The Prague Dependency Treebank: Annotation Structure and Support

Jan Hajič
Institute of Formal and Applied Linguistics
Faculty of Mathematics and Physics
Charles University
Malostranské nám. 25
CZ-118 00 Prague 1
e-mail: hajic@ufal.mff.cuni.cz

Barbora Hladká, Petr Pajas
Center for Computational Linguistics
Faculty of Mathematics and Physics
Charles University
Malostranské nám. 25
CZ-118 00 Prague 1
e-mail: {hladka, pajas}@ckl.mff.cuni.cz

Abstract

The contents of the Prague Dependency Treebank (recently released by the Linguistic Data Consortium in its version 1.0) is described, from morphology to surface syntax to the deep (underlying) syntax layers of annotation. For each layer, the basic assumptions are given, followed by a more detailed description of the annotation scheme. Annotation software currently in use is characterized and its distinguishing features are emphasized. Finally, the checking schema and procedures adopted for the release of the Prague Dependency Treebank version 1.0 are discussed.

1 Annotation Scheme

1.1 The Prague Dependency Treebank Project History

The Prague Dependency Treebank Project (PDT) has been designed in 1995-1996 as a cooperative project[1] of the Institute of Formal and Applied Linguistics (IFAL) at Charles University, Prague and several other institutions which provide data, feedback and support.

The project started in 1996 (Hajič et al., 1996), together with a Czech National Corpus (CNC) project (Čermák, 2001), which provides the initial (raw but clean) data for a subsequent linguistic annotation, and originally was meant for a mere specification of what should (and could) be annotated both morphologically and syntactically. Later in 1996, with additional funding for the newly constituted Linguistic Data Lab as a part of IFAL, the actual annotation began (Hajič, 1998), (Hajič et al., 2001); this first phase of the project was proposed to last for five years and to annotate about 1M words of data morphologically and (surface-)syntactically.

The project currently continues in the new Center for Computational Linguistics (CCL), also at Charles University, Prague (where almost all the funding is coming to) in a close collaboration with IFAL. The deeper level of syntactic annotation based on the Functional Generative Description of natural language (Sgall et al., 1986) should be specified and subsequently annotated before the end of 2004. (Hajičová et al., 2001) describe the latest developments of the Prague Dependency Treebank.

1.2 Annotation Contents

The Prague Dependency Treebank has been conceived as an open-ended, long-term project of linguistic annotation of Czech texts. Three layers of annotation have been defined so far. Certain basic "axioms" have also been stated to facilitate the specification of the contents of the annotation on the three layers:

1. Morphological layer
    (a) All text words (tokens), as defined by the Czech National Corpus tokenizer[2], are taken into account separately for morphological annotation, the result of which is a set of *[lemma,tag]* pairs;
    (b) After (manual) disambiguation, only one *[lemma,tag]* pair per word is allowed in the annotation.
2. Analytical layer (surface syntax)
    (a) A single-rooted *dependency tree* is being built for every sentence[3] as a result of

---

[1] The project acknowledges the support of the Grant Agency of the Czech Republic, grants 405/96/0198 and 405/96/K214, and the Ministry of Education of the Czech Republic, projects VS96151 and LN00A063.

[2] The tokenizer has been also designed by IFAL. It defines a token (roughly) as a string of letters and numbers delimited by spaces and punctuation, including periods, apostrophes, and hyphens. Every non-WS delimiter is a token, too, as usual.

[3] Sentences are recognized automatically by the CNC tok-

the annotation, with a single *(analytical) function* describing every dependency relation in the tree;

   (b) Every item (token) from the morphological layer gets exactly one node in the tree, and no nodes (except for the single "technical" root of the tree) are added;

   (c) The *order* of nodes in the *original sentence* is being preserved in an additional attribute; non-projective constructions are allowed;

   (d) Only one (manually assigned) analytical annotation (dependency tree) is allowed per sentence.

3. Tectogrammatical layer (underlying syntax, or linguistic meaning)

   (a) A single-rooted *dependency tree* is being built for every sentence, with dependency relations marked by *functors*[4] (deep syntactic relations);

   (b) As a general rule[5], only nodes labeled by *autosemantic* words are in the tree[6];

   (c) Insertion of (surface-elided) nodes is driven by the notion of *valency*: if a word is deemed to be used in a context in which some of its valency frames applies, then all the frame's slots are "filled" (using regular dependency relations between nodes) by either existing words or newly created nodes, annotated accordingly;

   (d) Every node of the tree is furthermore annotated by such a set of grammatical features that enables to fully capture the meaning of the sentence (and therefore, to generate[7] the original sentence);

   (e) Topic and focus are marked, together with so-called deep word order[8];

   (f) Grammatical and some textual coreference is resolved and marked.

