The Linguist's Guide to PLAIN – Part 2

Drawing up a Syntax Component

Peter Hellwig, University of Heidelberg

(Version July 2015)

Note: A former version of this paper had the title
"Testing the PLAIN IDE with German examples. How to program a parser in DRL"

Abstract

PLAIN (Programs for Language Analysis and Inference) is an integrated development environment (IDE) which provides comprehensive facilities to (computational) linguists for creating and processing lingware. PLAIN adheres to Dependency Unification Grammar (DUG), a particular linguistic approach to natural languages. DUG aims at a simple and, at the same time, broad coverage of linguistic phenomena.

Dependency Representation Language (DRL) is the formalism of DUG. DRL is so to say a programming language - to be used by a linguist in order to have the computer analyze natural language. In this paper we discuss the DUG approach to syntax and the resources that have to be drawn up for a parser. A linguistic guide to PLAIN with English examples is still in progress. For the time being, the article describes the data that is used in the *german_demo* project (to be downloaded from plain-nlp.de) We apologize for the lower benefit a reader might have who does not understand German.

Contents

Introduction

The PLAIN IDE provides an interpreter of resources written in Dependency Representation Language (DRL) for various purposes. The main functions are the Scanner, the Parser, the Transducer, and the Generator. The Scanner reads natural language input, consults the morphological resources and outputs a morpho-syntactic categorization of input segments. The Parser takes over the morpho-syntactic analysis, consults the grammatical resources ("synframes" and "templates") written in DRL and yields a syntactic description in DRL format. The Transducer reads a DRL construct, consults rules written in DRL and outputs new DRL constructs. The generator turns DRL constructs into natural language surface. It consults the morphological as well as grammatical resources.

One may call on the following publication for the theoretical background of PLAIN:

Peter Hellwig: *Dependency Unification Grammar.* In: V. Agel, L.M. Eichinger, H.-W. Eroms, P. Hellwig, H.-J. Heringer, H. Lobin: Dependency and Valency. An International Handbook of Contemporary Research. Mouton 2003.

This paper is about the following resource files of the *german_demo* project. These files are included in the package if PLAIN is downloaded from www.plain-nlp.de.

Definition of categories:
german_demo/load/de_catf.xml
Morpho-syntactic lexicon - words of closed classes, endings:
german_demo/load/de_lexbase.xml
Morpho-syntactic lexicon - words of open classes:
german_demo/load/de_demo_adjektive.xml
german_demo/load/de_demo_substantive.xml
german_demo/load/de_demo_verben.xml
Syntagmatic lexicon (complements and adjuncts):
german_demo/load/de_synframes.xml
Syntagmatic templates:
german_demo/load/de_templates.xml
Input file with test examples:
german_demo/test/de_parser.txt
Output with brief results:
german_demo/test_results/parser_output_short.txt
Output with complete results:
german_demo/test_results/parser_output_long.txt
Output for further processing:
german_demo/test_results/parser_output_drl.txt

Data in this paper is distinguished by color:

- black - comments
- green - testing examples (see file *de_parser.txt*)
- brown – morpho-syntactic lexicon
- red - synframes
- blue - templates

There is a special system to handle examples and organize tests. The grammar writer can add test material to each syntactic template he draws up. These examples are included between xml tags <test></test>). The Converter *Test Examples to Parser Input* creates a test bed from this material. The examples between tags are formated as follows:

- in brackets = this segment corresponds with the filler of the template,
- between slashes = this segment corresponds with the head of the template,
- anything else = context not covered by the template (described somewhere else).

Input marked with '*' is not grammatical; parsing such examples should fail. (Of course the asterisk is not included in the test file.) Input marked with '+' will yield a correct analysis only if the option "first match" is turned off.

The following should be read in sync with the test file *german_demo/test/de_parser.txt* and the output files *german_demo/test_results/de_parser_output...txt.* The aim of these examples is testing the functionality of the program rather than a broad coverage of the German grammar!

Theoretical premises

The unique selling point of Dependency Unification Grammar is its lexicalistic and functional approach. The basis for this peculiarity are the following hypotheses:

Syntax and lexical semantics are not independent.
Each word has a different meaning and hence behaves differently in compositions.
Lexical semantics is the source of syntax.

Many goals of natural language processing require syntactic analysis.
But in most of the cases the analysis must be functional, not just compositional.

An example may illustrate the connection of lexical meaning and syntagmatic function:

Der Finanzminister überredete den Präsidenten der Zentralbank dazu, die Zinsen zu erhöhen.

The minister of finance persuaded the president of the reserve bank to increase the interest rates.

überredete

persuaded

jemand	*jemanden*	*zu etwas*
someone	somebody	to do something
Lisa *Wer zu feige ist* *Der Finanzminister*	*ihren Freund* *jemand anderen* *den Präsidenten der Zentralbank*	*zu einer Aktion* *(dazu) abzuhauen* *(dazu) die Zinsen zu erhöhen*

Lexical roles (Commutation test)

"der Überreder“	"der Überredete“	"der Überredungsinhalt"
the persuader	the one persuaded	the content of persuasion

Abstract roles (Coordination test)

SUBJECT	OBJECT	PREPOBJECT

Figure 1: Syntagmatic roles – lexical and abstract ones

Syntagmatic function is concrete and transparent in the lexical context. The meaning of *persuade* requires mentioning a *persuader, someone persuaded,* and *an action to be performed by the persuaded.* Countless phrases can substituted in each role. The lexical roles (*persuader, persuaded* etc.) stay the same.

Lexical roles are crucial for understanding a sentence. However, abstract roles can be established as well. They are revealed by the coordination test. Lexical roles that can be coordinated must be in some sense similar. We assign the same syntagmatic role tags to them. Compare the following coordination tests:

Er rief ihn an und überredete ihn	Anrufer = Überreder = SUBJECT
Er hat den Präsidenten angerufen und überredet ...	Angerufener=Überredeter=OBJECT
Er hat ihn nicht nur dazu überredet sondern sogar davon überzeugt, die Zinsen zu erhöhen.	Inhalt des Überredens=Inhalt des Überzeugens=PREPOBJECT.

Assuming that dependency is a relationship between single words is a misconception. Obviously complements (e.g. subject, object and prepobject of *persuade*) consist of an arbitrary number of words. While the heads are terminals (e.g. *persuaded*), complements are non-terminal constituents! Each complement as a whole is characterized by a grammatical function; most morpho-syntactic features (case, agreement, word order) apply to the whole complement and not just to the heading word in it. In reality, it is not the edge between two nodes that represents the dependency relation but an edge between a node and a complete tree depending from that node. In the following figure the dependent trees are included in boxes in order to elucidate this fact. Each box embraces a complement. The dependency relation exists between words and such implicit boxes. As opposed to Lexical Functional Grammar (LFG), with its c-structure and f-structure, the representation is mono-stratal.

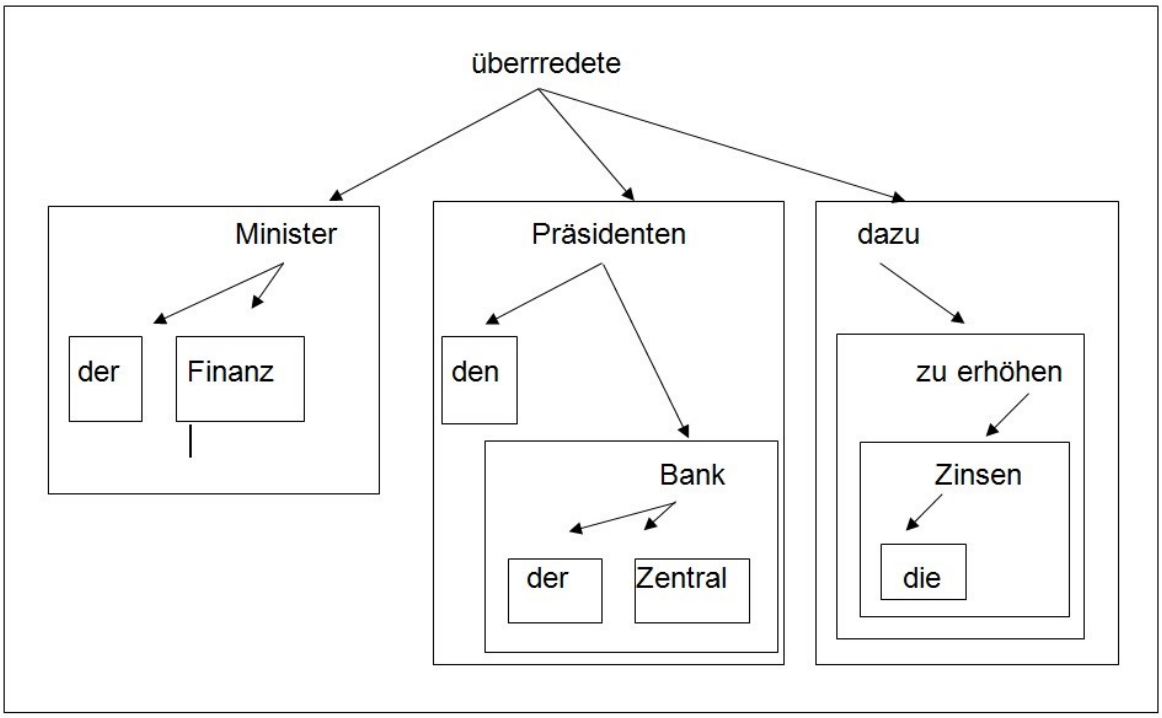


Figure 2: Dependency relation between words and complements

Role labels are the appropriate means to identify the constituents (the "boxes"). Let us add them!

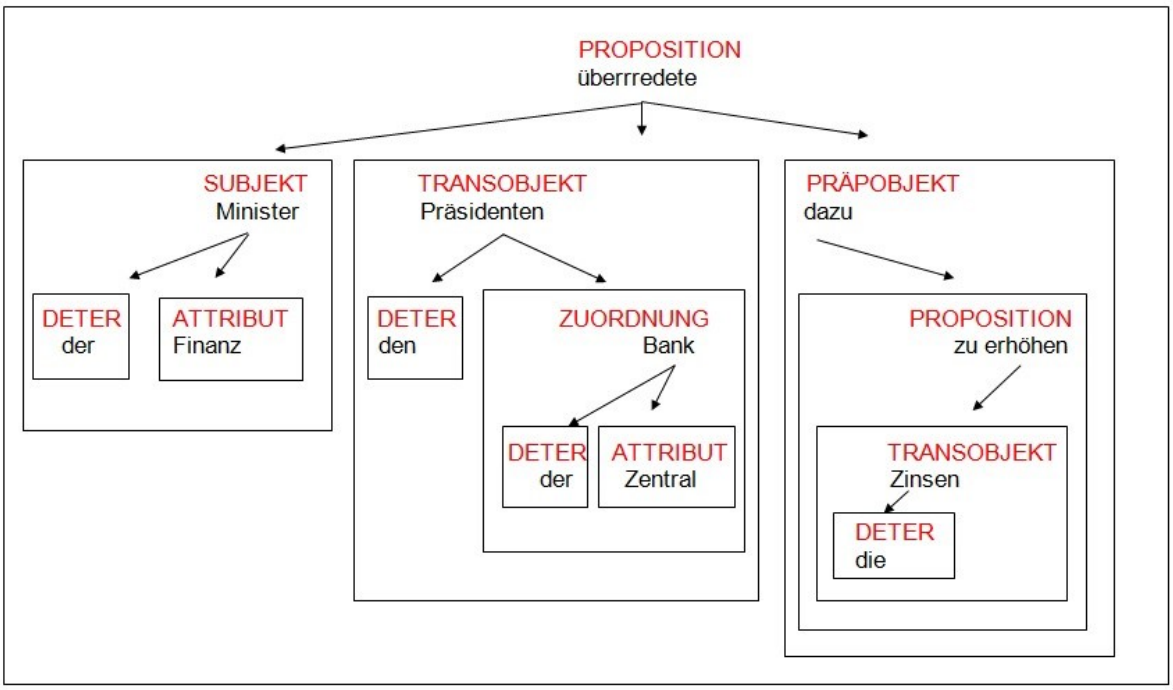


Figure 3: Dependency relation with syntagmatic role labels

Words which denote relation are unsaturated. They require certain complements to make sense. However, many add-ons may introduce other factors into the scene. These add-ons are called "adjuncts".

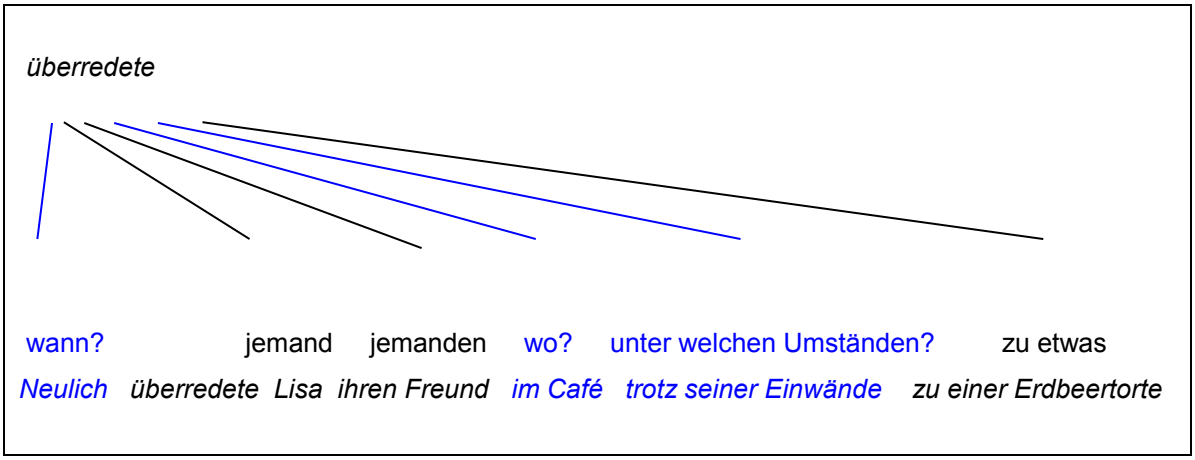


Figure 4: Adjuncts enrich the scene

Complements and adjuncts are both dependents, i.e. they are subordinated to a "head". The dependency relation is represented by edges in a directed graph (or "tree") , where each complement or adjunct node is subordinated to a head. A tree can also be represented by a bracketed expression which is the common DRL representation. On the surface of language, the strings representing complements or adjuncts are located to the left or to the right of the head's string. In the DRL format we use explicit attributes ('R' or 'L' for right or left) to represent this fact.

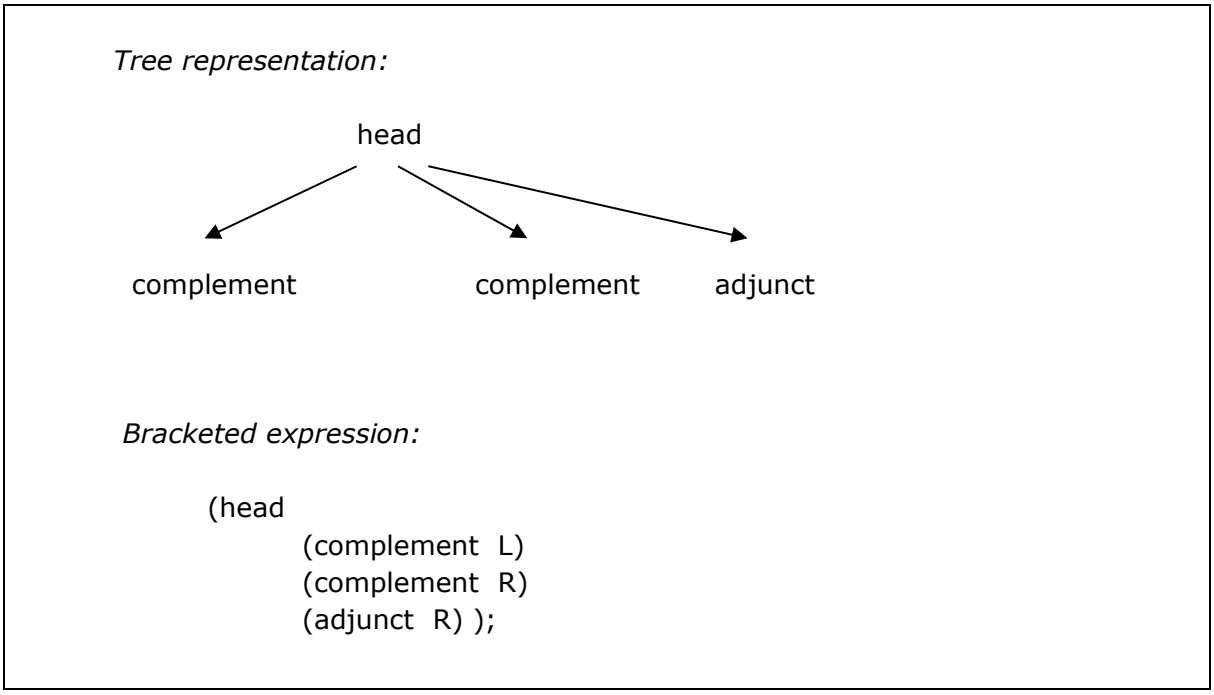


Figure 5: Representation of dependency

Categories

A node in the tree or a corresponding text in the bracketed expression is called a "term". Each term in DRL usually consists of a category. A category consists of a set of attributes. Each attribute consists of an attribute name and a set of values (including the empty set). The set of values is surrounded by square brackets. Values are represented by their name. Several values (usually denoting a disjunction) are separated by commas.

This notation is practical for translating informal linguistic descriptions into formal ones. Bear in mind that traditional linguistic is the primary source of the type of data we need. The ordinary school teacher's statement

> "The character string 'goes' is an inflectional form of the word 'go' indicating a finite verb in the third person singular present tense."

is written in DRL as follows:

```
(lexeme[go] category[verb] form[finite] person[third] number[singular]
tense[present])
```

Complex categories consisting of attribute-value pairs lend themselves to calculating agreement of words and phrases. The simplest case is the intersection of values of those attributes that must agree. If the intersection is empty then there is no agreement. Treating agreement in this way is similar to solving equations in mathematics and logic, rather than applying rules one after the other. This is in fact the essence of unification grammars.

Most grammars of the unification family possess one uniform mechanism of unification. Linguistic reality is too variable, though, to tar all phenomena with the same brush. That is why several equations are available in DUG for calculating agreement. Each attribute must make a choice on a particular method. There is not just one algorithm of unifying the categories of several words or phrases. Instead, different built-in routines are associated with attributes according to a type declaration. Each attribute in a complex category invokes a little program which interacts with the same or with different attributes in the complex category of another item.

PLAIN offers many types of attributes (see the file plain-xml.dtd). Not all of them play a role in syntax. Nevertheless, in order to give you an impression of the total framework, here is a list of the actual assortment:

```
Semantic features:

      lx  lexeme
      rd  reading
      hy  hypernym

Grammatical features:

      ut  utterance property, illocution
      rl  role, syntactic function
      mc  main syntactic category, part of speech
      df  disjunctive feature
      cf  conjunctive feature
      ef  exclusive feature
      of  overwriting feature

Surface form features:

      ch  character string
      qu  quotation
      lp  left punctuation mark
      rp  right punktuation mark
      cs  upper and lower case
      ud  utterance delimiter
```

```
Attribute excluding attributes:

      ne  unacceptable feature

Word order features:

      lt  left side dependent (within tree projection)
      rt  right side dependent (within tree projection)
      sc  numbered succession
      aj  adjacency
      mg  margin position

DUG constructs in syntax descriptions:

      tp   template name in a template
      sl   slot indicator in a template
      cp   complement in a synframe
      ad   adjunct in a synframe
      ea   expected adjunct in a synframe
      co   conjunct in a synframe
      nc   nucleus complement in a synframe
      rs   raising complements in a synframe
      tc   a trace of an elliptic conjunct

Logical constants and transducer rules:

      no   logical not
      tr   logical true
      fl   logical false
      rr   replacement rule
      er   expansion rule
```

Figure 6: Attribute types

Since attributes are processed differently, their type must be specified in the category definition file of each project. A fragment of this file in the german_demo project is the following:

```
<catdef>
    <lx>
        <name>lexem</name>
         <unrestricted/>
    </lx>
</catdef>
```

```
<catdef>
    <mc>
        <name>kategorie</name>
        <val>satz</val>
        <val>verb</val>
        <val>nomen</val>
        <val>adjektiv</val>
        <val>artikelwort</val>
        <val>praeposition</val>
        <val>konjunktion</val>
        <val>praefix</val>
        <val>partikel</val>
    </mc>
</catdef>

<catdef>
    <df>
        <name>form</name>
        <val>finit</val>
        <val>infinitiv</val>
        <val>partizip</val>
        <val>imperativ</val>
        <val>infinitiv_zu</val>
    </df>
</catdef>
```

Figure 7: Attribute declaration of the german_demo project

The XML marker of an attribute definition is <catdef>. First, the type of the attribute has to be declared, e.g. <lx>, <mc> or <df>. This type determines the built-in routine that is invoked if the attribute occurs. Then the name and, possibly, the values of the attribute must be specified. One is completely free as to what name the attributes should have. The names may follow the grammatical tradition of the language in question. Values may be declared as <unrestricted> or they must be listed.

