usually cannot be regarded as heads. These edge labels are among other things used for punctuation, apposi-

tions and particles. Furthermore, at the top level of the phrase structure, the edge label *MS* resides. In the case where there is more than one macro syntagm in a sentence, the algorithm chooses not to attach them to each other using a dependency relation. We motivate this by the fact that the relation between two macro syntagms is not really a dependency relation. These sentences will therefore be unconnected, i.e. they will have more than one root.

5 Final remarks

By and large, we have managed to reconstruct the original syntactically annotated corpus Talbanken, encoded in the MAMBA-protocol. At a reasonable cost, we have made an old linguistic resource more accessible by transforming Talbanken into a modern representation. During the three steps of the project, we have kept two ideas in mind, the fidelity idea and the idea of a theory-supporting treebank.

In the first step, we transformed the original format into a phrase structure representation leaving the phrase labels unspecified. Here we tried to comply with the fidelity idea without going to extremes. For example, we have been forced to introduce additional phrases to cope with coordination, and disregard the small fraction of semantics encoded in the MAMBA-format. Coordination is a problem in step 1, since the MAMBA-annotation sometimes is ambiguous and difficult to interpret. But mostly, we believe that the identification of phrases is performed successfully, which includes the handling of discontinuity and the alphabetical displacement. The refining part of this step is in a way a violation of the fidelity idea. However, all deletion of non-terminals and relocation of information has been performed without loosing too much information. On the other hand, this refinement is probably a better candidate as a theory-supporting treebank source.

Step two can be regarded as the transformation from a source treebank to a target treebank, i.e. a phrase structure treebank. This transformation is for our source only a matter of labeling the phrases. Obviously, there is a lot of freedom here. We have been influenced by the annotation in the Penn Treebank and the NEGRA Treebank. Moreover, coordination is tricky in this step as well, and requires a special treatment outside the ordinary labeling of phrases. It is only the phrase structure based target that directly adopts the phrases from the source. It is hard to imagine a source where phrases and their boundaries are left out, moving the responsibility to create phrases to transformation rules. The actual labels of the phrases are on the other hand inferred more easily.

We use the output of step 1 as input to step 3. Here we abandon the notion of phrases. The edge labels are extracted from the words and phrases to name the dependency labels of the relation in the dependency trees. The transformation from the source to a dependency representation is a set of head-finding rules, where the edge label *HD* has been the main head-finding clue. In this step, we have chosen to discard some of the edge labels when attaching a head to the head of a phrase. A possible source of errors can be that some phrases in the source treebank contain more than one *HD*-edge label. But it is possible to create another set of transformation rules, preserving all such edge labels in one dependency label. The latter set of transformation rules would better comply with the fidelity idea, whereas our solution better fits our needs. In this step,

coordination needs a special treatment. Their annotation is only a consequence of the encoding of the previous step and the design of the head-finding rules. The strategy to let the left-most one be the head might not always be the correct decision. Hence, when creating a theory-supporting treebank from scratch, a suggestion is to avoid having more than one head word per phrase.

Moreover, the source treebank has currently a rather flat annotation, which does not favour any specific theory with a deeper nesting. A deeper nesting would on the other hand probably make the transformation into a target treebank, especially dependency based ones, more accurate due to the fact that there are fewer terminals and non-terminals within each phrase to search among in order to find the head. Our target treebanks, the phrase structure treebank and the dependency treebank, thus do not have to be regarded as the final result. A number of post-processing steps on the source treebank after step 1 would probably improve the quality of the targets. For example, Samuelsson and Volk [12] suggest possible deepenings. Furthermore, depending on the settings for the treebank deepening of the source, while using the same head-finding rules, it is possible to conform to different dependency treebank theories. This may lead to interesting comparisons, e.g. finding the deepening setting that maximizes parsing accuracy.

To conclude, we believe that we successfully made a linguistic resource available. One of the problems in Talbanken is coordination, and one way to deal with this is probably to revise the original data. Another solution could be to modify the output of either step 1 or 2, since there exist annotation tools for the phrase structure representation we use. The converted versions of Talbanken can be found at *http://w3.msi.vxu.se/users/nivre/research/talbanken.html*.