1.2.1 The Morphological Layer

The annotation at the morphological layer is compatible with the IFAL's morphological analyzer (Hajič, 2001). The tagset is based on a full set of morphosyntactic categories known for Czech. 13 categories (Part of speech, Detailed POS, Gender, Number, Case, Possessor's Gender and Number, Person, Tense, Voice, Degree of Comparison, Negation and Variant) are being used. There are 4257 plausible combinations of category values, as present in the morphological dictionary (about 1/4 of them are actually found in the PDT).

Lemmas are also annotated at this layer. The notion of lemma is seen as a unique identification of the morphological dictionary entry the textual form belongs to. Therefore, even though (human-readable) strings similar to dictionary entry headwords are used, they are merely references (pointers) to the appropriate paradigm.

1.2.2 The Analytical Layer

Based on the "axioms" above, two things are the actual result of annotation at the analytical layer: the dependency relations, and the analytical functions (and the linear order index of every token). Given that the root of the dependency tree is added to insure single-rootedness of broken or non-predicative sentences, the number of dependencies in a sentence is always equal to the number of tokens in the original sentence.

The direction of the dependencies is determined according to the usual conventions used in dependency-based theories, and can by simply stated in general as follows: given a dependency relation holds between a pair of nodes, a node is dependent on another (so-called governor) node, if the deletion of the dependent node will not harm the "grammaticality" of the sentence. E.g., adjectives depend on nouns, subjects, objects and adverbials on verbs, etc (see Fig. 1). In cases when this rule cannot be applied because the relation is more complex (and thus not representable by the simple definition of a tree adopted), the direction of the depen-

---

enizer, but sentence break errors are manually corrected in the PDT.

[4] The set of functors is much richer than the set of analytical functions from the layer 2 of the annotation scheme.

[5] With the exception of cases in which dependencies are used rather technically because of their inability to capture certain "non-dependency-like" phenomena, such as coordination.

[6] Thus many nodes, found at the morphological and analytical layers, disappear (such as function words, prepositions, subordinate conjunctions, etc.) or, better to say, the information carried by the function words is "passed over" to the relevant attributes of the autosemantic nodes they belong to.

[7] Perhaps by a quite sophisticated (generation) algorithm, certainly not as straightforward as its counterpart at the analytical layer.

[8] The order of nodes, also marked at this layer, is in general different from the surface word order, and all the resulting trees are projective by the definition of the deep word order.

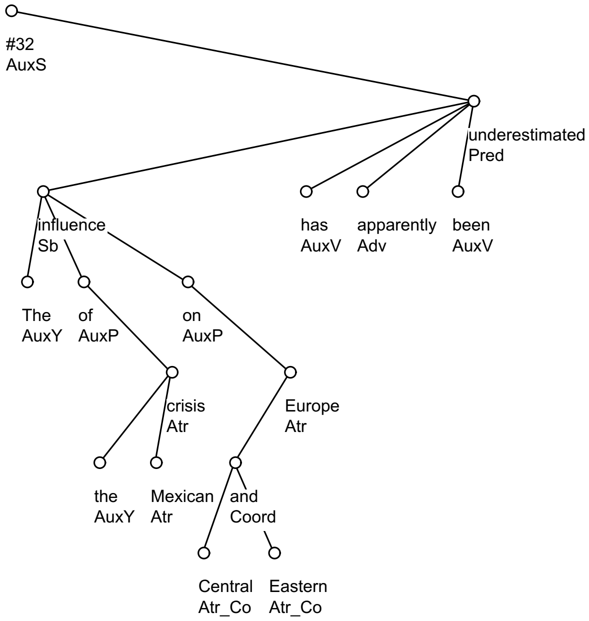
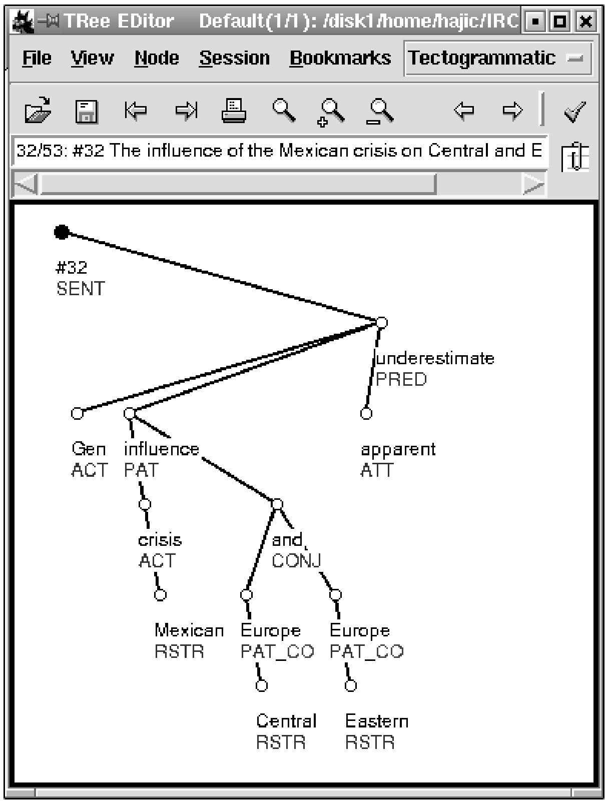