The list of attributes used in a project together with their types can be displayed with the command *Categories* > *Show Attributes* of the PLAIN IDE. We are going to explain the processing of the attributes of the german_demo below when they occur.

Morpho-syntactic lexicon, templates, synframes

The smallest unit of the syntax component is the "word". Words are the units at the interface between the scanner (which reads the text and classifies the segments) and the parser (which finds out the structure of these units). There is room for arbitrary decisions here. What is treated as a word in a concrete implementation is a matter of practicality rather than truth. Words are made available by the morpho-syntactic lexicon. How to create such a lexicon is described in *The Linguist's Guide to Plain – Part 1. Drawing up a Morphological Component.*

"Templates" are like rules in other types of grammars. A template is a tree that mirrors the relationship between a head term and one dependent term in an instance. If a head term can have several dependents then several templates have to be drawn up. The basic idea of a template is that the dependent is described in terms of a slot. This slot is associated with the head. It describes so-to-say the expectation of the head to see particular material in its surroundings. The parser retrieves suitable constructions that can fill the slot. In the examples below, templates are displayed in blue.

"Synframes" are like strict subcategorizations in other types of grammars. A synframe is a tree with a lexical item as head and a number of template references as dependent terms. In this way the valency of a lexical item (its expectations) is specified. In the examples below, synframes are displayed in red.

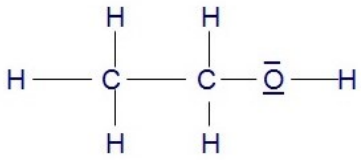
A footnote on complexity

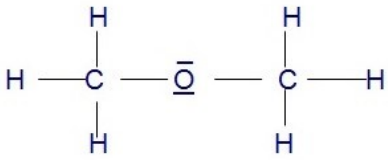
Why valency? Complexity is often estimated by analogy to the SAT (satisfiable) problem: How can one find out under which configuration a formula is true? If there is a sequence of inferences to prove the formula, the problem is P-complex. Such problems can be solved by a deterministic automation in polynomial time. If however all substitutions of variables must be tried out in combination (i.e. the SAT problem) then the problem is NP-complex. The effort to find a solution depends exponentially on the number of variables.

PSG rules | A → B1 + B2 + ... + Bn | **are suspiciously similar to the SAT-problem!**

If such a problem occurs it is likely that something is missing. According to Robert Berwick (*Computational Complexity and Natural Language* 1987*)*, SAT problems are unnatural. They imply that the problem has no specific structure which can be exploited to solve the problem.

There is another possibility for defining a formal system: Chemistry!





Atoms are the basis elements of chemistry (the vocabulary). Each atom has a valency, i.e. combination capacity due to the free electrons in its shell. By stipulating that in each combination one value of one atom must match one value of another atom and that no electron must be left over, the set of all possible molecules (the sentences) is defined.

Simple complements

Complements are constituents that are syntactically related to a word due to the "valency" of the word. Semantically, the heading word is a relation or property that opens up a slot for one or more "arguments".

Examples:

er verreist

wir verreisen

verreisen wir

verreist ihr

Phenomena:

- valency
- morpho-syntactic agreement
- word order

Solution:

- ✓ morpho-syntactic lexicon
- ✓ synframes (lexically triggered)
- ✓ templates
- ✓ attributes (constraints, unification, word order)

Parser Output:

```
wir verreisen

(lexem[verreisen] kategorie[verb] form[finit] numerus[plural]
perfekt[sein] person[erste] stellungstyp[verb_zweit] tempus[praesens]
wortlaut[verreisen] schreibung[klein] s_position[1,2]
    (L rolle[subjekt] lexem[sprecher_pl'] kategorie[nomen]
    determination[+] kasus[nominativ] numerus[plural,C] person[erste,C]
    pronomen[personal] wortlaut[wir ] schreibung[klein] s_position[1]));
```

Short form output: (i.e. just role values, lexeme values and the tree structure)

```
(verreisen
     (subjekt:  sprecher_pl'));
```

The following resources have been deployed to arrive at this result.

Morpho-syntactic lexicon lookup:

```
'verreisen '
(lexem[verreisen] kategorie[verb] form[infinitiv] schreibung[klein]);

'verreisen '
(lexem[verreisen] kategorie[verb] form[finit] numerus[plural]  person[erste,dritte]
 tempus[praesens] schreibung[klein]);
```

Synframe:

```
(lexem[verreisen]
        (complement[+subjekt]) );
```

Templates:

```
(template[+subjekt] kategorie[verb] form[finit]  s_position[2]
stellungstyp[verb_zweit,A]
        (L ____[complement]  rolle[subjekt,A] kategorie[nomen] kasus[nominativ]
        numerus[C] person[C]   w_pronomen[C] determination[+]  s_position[1]));

<test topic="+subjekt">(ich)  /verreise/; (er) /verreist/; (wer) /verreist/ </test>

(template[+subjekt] kategorie[verb] form[finit] s_position[2]
stellungstyp[verb_front,verb_zweit,A]
        (R ____[complement]  rolle[A,subjekt] kategorie[nomen] kasus[nominativ]
        numerus[C] person[C]  determination[+]  s_position[3,6]));

<test topic="+subjekt">/verreist/ (er)? ; /vereist/ (du)?</test>
```

Comments:

First, the words in the input text are recognized by comparing the input with the morpho-syntactic lexicon. Categories consisting of a set of attributes are retrieved. The words may be ambiguous. There may even be alternatives of the segmentation into words. Every alternative is stored in a working store called "chart".

The lexeme is the most important attribute. As mentioned earlier, the lexical meaning is the source of the syntactic power of a word. That is why the lexeme serves as the key to retrieve so-called synframes. A synframe associates syntactic templates with the word.

The 'template' attribute in the template serves for identification. The same template assignment in a synframe can trigger many different templates with the same name. Each one handles another configuration, e.g. the subject before or after the verb in German (er verreist versus verreist er).

The grammar writer may develop the following mental picture of a template:

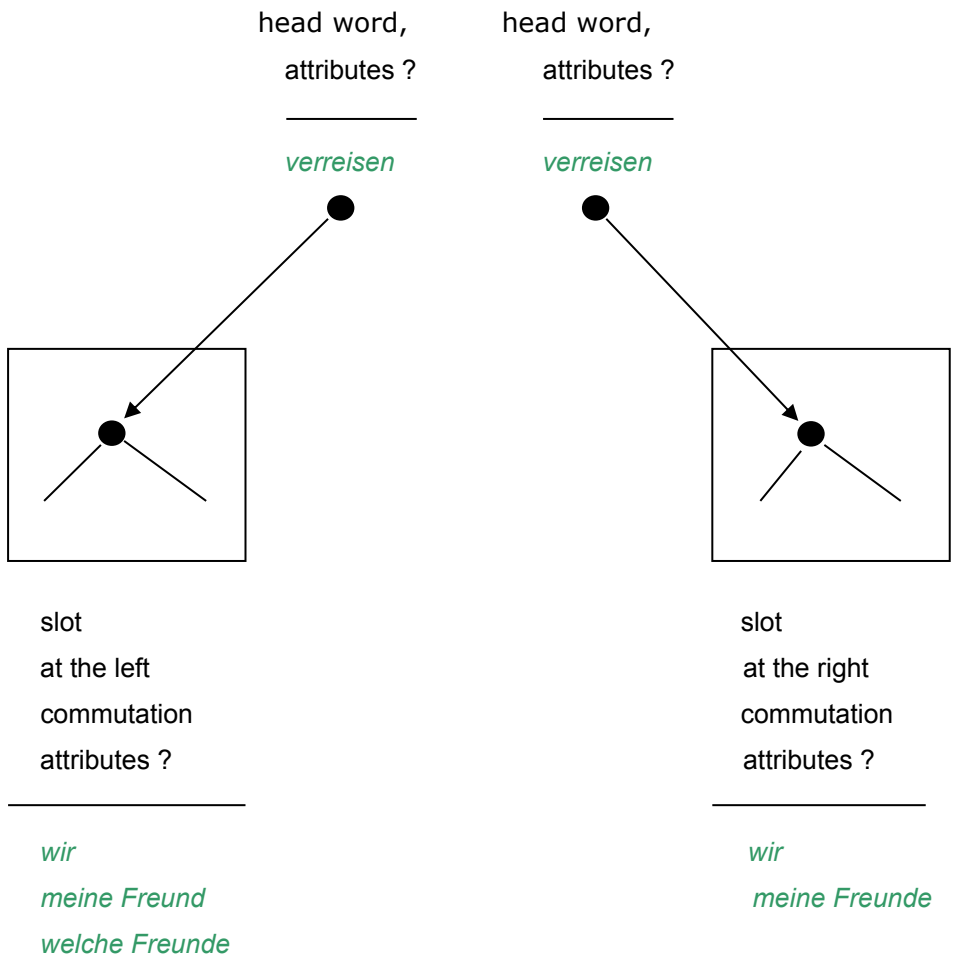


Figure 8: How to imagine a template (one with a slot to the right, one with a slot to the left)

In terms of the dependency tree, a template covers the edge between two nodes. The edge leads from one head node to one dependent node plus anything depending on that node. One could also say the edge connects a head node and a dependent tree. The dependent tree may represent a whole phrase (indicated by the boxes). However, only the top most term is visible. If need be, attributes have to be propagated from below to this surface node.

In terms of linguistic analysis, a template covers the combination capability between a word and one of its contexts. This context can be idntified as a set of substitutable phrases. It is labeled by a syntagmatic role. Substitution tests also reveal grammatical features and the requirement of agreement. Constraints may affect the head word as well as the dependent phrase. All of this information must be written down in the template, either in the head term or in the slot term. The analogy of slot and filler mirrors nicely the substitution mechanism characterizing the taxonomic heuristics.

A few more explanations may be welcome. The slot is marked by '___' followed by the slot type 'complement'. There are other types of slots which will be introduced later.

'L' of type <lt> means "the dependent is to the left of head" , 'R' of type <rt> means "the dependent is to the right of the head. 's_position' of type <sc> is a numbered sequence of dependents in the sentence. Note that word order is described by means of explicit attributes of the terms rather than by projection of the tree. Actually there are five types of word order attributes (as opposed to the single concatenation relation of constituent grammar):

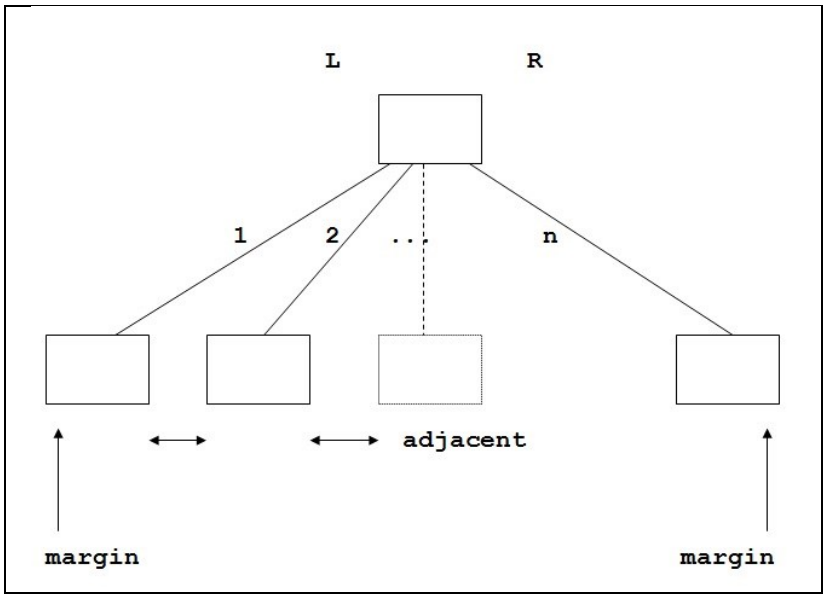


Figure 9: Word order attributes

In addition to the types <lt>, <rt> and <sc>, an attrribute indicates whether a dependent is adjacent to the head (or the head plus previously joined dependents) on the right or on the

left, i.e. type <aj>. Another attribute requires that a dependent is the first or the last of all dependents, i.e. type margin <mg>.

Slot and filler must also agree with respect to 'kategorie' of type <mc>, i.e. the main category or part of speech. Furthermore, 'form', 'kasus', ' numerus', 'person', 'w_pronomen', 'determination' must form a non-empty intersection of values, because these attributes are all declared as disjunctive grammatical features of type <df>.

The flag value 'C' enforces agreement of values between head term and dependent. (You can also think of the values being copied upwards to the head.) In the case of type <df>, the values of the attribute in head and dependent must form a non-empty intersection, too. This intersection can be narrowed down if several dependents copy their values of the same attribute to the head.

The flag value 'A' has the effect that the values in the attribute are not checked for agreement with the instance but just added to the resulting term. For example, 'rolle[subjekt,A]' does not state a condition the filler of the slot must meet, but assigns the attribute of being a subject to the successful assignment. In the sequel such added values can of course function as conditions if they appear without 'A'.

The following mappings exist between lexicon, synframe, template, chart element and filler:

Lexicon

(lexem[verreisen] kategorie[verb] form[infinitiv]);

(lexem[verreisen] kategorie[verb] form[finit] numerus[plural] person[erste,dritte] tempus[praesens]);

Synframe

(lexem[verreisen] (complement[+subjekt]));

Template

(template[+subjekt] kategorie[verb] form[finit] s_position[2] stellungstyp[verb_zweit,A]
(L ___[complement] rolle[subjekt,A] kategorie[nomen] kasus[nominativ] numerus[C]
person[C] w_pronomen[C] determination[+] s_position[1]));

Chart element

(lexem[verreisen] kategorie[verb] form[finit] numerus[plural] person[erste,dritte] tempus[praesens]
s_position[2] stellungstyp[verb_zweit,A]
(L ___[complement] rolle[subjekt,A] kategorie[nomen] kasus[nominativ] numerus[C]
person[C] w_pronomen[C] determination[+] s_position[1]));

Filler

'wir ' (lexem[sprecher_pl'] kategorie[nomen] pronomen[personal] determination[+]
kasus[nominativ] numerus[plural] person[erste]);

The word 'verreisen' is looked up in the morpho-syntactic lexicon. Two results are found (infinitive and finite verb). The lexemes of the words acts as keys to retrieve the synframe. The specified templates are looked up. They are merged with the original attributes from the lexicon in order to form a chart element.

The goal of the parser is to find a filler for the slot of a chart element among the other chart elements. For example the chart element with 'wir' is tried for the slot of the chart element of 'verreisen'. In the course of this check the built-in routines of all attributes are run. If one attribute fails the filling of the whole slot fails.

The chart keeps all the information about each segment at each stage of analysis. What is more, it maps each chart element onto the character sequence in the input by storing its left edge and its right edge. It is necessary for the template writer to understand how the parser, then, works through the chart

In principle, the parser proceeds from left to right. Each new word forms a new chart element (or maybe more if the word is ambiguous). Before continuing to the next word, the parser tries all the previous chart elements that are adjacent to the left of the new word, if either the new word has a slot for the neighbor or if the neighbor has a slot for the new word. If either case occurs then a new chart element is created, now covering a segment arching over the segments of the two previous elements. This new chart element is now made the actual one, looking for adjacent neighbors, and maybe creating new chart elements with broader coverage if a suitable slot can be filled. This process is continuing like a cascade as long as adjacent elements in the chart are untried. Only then, the parser adds the next word to the chart.

The grammarian should know what will happen first and what follows in the course of constructing a complete dependency tree. Only then he can control the process efficiently by constraints and agreement of attributes.

Here is an example:

hat	der	freundliche	Mann	der	Frau	das	Buch	gegeben ?
did	the	friendly	man	to the	women	the	book	give ?

(Did the friendly man give the book to the women?)

While parsing this sentence, the following segments are stored. Each line corresponds to one chart element. The boxes are the actual slots (to the right or to the left of the head phrase). The prerequisite of slot-filling is the the adjacency of segments.

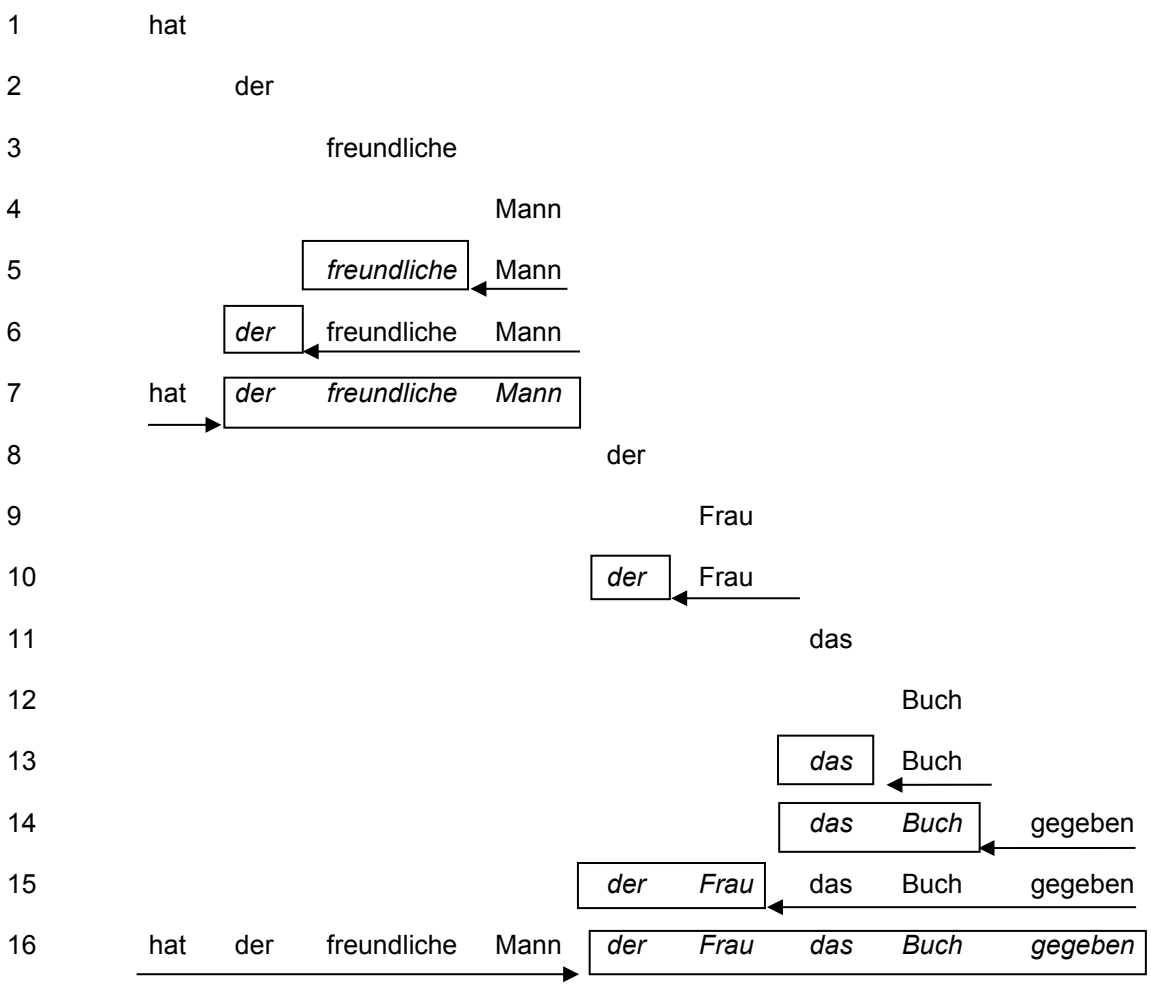


Figure 10: Thee so-called chart and the sequence of slot-fillings

Sentences

Examples:

Er verreist.

Verreist ihr?

Wer verreist?

Verreise!

Phenomena:

- Sentence meaning (in addition to word meaning) exist, e.g. assertion verus question.
- Punctuation marks

Problems:

- Are sentences in conflict with the lexicalistic approach of Dependency Grammar?
- How can sentence meaning be represented in dependency trees?

Solution:

- ✓ The property of being an assertion, question, exclamation is quasi lexicalized.
- ✓ Sentence illocution is represented by lexemes like the meaning of words.
- ✓ These meanings may be attributed to punctuation marks ("Satzzeichen").