References

[1] Sabine Brants, Stefanie Dipper, Silvia Hansen, Wolfgang Lezius, and George Smith. The TIGER treebank. In *Proceedings of the Workshop on Treebanks and Linguistic Theories*, pages 24–41, 2002.

[2] M. Collins. A new statistical parser based on bigram lexical dependencies. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 184–191, 1996.

[3] J. Einarsson. *Talbankens skriftspråkskonkordans*. Lund University, Department of Scandinavian Languages, 1976.

[4] J. Einarsson. *Talbankens talspråkskonkordans*. Lund University, Department of Scandinavian Languages, 1976.

[5] J. Järborg. Manual för syntaggning. Technical report, Göteborgs universitet: Institutionen för svenska språket (Språkdata), 1986.

[6] Matthias T. Kromann. Proposals for extensions and conventions in Tiger-XML within the nordic treebank network, March 2005. http://www.id.cbs.dk/~mtk/ntn/tiger-xml.html.

[7] David M. Magerman. Statistical decision-tree models for parsing. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pages 276–283, 1995.

[8] Igor Mel'cuk. *Dependency Syntax: Theory and Practice*. State University of New York Press, 1988.

[9] Andreas Mengel and Wolfgang Lezius. An XML-based representation format for syntactically annotated corpora. In *Proceedings of the 2nd International Conference on Language Resources and Evaluation (LREC2000)*, volume 1, pages 121–126, Athens, Greece, May 2000. ELRA.

[10] J. Nivre. Theory-supporting treebanks. In *Nivre, J. and Hinrichs, E. (eds.) Proceedings of the Second Workshop on Treebanks and Linguistic Theories*. Växjö University Press, 2003.

[11] J. Nivre, K. de Smedt, and M. Volk. *Treebanking in Northern Europe: A White Paper*, pages 97–112. Copenhagen: Museum Tusculanums Forlag, 2005.

[12] Yvonne Samuelsson and Martin Volk. Automatic node insertion for treebank deepening. In *Proc. of the Third Workshop on Treebanks and Linguistic Theories (TLT)*, pages 127–136, Tübingen, Germany, 2004.

[13] W. Skut, B. Krenn, T. Brants, and H. Uszkoreit. An annotation scheme for free word order language. In *Proceedings of the 5th Conference on Applied Natural Language Processing*, pages 88–95, Washington, DC, 1997.

[14] Ulf Teleman. *Manual för grammatisk beskrivning av talad och skriven svenska*. Lund: Studentlitteratur, 1974.

A The coarse lexical categories

Description	Lexical Category
Coordinating conj.	++
Adverb	AB
Adjective	AJ
Noun	PN, MN, AN
	VN, NN
Verb	AV, BV, BV
	WV, QV, MV
	KV, SV, GV
	FV, VV
Punctuation	I?, IC, IG
	IK, IP, IQ
	IR, IS, IT
	IU, PU
The identity tag	ID
Infinitive marker	IM
Pronoun	PO
Preposition	PR
Number	RO, EN
Participle	SP
	TP
Subordinate conj.	UK
Unknown tag	XX

Table 4: The lexical categories in the MAMBA-annotation format

Description	Dummy Category
Clauses	AC, CC, FC
	JC, KC, RC
	VC, ZC
Cond. clauses	FK, BK, OK
Phrases	AF, +F, KF
	IF, JF, PF
	MF, XF

Table 5: The hierarchic dummy categories in the MAMBA-annotation format

B The mapping to grammatical functions

The table below contains grammatical functions in MAMBA. The grammatical functions in italic in the *AD*-columns are not part of the original set of categories in MAMBA. They are instead inserted and used by the program internally in step 1. Also, in the other two steps, only the grammatical functions in the *GF*-columns are still seen in the phrase structure trees.