Figure 1: Analytical Tree Annotation of the sentence *The influence of the Mexican crisis on Central and Eastern Europe has apparently been underestimated.*



32/53: #32 The influence of the Mexican crisis on Central and E

Figure 2: Tectogrammatical Tree Annotation of the sentence from Fig. 1 (in the `TrEd` editing window).

dencies is determined by convention (e.g., prepositions govern their noun phrases, subordinate conjunctions govern the subordinate clauses etc.).

Coordination and apposition are phenomena that evade the usual assumptions about dependencies. They also have to be annotated by convention; we decided that the coordination conjunction (that includes commas, if necessary) "govern" the set of coordinated subtrees. Each of the coordinated subtrees is moreover marked as a "member of a coordination"; this allows for common modifications of the whole coordination construction to be annotated easily (by simply making them depend also on the coordinating conjunction node, but *not* marking them as the members of the coordination). The same scheme is used for apposition.

The set of analytical functions consists of 26 functions, most of them, however, being so-called "auxiliary" functions (for proper marking of prepositions, conjunctions, auxiliary verbs, reflexive particles, coordination, etc.). The main functions are only the predicate, subject, object, adverbial, attribute, and verbal attribute (complement). Therefore, neither e.g. objects nor adverbials are further (sub)classified; that task is postponed to the tectogrammatical layer.

1.2.3 The Tectogrammatical Layer

The tectogrammatical layer (for an example annotation see Fig. 2) can be best described as the "deep (underlying) syntax", as labeled in some theories; we use the term "linguistic meaning". Its basic characteristics are specified in Sect. 1.2, item 3. The tectogrammatical layer annotation scheme is divided into four sublayers: the core annotation (dependencies and functional annotation), the topic/focus and deep word-order annotation, the co-references, and finally the fully specified tectogrammatical annotation (completing the necessary grammatical information). Only the core annotation sublayer is described here.

The aim of the tectogrammatical layer is to go beyond the surface shape of the sentence with such notions as "subject" and "object", and to employ notions like "actor", "patient", "addressee" etc., while still being mostly driven by the language structure itself rather than by the general world knowledge.

The units of description are *autosemantic words* (by "autosemantic" we mean words that bear some kind of lexical meaning, not just grammatical function). Dependencies between units are seen as the basic tool to achieve this goal through a uniform description of all the relations (not just that of the predicate) between the (autosemantic) words in a sentence[9]. The dependencies are labeled by so-called *functors* (and their *syntactic grammatemes*, providing more detailed classification of functors if necessary), which describe the dependency relations. Every sentence is thus represented as a (dependency) tree, the nodes of which are the units of description (autosemantic words), and the (labeled) edges represent dependencies among them.

The words and dependencies alone would, however, form an incomplete representation (since the goal is to fully represent the linguistic meaning of the given sentence) in the sense that it would not be possible to uniquely generate a (surface) sentence with the original meaning from such a representation. We still need some grammatical information to achieve that goal: for example, we need the attribute "NUMBER" to know if a certain noun has the meaning of plural or singular.[10] On the other hand, with many nodes this information is unnecessary and in fact it might be "harmful" under certain circumstances[11], since it can be derived from other parts of the tree (by grammatical agreement rules, for example). Thus the NUMBER attribute is never specified for verbs, for example. Similarly, we need such attributes as tense, modality (both sentence and verbal), and gender, but again only where ambiguity could arise and its value cannot be determined from other attributes of the current node or other nodes in the representation.

We use a set of 72 functors. They are divided into several categories: arguments (actor, patient, addressee, origin, effect), free modifications (such as those of time, place, manner, etc.), and several smaller "technical" categories for expressing negation, coordination and apposition, foreign phrases etc. The assignment of functors is driven primarily by *valency frames*, and (if no valency frame is available for a given governing word) by default definitions of the functors. Valency frames are used for all verbs and verbal nouns and adjectives, and (at the final fourth sublayer of tectogrammatical annotation) also for all nouns and adjectives proper.