Parser Output:

```
Er verreist.

(illokution:  aussage'
    (proposition:  verreisen
       (subjekt:  anaphor_sgm')));

 Verreist ihr?

 (illokution:  frage'
     (proposition:  verreisen
        (subjekt:  adressat_pl')));

 Wer verreist?

 (illokution:  frage'
     (proposition:  verreisen
        (subjekt:  w_person')));

 Verreise!

 (illokution:  ausruf'
     (proposition:  verreisen));
```

Resources (lexicon, synframes, templates):

'.' (lexem[aussage'] kategorie[satz] aeusserung[+]);

'?' (lexem[frage'] kategorie[satz] aeusserung[+]);

'!' (lexem[ausruf'] kategorie[satz] aeusserung[+]);

```
(lexem[aussage']
        (complement[+aussage]));

(lexem[frage']
        (complement[+frage]));

(lexem[ausruf']
        (complement[+aufforderung]));


(template[+aussage] kategorie[satz] rolle[illokution,A]
        ( L ___[complement]  rolle[proposition,A] kategorie[verb] form[finit]
        stellungstyp[verb_zweit] randstellung[+]));

<test topic="+aussage">(Er verreist)/./</test>

(template[+frage] kategorie[satz] rolle[illokution,A]
        ( L ___[complement]  rolle[proposition,A] kategorie[verb] form[finit]
        stellungstyp[verb_front] randstellung[+]));

<test topic="+frage"> (Verreist er) /?/  </test>

(template[+frage] kategorie[satz] rolle[illokution,A]
        ( L ___[complement]  rolle[proposition,A] kategorie[verb] form[finit]
        stellungstyp[verb_zweit] w_pronomen[+] randstellung[+]));

<test topic="+frage">(Wer Verreist)/?/ </test>

(template[+aufforderung] kategorie[satz] rolle[illokution,A]
        ( L ___[complement]  rolle[proposition,A] kategorie[verb] form[imperativ]
        stellungstyp[verb_front,A] randstellung[+]));

<test topic="+aufforderung"> (Verreise)/!/ </test>
```

Comment:

Sentences are treated as complements of an illocution. In written language, the illocution of a sentence can be gathered from punctuation marks. Full stop '.' is associated in the lexicon with 'lexem[aussage']' (assertion), question mark '?' with 'lexem[frage]' (question), exclamation mark '!' with 'lexeme[ausruf']' (exclamation). These lexemes are associated with synframes which assign the appropriate templates.

The special positional feature 'randstellung' (i.e. margin type <mg>) requires that the filler of the slot reaches the edge of the input on the left (if L) or on the right hand side (if R). This attribute prevents the parser of trying to apply the templates to incomplete fragments of the sentence.

Elliptic complements

Examples:

verreise!

er ist verreist.

Problem:

- Sometimes complements must be removed from the valency, e.g. the subject from infinitives and participles.

Solution:

✓ A particular type of slot: ***elliptic.*** This slot is satisfied without a filler.

Template:

```
(template[+subjekt] kategorie[verb] form[infinitiv,partizip,imperativ,infinitiv_zu]
        (___[elliptic]));

<test topic="+subjekt, elliptic">()/verreise/ ! </test>
<test topic="+subjekt, elliptic">Er ist ()/verreist/. </test>
```

Comment:

In the case of the imperative as well as with the infinitive and participle the subject has to be removed from the set of complements of the verb. (The subject may be passed to other elements; see the section on raising below.) This is done by drawing up a subject template whose slot is declared as elliptic.

Syntactic frames and syntagmatic resolution of lexical ambiguity

Examples:

Der Mann gibt dem Kind das Buch. (The man gives the book to the child.)

Es gibt keinen Stau. (There is no traffic jam.)

Der Mann gibt sich freundlich. (The man behaves friendly.)

Phenomenon:

- The same word has various meanings and, hence, various syntactic environments.
- Usually the syntactic context reveals the appropriate lexical meaning.

Parser Output:

```
Der Mann gibt dem Kind das Buch.


 (illokution:  aussage'
    (proposition:  geben transferieren
       (subjekt:  mann
          (determination:  definit'))
       (dativ_objekt:  kind
          (determination:  definit'))
       (trans_objekt:  buch
          (determination:  definit'))));

Es gibt keinen Stau.


 (illokution:  aussage'
    (proposition:  geben existieren
       (subjekt_es:  es partikel)
       (intrans_objekt:  stau
          (determination:  kein))));


Der Mann gibt sich freundlich.

 (illokution:  aussage'
    (proposition:  geben darstellen sich
       (subjekt:  mann
          (determination:  definit'))
       (praedikativ:  freundlich)));
```

Solution:

- ✓ Various synframes for the same lexeme.
- ✓ A separate attribute (type <rd> representing the reading ('lesart') in addition to the lexeme.
- ✓ Disambiguation of the reading attribute due to the syntactic context

.

Synframes (inter alia):

```
(lexem[geben] lesart[transferieren]
        (complement[+subjekt])
        (complement[+dativ,N])
        (complement[+trans]));
```

The value 'N' turns the complement into an optional one (N = the alternative is nothing).

```
(lexem[geben] lesart[existieren]
        (complement[+subjekt_es])
        (complement[+intrans]));

(lexem[geben] lesart[darstellen]
        (complement[+subjekt])
        (complement[+reflexiv_akkusativ])
        (complement[+praed_attribut] ));
```

Templates :

```
(template[+subjekt] kategorie[verb] form[finit]  s_position[2] stellungstyp[verb_zweit,A]
        (L ___[complement]  rolle[subjekt,A] kategorie[nomen] kasus[nominativ] numerus[C]
        person[C]  w_pronomen[C] determination[+]  s_position[1]  ));

<test topic="+subjekjt">(Der Mann )/gibt/ dem Kind das Buch </test>
<test topic="+subjekt">(Wer) /gibt / dem Kind das Buch   </test>

(template[+subjekt_es] kategorie[verb] form[finit] numerus[singular] person[dritte] s_position[2]
stellungstyp[verb_zweit,A]
        (L ___[complement]  rolle[A,subjekt_es] lexem[es] lesart[partikel,A]   s_position[1]));

<test topic="+subjekt_es">(Es) /gibt/  keinen Stau </test>

(template[+dativ] kategorie[verb] form[finit, imperativ] s_position[2]
        (R ___[complement]  rolle[A,dativ_objekt] kategorie[nomen] kasus[dativ]      determination[+]
        stellungstyp[verb_front, verb_zweit,C,A]  s_position[7,8] ));

<test topic="+dativ"> Der Mann  /gibt/ (dem Kind) das Buch.</test>
<test topic="+dativ"> Der Mann  /gibt/ das Buch(dem Kind).</test>

(template[+trans] kategorie[verb] form[finit, imperativ] s_position[2]
        (R ___[complement]  rolle[A,trans_objekt] kategorie[nomen] kasus[akkusativ]
        determination[+] stellungstyp[verb_front, verb_zweit,C,A]  s_position[7,8] ));

<test topic="+trans"> Der Mann  /gibt/ dem Kind (das Buch).</test>
<test topic="+trans"> Der Mann  /gibt/ (das Buch) dem Kind.</test>

 (template[+intrans] kategorie[verb] form[finit, imperativ] s_position[2]
        (R ___[complement]  rolle[A,intrans_objekt] kategorie[nomen] kasus[akkusativ]
        determination[+] stellungstyp[verb_front, verb_zweit,C,A] s_position[9] ));

<test topic="+intrans"> Es /gibt/ (keinen Stau). </test>
<test topic="+intrans"> Sie /bekommt/ (ein Kind). </test>

(template[+reflexiv_akkusativ] kategorie[verb] form[finit, imperativ] s_position[2]
        (R ___[complement, remove]  rolle[A,reflexiv] kategorie[nomen] kasus[akkusativ]
        pronomen[reflexiv] person[C] stellungstyp[verb_front, verb_zweit,C,A]  s_position[7]
        lexem_zusatz[sich,A,C]));

<test topic="+reflexiv_akkusativ"> Er  /gibt/ (sich)  freundlich. </test>
<test topic="+reflexiv_akkusativ"> Du  /gibst/ (dich)  freundlich. </test>
```

Slot type '[remove]' excludes the word 'sich' from becoming a separate node in the dependency tree. The reflexive pronoun is part of the lemma 'sich geben' rather than a separate morpheme (e.g. in 'sich waschen').

```
(template[+praed_attribut] kategorie[verb] form[finit] s_position[2]
        (R ___[complement] rolle[praedikativ,A] kategorie[partikel] verwendung[praedikativ]
        s_position[14] stellungstyp[verb_front,verb_zweit,A,C]));

<test topic="+praed_attribut"> Er /gibt/ sich (freundlich).</test>
```

Comment:

The same lexeme usually has several syntactic frames which are often associated with different readings of the lexeme. The readings are represented in the synframe by the attribute 'lesart'. So, the exact meaning of a word is jointly represented by two attributes: the lexeme, type <lx>, and the reading, type <rd>. The parser achieves disambiguation of the readings by finding the complements that are predicted in the alternative synframes.

Optional and mandatory complements

Examples:

Der Mann gibt der Frau Geld.
Der Mann gibt Geld.
* Der Mann gibt der Frau.

Phenomenon:

- There are mandatory and optional complements in a frame.

Solution:

✓ Alternative 'N' value ("nothing") for optional complements.

Synframe:

```
(lexem[geben] lesart[transferieren]
        (complement[+subjekt])
        (complement[+dativ, N])
        (complement[+trans]));
```

Parser Output:

```
Der Mann gibt Geld.

(illokution:  aussage'
    (proposition:  geben transferieren
       (subjekt:  mann
          (determination:  definit'))
       (trans_objekt:  geld)));

Der Mann gibt der Frau.
No complete parsing achieved
```

If a complement is specified in a synframe then it must occur in the input, except if 'N' is one of the values. 'N' turns the complement into an optional one.

Subclause as complement

Examples:

Die Frau überrascht, dass er verreist.

Dass er verreist, überrascht die Frau.

Phenomena:

- A complement (e.g. the subject) can consist of a clause, because it refers to a fact rather than to a thing.
- There must be a way to deal with punctuation marks.

Parser Output:

```
Die Frau überrascht, dass er verreist.

 (illokution:  aussage'
    (proposition:  überraschen
       (trans_objekt:  frau
          (determination:  definit'))
       (subjekt:  dass
          (proposition:  verreisen
             (subjekt:  anaphor_sgm')))));

Dass er verreist, überrascht die Frau.

 (illokution:  aussage'
    (proposition:  überraschen
       (subjekt:  dass
          (proposition:  verreisen
             (subjekt:  anaphor_sgm')))
       (trans_objekt:  frau
          (determination:  definit'))));
```

The following synframe adds a subject clause ('+subjekt_dass ') to the valency of *überraschen* (surprize)*:*

```
(lexem[überraschen]
    (complement[+subjekt, +subjekt_dass])(complement[+trans]));
```

The following synframe and template combine the conjunction and the rest of the clause:

```
(lexem[dass]
        (complement[+konjunktional_satz]));

(template[+konjunktional_satz] kategorie[konjunktion]
         ( R ___[complement]  rolle[proposition,A] kategorie[verb] form[finit] stellungstyp[verb_end]
        ));

<test topic="+konjunktional_satz">/dass/ (er verreist) </test>
<test topic="+konjunktional_satz">/weil/ (er dem Kind das Buch gibt) </test>
```

The following entries in the morpho-syntactic lexicon belong to the network 'wortende' (end of word). They recognize punctuation marks at the end of the word and, as a result, assign the attribute 'zeichen_rechts' (punctuation mark to the right) to the word. The values of this attribute are the punctuation marks which can terminate a word. The first entry in the list is the space character which is the most frequent word delimiter. It does not introduce an attribute.

```
<form>  <char> </char> </form>
<form> <char>)</char>  <drl>(zeichen_rechts[klammer])</drl> </form>
<form> <char>,</char>  <drl>(zeichen_rechts[komma])</drl> </form>
<form>  <char>.</char>  <drl>(zeichen_rechts[punkt])</drl> </form>
```

The following templates subordinate the subject clauses to the verb of the main clause.

```
(template[+subjekt_dass] kategorie[verb] s_position[2]
        (L ___[complement]  rolle[A,subjekt] kategorie[konjunktion] zeichen_rechts[komma]
        stellungstyp[verb_zweit,C,A]  s_position[1] )));

<test topic="+subjekt_dass">(Dass sie verreist, )/gibt/  dem Mann zu denken. </test>

(template[+subjekt_dass] kategorie[verb] s_position[2] zeichen_rechts[komma]
        (R ___[complement]  rolle[A,subjekt] lexem[dass] kategorie[konjunktion]
        zeichen_rechts[komma, punkt]  stellungstyp[verb_front,verb_zweit,C,A] s_position[16])));

<test topic="+subjekt_dass"> Dem Mann  /gibt/  zu denken, (dass sie verreist.)</test>
```

Comment:

In general, punctuation marks are treated in the morpho-syntactic lexicon as word delimiters in the same way as space. If words are looked up in the morpho-syntactic lexicon then the attributes of the punctuation marks may be added to the attributes of the word they delimit. There are left punctuation marks, type <lp>, and right punctuation markd, type <rp>). Attributes of type <lp> and <rp> are propagated upwards at the outer edges of a construction automatically. It is not necessary to specify 'C'. Eventually they serve as constraints in templates for clauses.

Theoretical remark: This is an example how DUG tries to stick by the dependency structure of sentences but nevertheless to satisfy the linear grouping of constituents which can be observed as well. Another case is coordination (see below).

The first template above requires a filler with a comma at the right edge. This is encoded in the slot as 'zeichen_rechts[komma] '. A suitable filler is ' Dass sie verreist, ' . The second template requires that the main clause is delimited by a comma ('zeichen_rechts[komma]'. This is encoded in the head term. An instance is ' Dem Mann /gibt/ zu denken,' . The

subclause must be finished by a punctuation mark too. According to 'zeichen_rechts[komma, punkt]' in the slot, it can be a comma or a full stop. A suitable filler is ' dass sie verreist.' .

Compounds

Examples:

Staubecken
die Staubecken
die Staubecke
das Staubecken
abgeben
dreiundzwanzig

Phenomenon:

- In German compounds are written as one word. Hence, a compound is a word with several lexemes.
- A compound has an internal dependency structure. There can be complements as well as adjuncts.

Problem:

- There might be alternative segmentations.
- There are quite different dependency structures.

Solution:

- ✓ Since the morpho-syntactic lexicon is implemented as a transition network of letters, alternative transitions are possible and alternative segments can be detected. Segmentation of compounds is a matter of the morphology component.
- ✓ Synframes and templates allow for the "reconstruction" of the compound. The structure of compounds is a matter of the parser.

Parser output:

```
die Staubecken

( becken
    (determination:  definit')
    (zugeordnet:  stau));

( ecke
    (determination:  definit')
    (zugeordnet:  staub));
```

```
die Staubecke

( ecke
    (determination:  definit')
    (zugeordnet:  staub));
```

Resources (lexicon, synframes, templates):

```
1  'Stau'
      (lexem[stau] kategorie[partikel] kompositum[+] numerus[singular]
      verwendung[attributiv] schreibung[gross]);

1  'Staub'
      (lexem[staub] kategorie[partikel] kompositum[+] numerus[singular]
      verwendung[attributiv] schreibung[gross]);

5  'becken '
      (lexem[becken] kategorie[nomen] genus[neutrum]
      kasus[nominativ,dativ,akkusativ] numerus[singular] person[dritte]
      pronomen[nein] schreibung[klein]);


6  'ecken '
      (lexem[ecke] kategorie[nomen] kasus[nominativ,genitiv,dativ,akkusativ]
      numerus[plural] person[dritte] pronomen[nein] schreibung[klein]);

5  'becken '
      (lexem[becken] kategorie[nomen]
      kasus[nominativ,genitiv,dativ,akkusativ] numerus[plural] person[dritte]
      pronomen[nein] schreibung[klein]);

(lexem[becken]
        (complement[+bestimmungswort,N]));

(lexem[ecke]
        (complement[+bestimmungswort,N]));

(template[+bestimmungswort] kategorie[nomen] schreibung[klein]
         L ___[complement] rolle[A,zugeordnet] kategorie[nomen] kompositum[+] ));

<test topic="+bestimmungswort"> (Staub)/ecken/; (Stau)/becken/ </test>
```

Comment:

Compounds (in German written as one word) are decomposed by the lexicon lookup mechanism (the "scanner"). The parser puts the segments of compounds together in the same way as separate words. We assume valency as the driving force in compounds, which is to be encoded by synframes and templates. The syntactic context resolves the ambiguous segmentation, like 'Stau-becken' (reservoir) versus 'Staub-ecken' (dusty corners).

The separable prefix is another instance of a compound. If the verb is at the end of the sentence the prefix is merged with it into one word. Complex cardinal numbers in German are written as one word too.

Portmanteau morphs

Examples:

am Sonntag
im Buch
an dem Sonntag
in dem Buch

Phenomenon:

- A portmanteau morph represents several lexemes that can not be mapped to separate segments of the word. (e.g. am =an dem, im=in dem, zum=zu dem)

Problem:

- The elements of a portmanteau morph might play diverging roles in the dependency structure. **(** e.g. am =an dem, if we have am Sonntag= an is head of Sonntag, dem is dependent of Sonntag).

Solution:

✓ Several terms are associated with one word in the lexicon. The parser treats these terms like independent words.

Parser output:

```
am Sonntag

 ( an
    (komplement:  sonntag
       (determination:  definit')));

an dem Sonntag

( an
    (komplement:  sonntag
       (determination:  definit')));
```

Lexicon:

```
'am '
        (lexem[an] kategorie[praeposition] genus[maskulin,neutrum]
        kasus[dativ] numerus[singular] schreibung[klein])

        (lexem[definit'] kategorie[artikelwort] flexion[stark-schwach]
        genus[maskulin,neutrum] kasus[dativ] numerus[singular]);
```

Two terms are associated with 'am', one like 'an' and one like 'dem'.

The fist term contains 'lexem[an]'. This will trigger the synframe::

```
(lexem[an] kategorie[praeposition]
        (complement[+nominalphrase])
        (adjunct[%lokaladverb] kasus[dativ])
        (adjunct[%richtungsadverb] kasus[akkusativ])
        (adjunct[%temporaladverb] kasus[dativ]));
```

The second term contains 'lexem[definit']' which invokes the synframe:

```
lexem[definit']
        (adjunct[%individuativ]) );
```

'am Sonntag' is analyzed as follows. The second portmanteau morpheme 'lexem[definit']' is subordinated via the '%individuativ' template to 'Sonntag'. The resulting 'Sonntag + definit' is subordinated to the first portmanteau morpheme 'lexem[an]' via the '+nominalphrase' template. So we get the same result as for 'an dem Sonntag'.

Some technical precautions of the parser are necessary, which must not bother the user though.

Multi-word lexemes

Examples:

mit Hilfe des Buches

Phenomenon:

- While a compound and a portmanteau morph is one word with more than one lexeme there may be more than one word which we want to represent just by one lexeme.

No Problem:

- Since the lexicon is a letter tree including the space character, a single entry can consist of several strings.

Parser output:

```
mit Hilfe des Buches

( mit_hilfe
    (komplement:  buch
       (determination:  definit')));
```

Resources:

```
1  'mit '
      (lexem[mit] kategorie[praeposition] kasus[dativ] schreibung[klein]);

5  'Hilfe '
      (lexem[hilfe] kategorie[nomen] genus[feminin]
      kasus[nominativ,genitiv,dativ,akkusativ] numerus[singular]
      person[dritte] pronomen[nein] schreibung[gross]);

1  'mit Hilfe '
      (lexem[mit_hilfe] kategorie[praeposition] kasus[genitiv]
      schreibung[klein]);
```

Here we treat 'mit Hilfe' as a simple preposition with 'kasus[genitiv]'.

.

```
(lexem[mit_hilfe]  (complement[+nominalphrase]) )

(template[+nominalphrase] kategorie[praeposition]
          (R ___[complement] rolle[A,komplement] kategorie[nomen] determination[+] kasus[C]
         genus[C] numerus[C] angrenzend[+]  hyperonym[C]));

<test topic="+nominalphrase">/an/ (der Kirche)</test>
<test topic="+nominalphrase">/am/ (Sonntag)</test>
<test topic="+nominalphrase">/mit Hilfe/ (des Buches)</test>
```

Selectional restrictions

Examples:

Er gibt nie an.

Der Mann gibt das Buch ab.

Er gab den Versuch auf.

Phenomenon:

- Some words expect other words with a particular lexeme in a given construction.

Solution:

- ✓ A new type of slot: ***'select'.***
- ✓ The corresponding selectional restriction must be present in the synframe.

Parser output:

```
Er gibt nie an.

(illokution:  aussage'
    (proposition:  geben sich_brüsten
       (subjekt:  anaphor_sgm')
       (satzadverb:  nie)
       (praefix:  an)));

Der Mann gibt der Frau das Buch ab.


(illokution:  aussage'
    (proposition:  geben überlassen
       (subjekt:  mann
          (determination:  definit'))
       (dativ_objekt:  frau
          (determination:  definit'))
       (trans_objekt:  buch
          (determination:  definit'))
       (praefix:  ab)));
```

Resources:

```
(lexem[geben] lesart[sich_brüsten]
        (complement[+subjekt])
        (complement[+praefix] lexem[an]));

(lexem[geben] lesart[überlassen]
        (complement[+subjekt])
        (complement[+dativ,N])
        (complement[+trans])
        (complement[+praefix] lexem[ab]));
```

```
(template[+praefix] kategorie[verb] form[finit] s_position[2]
        (R ___[complement, select] rolle[A,praefix] kategorie[praefix] kompositum[-]
        stellungstyp[verb_front,verb_zweit,C,A] s_position[14]));

<test topic="+praefix"> Sie  /gibt/ (an). </test>
<test topic="+praefix"> Der Mann  /stellt/ das  Radio (ab). </test>
```

Slot type 'select' in the template requires a lexeme functioning as a selectional restriction to be specified in the synframe. In the above synframes for 'geben' the selectional restriction for the prefix is 'lexem[an]' and 'lexem[ab]' which will be enforced when processing the template.