GF	AD	GF	AD	GF	AD
++	++	FO	FO		*MS5*
+A	+A		FP		*MS6*
	+B	FS	FS	NA	NA
+F	+F	FV	FV	OA	OA
	+F1	GM	GM		OB
	+F2	GX	GX	OO	OO
	+F3	I?	I?		OP
	+F4	IC	IC		OQ
	+F5	IG	IG	PL	PL
	+F6	IK	IK	PR	PR
	+F7	IM	IM	PT	PT
	+F8	IO	IO	RA	RA
	+F9	IP	IP		RB
AA	AA	IQ	IQ		RC
	AB	IR	IR	SP	SP
	AC	IS	IS		SQ
AG	AG	IT	IT	SS	SS
AN	AN	IU	IU	ST	ST
	AO	IV	IV	TA	TA
AT	AT		IX		TB
	AU	JC	JC		TC
	AV	JG	JG	TT	TT
CA	CA	JR	JR	TX	TX
	CB	JT	JT	UK	UK
DB	DB	KA	KA	VA	VA
DT	DT		KB	VO	VO
	DU	MA	MA	VS	VS
	DV		MB	XA	XA
EF	EF	MS	MS	XF	XF
EO	EO		*MS1*	XT	XT
ES	ES		*MS2*	XX	XX
ET	ET		*MS3*		XY
	EU		*MS4*	YY	YY
	EV				

Table 6: The grammatical functions in MAMBA (GF = grammatical function, AD = alphabetical displacement)

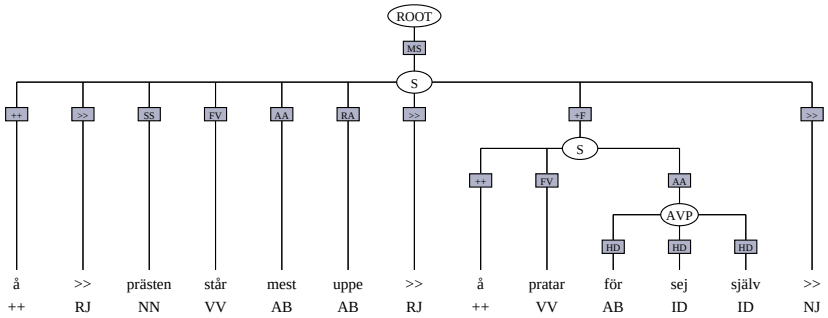

Figure 34: An example of the phrase structure tree of a spoken utterance: ”and >> the priest is mostly standing up there >> and talks to himself >>

C Reconstructing G, SD and IB

Only a few adjustments of the original reconstruction program were needed to make it possible to get a quite nice looking result for the parts G, SD and IB of Talbanken. However, depending on the needs, the output might require follow-up.

The part G, which contains unchanged text written by students, required in principle no adjustments in order to run through all steps of the reconstruction program. It contained a number of annotation errors, making changes in the original files unavoidable. Annotation errors exist in the other three parts as well. We have therefore documented the changes we made in the original files whenever we have found apparent errors and inconsistencies. The documents can be found at *http://w3.msi.vxu.se/users/nivre/research/talbanken.html*.

The adjustments for SD and IB, containing spoken language, implied some changes to the reconstruction program. Most notably, the corpora contain no graphical sentences and macro syntagms, which occur only in written language. Since these markers are important for the split of sentences, the corpora are instead divided into utterances. These are marked with dummy words having the grammatical function <<, and are not present in P and G. To keep the division simple, we chose to treat these dummy words in much the same way as *GM*. Thus, SD and IB are by and large divided into utterances according to the original MAMBA-annotation.

SD and IB also treat pauses, hesitations and false starts, etc. as parts of the syntactic structure. Pauses and hesitations are annotated like words and have their own parts-of-speech. There are actually three kinds of pauses, and these things imply changes to the tagset. Here we have kept the transformation simple and kept pauses and hesitation. Furthermore, when the words were transcribed, the annotators tried to capture simplifications that speakers make, not write the word as it usually is written. For example, the word *jag* is often pronounced *ja*. A sentence picked from SD is shown in figure 34.

Växjö
universitet

Matematiska och systemtekniska institutionen
SE-351 95 Växjö

tel 0470-70 80 00, fax 0470-840 04
www.msi.vxu.se