A valency frame contains a set of slots, each of them characterized by a single functor (such as "addressee"), list of possible surface expressions (such as the required case, sometimes together with a preposition, etc.), and an indicator whether the slot is (semantically) obligatory or not. Certain transformations of the surface expression form (passivization, reflexivity, nominal derivations) are considered implicit and such ("derived") frames are not explicitly present. Each word can have several valency frames; at this point, we do not distinguish word senses or group frames into sense groups, but we do group frames into "superframes": a *superframe* is a set of frames that are indistinguishable if only obligatory slots are considered. A finer distinction of "obligatory" and "facultative" slots is currently being developed (Skoumalová et al., 2001).

Valency frames could, in theory, be easily collected from an annotated corpus. However, the theory of valency frames is quite elaborate and it is not feasible that the job of creating all valency frames be left to the annotators only, if for anything else than consistency reasons. Therefore, we are creating a valency lexicon on-the-go, as new verbs (nouns, adjectives) are encountered during annotation, and the valency lexicon, a "static" and separate data source, will be an important part of the Prague Dependency Treebank at the tectogrammatical layer of annotation. It will contain not only frames collected from the PDT, but also manually determined frames based on a larger unannotated corpus.

Ellipsis (on the surface) is an omnipresent phenomenon in actual language use, in fact more common than one usually thinks. Apart from obvious cases of ellipsis, such as omitted verb (or noun) in the second part of a sentence (or nominal) coordination or in direct speech (especially in short answers to questions), there are problematic cases for the solution of which we have to consider when the missing node in the dependency tree should actually be created "from scratch" and labeled. We use the valency frames in such cases to determine if a node should be created, as well as what function it serves.

---

[9] By sentence we mean a sentence in the corpus ("running text") sense, "from period to period."

[10] In order to generate the correct grammatical number on the surface. However, we do not go that far to require an exact numerical specification to every noun; it is often unknown and hardly determinable.

[11] Such as in machine translation using such structures, when the grammatical number in the target language is determined by its own agreement rules rather than by the grammatical number of the corresponding word in the source language.

1.3 Annotation Format

Although we still use a "legacy" format that is very compact but not very programming-friendly[12] in the annotation tool `TrEd` (see Sect. 2), we currently also support an SGML version of the data for data interchange[13]. The current annotation structure is based on an "all-inclusive" design: everything including the original text, level-0 markup, morphological, analytical (surface-syntactic) and even tectogrammatical markup is present in a single document. The annotation markup is in fact a superset of the markup used in the Czech National Corpus, from which we took the original data. It is quite clear that such a structure, with the complexity of the markup, is becoming inadequate. Our (hopefully short-term) plans and ideas about transforming the annotation to a new scheme based on XML are described later in this section.

The current formats correspond to a large extent to the annotation strategy. For example, individual text tokens become target of the morphological markup. Each morphologically annotated token forms a node of an analytical tree the dependency structure of which is represented either explicitly (in the legacy data format) or using pointers from dependent to governing nodes (in the SGML version). On the tectogrammatical layer, a tectogrammatical tree "borrows" some nodes from the analytical tree (in fact, only those nodes which represent auto-semantic words). Beside that, a tectogrammatical tree may contain additional nodes, newly created to represent words elided on the surface level. Though the structure of an analytical tree and that of its tectogrammatical counterpart are similar, they are not identical and must therefore be represented separately. In the SGML format, this can easily be handled by having separate markup for each type of dependency pointers. On the other hand, there is no way to markup both of the trees at the same time in the structure-based format (since the only means to markup a dependency in that format is parentheses). Therefore, when using this data format, one must choose which of the trees is to be represented explicitly (i.e. which will be, for example, displayed as a tree in the annotation tools described in Section 2) and which will be encoded into pointers. In spite of the fact that the sets of nodes over which the trees are built overlap, they are not identical. To solve this technically, we use so-called *hidden nodes*. Such nodes, albeit technically present in the tree, are not considered to belong to it. It does not matter to which node they are attached, and when a node is hidden then every element in its sub-tree is hidden, too. In the visual representation on a computer screen these nodes are either completely omitted or at least visually distinguished (annotators may switch between these two states since there are cases when, for technical reasons, we want to be able to see all the nodes, regardless to which layer they actually belong to).

Beside the dependency differences, the node order is different in the two types of trees. An analytical tree is ordered according to the surface ordering of the sentence. Thus, analytical trees may be non-projective. In the tectogrammatical tree, the nodes are ordered by the deep word order, which is related to topic-focus articulation. These trees are always projective. For technical reasons there is one more ordering defined on nodes, which orders nodes belonging to both types of trees. This ordering is identical to the surface order on analytical nodes, but all nodes newly created on tectogrammatical layer follow the analytical nodes, and among themselves they appear in arbitrary order.