Besides, the prefix can form a compound with the verb depending on word order. In this case the following template (with the same name!) applies.

```
(template[+praefix] kategorie[verb] form[infinitiv, partizip]
        (L ___[complement, select]  rolle[A,praefix] kategorie[praefix] kompositum[+]
        angrenzend[+]));

<test topic="+praefix"> Er hat das Geld  (ab)/gegeben/.</test>
<test topic="+praefix"> Kannst du mal das Radio  (ab)/stellen/?) </test>
```

Idiomatic expressions

Examples:

Mir standen die Haare zu Berge.

weil mir die Haare zu Berge standen

Dies steht außer Frage.

Dass sie verreist, hat außer Frage gestanden.

Phenomena:

- Idiomatic expressions have a meaning different from the meaning of the words in the expression. (They violate more or less Frege's principle of compositionality.)

Problem:

- They can not be treated as multi-word lexeme, because the expression is subject to syntactic permutation.
- It is tedious to include fragmentary elements, like ' zu Berge', in the lexicon.

Solution:

- ✓ Particular types of slot: '***quote', 'remove'.*** Attribute type *quote* <qu>.
- ✓ Quoted expressions do not have to be in the lexicon. They are directly matched with the input.
- ✓ Removed slots do not form a node in the dependency tree.

Parser output:

```
Mir standen die Haare zu Berge.

 (illokution:  aussage'
    (proposition:  stehen idiom die_Haare_zu_Berge
       (dativ_objekt:  sprecher_sg')));

weil mir die Haare zu Berge standen

 ( weil
    (proposition:  stehen idiom die_Haare_zu_Berge
       (dativ_objekt:  sprecher_sg')));

Dass sie verreist, hat außer Frage gestanden.

(illokution:  aussage'
    (proposition:  haben perfekt_hilfsverb
       (subjekt:  dass
          (proposition:  verreisen
             (subjekt:  anaphor_sgf')))
       (praedikativ:  stehen idiom außer_Frage)));
```

Resources:

```
(lexem[stehen]  lesart[idiom]
        (complement[+dativ])
        (complement[+verbzusatz] lexem_zusatz[die_Haare_zu_Berge]));

(lexem[stehen]  lesart[idiom]
        (complement[+subjekt, +subjekt_dass])
        (complement[+verbzusatz] lexem_zusatz[außer_Frage]));

 (template[+verbzusatz] kategorie[verb] form[finit, imperativ] s_position[2]
        (R ___[complement, quote, remove] s_position[14] stellungstyp[verb_front, verb_zweit, C,A]
        lexem_zusatz[C]));
```

Comment:

Slot type 'quote' in the template requires an attribute of the type <qu> in the synframe. The value of this attribute mirrors the surface form of context that fits into the slot, except that spaces are substituted by underscores. This device is convenient for idiomatic expressions. In the above example the attribute of type <qu> in question is 'lexem_zusatz' (additional lexeme).

The example also illustrates the slot type 'remove'. This has the effect, that no node is built in the dependency tree for the dependent in question. In the example this type of slot is provided for the fixed idiomatic phrase. Since the attribute 'lexem_zusatz' is copied ('C') to the head, the idiomatic meaning is included in the heading verb node. (You have to decide! Although semantically plausible, this solution is syntactically less transparent.)

Using the unknown-word function for proper names

Example:

Herr Frank Walter Steinmaier verreist.

Problem:

- It is not possible to have all proper names in the lexicon.

Solution:

✓ Use the unknown-word attribute as default for proper names.

***Parser output*:**

```
Außenminister Frank Walter Steinmeier verreist.

Warning: This result includes an unknown word

 (illokution:  aussage'
    (proposition:  verreisen
       (subjekt:  aussenminister
          (name:  frank)
          (name:  walter)
          (name:  steinmeier))));
```

Synframes:

(lexem[herr] (complement[+name]));
(lexem[frau] (complement[+name]));
(lexem[aussenminister] (complement[+name]));

Template:

(template[+name] determination[A,+] n_position[3]
(R ___[complement, multiple] rolle[A,name] unbekannt n_position[4]));

<test topic="+name">/ Bundeskanzlerin/ (Angela Merkel) </test>

Comment:

If a word is not found in the lexicon then the unknown word function is executed. The function creates a term with the unknown attribute, i.e. type <yy>. The name of this attribute must be defined in the file *catf.xml.* The attribute has no values. It can be used in templates as selectional restriction. In the example above the attribute is named 'unbekannt' (unknown).

The template treats unknown words to the right of 'herr' (mister), 'frau' (missis) or 'aussenminister' (secretary of state) as names. At least the first occurrence of a proper name in a text is often paired with such a functional description of the person.

Another peculiarity of this template is the slot type 'multiple'. As opposed to normal slots, which can be filled only once, multiple slots can be filled several times, for example if a name consists of several words, like Frank + Walter + Steinmaier.

Variation of word order

Phenomena:

In German each complement of the verb has a relative fixed place in the sentence. In the current fragment we assume, among others, the following sentence positions:

1	Vorfeld, any complement can move to this place
2	finite Verb
3	subject if not in the Vorfeld
7	dative object, transitive accusative object (also as pronouns)
8	dative object, transitive accusative object (pronoun with special conditions only)
9	intransitive accusative object
13	adverbs
14	verb prefix, idiomatic phrase, predicative adjective
15	finite verb or participle corresponding with auxiliaries
16	extrapositions, e.g. subclauses

The numerical sequence attribute, type <sc>, can be used directly to assign these positions to the complements. In the templates below this attribute is named ' s_position'.

Two exceptions of the fixed position of complements give the impression of German as a language with variable word order:

1. Almost any complement can move to the "Vorfeld" (the place in front of the finite verb in the main clause), due to discourse regularities.
2. The verb can move around from front position ('stellungstyp[verb_front]'), second position ['stellungstyp[verb_zweit]' to end position ('stellungstyp [verb_end]').

As a consequence, there must be several templates for the same complement at its fixed place, some for right attachment to the verb, some for left attachment to the verb. Usually the same template names are chosen for various realizations of the same syntactic role including word order. The synframes need not to be augmented.

The following examples are dealing with complements in the Vorfeld:

Dem Kind gibt der Mann das Buch.
Das Buch gibt der Mann dem Kind.
Was gibt der Mann dem Kind?
Wem gibt das Buch zu denken?

Parser output:

```
Dem Kind gibt der Mann das Buch.

      (illokution:  aussage'
          (proposition:  geben transferieren
             (dativ_objekt:  kind
                (determination:  definit'))
             (subjekt:  mann
                (determination:  definit'))
             (trans_objekt:  buch
                (determination:  definit'))));
```

Resources:

```
(template[+dativ] kategorie[verb] form[finit] s_position[2]
        (L ___[complement]  rolle[A,dativ_objekt] kategorie[nomen] kasus[dativ]
       determination[+] fokus[+,A] stellungstyp[verb_zweit,C,A] s_position[1]
       w_pronomen[C]));

<test topic="+dativ, im Vorfeld">(Dem Kind) /gibt/  der Mann das Buch </test>
<test topic="+dativ, im Vorfeld">(Wem) /gibt/  der Mann das Buch? </test>

(template[+trans] kategorie[verb] form[finit] s_position[2]
        (L ___[complement]  rolle[A,trans_objekt] kategorie[nomen] kasus[akkusativ]
       determination[+] fokus[+,A] stellungstyp[verb_zweit,C,A]  s_position[1]
       w_pronomen[C]));

<test topic="+trans, im Vorfeld">(Das Buch) /gibt/  er den Kindern.</test>
<test topic="+trans, im Vorfeld">(Was) /gibt/  er den Kindern ?</test>

 (template[+intrans] kategorie[verb] form[finit] s_position[2]
        (L ___[complement]  rolle[A,intrans_objekt] kategorie[nomen]
       kasus[akkusativ] determination[+] stellungstyp[verb_zweit,C,A]  s_position[1]
       w_pronomen[C]));

<test topic="+intrans, im Vorfeld">(Ein Kind) /bekommt/ sie nicht.</test>
<test topic="+intrans, im Vorfeld">(Was) /bekomme/ ich ?</test>
```

The subject is often in the Vorfeld. In English this is almost always the case. While in German the Vorfeld is occupied by some other complement there must be another spot for the subject, to the right of the main verb:

```
(template[+subjekt] kategorie[verb] form[finit] s_position[2]
stellungstyp[verb_front,verb_zweit,A]
        (R ___[complement]  rolle[A,subjekt] kategorie[nomen] kasus[nominativ]
       numerus[C] person[C]   determination[+]  s_position[3,6]));

<test topic="+subjekt, nach dem Verb"> Das Buch  /gibt/ (der Mann)  dem Kind .
</test>
<test topic="+subjekt, nach dem Verb"> Das Buch  /gibt/ dem Kind (der Mann).
</test>
```

```
(template[+subjekt_es] kategorie[verb] form[finit] numerus[singular] person[dritte]
s_position[2]
        (R ___[complement]  rolle[A,subjekt_es] lexem[es] lesart[partikel,A]
        stellungstyp[verb_front,verb_zweit,C,A] angrenzend[+]));

<test topic="+subjekt_es, nach dem Verb"> Heute  /regnet/ (es). </test>
```

Within relative clauses, the relative pronoun is in front and the verb at the end:

welchem die Frau das Buch gibt

welches die Frau dem Mann gibt

welcher der Frau das Buch gibt

welches es gibt

Parser output:

```
welchem die Frau das Buch gibt

 ( geben transferieren
    (dativ_objekt:  relativ_sgn')
    (subjekt:  frau
       (determination:  definit'))
    (trans_objekt:  buch
       (determination:  definit')));
```

Resources:

```
(template[+subjekt] kategorie[verb] form[finit] s_position[15] stellungstyp[verb_end,A]
         (L ___[complement]  rolle[A,subjekt] kategorie[nomen] kasus[nominativ]  numerus[C]
        person[C] pronomen[relativ]  relativ_pronomen[C]  s_position[1]));

<test topic="+subjekt, Relativpronomen">(der) dem Kind das Buch/gibt/</test>
<test topic="+subjekt, Relativpronomen">(welches)  dem Mann zu denken /gibt/</test>

(template[+dativ] kategorie[verb] form[finit] s_position[15]
         (L ___[complement]  rolle[A,dativ_objekt] kategorie[nomen] kasus[dativ]  pronomen[relativ]
        relativ_pronomen[C] stellungstyp[verb_end,C,A] s_position[1]));

<test topic="+dativ, Relativpronomen">(dem)  er das Buch/ gibt /</test>

(template[+trans] kategorie[verb] form[finit] s_position[15]
         (L ___[complement]  rolle[A,trans_objekt] kategorie[nomen] kasus[akkusativ]
        pronomen[relativ] relativ_pronomen[C]  stellungstyp[verb_end,C,A] s_position[1]));

<test topic="+trans, Relativpronomen">(die)  er den Kindern /gibt/ </test>


(template[+intrans] kategorie[verb] form[finit] s_position[15]
        (L ___[complement]  rolle[A,intrans_objekt] kategorie[nomen] kasus[akkusativ]
        pronomen[relativ] relativ_pronomen[C]  stellungstyp[verb_end,C,A] s_position[1]));

<test topic="+intrans, Relativpronomen">(den)  es /gibt/ </test>
```

The verb at the end in subclauses:

weil er der Frau das Buch gibt

weil es keinen Stau gibt

Parser output:

```
weil er der Frau das Buch gibt

( weil
    (proposition:  geben transferieren
       (subjekt:  anaphor_sgm')
       (dativ_objekt:  frau
          (determination:  definit'))
       (trans_objekt:  buch
          (determination:  definit'))));
```

Resources:

```
(template[+subjekt] kategorie[verb] form[finit] s_position[15] stellungstyp[verb_end,A]
        (L ___[complement]  rolle[A,subjekt] kategorie[nomen] kasus[nominativ]  numerus[C]
       person[C]   s_position[3]));

<test topic="+subjekt, Nebensatz"> weil  (er) dem Kind das Geld /gibt/ </test>

(template[+subjekt_es] kategorie[verb] form[finit] numerus[singular] person[dritte] s_position[15]
 stellungstyp[verb_end,A]
       (L ___[complement]  rolle[A,subjekt_es] lexem[es] lesart[partikel,A]  s_position[3] ));

<test topic="+subjekt_es, Nebensatz"> weil (es) heute /regnet/ </test>

(template[+intrans] kategorie[verb] form[finit, infinitiv, partizip] s_position[15]
        (L ___[complement]  rolle[A,intrans_objekt] kategorie[nomen] kasus[akkusativ]
       determination[+] stellungstyp[verb_zweit, verb_end,C,A] s_position[9]));

<test topic="+intrans, Nebensatz"> weil sie (ein Kind) /bekommt/ </test>

 (template[+reflexiv_akkusativ] kategorie[verb] form[finit, infinitiv, partizip] s_position[15]
        (L ___[complement, remove]  rolle[A,reflexiv] kategorie[nomen] kasus[akkusativ]
       pronomen[reflexiv] person[C]   stellungstyp[verb_end,C,A] s_position[7]
       lexem_zusatz[sich,A,C]));

<test topic="+reflexiv_akkusativ, Nebensatz"> weil du (dich)  freundlich /gibst/ </test>

(template[+verbzusatz] kategorie[verb] form[finit, infinitiv, partizip]
       (L ___[complement, remove] angrenzend[+] stellungstyp[verb_end, C,A] lexem_zusatz[C]));

<test topic="+verbzusatz, nebensatz"> weil das Buch dem Mann (zu denken) /gibt/ </test>
<test topic="+verbzusatz"> Das Buch hat dem Mann (zu denken) /gegeben/. </test>
```

Note: The attribute 'angenzend[+]' (adjacent, i.e. type <aj>) is another word order feature.

***The positions of dative and accusative objects are dependent on each other if pronouns are among them*:**

Er gibt dem Kind das Buch.

Er gibt das Buch dem Kind.

Er gibt es dem Kind.

***Er gibt dem Kind es.**

Er gibt ihm das Buch.

***Er gibt das Buch ihm**.

weil er es dem Kind gibt

***weil er dem Kind es gibt**

Er gibt es ihr.

***er gibt ihr es**

weil er es ihr gibt

***weil er ihr es gibt**

Parser output:

```
Er gibt es ihr.

(illokution:  aussage'
    (proposition:  geben transferieren
       (subjekt:  anaphor_sgm')
       (trans_objekt:  anaphor_sgn')
       (dativ_objekt:  anaphor_sgf')));
```

Resources:

```
(template[+dativ] kategorie[verb] form[finit, imperativ] s_position[2]
        (R ___[complement]  rolle[A,dativ_objekt] kategorie[nomen] ohne[pronomen] kasus[dativ]
       determination[+] stellungstyp[verb_front, verb_zweit,C,A]  s_position[7,8]));

<test topic="+dativ"> Er /gibt/ (dem Kind)  das Buch. </test>
<test topic="+dativ"> Er /gibt/  das Buch (dem Kind). </test>

(template[+dativ] kategorie[verb] form[finit, infinitiv, partizip] s_position[15]
        (L ___[complement]  rolle[A,dativ_objekt] kategorie[nomen] ohne[pronomen]  kasus[dativ]
       determination[+] stellungstyp[verb_end,C,A]  s_position[7,8]));

<test topic="+dativ, Nebensatz"> weil der Mann (dem Kind)  das Buch /gibt/</test>
<test topic="+dativ, Nebensatz"> weil er das Buch (dem Kind) /gibt/</test>

(template[+trans] kategorie[verb] form[finit, imperativ] s_position[2]
        (R ___[complement]  rolle[A,trans_objekt] kategorie[nomen] ohne[pronomen]
       kasus[akkusativ] determination[+] stellungstyp[verb_front, verb_zweit,C,A]  s_position[7,8] ));

<test topic="+trans"> er) /gibt/  dem Kind (das Buch)</test>
<test topic="+trans"> er) /gibt/ (das Buch)  dem Kind </test>
```

```
(template[+trans] kategorie[verb] form[finit, infinitiv, partizip] s_position[15]
        (L ___[complement]  rolle[A,trans_objekt] kategorie[nomen] ohne[pronomen]
        kasus[akkusativ] determination[+] stellungstyp[verb_end,C,A]   s_position[7,8]));

<test topic="+trans, Nebensatz"> weil der Mann dem Kind (das Buch) /gibt/ </test>
<test topic="+trans, Nebensatz"> weil er (das Buch)  dem Kind /gibt/ </test>
```

So far, if both the transitive object and the dative object are noun phrases they can each either occur first or second ('s_position[7,8]').

```
(template[+dativ] kategorie[verb] form[finit, imperativ] s_position[2]
        (R ___[complement]  rolle[A,dativ_objekt] kategorie[nomen] kasus[dativ]
        pronomen[personal,demonstrativ,reflexiv]   stellungstyp[verb_front, verb_zweit,C,A]
        s_position[7]));

<test topic="+dativ, Pronomen"> Er /gibt/  (ihr)  das Buch. </test>
<test topic="+dativ, Pronomen"> Er /hilft/ (sich)  selbst .</test>

(template[+dativ] kategorie[verb] form[finit, infinitiv, partizip] s_position[15]
        (L ___[complement]  rolle[A,dativ_objekt] kategorie[nomen] kasus[dativ]
        pronomen[personal,demonstrativ, reflexiv]  stellungstyp[verb_end,C,A] s_position[7,8]));

<test topic="+dativ, Pronomen, Nebensatz"> weil er  (ihr) das Buch /gibt/ </test>
<test topic="+dativ, Pronomen, Nebensatz"> weil er  (sich)  selbst /hilft/ </test>
 <test topic="+dativ, Pronomen, Nebensatz"> weil er es (ihr) /gibt/ </test>

(template[+dativ] kategorie[verb] form[finit, imperativ] s_position[2] co_pronomen[trans]
        (R ___[complement]  rolle[A,dativ_objekt] kategorie[nomen] kasus[dativ]
        pronomen[personal,demonstrativ]  stellungstyp[verb_front, verb_zweit,C,A] s_position[8]));

<test topic="+dativ, Pronomen. Inversion"> er /gibt/   es (ihr) </test>
<test topic="+dativ, Pronomen, Inversion">/gib/  es (ihm) </test>
```

The dative object as a pronoun must be in the last possible position ('s_position[8]') if the transitive object is also a pronoun (i.e. 'co_pronomen[trans]')

```
(template[+trans] kategorie[verb] form[finit, imperativ] s_position[2]
        (R ___[complement]  rolle[A,trans_objekt] kategorie[nomen] kasus[akkusativ]
        pronomen[personal,demonstrativ,reflexiv]  stellungstyp[verb_front, verb_zweit,C,A]
        co_pronomen[trans,A,C] s_position[7]));

<test topic="+trans, Pronomen"> Er /gib/t (es)  der Frau.</test>
<test topic="+trans, Pronomen"> Er /gibt/ (es)  ihr.</test>
<test topic="+trans, Pronomen"> Er /wäscht/ (sich) .</test>

(template[+trans] kategorie[verb] form[finit, infinitiv, partizip] s_position[15]
        (L ___[complement]  rolle[A,trans_objekt] kategorie[nomen] kasus[akkusativ]
        pronomen[personal,demonstrativ,reflexiv]   stellungstyp[verb_end,C,A] s_position[7]));

<test topic="+trans, Pronomen, Nebensatz"> weil er (es)  dem Kind /gibt/ </test>
<test topic="+trans, Pronomen, Nebensatz"> weil er (sich) /wäscht/ </test>
```

The transitive object as a pronoun always occupies the first possible position ('s_position[7]').

Nucleus complement and raising

Examples with auxiliaries, perfect tense:

Der Mann hat dem Kind das Buch gegeben.
Dass die Frau verreist ist, hat den Mann überrascht.
Ich bin verreist.

Phenomenon:

- The auxiliary and the main verb form a unit with respect to the complements. Traditionally this unit is called "nucleus".

Problem:

- The complements, including the subject, are introduced in the synframes of the (infinite) main verb. For example, the type of subject (+subjekt, +subjekt_es, +subjekt_dass) depends on the main verb rather than on the auxiliary.
- On the other hand the subject phrase and the (finite) auxiliary verb must agree in person and number. That is why the subject slot must be subordinated to the auxiliary.

Solution:

✓ New type of slot: **'nucleus',** and a new term in synframes: 'raising_complement'.
.

Parser output:

```
Dass die Frau verreist ist,  hat den Mann überrascht.

(illokution:  aussage'
    (proposition:  haben perfekt_hilfsverb
       (subjekt:  dass
          (proposition:  sein perfekt_hilfsverb
             (subjekt:  frau
                (determination:  definit'))
             (praedikativ:  verreisen)))
       (praedikativ:  überraschen
          (trans_objekt:  mann
             (determination:  definit')))));
```

Synframes:

```
(lexem[sein] lesart[perfekt_hilfsverb] perfekt[sein]
        (nucleus_complement[+praed_perfekt])
        (raising_complement[+subjekt,+subjekt_es,+subjekt_dass]));

(lexem[haben] lesart[perfekt_hilfsverb] perfekt[haben]
        (nucleus_complement[+praed_perfekt])
        (raising_complement[+subjekt,+subjekt_es,+subjekt_dass]));
```

These synframes result in a nucleus slot for the predicative ('+praed_perfekt') and in alternative raising complements ('+subjekt,+subjekt_es,+subjekt_dass'). If one of the subject slots is filled then the program checks whether this type of subject is part of the valency of the verb that fills the nucleus slot.