As mentioned above, we are preparing a complete rewriting of the annotation scheme. It should obey the following principles:

1. Extended Markup Language (XML) will be used to represent all markup layers.
2. The markup will follow the stand-off-annotation principle. The original data and individual markup layers will be separate.
3. The original data will be marked in the basic TEI XML markup according to TEI P4X guidelines. The data will be kept constant and separate.
4. The DTD for each layer will be designed to adequately and consistently represent the structures described in the theory. Compatible solutions will be chosen wherever possible, so that markup originating from several layers can be combined together when needed.
5. Extended pointers will be used to connect the annotation with the original data as well as to link the various layers with each other.

---

[12] Derived originally from the dependency-oriented formalism of Colmerauer's Q-SYSTEMS with an implicitly marked tree structure.

[13] Some tools can work with the SGML format directly.

6. If possible, consistent approaches will be used to solve similar problems on various layers (alternative markup for example).

We believe that the stand-off approach described above will make it much easier to develop adequate annotation markup and that it will simplify many tasks which arise during the corpus-annotation process. For example, since the layers of markup will be stored separately, large changes may be performed on one of them if needed without a fear of affecting or damaging the other ones (provided that certain rules are applied which prevent or make it easy to repair any link disconnections that may appear during such a process).

Obviously, with this approach some of the human annotation as well as post-annotation checking may be done independently on different layers with no need of subsequent complicated merging.

On the technical background, the XML format brings the advantage of existing and already well-implemented standards. For example, assuming that a well-designed XML representation of the annotation on analytical and tectogrammatical layers is adopted, XML-related technologies like XPath or XSLT may be used even for simple structure queries and transformations.

2 Support

2.1 General Annotation Tree Editor

The central annotation tool is a language-independent[14] graphical tree editor called `TrEd`[15].

The `TrEd` editor

- allows to create and modify the dependency sentence structure (or any tree-based structure)[16],
- can be "internally" programmed using a powerful macro language based on the Perl programming language, using a defined API for its internal data structures,
- can communicate "externally" with other programs running over the network, effectively allowing for client-server type application to be easily integrated into it[17],
- may be used in an interactive mode (for the usual manual annotation) as well as in a batch mode (for post-processing or any other automated task to be performed on a number of files),
- contains features allowing for batch as well as interactive visual checking and comparison of doubly-annotated data
- allows extensive customization of editing possibilities (e.g. to enable/disable certain features for different types of annotation) and customization of the visual layout to use effectively the screen's real estate to fit the task at hand as well as individual annotator's preferences.

2.2 Macro Programming Capability

TrEd is designed to be extensible by user-defined macros, which may not only add new functionality to it but also alter its default behavior. This design gives the editor the power of a general annotation tool that can be highly customized for any specific annotation task. In order to make the human annotation on analytical and tectogrammatical layers as easy and effective as possible, a large set of specialized macros was created. Their functionality ranges from simple tasks, such as assigning a specific analytical function or tectogrammatical functor to a single node, to rather complicated ones, which are able of automatic assignment of the statistically most probable function or functor to each node in a tree. Though the macros for automatic function or functor assignment are not 100% successful (if they were, there would be no need for human annotation!), at least they do save the annotators some typing and in some cases they may even guide them.

The macros as well as the editor itself are written in Perl programming language (which was chosen for its multi-platform portability and also for the reason of rapid development). TrEd can be used either as an interactive tree editing tool (which is how it serves during the annotation process) or as an off-line processing tool, in which case its only task is

---

[14] Current version of `TrEd` only supports languages with 8-bit encodings but it is writing-system-independent (i.e. it can handle right-to-left scripts).

[15] For a sample screen shot, see Fig. 2.

[16] `TrEd` is currently used for the annotation on the tectogrammatical layer, but it can be — and it will be, if necessary — used also for the analytical layer annotation, even though the analytical layer has been annotated in its greater part by a simpler, MS-Windows-only specialized graph editor called `GRAPH` (Křen, 1996).

[17] Such as a speech-enabled interface based on a third-party client-server enabled software, should such thing be deemed effective for the annotation task.

to evaluate a specified macro over a set of trees. The latter, non-interactive mode, is used both during the automatic step of transformation from analytical trees to their tectogrammatical shapes, and in the process of post-annotation checking, during which a set of rules is applied to verify the correctness and consistency of the annotation (see Sect. 3).

In the editor, the annotation markup appears either as a dependency (explicitly displayed by edges connecting the nodes) or in form of named labels called attributes. Recently, there are 84 attributes defined on the nodes of tectogrammatical trees. The user may specify which of the attributes are actually shown on the screen; the rest of them is displayed on demand only.

2.3 Restrictive Task Customization

To prevent data damage caused by mistakes, annotators usually obtain a restricted version of the editor[18] where only actions relevant to the annotation on the specific layer are permitted. Moreover, only macros related to their individual annotation tasks are enabled. Thus, the annotators of the tectogrammatical layer are limited to read-only access to the attributes containing morphological and analytical markup, and the annotators of the analytical layer have read-only access to the morphological-markup attributes.

2.4 Integration of the Valency Lexicon

We have already said that assignment of functors on the tectogrammatical annotation is tightly connected with the valency frames. Therefore, for the purposes of tectogrammatical annotation, a set of tools was developed to ease the searching and appending the valency frame lexicon. These tools provide a graphical user interface to the lexicon that is itself stored in the form of an XML file. The interface is connected with the tree editor through several dedicated macros. These macros allow the annotators to assign a valency frame to a node by choosing it from a list of frames from the lexicon. When a valency frame from the lexicon is assigned to a node, its identifier is stored as one of the attributes in the data. In the near future, macros for automatic functor assignment and elided-node generation based on the selected valency frame will be created. We hope that, among other purposes, the connection between the data and the lexicon will be useful for future consistency checking.

Since many nouns in Czech are derived from verbs, they share some of the valency frames with the verb they derive from[19]. In such a case the annotators have the opportunity to choose either a specific frame for the noun, or any frame from the verb it is derived from according to the morphological annotation.

The user interface lets the annotator add new items to the lexicon if no existing frame fits the context (or if there is yet no entry for the word concerned in the lexicon). It is a general rule that each lexicon entry should contain an example which may be optionally supplemented by annotator's note or comment. Since there are more annotators working on the tectogrammatical annotation (and not all of them may work on-line) merging their private lexicon additions must be done at regular intervals.

There are certain facilities which give the annotators the possibility of modifying, deleting or replacing existing valency frames that are incorrect or incomplete. However, since such frames may have already been assigned to some nodes, they are never removed from the lexicon permanently but rather marked as obsolete. Every modification of a frame results in actual addition of a new entry to the lexicon, while the old entry is marked as obsolete and either hidden from the annotators completely, or at least visually distinguished from the valid entries. In the process of post-annotation checking all the nodes containing a reference to an invalid or obsolete valency-lexicon entry will be checked and (where possible, automatically) corrected.

The choice of the correct valency frame is not always as easy as it could seem. For that reason it is allowed to assign more than one valency frame to a node in those cases where the decision cannot be made by the annotator. As was already mentioned in Sect. 1.2.3, we group frames into so-called *superframes*, which group frames that are indistinguishable as long as only obligatory slots are considered. An annotator may choose such a superframe instead of a single frame in case he or she is in doubts which of the frames it contains actually corresponds to the text being annotated. As a result, identifiers of all frames belonging to the same superframe are stored in the data. The question if they should or should not be disambiguated in the end (and even if such disambiguation will ever be possible) is postponed and certainly requires further research.

---

[18] As the restrictions can be applied from the macros, there is no need to actually create different versions of the program.

[19] There is a canonical transformation of a verb's valency frame to a frame of its noun derivative.

2.5 Visualization of Inter-annotator Discrepancies

Since the consistency between annotators is an extremely important factor of corpus annotation, consistency tests are performed in regular intervals during the annotation of tectogrammatical layer. In the test, each annotator obtains the same piece of data for annotation, their annotations are compared, and the results evaluated. A special program was developed to effectively find differences in both the tree structure and its labeling (i.e. node attributes).

Although the output of such comparison is suitable enough to get an overall idea about the consistency and even for further automatic processing, it does not contain the very trees concerned and therefore it cannot be used by the annotators, who usually want to see the actual places in the trees where they erred or diverged. Because of that, a version of the same program in a form of a tree-editor macro was created and extended to visually display the differences in the data interactively allowing the annotators to make corrections immediately.

2.6 Annotation Maintenance Tools

The planned translation of the Prague Dependency Treebank to XML format based on the principles mentioned in Sect. 1.3 brings about a need to create tools that would handle the specific tasks of this approach. These include support libraries to simplify parsing and link dereferencing, as well as pre- and post- processing tools which could be used as wrappers for legacy or close-source applications unable to resolve the pointer cross-references between annotation layers or the links to the pure data. Such tools could merge the markup of several annotation layers into a single file and feed that file to the application. Note that this may not be straightforward under all circumstances since the structures being merged may overlap. Hopefully, the need to do this will be rare. The reverse process of splitting a merged markup to the individual layers in the standoff format hierarchy should also be possible.

3 Data Checking

In a manually annotated corpus, the most important issue which must be pursued is *consistency* either guaranteed by the annotation guidelines, or by the limited number of annotators. No matter which way is chosen, it is naturally necessary to devote a large proportion of time to corpus checking in addition to the annotation process itself.

3.1 Morphological Layer

There was no annotation manual on the morphological layer (the categories in the tagset used correspond directly to what every high school graduate knows about Czech morphology); the absence of a manual was thus replaced by a consistency-driven distribution of annotators across the whole annotation process. The data was, first, double-tagged; while nine annotators have been involved in the first pass of the annotation, only two annotators have participated in the checking step. The discrepancies coming from two annotated versions of the same file (4–5% on average) were checked and disambiguated by only one annotator.