Templates:

```
(template[+praed_perfekt] kategorie[verb] form[finit] s_position[2]
        (R ___[nucleus] rolle[praedikativ,A] kategorie[verb] form[partizip] perfekt[C]
        s_position[15]));

<test topic="+praed_perfekt"> der Mann /hat/  dem Kind das buch (gegeben) </test>
<test topic="+preaed_perfekt"> der Mann /ist/ (verreist) </test>

(template[+praed_perfekt] kategorie[verb] form[finit] s_position[15]
        (L ___[nucleus] rolle[praedikativ,A] kategorie[verb] form[partizip] perfekt[C]
        angrenzend[+]));

<test topic="+praed_perfekt, Nebensatz"> weil der Mann dem Kind das Buch
(gegeben) /hat/ </test>
<test topic="+praed_perfekt, Nebensatz"> weil der Mann (verreist) /ist/ </test>
```

The attribute ' perfekt' (either 'haben' or 'sein' auxiliary) must be assigned to each verb by means of a synframe:

```
(lexem[geben,A]  kategorie[verb] perfekt[haben]);
(lexem[überraschen,A]  kategorie[verb] perfekt[haben]);
(lexem[verreisen,A]  kategorie[verb] perfekt[sein]);
```

Note: Synframes allow to introduce additional attributes which are not present in the morphological lexicon. See section *Introducing additional attributes* below.

Raised complements must be removed from the original verb.This is done by means of elliptic slots:

```
(template[+subjekt] kategorie[verb] form[infinitiv,partizip,imperativ,infinitiv_zu]
        (___[elliptic] ));

(template[+subjekt_es] kategorie[verb] form[infinitiv,partizip,imperativ,infinitiv_zu]
        (___[elliptic] ));

 (template[+subjekt_dass] kategorie[verb] form[infinitiv,partizip,imperativ,infinitiv_zu]
        (___[elliptic] ));
```

Examples with copula and predicative adjectives:

Der Mann ist freundlich.

Die Frau ist dem Mann überlegen.

Dass der Mann verreist, ist der Frau unheimlich.

Phenomena:

- If there is a predicative adjective then the complements depend on the adjective's valency. However the subject slot must be subordinated to the copula auxiliary, because both must agree in person and number.

Solution:

✓ The predicative adjective and the auxiliary verb 'sein' form a nucleus. The subjects are raised.

Parser output:

```
Die Frau ist dem Mann überlegen.

 (illokution:  aussage'
    (proposition:  sein kopula
       (subjekt:  frau
          (determination:  definit'))
       (praedikativ:  überlegen
          (dativ_objekt:  mann
             (determination:  definit')))));
```

Note: In the following examples, the category 'kategorie[partikel]' has been chosen for the uninflected form of the adjective (i.e. uninflected adjectives are treated as particles). This is arranged for in the morpho-syntactic lexicon.

Resources:

```
(lexem[sein] lesart[kopula]
        (nucleus_complement[+praed_adjektiv])
        (raising_complement[+subjekt,+subjekt_es, +subjekt_dass]));

(lexem[freundlich] kategorie[partikel]
        (complement[+subjekt]));

(lexem[überlegen] kategorie[partikel]
        (complement[+subjekt])
        (complement[+dativ]));

(lexem[unheimlich] kategorie[partikel]
        (complement[+subjekt,+subjekt_dass])
        (complement[+dativ]));
```

```
(template[+praed_adjektiv] kategorie[verb] form[finit] s_position[2]
        (R ___[nucleus] rolle[praedikativ,A] kategorie[partikel]
        verwendung[praedikativ] s_position[14]));

<test topic="+praed_adjektiv"> der Mann /ist/ (freundlich) </test>

(template[+praed_adjektiv] kategorie[verb] form[finit] s_position[15]
        (L___[nucleus] rolle[praedikativ,A] kategorie[partikel] verwendung[praedikativ]
        s_position[14]));

<test topic="+praed_adjektiv, Nebensatz"> weil der Mann (freundlich) /ist/ </test>

(template[+subjekt] kategorie[partikel]
        (___[elliptic] ));

<test topic="+subjekt, elliptisch bei prädikatischem Adjektiv (partikel)"> der Mann ist
()/freundlich/ </test>
```

Complement of the predicative adjective that isnot raised:

```
(template[+dativ] kategorie[partikel] s_position[15]
        (L ___[complement]  rolle[A,dativ_objekt] kategorie[nomen] ohne[pronomen]
        kasus[dativ] determination[+]  s_position[7,8] ));

<test topic="+dativ, zum praed. Adjektiv"> die Frau ist (dem Mann) /überlegen/
</test>
<test topic="+dativ, zum praed. Adjektiv"> die Frau ist (dem Mann)  das Geld)
/schuldig/ </test>
<test topic="+dativ, zum praed. Adjektiv"> weil die Frau ist (dem Mann)  Geld
/schuldig/  ist </test>
```

Introducing additional attributes

It is desirable to introduce attributes in addition to the ones looked up in the morpho-syntactic lexicon. This can be done within synframes. On the one hand one can simply add attributes in the head term of the synframe:

```
(lexem[geben] lesart[transferieren]
        (complement[+subjekt])
        (complement[+dativ,N])
        (complement[+trans]));

(lexem[geben] lesart[existieren]
        (complement[+subjekt_es])
        (complement[+intrans]));
```

The attribute 'lesart' (reading) is introduced in the synframe rather than in the lexicon, because the reading is paired with a syntagmatic behavior (the valency). The attribute 'lexem', on the other hand, is assigned in the lexicon. It is a first approximation to meaning but it might still be ambiguous.

One can write synframes without complements. Their purpose is solely to introduce attributes.

```
(lexem[geben,A]  kategorie[verb] perfekt[haben]);
(lexem[verreisen,A]  kategorie[verb] perfekt[sein]);
```

We need the attribute 'perfekt' in order to determine whether the perfect of a particular verb is formed with the auxiliary 'haben' or 'sein'. The flag 'A' indicates that the following attributes are to be inherited by all existing synframes of the respective lexeme. If 'A' is omitted, a new syntactic frame with no complements would be created. The attributes would not be passed to the existing frames.

There are many other reasons for introducing attributes by synframes. Here semantic features are introduced that serve to identify temporal adverbs:

```
(lexem[sonntag] hyperonym[zeit]);
(lexem[montag] hyperonym[zeit]);
(lexem[dienstag] hyperonym[zeit]);
```

The following s one possibility to fit continuativa (like 'Geld') or pronouns into slots that usually require a phrase with a determiner ('er gibt der Frau *Buch/das Buch/Geld'). Assigning the attribute 'determination[+]' is so-to-say determination by default.

```
(lexem[geld] determination[+]);
(lexem[w_person'] determination[+]);
(lexem[w_neutr'] determination[+]);
```

Discontinuous constituents

Examples:

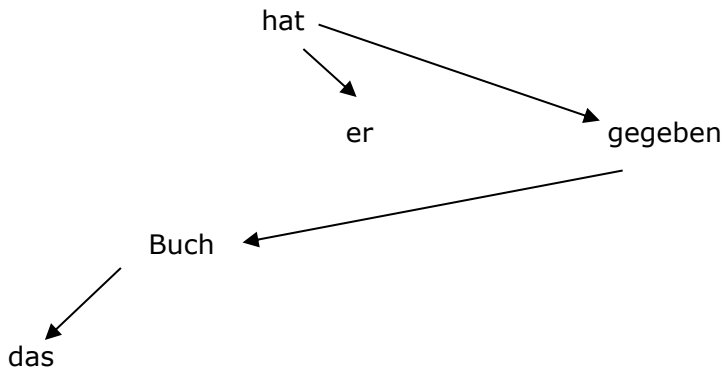
Das Buch hat er dem Mann gegeben.

Dem Mann hat er Geld gegeben.

Dem Mann das Geld hat er genommen.

Phenomenon:

- There are linear word order mechanisms (e.g. topicalization, extraposition, scrambling) which traverse the dependency relations.



Problem:

- These mechanisms lead to projectivity violations (crossing arcs) of the dependency trees. The chart parser works strictly bottom-up and can only combine adjacent constituents.

Unfavorable solution:

Attaching the displaced constituent to some other head, e.g. 'das Buch' to 'hat'. This would blur its syntagmatic role.

Solution:

✓ A special type of slot: '***discontinuous***', the addition of a so-called *mother term* to the template.

Parser Output:

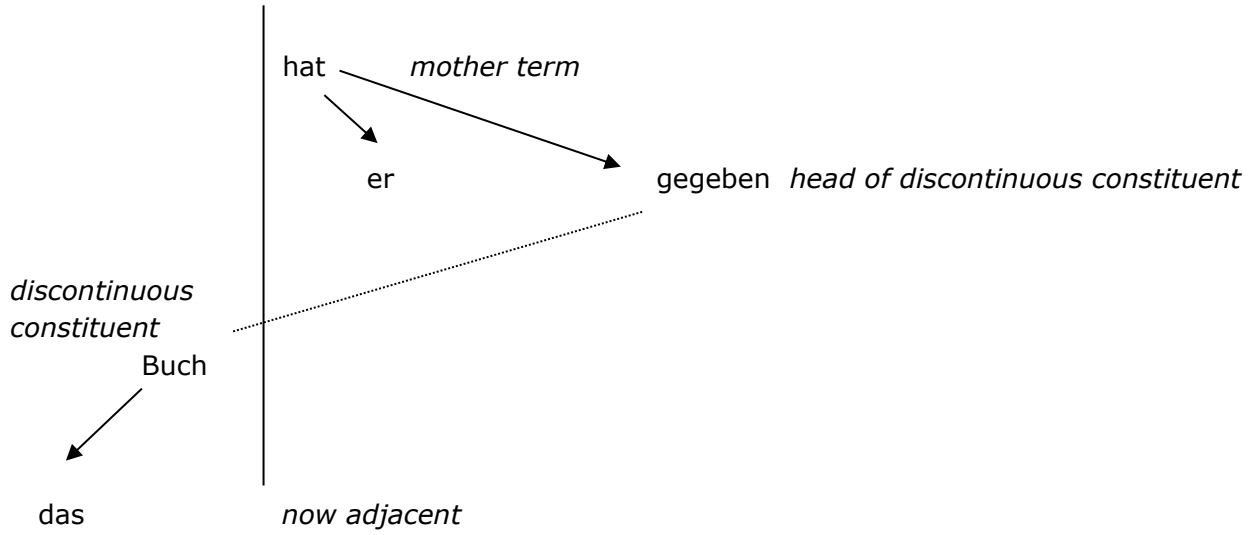
```
Dem Mann das Geld hat er genommen.

(illokution:  aussage'
    (proposition:  haben perfekt_hilfsverb
       (subjekt:  anaphor_sgm')
       (praedikativ:  nehmen
          (dativ_objekt:  mann
             (determination:  definit'))
          (trans_objekt:  geld
           (determination:  definit'))))));
```

Comment:

Long distance dependencies may involve discontinuity of heads and complements. For example, 'Das Buch' in 'Das Buch hat er dem Mann gegeben' is the direct object of 'gegeben' and should be subordinated to this term. However, the finite verb and the raised subject 'hat er' (which in turn is the head of 'gegeben') are located in between and hence separate the complement 'Buch' from its head 'gegeben'.

This causes a technical problem since the parser combines adjacent substrings only. Anything else would result in a "combinatorial explosion". The solution is the special slot type 'discontinuous'. This type of slot needs not to be filled right away. Instead it is propagated upward from one intermediate result to the other until it is passed to a so-called "mother term". A mother term dominates the head of a discontinuous constituent which is somewhere down in the tree. If a filler is adjacent to the mother term and can fill the slot, then the filler is subordinated to the original head of the slot.



Templates:

```
(template[+trans] kategorie[verb] form[partizip] rolle[praedikativ]
        (L ___[discontinuous]  rolle[A,trans_objekt] kategorie[nomen] kasus[akkusativ]
       fokus[+,A] relativ_pronomen[C] ))
(kategorie[verb] form[finit] stellungstyp[verb_zweit] lesart[perfekt_hilfsverb]);

<test topic="+trans, diskontinuierlich">(Das Buch)  hat er den Kindern /gegeben/
</test>

(template[+dativ] kategorie[verb] form[partizip]
        (L ___[discontinuous]  rolle[A,dativ_objekt] kategorie[nomen] kasus[dativ]
       fokus[+,A] relativ_pronomen[C] ))
(kategorie[verb] form[finit]);
```

```
<test topic="+dativ, diskontinuierlich">(Den Kindern)  hat er das Buch /gegeben/.
</test>
```

A template for a discontinuous complement contains the head term, the slot description and, as a third term, the mother term. Combining heads and dependents continues until the latest head unifies with the mother term of a waiting discontinuous template. Now, normal adjacent attachment is tried. If successful, the filler of the discontinuous slot is subordinated to the original head rather than to the mother term.

Note that we have provided the element in extraposition with the attribute 'fokus[+]'. This is a touch of discourse analysis which has to be accomplished later.

Adjuncts

Examples of prepositional adverbials as typical adjuncts:

Der Mann gibt dem Kind das Buch in der Kirche.
Er geht in die Kirche.
Es steht im Buch.
Er steht an der Kirche.
Er verreist am Sonntag.

Phenomena:

- Adjuncts are dependents that can occur with many heads.
- Their syntagmatic role is defined by their own meaning rather than by the valency of the head.
- An adjunct might be ambiguous with respect to role.

Solution:

- ✓ Enhancement of synframes: introduction of 'adjunct' rterms in addition to complements..
- ✓ Semantic attributes may be used within templates.

Parser output:

```
Er geht in die Kirche.

 (illokution:  aussage'
    (proposition:  gehen
       (subjekt:  anaphor_sgm')
       (richtung:  in
          (komplement:  kirche
             (determination:  definit')))));


Er steht an der Kirche.

 (illokution:  aussage'
    (proposition:  stehen
       (subjekt:  anaphor_sgm')
       (ort:  an
          (komplement:  kirche
             (determination:  definit')))));


Er verreist am Sonntag.

 (illokution:  aussage'
    (proposition:  verreisen
       (subjekt:  anaphor_sgm')
       (zeit:  an
          (komplement:  sonntag
             (determination:  definit')))));
```

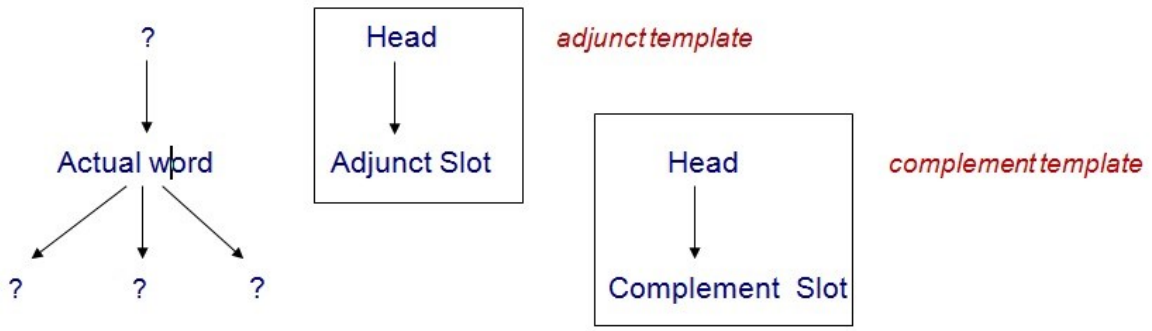


Figure 11: How adjunct and complement templates correspond with the dependency tree

Adjuncts are dependents like complements. Their templates are much the same too: A dependency relationship is established between a head and a dependent. However, while complements are considered as the result of the lexical "valency" of a head and hence are specified in a synframe of the head, adjuncts are considered to be independent. They can occur with many heads. Therefore the template for an adjunct is associated in a synframe with

the adjunct itself. The aim of the adjunct template is to find a head that allows the attachment of the adjunct.

There are borderline cases. Sometimes the choice between complement and adjunct is simply a matter of efficiency. If a particular dependent is often absent then it is more efficient to treat it as adjunct. An adjunct attachment is tried only if the adjunct is present. A complement slot is always tried even if the context is completely unsuitable.

Synframes for adverbial prepositions:

Note: As a convention (technically not necessary) we start the names of templates of complements with '+' and adjuncts with '%'.

```
(lexem[in]
        complement[+nominalphrase])
        (adjunct[%lokaladverb] kasus[dativ])
        (adjunct[%richtungsadverb] kasus[akkusativ]));

(lexem[an] kategorie[praeposition]
        (complement[+nominalphrase])
        (adjunct[%lokaladverb] kasus[dativ])
        (adjunct[%richtungsadverb] kasus[akkusativ])
        (adjunct[%temporaladverb] kasus[dativ]));
```

Template for the noun phrase:

```
(template[+nominalphrase] kategorie[praeposition]
         (R ___[complement] rolle[A,komplement] kategorie[nomen] determination[+] kasus[C]
        genus[C] numerus[C] angrenzend[+] hyperonym[C]));

<test topic="+nominalphrase">/an/ (der Kirche) </test>
<test topic="+nominalphrase">/in/ (die Kirche) </test>
<test topic="+nominalphrase">/in/ (der Kirche) </test>
```

Templates for adverbial adjuncts:

```
(template[%temporaladverb] kategorie[verb] s_position[2]
         (R ___[adjunct] rolle[zeit,A] s_position[13]  hyperonym[zeit]));

<test topic="%temporaladverb ">Er /verreist/ (am Sonntag). </test>

(template[%richtungsadverb] kategorie[verb] hyperonym[sich_bewegen] s_position[2]
         (R ___[adjunct] rolle[richtung,A] s_position[13] ));

<test topic="%richtungsadverb ">Er /geht/( in die Kirche). </test>

(template[%lokaladverb] kategorie[verb] s_position[2]
         (R ___[adjunct] rolle[ort,A] s_position[13] ohne[hyperonym] ));

<test topic="%lokaladverb ">Er /steht/ (an der Kirche). </test>
```

Semantic attributes:

(lexem[gehen] hyperonym[sich_bewegen]);

(lexem[sonntag] hyperonym[zeit]);

(lexem[montag] hyperonym[zeit]);

(lexem[dienstag] hyperonym[zeit]);

The attribute 'hyperonym[zeit]' (i.e. hypernym[time]) is a prerequisite of an adjunct serving as temporal adverb. The attribute hyperonym[sich_bewegen] (i.e. movement) must adhere to the verbal head (!), if an adjunct is to qualify as directional adverbial. ohne[hyperonym] (i.e. without hypernym) is an attribute of type <ne>. The value of this type of attribute can be any other attribute, which then must not occur in the filler. We use this as a default specification for local adverbials. Rather than assigning a hypernym 'ort' (place) to all nouns that exist in space we assume the role 'ort' if no hypernym is specified.

Examples of determiners as adjuncts:

We treat the determiners in German as adjuncts. This is most efficient, since in this way we avoid searching for a determiner if none is present (e.g. indefinite plural).

der Mann

sein Buch

keinen Stau

ein Kind

Resources:

```
(lexem[ein] kategorie[artikelwort]
        (adjunct[%individuativ]) );

(lexem[anaphor_sgm'] kategorie[artikelwort]
        (adjunct[%individuativ]) );

(lexem[kein] kategorie[artikelwort]
        (adjunct[%individuativ]) );

(lexem[definit']
        (adjunct[%individuativ]) );
```

The artificial lexeme '[anaphor_sgm']' represents 'sein, seinem, seiner' (his), the lexeme '[definit']' represents 'der, die, das' .

```
(template[%individuativ] kategorie[nomen] ohne[pronomen] n_position[3]
        (L ___[adjunct] rolle[A,determination] kategorie[artikelwort] kasus[C]
        flexion[C] genus[C]  numerus[C]  w_pronomen[C] determination[+,C,A]
        n_position[1] ));

<test topic="%individuativ">(der) /Mann/; (sein) /Buch/ </test>
```

'n_position' means position in a noun phrase. Like 's_position' this is an attribute of type <sc>. The word order of noun phrase components is fixed in German. The crucial attribute is 'determination[+]' which is passed upwards to the noun if a determiner fills the slot. A noun phrase can refer to things as subjects and objects only if the attribute 'determination[+]' is present. In the case of mass nouns like 'Geld' this attribute is inherent due to an attribute assignment by a synframe:

```
(lexem[geld] determination[+]);
```

Examples of propositional adverbs:

Der Mann verreist nicht.

Der Mann verreist wahrscheinlich nie.

*Der Mann verreist nie nicht.

Resources:

```
(lexem[nicht]  kategorie[partikel] negation[+]
        (adjunct[%satzadverb] negation[+] ) );

(lexem[nie] negation[+]
        (adjunct[%satzadverb] ) );

(lexem[beinahe]
        (adjunct[%satzadverb] ) );

(lexem[wahrscheinlich]
        (adjunct[%satzadverb] ) );

(template[%satzadverb] kategorie[verb] s_position[2]
        (R ___[adjunct,multiple] rolle[satzadverb,A] s_position[13]  negation[C]));

<test topic="%satzadverb "> Der Mann /verreist/ (wahrscheinlich) (nie). </test>
```

Comment:

'negation' is an exclusive feature (type <ef>). Such an attribute must occur only once in the unification of values. By propagating this attribute (with 'C') one can exclude double negation from grammatical sentences.

While a slot usually allows a single filler only, 'multiple' slots can be filled several times. As oposed to complements, several adjuncts with the same role can modify a verb.

Expected adjuncts

Examples:

Die Frau befindet sich in der Kirche.

* Die Frau befindet sich.

Phenomena:

- Some adverbials are obligatory with certain verbs to form a grammatical sentence.

Problem:

- Since they are obligatory, these elements should be included as complements in the valency of the head. This is at least the point of some scholars.
- On the other hand their form and function is identical with free adjuncts.

Solution:

✓ It is allowed to include adjunct templates in the synframe of the heading verb as so-called "expected_adjuncts". In this case they function like complements, i.e. they come with the head and look for a dependent. The same templates as those for "free" adjuncts may be used.

Parser output:

```
Die Frau befindet sich in der Kirche.

 (illokution:  aussage'
    (proposition:  befinden sich
       (subjekt:  frau
          (determination:  definit'))
       (ort:  in
          (komplement:  kirche
             (determination:  definit')))));


Die Frau befindet sich.

No complete parsing achieved
```

Synframe:

```
(lexem[befinden]
        (complement[+subjekt])
        (complement[+reflexiv_akkusativ])
        (expected_adjunct[%lokaladverb]) );
```

General synframes for adjuncts

Example:

der freundliche Mann

Der freundliche Mann, welcher dem Kind ein Buch gegeben hat, verreist.

Das Buch, welches der Mann der Frau gibt, ist spannend.

Phenomenon:

- Almost any noun may have an attributive adjective and can be augmented by a relative clause.

Problem:

- Encoding an adjunct template with every adjective or a relative clause template with every verb is inconvenient.

Solution:

✓ We deviate from the lexicalistic approach of DGs and allow lexeme variables in synframes.

Parser output:

```
Der freundliche Mann, welcher dem Kind ein Buch gegeben hat, verreist.

 (illokution:  aussage'
    (proposition:  verreisen
       (subjekt:  mann
          (attribut:  relativsatz'
             (proposition:  haben perfekt_hilfsverb
                (subjekt:  relativ_sgm')
                (praedikativ:  geben transferieren
                   (dativ_objekt:  kind
                      (determination:  definit'))
                   (trans_objekt:  buch
                      (determination:  ein)))))
          (determination:  definit')
          (attribut:  freundlich))));
```

Resources:

```
(lexem[#,A]  kategorie[adjektiv]
       (adjunct[%attr_adjektiv]) );

(lexem[#,A]  kategorie[verb]
       (adjunct[%relativsatz]) );
```

The variable '#' in these synframes covers any lexeme. 'A' indicates that the specified adjunct is inherited by all synframes called by the lexem. The part of speech attribute 'kategorie' (type <mc>) constrains the applicability.

```
(template[%attr_adjektiv] kategorie[nomen] n_position[3]
        (L ___[adjunct] rolle[A,attribut] kategorie[adjektiv] kasus[C] flexion[C]
       genus[C]  numerus[C]  n_position[2] ));

<test topic="%attrr_adjektiv"> der (freundliche) /Mann/ </test>

(template[%relativsatz] kategorie[nomen] genus[maskulin] numerus[singular]
zeichen_rechts[komma]
       (rolle[attribut] lexem[relativsatz]
               (R ___[adjunct] rolle[A,proposition] kategorie[verb] form[finit]
              relativ_pronomen[maskulin] angrenzend[+]
              zeichen_rechts[komma,punkt])));

<test topic="%relativsatz, maskulinum"> der /Mann/(, welcher verreist )</test>

(template[%relativsatz] kategorie[nomen] genus[neutrum] numerus[singular]
zeichen_rechts[komma]
        (rolle[attribut] lexem[relativsatz]
               (R ___[adjunct] rolle[A,proposition] kategorie[verb] form[finit]
              relativ_pronomen[neutrum] angrenzend[+]
              zeichen_rechts[komma,punkt])));

<test topic="%relativsatz, neutrum"> das/ Buch/(, welches der Mann der Frau gibt
)</test>

(template[%relativsatz] kategorie[nomen] genus[feminin] numerus[singular]
zeichen_rechts[komma]
        (rolle[attribut] lexem[relativsatz]
               (R ___[adjunct] rolle[A,proposition] kategorie[verb] form[finit]
              relativ_pronomen[feminin] angrenzend[+]
              zeichen_rechts[komma,punkt])));

<test topic="%relativsatz, femininum"> die /Frau/(, welche verreist )</test>

(template[%relativsatz] kategorie[nomen] numerus[plural] zeichen_rechts[komma]
        (rolle[attribut] lexem[relativsatz]
               (R ___[adjunct] rolle[A,proposition] kategorie[verb] form[finit]
              relativ_pronomen[plural] angrenzend[+]
              zeichen_rechts[komma,punkt])));

<test topic="%relativsatz, plural"> die /Männer/(, welche verreisen )</test>
```

The considerable number of templates is due to the fact that there must be a correspondence between the gender of the head noun, encoded in the attribute 'genus', and the gender of the relative pronoun, encoded in the attribute 'relativ_pronomen'. There is no mechanism to unify the values of different attributes directly.

Discontinuous adjuncts

Example:

Er hat dem Mann das Buch gegeben, welcher verreist.

Resources:

```
(template[%relativsatz] kategorie[nomen] genus[maskulin] numerus[singular]
        (rolle[attribut] lexem[relativsatz]
                (R ___[discontinuous] rolle[A,proposition] kategorie[verb] form[finit]
               relativ_pronomen[maskulin] )) )
(kategorie[verb] form[finit] lesart[perfekt_hilfsverb,kopula] zeichen_rechts[komma] );

<test topic="%relativsatz, diskontinuierlich">Er hat  dem /Mann/  das Buch gegeben(,
welcher verreist ).</test>
```

The relative clause 'welcher verreist' in the above example is separated from its head noun 'Buch' by the participle 'gegeben'. The templates for discontinuous adjuncts are the same as for discontinuous complements. There is a head and a slot for a discontinuous constituent and a mother term which must dominate the intermediate result when the slot is filled.

Parser output:

```
Er hat dem Mann das Buch gegeben, welcher verreist.

(illokution:  aussage'
    (proposition:  haben perfekt_hilfsverb
       (subjekt:  anaphor_sgm')
       (praedikativ:  geben transferieren
          (dativ_objekt:  mann
             (attribut:  relativsatz'
                (proposition:  verreisen
                   (subjekt:  relativ_sgm')))
             (determination:  definit'))
          (trans_objekt:  buch
             (determination:  definit')))));
```

Intermediate term between head and slot

The templates for relative clauses above ('%relativsatz') include an intermediate term between the head term and the slot term: '(rolle[attribut] lexem[relativsatz'])'. The reason is the following. Sometimes the role of a constituent depends on the perspective. From the perspective of the noun the relative clause denotes an attribute of the noun. From the perspective of the relative sentence it denotes a proposition. If the dominating verb of the relative clause would be subordinated directly to the noun then the top-down perspective could not be represented. That is why we introduced an artificial term into the template and hence into the dependency tree. In the case of subclauses with conjunction the conjunction can carry the role of the top-down perspective. Here no artificial term is needed.

Coordination

Examples of infix coordination:

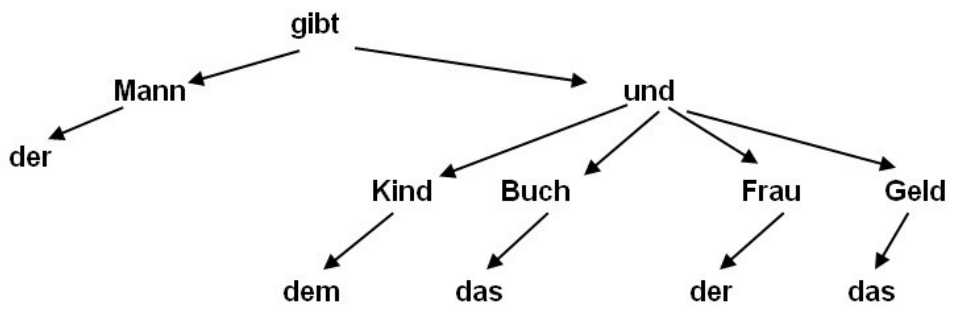
Er hat dem Kind das Buch oder das Geld gegeben.
Der Mann gibt dem Kind das Buch und der Frau das Geld.
weil der Mann dem Kind das Buch und der Frau das Geld am Sonntag gibt

Phenomena:

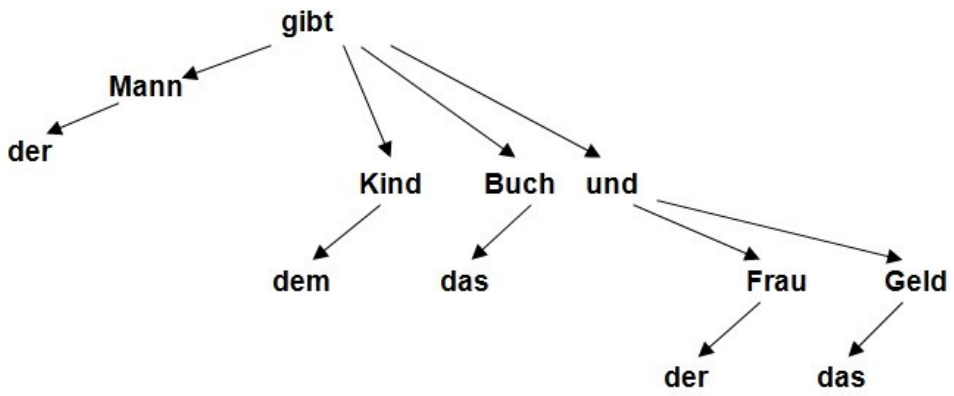
- Dependency is a hierarchy of heads and complements or adjuncts..
- Coordination introduces a linear grouping of elements orthogonal to the dependency relations.
- The grouped elements in one conjunct must have the same roles as the ones in the other conjunct(s).

Problem:

- We have a representation problem. How can we integrate coordination in a dependency tree?
- Treating the conjunction as a dependency head of the whole coordination **does not work:**



Solution:



The representation of coordination in the DUG is guided by the following principles:

- ✓ A coordination consists of (usually two) conjuncts. The conjunction dominates just one conjunct (rather than the whole coordination). In German there are two possibilities:

  a) The first conjunct has no conjunction. The second conjunct is headed by a conjunction.

  Er gibt dem Kind das Buch und der Frau das Geld am Sonntag.

  first conjunct *second conjunct*

  b) Both conjuncts are headed by a conjunction.

  Er gibt entweder dem Kind das Buch oder der Frau das Geld am Sonntag.

  first conjunct *second conjunct*

  a) is known as infix conjunction,.
  b) is known as prefix conjunction.

- ✓ If the conjunct has no conjunction (e.g. the first conjunct in a) then the complements of the conjunct are directly subordinated to the head as if there were no coordination.

- ✓ If a conjunction is present then the conjunction is subordinated in the tree at the place where the dependents would belong if they were not coordinated . Consequently, the coordinated dependents get one level lower. Role labels nevertheless ensure the right interpretation.

- ✓ The conjunction is "transparent" with respect to valency, roles, and agreement. One can so-to-say look through the conjunction upwards to see the "real" head with its valency and agreement requirements.

Parser output:

```
Er gibt der Frau das Geld und dem Kind das Buch am Sonntag.

(illokution:  aussage'
    (proposition:  geben transferieren
       (subjekt:  anaphor_sgm')
       (dativ_objekt:  frau
          (determination:  definit'))
       (trans_objekt:  geld
          (determination:  definit'))
       (koordination:  und
          (dativ_objekt:  kind
             (determination:  definit'))
          (trans_objekt:  buch
             (determination:  definit')))
       (zeit:  an
          (komplement:  sonntag
             (determination:  definit')))));
```

The dative object 'Frau' and the transitive object 'Geld' are the first conjunct. Since there is no conjunction they are direct complements of 'geben'. The conjunction 'und' is transparent. If one "looks through it" up the tree then one sees again 'geben' which is the "real" head of the dative object 'Kind' and the transitive object 'Buch' in the second conjunct.

Writing a grammar of coordination

Synframe for the conjunction und:

```
(lexem[und] kategorie[konjunktion]
        (complement[+subjekt_k,N])
        (complement[+dativ_k,N])
        (complement[+trans_k,N])
        (complement[+intrans_k,N])
        (conjunct[%satzglied_konjunkt]) );
```

The conjunction must have the same complements for subject, dative and transitive or intransitive objects as they occur with verbs. In a construction with the conjunction these complements have another syntax than in the sentence, though. Therefore we need a '_k' version of each verbal complement as soon as it dominated by the conjunction. Luckily we do dot need synframes for each variant of valency. One synframe is sufficient whose complements are all optional ('N') . Any choice is in order, the exact matching of original valency and coordinated elements is checked by the program anyway.

Complement templates for conjunctions:

```
(template[+subjekt_k] kategorie[konjunktion]
        (R ___[complement]  rolle[subjekt,A] kategorie[nomen] ohne[relativ_pronomen]
        kasus[nominativ] koord_subjekt[+,C,A]));

<test topic="+subjekt_k">/und/ (der Mann) </test>

(template[+dativ_k] kategorie[konjunktion]
        (R ___[complement]  rolle[dativ_objekt,A] kategorie[nomen] ohne[relativ_pronomen]
        kasus[dativ] ));

<test topic="+dativ_k">/und/ (dem Mann) </test>

(template[+trans_k] kategorie[konjunktion]
         (R ___[complement]  rolle[trans_objekt,A] kategorie[nomen] ohne[relativ_pronomen]
        kasus[akkusativ] ));

<test topic="+trans_k">/und/ (den Mann)  sieht) </test>

template[+intrans_k] kategorie[konjunktion]
        ( R ___[complement]  rolle[intrans_objekt,A] kategorie[nomen] ohne[relativ_pronomen]
        kasus[akkusativ] ));
<test topic="+intrans_k">/und/ (das Buch)   enthält </test>
```

If the conjunct is complete, the conjunction (and with it the whole conjunct) has to be attached to the head. This is guided by a 'conjunct' specification in the synframe. Technically conjunct attachment is completely the same as adjunct attachment. Like an adjunct, the conjunct is appending itself to a suitable head. The new slot type ___[conjunct] guarantees this effect.

Conjunct templates:

```
(template[%satzglied_konjunkt] kategorie[verb]
        (L ___[conjunct] rolle[koordination,A]  koord_subjekt[C] ));

<test topic="%satzglied_konjunkt"> dem Kind das Buch (und der Frau das Geld )/gibt/
</test>

(template[%satzglied_konjunkt] kategorie[verb]
        (R ___[conjunct] rolle[koordination,A]  koord_subjekt[C]));

<test topic="%satzglied_konjunkt">/gibt/  dem Kind das Buch (und der Frau das Geld )
</test>
```

Built-in functions:

Usually no linguistic properties are built-in in the PLAIN programs. Every phenomenon has to be described explicitly by the linguist in some DRL resources. In the case of coordination we had to make exceptions. Important principles of coordination are built-in. If a conjunct with a heading conjunction occurs then the program

- checks whether the dependents of the conjunction fit the valency of the "real" head;
- checks whether the dependent roles of the conjunction (i.e. attribute of type <rl>) have occurred in the other conjunct, where the latter is either directly under the real head or under another conjunction.

Infix coordination is a bit tricky due to the change between complement attachment and conjunct attachment either to the right or to the left of the "real" head. Arcs below the sentences in the following figures represent complements (the head looks for complements), arcs above represent conjunct attachment (the conjunction looks for a head). The numbers display the sequence of actions. The template writer has to ensure that the respective heads are indeed available on the right dependency level at the moment of attachment. There are diagnosis tools to check on this.

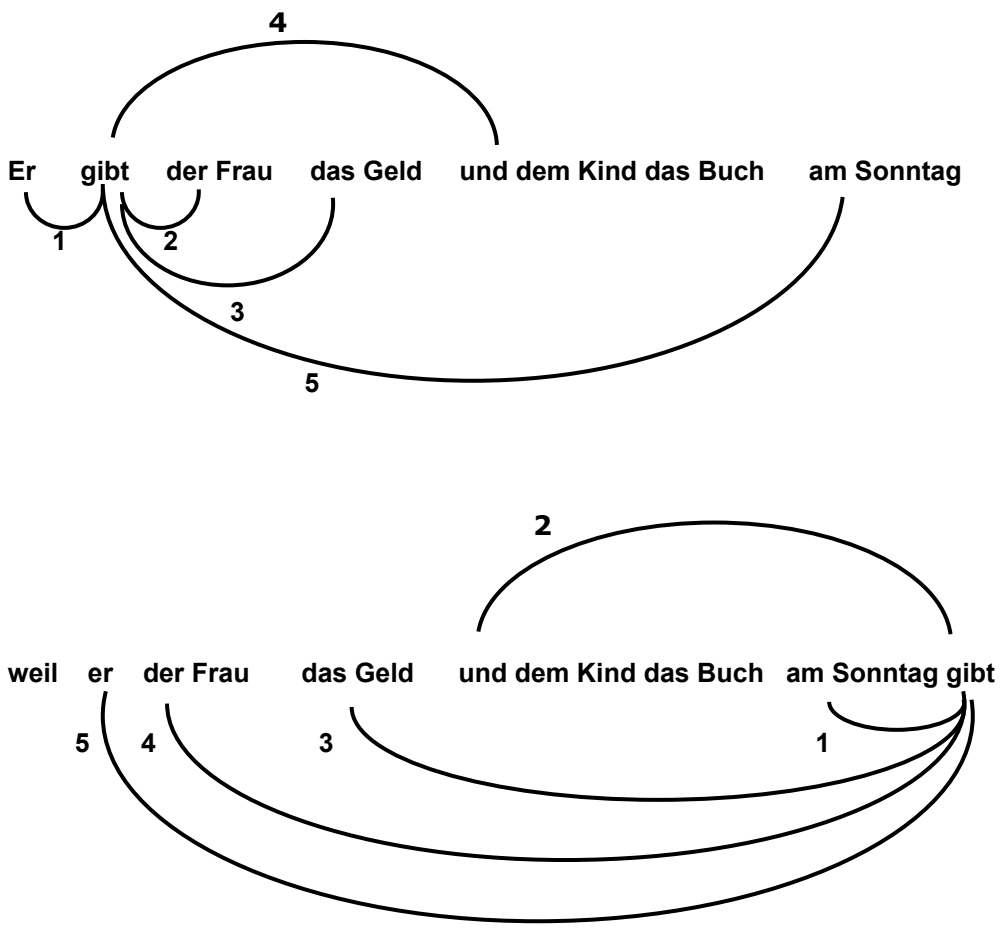


Figure 12: The parser steps with conjuncts to the right and to the left of the "real" head

Examples of coordinated verb complements with prejuncts and postjuncts:

Er gibt entweder dem Kind oder der Frau das Buch.

Er hat entweder der Frau oder dem Kind das Geld gegeben

Er hat der Frau entweder das Buch oder das Geld gegeben

Er hat entweder der Frau das Buch oder dem Kind das Geld gegeben.

weil er entweder der Frau oder dem Kind das Geld gegeben hat

weil er entweder der Frau das Buch oder dem Kind das Geld gegeben hat

weil er der Frau entweder das Buch oder das Geld gegeben hat

Parser output:

```
Er hat entweder der Frau das Buch oder dem Kind das Geld gegeben.

(illokution:  aussage'
    (proposition:  haben perfekt_hilfsverb
       (subjekt:  anaphor_sgm')
       (praedikativ:  geben transferieren
          (koordination:  entweder
             (dativ_objekt:  frau
                (determination:  definit'))
             (trans_objekt:  buch
                (determination:  definit')))
          (koordination:  oder
             (dativ_objekt:  kind
                (determination:  definit'))
             (trans_objekt:  geld
                (determination:  definit')))))));
```

Resources:

```
(lexem[entweder] kategorie[konjunktion] prejunkt[entweder] postjunkt[oder]
        (conjunct[%satzglied_prejunkt])
        (complement[+subjekt_k,N])
        (complement[+dativ_k,N])
        (complement[+trans_k,N])
        (complement[+intrans_k,N]));

(lexem[oder] kategorie[konjunktion] prejunkt[entweder] postjunkt[oder]
        (conjunct[%satzglied_konjunkt])
        (conjunct[%satzglied_postjunkt] )
        (complement[+subjekt_k,N]) (
        complement[+dativ_k,N])
        (complement[+trans_k,N])
        (complement[+intrans_k,N]) );

(template[%satzglied_prejunkt] kategorie[verb]
        (L ____[prejunct] prejunkt[C] postjunkt[C] rolle[koordination,A] numerus[C]));

<test topic="%satzglied_prejunkt"> weil er (entweder dem Kind)  oder der Frau das
Buch /gibt/ </test>

(template[%satzglied_prejunkt] kategorie[verb]
         (R ____[prejunct] prejunkt[C] postjunkt[C] rolle[koordination,A] numerus[C]));

<test topic="%satzglied_prejunkt">  Er /gibt/ (entweder dem Kind)  oder der Frau das
Buch. </test>
 (template[%satzglied_postjunkt] kategorie[verb]
        (L ____[postjunct] prejunkt[C] postjunkt[C] rolle[koordination,A] ));

<test topic="%satzglied_postjunkt">  weil er entweder dem Kind (oder der Frau)  das
Buch /gibt/ </test>
```

(template[%satzglied_postjunkt] kategorie[verb]
(R ___[postjunct] prejunkt[C] postjunkt[C] rolle[koordination,A]));

<test topic="%satzglied_postjunkt"> Er /gibt/ entweder dem Kind (oder der Frau) das Buch. </test>

Comment:

A "prejunct" is the first of two conjuncts with a conjunction, a "postjunct" is the second conjunct. Two new slot types, **'prejunct'** and **'postjunct'**, ensure the appropriate attachment to the "real" head of the conjuncts. Usually the conjunctions are lexically correlated. e.g. 'prejunkt[entweder] postjunkt[oder]' (either – or). The right lexemes are enforced by means of agreement of the attributes prejunkt[C] postjunkt[C]. These attributes are type <df>.