Each file has than been checked against the automatic morphological analyzer (AMA), mainly because the coverage of AMA's dictionary had been changed during the time span of the annotation. For each word form, the AMA generates a set of [*lemma*, *tag*] pairs that is compared against the manually assigned [*lemma, tag*] pair.

Our strategy is to miss as few annotation errors as possible. If the manually assigned pair is found in the AMA's set, we know that the given word is annotated consistently with the current AMA (although it might still be wrong given its context). If the manually assigned pair is not found in the set, then either the AMA is wrong or incomplete, or the manual annotation is wrong and has to be revised manually. Altogether, in the case of PDT 1.0 the AMA check has been done twice.

Besides the ”visible” discrepancies, we had to revise those word forms that have never been recognized by the AMA, this concerned 19K out of 1.7M words annotated on the morphological layer.

The annotation problems found here are, however, not just plain annotation errors; misspellings in the original text and formatting errors (such as wrongly split or joined word forms) are discovered and corrected (3.6K words in all), keeping the original input (properly marked) for further exploitation, such as spelling error analysis.

3.2 Syntactic-Analytic Layer

The analytical annotation task was certainly much more complex for the annotators than the morphological one. Thus it was not possible to use the double-tagging strategy used at the morphological layer. When the annotation on the analytical layer started, the guidelines for analytical annotation (Hajič et al., 2001) were only rudimentary, and it was decided that they would be improved throughout the whole period of the annotation. The strategy chosen was to always annotate in accord with the *most recent* version of the guidelines. We were aware that as the guidelines were evolving, many inconsistencies in the data were arising and that it was technically impossible to resolve them on an ongoing basis. Therefore we were preparing a post-annotation checking phase by collecting the annotation rules that were known to have changed, or that were found to be constantly problematic. The actual checking began when the development of the annotation guidelines somewhat stabilized (about halfway through the analytical-layer annotation).

The post-annotation checking procedure consisted of more than 50 steps of varying complexity, each of them checking correctness and/or consistency of the annotation of a specific issue. Some of the issues are technical, others are linguistically-oriented. These issues could be divided into three groups, according to the amount of manual work needed to correct an annotation that failed a given test: the fully automatically correctable issues, automatically locatable and subsequently manually corrected issues, and only manually locatable and correctable issues. Unfortunately, only few issues fell into the first category requiring no human action (besides writing the testing and auto-correcting procedure, of course). On the other hand, there was no issue that would fall into the third category. We were always able to at least automatically distinguish places potentially affected by the problem, and we never had to "read" the corpus once again in full. The only exception was that of Czech reflexive pronoun '*se*': this word may be marked with one of three different analytical functions according to its context. In this case we did not find a better solution that would lead to consistent results, than letting one person annotate separately every occurrence of this word in the whole treebank.

Thus most of the tests fell into the second group where only a certain amount of "suspicious" nodes was to be manually checked.

A great part of post-annotation checking was to resolve problems already marked in the data by annotators. These concerned mostly broken or strange sentences originating often from advertisements, or sport and TV program newspaper columns. Among the technical issues, sentence-breaking errors were the most frequent. There were several methods to locate such errors which often complemented one another, e.g. multiple predicates in one tree, more than one node in a tree marked with the auxiliary function for full-stop, etc.

3.3 Morphological vs. Syntactic-Analytic Layer

So far, we have provided the description of consistency-driven annotation *within* a particular layer of annotation. Since the PDT has a three-layer structure, we have to pursue the consistency also *across* the layers. In case of the version 1.0, when the separate layer-driven corrections were finished, a mutual revision of the morphological vs. syntactic-analytic annotations could start. This mutual checking has been practically realized as a set of regular expression-based rules.

In a dependency tree (as described in Sect. 1.2, item 2), let w_i and w_j be two dependent nodes corresponding to the *i*-th and *j*-th, respectively, let $j < i$, words (without loss of generality) in a particular sentence; be w_i a governing node of w_j. The mutual revision through the tree structures offers a way to incorporate a sentence context when looking for incorrect (not all of them, obviously) annotations. In our case, we concentrate on two aspects: (a) agreement in *case, gender* and *number* between predicate and depending subject, as well as between attribute (`Atr`) and its governing node (see Fig. 3), and (b) *case* agreement between a preposition and depending noun, pronoun, adjective or numeral (see Fig. 4).