Examples of coordination with adjuncts:

Der Mann hat dem Kind das Buch am Montag oder am Dienstag gegeben.

Der Mann gibt dem Kind das Buch am Montag und das Geld am Dienstag.

Er geht entweder in die Kirche oder zu den Kindern.

Parser output:

```
Er geht entweder in die Kirche oder zu den Kindern.

 (illokution:  aussage'
    (proposition:  gehen
       (subjekt:  anaphor_sgm')
       (koordination:  entweder
          (richtung:  in
             (komplement:  kirche
                (determination:  definit'))))
       (koordination:  oder
          (richtung:  zu
             (komplement:  kind
                (determination:  definit')))))));
```

Resources:

Adjuncts attach themselves to heads. If adjuncts should not only augment verbal frames but also conjuncts then we need templates for adjuncts with a conjunction as head.

(template[%richtungsadverb] kategorie[konjunktion]
(R ___[adjunct] rolle[richtung,A]));

<test topic="%richtungsadverb"> /und/ (in die Kirche) </test>

```
(template[%lokaladverb] kategorie[konjunktion]
        (R  ___[adjunct] rolle[ort,A] ohne[hyperonym]));
```

<test topic="%lokaladverb"> /und/ (an der Kirche) </test>

```
(template[%temporaladverb] kategorie[konjunktion]
        (R  ___[adjunct] rolle[zeit,A] hyperonym[zeit]));
```

<test topic="%temporaladverb"> /und/ (am Sonntag) </test>

Examples of nested coordination

If the option "first_match" is on then only one of the many possible readings is displayed. Often there is an ambiguity of 'oder' as a prefix or as an infix coordinator.

Der Mann und das Kind oder der Mann und die Frau verreisen.

Er gibt entweder der Frau und dem Mann das Buch oder dem Kind das Geld.

Er gibt entweder der Frau oder dem Mann das Buch oder dem Kind das Geld und das Buch.

Er gibt entweder der Frau und dem Mann das Buch oder dem Kind das Geld.

Parser output:

```
Er gibt entweder der Frau und dem Mann das Buch oder dem Kind das Geld.

(illokution:  aussage'
    (proposition:  geben transferieren
       (subjekt:  anaphor_sgm')
       (koordination:  entweder
          (dativ_objekt:  frau
             (determination:  definit'))
          (koordination:  und
             (dativ_objekt:  mann
                (determination:  definit')))
          (trans_objekt:  buch
             (determination:  definit')))
       (koordination:  oder
          (dativ_objekt:  kind
             (determination:  definit'))
          (trans_objekt:  geld
             (determination:  definit')))));
```

Resources:

```
(template[%satzglied_konjunkt] kategorie[konjunktion]
        (R ___[conjunct] rolle[koordination,A] ));
```

<test topic="%satzglied_konjunkt"> /und/ dem Mann (oder dem Kind) </test>

```
(template[%satz_konjunkt] kategorie[konjunktion]
        (R ___[conjunct] rolle[koordination,A] ));

<test topic="%satz_konjunkt"> /und/  dem Mann genommen (oder der Frau gegeben )
</test>

(template[%satz_prejunkt] kategorie[konjunktion]
        (R ___[prejunct] rolle[koordination,A] ));

<test topic="%satz_prejunkt"> /und/ (entweder dem Mann genommen)  oder der Frau
gegeben </test>

(template[%satz_postjunkt] kategorie[konjunktion]
        (R ___[postjunct] rolle[koordination,A] ));

<test topic="%satz_postjunkt"> /und/  entweder dem Mann genommen (oder der Frau
gegeben) </test>
```

Examples of discontinuous constituents coordinated:

Das Geld und das Buch hat er genommen.

Entweder das Geld oder das Buch hat er genommen.

Entweder der Frau das Buch oder dem Kind das Geld hat er gegeben.

Parser output:

```
Das Geld und das Buch hat er genommen.

 (illokution:  aussage'
    (proposition:  haben perfekt_hilfsverb
       (subjekt:  anaphor_sgm')
       (praedikativ:  nehmen
          (trans_objekt:  geld
             (determination:  definit'))
          (koordination:  und
             (trans_objekt:  buch
                (determination:  definit'))))));
```

No additional resources necessary.

Coordination of the predicative complements of auxiliaries:

This is coordination together with nucleus and raising.

Hat er der Frau ein Buch gegeben und dem Mann das Geld genommen?

Parser output:

```
Hat er der Frau ein Buch gegeben und dem Mann das Geld genommen?

(illokution:  frage'
    (proposition:  haben perfekt_hilfsverb
       (subjekt:  anaphor_sgm')
       (praedikativ:  geben transferieren
          (dativ_objekt:  frau
             (determination:  definit'))
          (trans_objekt:  buch
             (determination:  ein)))
       (koordination:  und
          (praedikativ:  nehmen
             (dativ_objekt:  mann
                (determination:  definit'))
             (trans_objekt:  geld
                (determination:  definit'))))));
```

Resources:

```
(lexem[und] kategorie[konjunktion]
        (conjunct[%praedikativ_konjunkt])
        (complement[+praedikativ_k]));

(lexem[entweder] kategorie[konjunktion] prejunkt[entweder] postjunkt[oder]
         (conjunct[%praedikativ_prejunkt])
        (complement[+praedikativ_k]));

(lexem[oder] kategorie[konjunktion] prejunkt[entweder] postjunkt[oder]
        (conjunct[%praedikativ_postjunkt])
         (complement[+praedikativ_k]));

(template[+praedikativ_k] kategorie[konjunktion]
         ( R ___[complement]  rolle[praedikativ,A] kategorie[verb] form[partizip]
        stellungstyp[C] ));

<test topic="+praedikativ_k"> /und/   dem Mann das Geld (genommen)  </test>

(template[%praedikativ_konjunkt] kategorie[verb] form[finit]
lesart[perfekt_hilfsverb,A]
        (R ___[conjunct] rolle[koordination,A]  ));

<test topic="%praedikativ_konjunkt">  Er /hat/  der Frau ein Buch gegeben (und dem
Mann das Geld genommen ) </test>

(template[%praedikativ_konjunkt] kategorie[verb] form[finit]
lesart[perfekt_hilfsverb,A]
        (L ___[conjunct] rolle[koordination,A]  ));

<test topic="%$praedikativ_konjunkt">  weil  er der Frau ein Buch gegeben( und dem
Mann das Geld genommen ) /hat/ </test>
```

(template[%praedikativ_prejunkt] kategorie[verb] form[finit]
lesart[perfekt_hilfsverb,A]
(R ___[prejunct] rolle[koordination,A]));

<test topic="%prraedikativ_prejunkt"> Er /hat/ (entweder der Frau ein Buch gegeben) oder dem Mann das Geld genommen </test>

(template[%praedikativ_prejunkt] kategorie[verb] form[finit]
lesart[perfekt_hilfsverb,A]
(L ___[prejunct] rolle[koordination,A]));

<test topic="%praedikativ_prejunkt"> weil er (entweder der Frau ein Buch gegeben) oder dem Mann das Geld genommen /hat/ </test>

(template[%praedikativ_postjunkt] kategorie[verb] form[finit]
lesart[perfekt_hilfsverb,A]
(R ___[postjunct] rolle[koordination,A]));

<test topic="%praedikativ_postjunkt"> Er/ hat/ entweder der Frau ein Buch gegeben (oder dem Mann das Geld genommen) </test>

(template[%praedikativ_postjunkt] kategorie[verb] form[partizip]
lesart[perfekt_hilfsverb,A]
(L ___[postjunct] rolle[koordination,A]));

<test topic="+praedikativ_postjunkt"> weil er entweder der Frau ein Buch gegeben (oder dem Mann das Geld genommen) /hat/ </test>

Coordination of predicates:

Er gibt das Geld und er nimmt das Buch.

Entweder er gibt das Geld oder er nimmt das Buch.

Parser output:

```
Er gibt das Geld und er nimmt das Buch.

(illokution:  aussage'
    (proposition:  geben transferieren
       (subjekt:  anaphor_sgm')
       (trans_objekt:  geld
          (determination:  definit')))
    (koordination:  und
       (proposition:  nehmen
          (subjekt:  anaphor_sgm')
          (trans_objekt:  buch
             (determination:  definit')))));
```

Resources:

```
(lexem[und] kategorie[konjunktion]
        (conjunct[%satz_konjunkt])
         (complement[+aussage_k]));

(lexem[entweder] kategorie[konjunktion] prejunkt[entweder] postjunkt[oder]
        (conjunct[%satz_prejunkt])
        (complement[+aussage_k]));

(lexem[oder] kategorie[konjunktion] prejunkt[entweder] postjunkt[oder]
        (conjunct[%satz_postjunkt])
        (conjunct[%satz_konjunkt])
        (complement[+aussage_k]));


(template[+aussage_k] kategorie[konjunktion]
         ( R ___[complement]  rolle[proposition,A] kategorie[verb] form[finit]
        stellungstyp[C] ));

<test topic="+aussage_k"> /und/ (er nimmt das Buch)  </test>

(template[%satz_konjunkt] kategorie[satz]
         (L ___[conjunct] rolle[koordination,A] ));

<test topic="%satz_konjunkt">  Er gibt das Geld (und er nimmt das Buch/)/./</test>

(template[%satz_prejunkt] kategorie[satz]
         (L ___[prejunct] rolle[koordination,A] stellungstyp[verb_zweit] ));

<test topic="%satz_prejunkt"> (entweder er gibt das Geld)  oder er nimmt das Buch
/. /</test>

(template[%satz_postjunkt] kategorie[satz]
        (L ___[postjunct] rolle[koordination,A] stellungstyp[verb_zweit] ));

<test topic="%satz_postjunkt">  Entweder er gibt das Geld) (oder er nimmt das
Buch)/./</test>
```

Comment:

In our syntax fragment, each coordination requires a head that governs the conjuncts. In the case of sentence coordination, the head of the sentences is "sentencehood" expressed by full stop, question mark, etc. This looks like an artificial solution, but it is not unreasonable. (In German punctuation marks are called "Satzzeichen", i.e. sentence marks.) As a consequence, coordinated sentences can not be parsed as long as the punctuation mark is still missing.

Ellipsis

Examples:

Er gibt das Geld und nimmt das Buch.
Er gibt und nimmt Geld.
Entweder der Mann gibt oder das Kind nimmt das Geld.
weil er dem Kind Geld gibt und nimmt

Phenomena:

- Complements and adjuncts can be omitted at their proper place in a coordination.
- They are mentally copied from one conjunct to the other (forward and backward).

Problem:

- If we want to save the principle of matching roles in coordination we have to reconstruct elliptic complements. The mechanism is built in.

Parser output:

```
Er gibt und nimmt Geld.

(illokution:  aussage'
    (proposition:  geben transferieren
       (trans_objekt: ellipse)
       (subjekt:  anaphor_sgm'))
    (koordination:  und
       (proposition:  nehmen
          (subjekt: ellipse)
         trans_objekt:  geld))));

weil er dem Kind Geld  gibt und nimmt

( weil
    (proposition:  geben transferieren
       (subjekt:  anaphor_sgm')
       (dativ_objekt:  kind
          (determination:  definit'))
       (trans_objekt:  geld))
    (koordination:  und
       (proposition:  nehmen
          (subjekt: ellipse)
          (dativ_objekt: ellipse)
         trans_objekt: ellipse))));
```

Built-in functions:

There are the following states and reactions of the program:

Case 1: A conjunct is subordinated to a head but it contains a term that has open slots, e.g. 'und nimmt', 'nimmt' is still lacking the complements '+subjekt' and '+trans'.

- The roles of the open slots are stored in a list of elliptic complements.

Case 2: Another conjunct is subordinated to a head.

- If the conjunct comprises a list of elliptic complements then the first conjunct is inspected for realized counterparts. The same is done vice versa. A copy of the realized complements is inserted and marked as elliptic in the conjunct where they were missing.

A problem with prefix coordination:

Er hat entweder der Frau das Geld gegeben oder dem Kind das Buch genommen.
Er hat der Frau entweder das Geld gegeben oder das Buch genommen.
Er hat der Frau das Geld entweder gegeben oder genommen.

In the case of prefix coordination there may some dependents of the coordinated items be left out of the scope of the coordination. ' Er hat der Frau entweder das Geld gegeben oder das Buch genommen'. 'der Frau' is outside of the scope, however its head 'gegeben' is within the scope of coordination. Our representation requires that 'Frau' is subordinated to 'gegeben' and (elliptically) to 'genommen'. What we do is treat such complements as discontinuous while the conjunction is the mother term.

Templates

```
(template[+subjekt] kategorie[verb] form[finit]
        (L ___[discontinuous]  rolle[A,subjekt] kategorie[nomen] kasus[nominativ]
        numerus[C] person[C]))
(kategorie[konjunktion] lexem[entweder]);

<test topic="+subjekjt, außerhalb Koordination">weil  (er) entweder die Hefte /gibt/
</test>
```

```
(template[+dativ] kategorie[verb] form[finit,partizip]
        (L ___[discontinuous]  rolle[A,dativ_objekt] kategorie[nomen] kasus[dativ]))
(kategorie[konjunktion] lexem[entweder]);

<test topic="+dativ, außerhalb Koordination "> (der Frau) entweder die Hefte
/gegeben/</test>

(template[+trans] kategorie[verb] form[finit, partizip]
        (L ___[discontinuous]  rolle[A,trans_objekt] kategorie[nomen]
        kasus[akkusativ]))
(kategorie[konjunktion] lexem[entweder]);

<test topic="+trans, außerhalb Koordination ">der Frau  (das Buch)  entweder
/gegeben/</test>
```

Parser output:

```
Er hat der Frau das Geld entweder gegeben oder genommen.

(illokution:  aussage'
    (proposition:  haben perfekt_hilfsverb
       (subjekt:  anaphor_sgm')
       (koordination:  entweder
          (praedikativ:  geben transferieren
             (dativ_objekt:  frau
                (determination:  definit'))
             (trans_objekt:  geld
                (determination:  definit'))))
       (koordination:  oder
          (praedikativ:  nehmen
             (dativ_objekt: ellipse)
             (trans_objekt: ellipse)))));
```

Note: We do not substitute the elliptic term by the instance, e.g. `'(dativ_objekt: ellipse'` by `'(dativ_objekt:  frau)'` (which would have been easy). The reason is the transparency of the analysis. It should be clear whether a particular term is present on the surface or not.

PLAIN has another component, called "Transducer". We leave it to this program to substitute elliptic terms by instances or even derive several complete sentences from a coordination with ellipsis. The parser output above is sufficient for this purpose. The transducer can easily change

```
Er hat der Frau das Geld entweder gegeben oder genommen.
```

into

```
Entweder er hat der Frau das Geld gegeben
oder er hat der Frau das Geld genommen.
```

Orphans

Examples:

Der freundliche gibt Geld.

Der freundliche Mann gibt und der glückliche nimmt Geld.

Phenomenon:

- Sometimes it is the head in a dependency relation that is missing due to ellipsis. We call the left-alone dependents "orphans".

Problem:

- The heads of orphans must be reconstructed in order to yield a well-formed dependency tree.

Solution:

- ✓ An general adjunct template for possible orphans.
- ✓ A new type of slot: '***orphan'.***

Parser output:

```
Der freundliche gibt Geld.

(illokution:  aussage'
    (proposition:  geben transferieren
       (subjekt: ellipse
          (determination:  definit')
          (attribut:  freundlich))
       (trans_objekt:  geld)));
```

Synframe for adjective ophans:

```
(lexem[#,A]  kategorie[adjektiv]
        (adjunct[%attr_adjektiv]) );
```

This is a general synframe. Due to the variable lexeme, it applies to any adjective.

Corresponding Template:

```
(template[%attr_adjektiv] kategorie[nomen] n_position[2,3]
        (___[orphan] rolle[A,attribut] kategorie[adjektiv] kasus[C] flexion[C] genus[C]  numerus[C] ));

<test topic="%attr_adjektiv, orphan ">der (freundliche) //</test>
```

The slot type 'orphan' results in a special processing. The parser does not try to attach the adjunct to an existing head. Instead a head is created according to the template's head (kategorie[nomen] n_position[2,3]) and the adjunct is subordinated to this artificial term. The result is stored in the chart. There the entry with the artificial noun can combine with other elements, for example with the determiner '*der*' flexion[stark-schwach] so that we get '*der freundliche*' as a noun phrase. If we had '*ein*' und '*freundlicher*' the attribute flexion[schwach-stark] would make for agreement of '*ein freundlicher*' as a noun phrase although no noun is present.

(Do orphans exist in English? 'Ein freundlicher' is translated into 'a friendly one', the ellipsis is realized by a pronoun.)

Tools

The PLAIN IDE is made for facilitating the linguist's work. Many functions serve this purpose. Please, read the following sections in sync with *The PLAIN User's Guide.* There you will find more instructions.

Output

Parsing result can be displayed in various formats. The default format is *Show Short Result*:

```
Input:
Die Kinder sind verreist.

Parsing result - first match only (offset 1 to 25, chart element 11):

 (illokution:  aussage'
    (proposition:  sein perfekt_hilfsverb
       (subjekt:  kind
          (determination:  definit'))
       (praedikativ:  verreisen)));

Statistics: 5 words, 11 bulletins, 22 percent overgenerated
```

Figure 14: Short Result

The purpose of this format is to gain a quick overview about the structure of the analysis. Just role values (attribute type <rl>), a colon, lexeme values (attribute type <lx>), reading values (attribute type <rd>), quoted expressions (attribute type <qu>) and the trace of elliptic terms (attribute type <tc>) are displayed.

In the above example, the 'first match only' option is in effect. It must be turned off if all results of homonymous input should be detected. 'offset' is calculated in characters from the beginning of the input. The indicated chart element is the one that contains the complete result. 'Statistics' include the number of words and the number of "bulletins". A bulletin is a package of information stored with a chart element. Overgeneration is an important factor of efficiency. The displayed measure starts from the minimum of nodes necessary to construct a complete dependency tree. This is twice the number of words minus one. So, you can expect at least 9 chart elements if there are 5 words in the input. Anything exceeding this number is overgeneration. It is measured in percent of the minimum. Overgeneration can be very harmful because it may lead to "combinatorial explosion". The degree of overgeneration, i.e. the number of useless intermediate results, depends very much on the sophisticated use of constraints in form of attributes in the lexicon, synframes and templates.

The output format *Long Result* looks as follows:

```
Input:
Die Kinder sind verreist.

Parsing result - first match only (offset 1 to 25, chart element 11):

 (rolle[illokution] lexem[aussage'] kategorie[satz] aeusserung[+]
 wortlaut[.] schreibung[klein]
    (L rolle[proposition] lexem[sein] lesart[perfekt_hilfsverb]
    kategorie[verb] form[finit] numerus[plural] perfekt[sein]
    person[dritte] stellungstyp[verb_zweit] tempus[praesens]
    hilfsverb[perfekt] wortlaut[sind ] schreibung[klein] randstellung[+]
       (L rolle[subjekt] lexem[kind] kategorie[nomen] determination[+]
       flexion[stark-schwach] kasus[nominativ] numerus[plural,C]
       person[dritte,C] pronomen[nein] wortlaut[Kinder ] schreibung[gross]
       s_position[1]
          (L rolle[determination] lexem[definit'] kategorie[artikelwort]
          determination[+,C] flexion[stark-schwach,C]
          kasus[nominativ,akkusativ,C] numerus[plural,C] wortlaut[Die ]
          schreibung[gross] n_position[1]))
       (R rolle[praedikativ] lexem[verreisen] kategorie[verb]
       form[partizip] perfekt[sein,C] stellungstyp[verb_front,
       verb_zweit,verb_end] wortlaut[verreist] schreibung[klein]
       s_position[15])));

Statistics: 5 words, 11 bulletins, 22 percent overgenerated
```

Figure 15: Long Result

The long result adheres to the DRL syntax. All attributes, values und flags are shown. However, some constraints are rather technical and may not be needed anymore after a parsing result is found. One can tune the Long Result output to the attributes which are interesting and hide the rest. This is done with the *Attribute Filtering Command:*

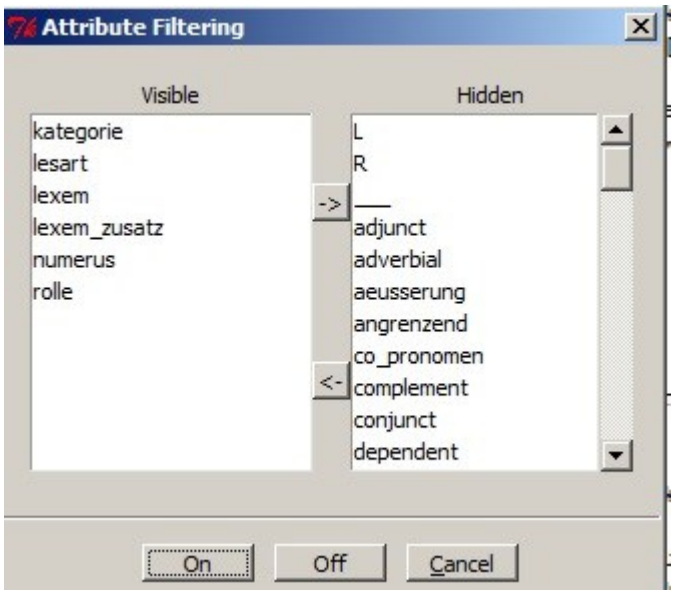


Figure 16: Attribute Filtering

Only those attributes are displayed that are declared as visible. The parsing result now looks as follows:

```
Input:
Die Kinder sind verreist.

Parsing result - first match only (offset 1 to 25, chart element 11):

 (rolle[illokution] lexem[aussage'] kategorie[satz]
    (rolle[proposition] lexem[sein] lesart[perfekt_hilfsverb]
    kategorie[verb] numerus[plural]
       (rolle[subjekt] lexem[kind] kategorie[nomen] numerus[plural,C]
          (rolle[determination] lexem[definit'] kategorie[artikelwort]
          numerus[plural,C]))
       (rolle[praedikativ] lexem[verreisen] kategorie[verb])));
```

Figure 17: Long Result with attribute filtering

The following format is called *Full Result*:

```
Input:
Die Kinder sind verreist.

Parsing result - first match only (offset 1 to 25, chart element 11):

25   '.'
 (rolle[illokution] lexem[aussage'] kategorie[satz] aeusserung[+]
 wortlaut[.] schreibung[klein]
12   'sind '
 (L rolle[proposition] lexem[sein] lesart[perfekt_hilfsverb]
 kategorie[verb] form[finit] numerus[plural] perfekt[sein] person[dritte]
 stellungstyp[verb_zweit] tempus[praesens] hilfsverb[perfekt]
 wortlaut[sind ] schreibung[klein] randstellung[+]
5   'Kinder '
 (L rolle[subjekt] lexem[kind] kategorie[nomen] determination[+]
 flexion[stark-schwach] kasus[nominativ] numerus[plural,C]
 person[dritte,C]  pronomen[nein] wortlaut[Kinder ] schreibung[gross]
 s_position[1]
1   'Die '
 (L rolle[determination] lexem[definit'] kategorie[artikelwort]
 determination[+,C] flexion[stark-schwach,C] kasus[nominativ,akkusativ,C]
 numerus[plural,C] wortlaut[Die ] schreibung[gross] n_position[1]))
17   'verreist'
 (R rolle[praedikativ] lexem[verreisen] kategorie[verb] form[partizip]
 perfekt[sein,C] stellungstyp[verb_front,verb_zweit,verb_end]
 wortlaut[verreist] schreibung[klein] s_position[15])));
```

Figure 18: Full Result

The offset of each word is recorded together with its character string. There is no indenting. The latter might be confusing if the structure is complex and large.

Eventually, there is the output in DRL. This format is machine readable. Files can be passed on to other components of PLAIN in this format:

```
<instance><drl>
 (rolle[illokution] lexem[aussage'] kategorie[satz] (rolle[proposition]
lexem[sein] lesart[perfekt_hilfsverb] kategorie[verb] numerus[plural]
(rolle[subjekt] lexem[kind] kategorie[nomen] numerus[plural,C]
(rolle[determination] lexem[definit'] kategorie[artikelwort]
numerus[plural,C])) (rolle[praedikativ] lexem[verreisen]
kategorie[verb])));
</drl></instance>
```

Figure 19: DRL machine readable format

Tests

If a large segment of a language is to be implemented, it is important to organize tests in a systematic way. Improvements of the lexicon, synframes and templates might have side effects. All of a sudden, proven examples don't work any more. That is why one should have test files which are parsed anew as soon as something has been changed. The output must be compared with former output. Many editors have facilities for this task.

It is desirable to relate the contents of input and output files. The following xml tag can be placed anywhere in the test input:

<test topic="x"/>

"x" may be substituted by any comment. The tag can help to arrange the test material, similar to headings. The standard test file of the german-demo project starts like this:

```
<test topic="Simple complements, agreement, sentences"/>
Ich verreise.
Du verreist.
Er verreist.
Verreist ihr?
Wer verreist?
<test topic="Elliptic complement, no subject">
Verreise!
<test topic="Syntagmatic frames, lexical disambiguation"/>
Der Mann gibt dem Kind das Buch.
```

Figure 20: Parser input with test tags

The output contains the same tags:

```
<test topic="Simple complements, agreement, sentences"/>

Input:
Ich verreise.

Parsing result - first match only (offset 1 to 13, chart element 7):

 (rolle[illokution] lexem[aussage'] kategorie[satz]
    (rolle[proposition] lexem[verreisen] kategorie[verb] numerus[singular]
       (rolle[subjekt] lexem[sprecher_sg'] kategorie[nomen]
numerus[singular,C])));

Statistics: 3 words, 7 bulletins, 40 percent overgenerated
```

```
<test topic="Simple complements, agreement, sentences"/>

Input:
Du verreist.

Parsing result - first match only (offset 1 to 12, chart element 7):

 (rolle[illokution] lexem[aussage'] kategorie[satz]
    (rolle[proposition] lexem[verreisen] kategorie[verb] numerus[singular]
       (rolle[subjekt] lexem[adressat_sg'] kategorie[nomen]
numerus[singular,C])));

Statistics: 3 words, 7 bulletins, 40 percent overgenerated

<test topic="Elliptic complement, no subject">

Input:
Verreise!

Parsing result - first match only (offset 1 to 9, chart element 4):

 (rolle[illokution] lexem[ausruf'] kategorie[satz]
    (rolle[proposition] lexem[verreisen] kategorie[verb]
numerus[singular]));

Statistics: 2 words, 4 bulletins, 33 percent overgenerated
```

Figure 21: Parser output with test tags

Usually templates need a lot of debugging. It is advisable to append examples to each template one draws up. This is the easiest way to remember what the template is about. The following tag should be used for this purpose:

```
<test topic="x">y</test>
```

x and y is any text. x is supposed to indicate the grammatical phenomenon in question, and y is an example. The PLAIN IDE provides the converter *Test Example to Parser Input.* This program creates a test file with <test topic="x"/> as tags and 'y' as example.

For example, the following templates have been drawn up for dative and transitive object in front position:

```
(template[+dativ] kategorie[verb] form[finit] s_position[2]
        (L ___[complement]  rolle[A,dativ_objekt] kategorie[nomen] kasus[dativ]
       determination[+] fokus[+,A] stellungstyp[verb_zweit,C,A] s_position[1]
       w_pronomen[C]));

<test topic="+dativ, im Vorfeld">(Dem Kind) /gibt/  der Mann das Buch </test>
<test topic="+dativ, im Vorfeld">(Wem) /gibt/  der Mann das Buch? </test>
```

```
(template[+trans] kategorie[verb] form[finit] s_position[2]
        (L ___[complement]  rolle[A,trans_objekt] kategorie[nomen] kasus[akkusativ]
       determination[+] fokus[+,A] stellungstyp[verb_zweit,C,A]  s_position[1]
       w_pronomen[C]));

<test topic="+trans, im Vorfeld">(Das Buch) /gibt/  er den Kindern.</test>
<test topic="+trans, im Vorfeld">(Was) /gibt/  er den Kindern ?</test>
```

Figure 22: Tests appended to templates

The converter creates the following test file from the examples in the template file:

```
<test topic="+dativ, im Vorfeld">
Dem Kind gibt  der Mann das Buch
</test>
<test topic="+dativ, im Vorfeld">
Wem gibt  der Mann das Buch?
</test>
<test topic="+trans, im Vorfeld">
Das Buch gibt  er den Kindern.
</test>
<test topic="+trans, im Vorfeld">
Was gibt  er den Kindern ?
</test>
```

Figure 23: Test file derived from examples appended to templates

The file in Figure 23 is fed to the parser. The output still preserves the topic tags from the templates. The linguist can check whether the template worked to her satisfaction.

```
<test topic="+dativ, im Vorfeld">

Input:
Dem Kind gibt  der Mann das Buch

Parsing result - first match only (offset 1 to 33, chart element 14):

 ( geben transferieren
    (dativ_objekt:  kind
       (determination:  definit'))
    (subjekt:  mann
       (determination:  definit'))
    (trans_objekt:  buch
       (determination:  definit')));

Statistics: 7 words, 14 bulletins, 7 percent overgenerated
```

```
<test topic="+dativ, im Vorfeld">

Input:
Wem gibt  der Mann das Buch?

Parsing result - first match only (offset 1 to 28, chart element 13):

 (illokution:  frage'
    (proposition:  geben transferieren
       (dativ_objekt:  w_person')
       (subjekt:  mann
          (determination:  definit'))
       (trans_objekt:  buch
          (determination:  definit'))));

Statistics: 8 words, 14 bulletins - no overgeneration
```

Figure 24: Parser output for tests appended to templates

Show chart elements

The principal structure of the chart has been illustrated in Figure 10 above. In the course of debugging, it is often necessary to check how the elements combine with each other (or do not). Taking a look in the chart is revealing. Let us return to the example of Figure 14.

```
Chart:

  1:   1-   4   (Die )
                left:  0  right:  0
  2:   5-  11   (Kinder )
                left:  0  right:  0
  3:   1-  11   (Kinder (Die ))
                left:  1  right:  2   L:  %individuativ
  4:  12-  16   (sind )
                left:  0  right:  0
  5:   1-  16   (sind (Kinder (Die )))
                left:  3  right:  4   L:  +subjekt
  6:   1-  16   (sind (Kinder (Die )))
                left:  3  right:  4   L:  +subjekt
  7:  17-  24   (verreist)
                left:  0  right:  0
  8:  17-  24   (verreist)
                left:  0  right:  0
  9:  25-  25   (.)
                left:  0  right:  0
 10:   1-  24   (sind (verreist)(Kinder (Die )))
                left:  6  right:  8   R:  +praed_perfekt
 11:   1-  25   (.(sind (verreist)(Kinder (Die ))))
                left: 10  right:  9   L:  +aussage
```

Figure 25: Survey of the chart

The command *Show All Chart Elements* displays all chart elements in the sequence of their emergence. (There is also an option to show only those chart elements which have been involved to build a particular element, e.g. the final result. This is so-to-say the chart without overgeneration.) Figure 25 shows the numbers of the chart elements, the associated strings in terms of character offsets, the resulting dependency structure in form of a bracketed expression with the words as terms. The numbers after 'left' and 'right' are the numbers of chart elements that provide the segments from which the new construction is built. The name of the template that has been used is recorded. If it is preceded by 'L:' then the filler is on the left, if it is preceded by 'R:' then the filler is on the right of the head.

Show sorted chart elements

The command *Show Sorted Chart Elements* triggers the following display:

```
Chart sorted by length of segments:

 11:   1-  25   (.(sind (verreist)(Kinder (Die ))))
 10:   1-  24   (sind (verreist)(Kinder (Die )))
  6:   1-  16   (sind (Kinder (Die )))
  5:   1-  16   (sind (Kinder (Die )))
  3:   1-  11   (Kinder (Die ))
  1:   1-   4   (Die )
  2:   5-  11   (Kinder )
  4:  12-  16   (sind )
  8:  17-  24   (verreist)
  7:  17-  24   (verreist)
  9:  25-  25   (.)
```

Figure 26: Chart elements sorted according to their length

This is mainly a debugging tool. If no complete analysis is achieved, one can at least see how far the parser got. It is kind of a documentation of partial parsing.

Show bulletin

A bulletin is an object that collects all information about a chart element. The command *Show Bulletin* displays this information. If we choose chart element 6 of Figure 26 we get:

```
Bulletin:
001 Chart   6:   1-  16   (sind (Kinder (Die )))
002 Constituents left:  3  right:  4  template: +subjekt

003 Heads:
 12   'sind '
    (lexem[sein] kategorie[verb] form[finit] numerus[plural]
     person[dritte] stellungstyp[verb_zweit] tempus[praesens]
     wortlaut[sind ] schreibung[klein] s_position[1,2]);

004 Dependents:
5   'Kinder '
   (L rolle[subjekt] lexem[kind] kategorie[nomen] determination[+]
    flexion[stark-schwach] kasus[nominativ] numerus[plural,C]
    person[dritte,C] pronomen[nein] wortlaut[Kinder ] schreibung[gross]
    s_position[1]
1      'Die '
       (L rolle[determination] lexem[definit'] kategorie[artikelwort]
       determination[+,C] flexion[stark-schwach,C]
       kasus[nominativ,akkusativ,C] numerus[plural,C] wortlaut[Die ]
       schreibung[gross] n_position[1]));

005 Frames:
   ( 1) f_able=0  lesart[kopula] kategorie[verb]
     stellungstyp[verb_front,verb_zweit,verb_end]
     raising_complement[+subjekt]
   ( 2) f_able=0  lesart[perfekt_hilfsverb] perfekt[sein] kategorie[verb]
     stellungstyp[verb_front,verb_zweit,verb_end]
     raising_complement[+subjekt]

006 Complements:
     1    0  +praed_adjektiv rolle[praedikativ]
     1    0  +praed_adjektiv rolle[praedikativ]
     0    1  +praed_perfekt rolle[praedikativ]
     0    1  +praed_perfekt rolle[praedikativ]

007 Adjuncts and conjuncts:
   101  101  %relativsatz rolle[attribut]
   101  101  %relativsatz rolle[attribut]
   101  101  %relativsatz rolle[attribut]
   101  101  %relativsatz rolle[attribut]
   101  101  %relativsatz rolle[attribut]

008 Parameters:
   seg_type:    1 origin:      3 head_rear:   12 head_front:  16
   l_blocked:   0 r_blocked:   0 first_morph: 1 last_morph:  1
   unknown:     0 utterance:   0 conjunct:    0 prepost:     0
   discont:     0 quote:       0 orphan:      0
   f_able:      0 fh_able:     1 hf_able:     1

011 Dependent roles:   1 subjekt
```

Figure 27: Contents of a bulletin

Bulletins are objects of the PLAIN implementation. However, if the parser fails, this can be the fault of the linguistic resources as well as a bug in the software. That is why the grammar writer has to get involved with bulletins to the extent that he is able to debug linguistic errors. Figure 27 is to be understood as follows:

`001 Chart:`	The number of the chart element, the edges of the associated input segment and the dependency structure of the words are displayed.
`002 Constituents:`	The left and right constituents are indicated in terms of the numbers of the chart elements that represent them. The template is mentioned that caused the subordination of the dependent to the head.
`003 Heads:`	Offset, string and DRL of the head in the construction is shown, in this case the attributes of the word 'sind'. If a word is a homonym then more than one head can be listed.
`004 Dependents:`	Offset, string and DRL of the dependents in the construction are shown, actually the attributes of the words 'Kinder' and 'Die'. 'Die' has been subordinated to 'Kinder' in another bulletin.
`005 Frames:`	A frame is understood here as the syntagmatic constellation of elements introduced by a synframe. There are two alternative frames: (1) the word 'sind' can function as a copula, e.g. 'die Kinder sind – schlau' (2) the word 'sind' can function as a perfect auxiliary, e.g. 'die Kinder sind – verreist'. 'f_able' means usable as a filler. This property does not apply as long as the predicative adjective or predicative participle have not yet been attached and, hence, the frame is incomplete.
`006 Complements:`	Here the templates are listed that describe the expected complements. The lines are indexed with so-called cluster numbers, separate for each frame. Hence, the first frame requires '`+praed_adjektiv'` with two alternatives. The second frame requires '`+praed_perfekt'` with two alternatives too.
`007 Adjuncts and conjuncts:`	Here templates are listed that describe the use of the chart element as adjunct. The whole verb phrase is a candidate as '`%relativsatz'` (relative clause). De facto this role would fail, though.

008 Parameters:	seg_type (segment type) = 1 word, 2 part of a compound, 3 to be ignored. origin (the way the bulletin was built) = 1 by lexicon lookp, 2 by elliptic complement filling, 3 by complement filling, 4 by adjunct filling, 5 by conjunct filling. head_rear and head_front = edges of the head word. l_blocked and r_blocked = no more filler to the left or to the right. first_morph, last morph - in a compound. unknown = 1 the result contains a word that has not been found in the lexicon. utterance = 1 an utterance marker or EOF follows. conjunct = 1 the head is coordinating conjunction. prepost = 1 a prejunct is expected, 2 a postjunct is expected. discont = a discontinuous slot exists, the filler is expected 2 to the left, 3 to the left or the right, 4 to the right. quote = a quote type slot exists, the filler is expected 2 to the left, 3 to the left or the right, 4 to the right. orphan = a orphan type slot exists. f_able = 1 the bulletin is usable as filler. fh_able = 1 the bulletin is usable as head to the right of a filler. hf_able = 1 the bulletin is usable as head to the right of a filler.
011 Dependent roles:	The following syntagmatic roles occurred. This is relevant in coordination.

It is possible to show all bulletins of a result. If the final result is chosen, one gets kind of a history of the course of parsing. The command *Compare Bulletins* displays two bulletins in parallel. This tool is useful if some entries in the chart look alike and one would like to know the difference.

Show diagnosis

You will be prompted for the number of two chart elements. *Show Diagnosis* resets the parser to the situation of processing these two elements. The parser tries anew to combine the corresponding constituents in order to form a new constituent covering both. All possibilities according to the bulletin data are tried, e.g. interpreting one constituent as head and the other one as complement, or one as head the other one as adjunct etc. Let us test the tool with an ungrammatical input:

```
Input:
das  Kinder

No complete parsing achieved

Statistics: 2 words, 2 bulletins - no overgeneration

Chart:

  1:   1-   4   (das )
                left:  0  right:  0
  2:   5-  12   ( Kinder )
                left:  0  right:  0
```

Figure 28: The chart after ungrammatical input

The diagnosis tool finds out why element 1 and element 2 do not combine:

```
Checking compatibility:

chart   1:   1-   4   (das )

1   'das '
 (lexem[definit'] kategorie[artikelwort] flexion[stark-schwach]
genus[neutrum] kasus[nominativ,akkusativ] numerus[singular] wortlaut[das ]
schreibung[klein]);

chart   2:   5-  12   ( Kinder )

5   ' Kinder '
 (lexem[kind] kategorie[nomen] kasus[nominativ,genitiv,akkusativ]
 numerus[plural] person[dritte] pronomen[nein] wortlaut[ Kinder ]
 schreibung[klein]);

*** Trying left complement
    1  +bestimmungswort = slotfiller unmatched: kompositum[+]
    2  +praepattribut = conflicting direction

*** Trying right complement = not suitable
    (L not hf_able or  R not f_able)
    (not: L hf_able and R discont)
    (not: L conjunct and R fh_able)
```

```
*** Trying left adjunct
    3  %individuativ = head-filler unmatched: numerus[singular,C]

*** Trying right adjunct = not suitable
   (R not f_able or  not adjunct or conjunct)

Involved templates:

  1  +bestimmungswort   slotdescription:  complement
 (kategorie[nomen] schreibung[klein]);

 (L rolle[zugeordnet,A] kategorie[partikel] kompositum[+] angrenzend[+]);


  2  +praepattribut   slotdescription:  complement
 (kategorie[nomen]);

 (R rolle[zugeordnet,A] kategorie[praeposition] angrenzend[+]);


  3  %individuativ   slotdescription:  adjunct
 (kategorie[nomen] pronomen[nein] n_position[3]);

 (L rolle[determination,A] kategorie[artikelwort] determination[+,C,A]
 flexion[C] genus[C] kasus[C] numerus[C] w_pronomen[C] n_position[1]);
```

Figure 29: Diagnosis

At first the attribute of both elements are shown. Then all kinds of attachments (complement to the left, to the right, adjunct to the left, to the right) are tried. The numbers refer to the templates that are sketched under the heading `'Involved templates:'` Each template contains two sets of attributes: those of the head and those of the slot. Template no. 3 would be the right one to attach the determiner to the noun as an adjunct (`'%individuativ'`). It requires agreement '`numerus[C]`'. Since the determiner is '`numerus[singular]`' and the noun is '`numerus[plural]`' the diagnosis reads:

```
*** Trying left adjunct
    3  %individuativ = head-filler unmatched: numerus[singular,C]
```

Of course, ungrammatical input is not the main application of the diagnosis tool. This tool is most useful if the linguist is puzzled why the parser does not work as expected for a particular grammatical construction. It is the best way of detecting bugs. It is also helpful to understand why useless intermediate results are created. Subsequently, templates can be made more efficient.