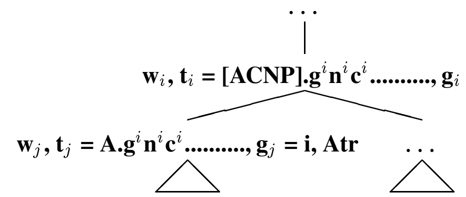


Figure 3: *case, gender, number* Agreement of Depending Nodes

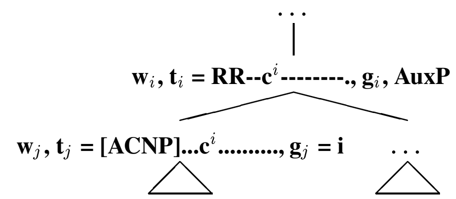


Figure 4: *case* Agreement in a Prepositional Phrase

The mutual revision through the analytical functions presents a tool to detect the errors on both sides of comparison — from this point of view, the analytical functions `Obj` (object), `Sb` (subject) and `Pred` (predicate) were tested against their morphological tags. Fig. 5 visualizes one of the checking rules — the one testing whether `Sb` (subject) has a relevant morphological tag (adjective or numeral or noun or pronoun in nominative ('1' in the fifth position) or in any *case*('X' in the fifth position), or verb, infinitive or number as a string of digits).

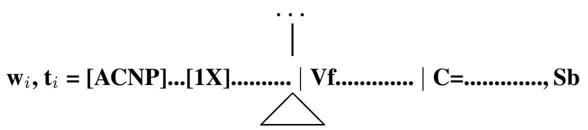


Figure 5: Subject and its Possible Morphological Tags

4 Conclusions

We have described the current status of the annotation scheme, the main annotation tool, and some experience with corpus checking. We will briefly mention the three remaining sublayers of the tectogrammatical layer, since their specification is still being developed.

The *deep word order* and the *topic/focus* attribute represents an attempt to annotate the effect of discourse structure inside a sentence. The "old" information, or the topic, is put to the left, and the "new" information, introduced by the given sentence, is moved to the right[20] at every level of the dependency tree.

Co-reference is supposed to be marked only to a certain extent: all of *grammatical co-reference* will be marked (e.g. the co-reference between a relative pronoun "which" and its antecedent), and some textual, too. Co-reference pointers can cross sentence boundaries, but in general only such types of co-reference that usually do not are being considered.

The full set of grammatical attributes will be annotated last, and we assume that only a certain portion of the whole treebank will be annotated at this final layer because of its complexity.

We will also prepare a "differential" specification of the tectogrammatical layer for English and Arabic, and we will annotate text samples in these languages for experiments with machine translation based on such deep structural analysis.

---

[20] In Czech, only very little change is needed, since in general the surface word order obeys this order.

References

Jan Hajič, Eva Hajičová, and Alexandr Rosen. 1996. Formal Representation of Language Structure. In *TELRI Newsletter No.3*, pages 12–19. TELRI Consortium.

Jan Hajič, Alla Bémová, Eva Buráňová, Jiří Kárník, Petr Pajas, Jarmila Panevová, Jan Štěpánek, and Zdeňka Urešová. 2001. A Manual for Analytic Layer Tagging of the Prague Dependency Treebank. Technical Report Translation of TR-1997-03, ÚFAL MFF UK, Prague, Czech Republic. English translation of the original Czech version by Eva Hajičová and Zdeněk Kirschner.

Jan Hajič. 1998. Building a Syntactically Annotated Corpus: The Prague Dependency Treebank. In Eva Hajičová, editor, *Issues of Valency and Meaning. Studies in Honor of Jarmila Panevová*, pages 106–132. Prague Karolinum, Charles University Press.

Jan Hajič. 2001. *Disambiguation of Rich Inflection - Computational Morphology of Czech*, volume I. Karolinum, Charles University Press, Prague. 334 pp.

Eva Hajičová, Jan Hajič, Barbora Hladká, Martin Holub, Petr Pajas, Veronika Řezníčková, and Petr Sgall. 2001. The Current Status of the Prague Dependency Treebank. In V. Matoušek et al., editor, *Proceedings of the Third Workshop on Text, Speech, Dialogue*, pages 11–20, Železná Ruda, Czech Republic.

Michal Křen. 1996. Graph editor. Master's thesis, Charles University in Prague.

Petr Sgall, Eva Hajičová, and Jarmila Panevová. 1986. *The Meaning of the Sentence and Its Semantic and Pragmatic Aspects*. Academia/Reidel Publishing Company, Prague, Czech Republic/Dordrecht, Netherlands.

Hana Skoumalová, Markéta Straňáková, and Zdeněk Žabokrtský. 2001. Enhancing the Valency Dictionary of Czech Verbs: Tectogrammatical Annotation. In V. Matoušek et al., editor, *Proceedings of the Third Workshop on Text, Speech, Dialogue*, pages 142–149, Železná Ruda, Czech Republic.

František Čermák. 2001. Language Corpora: The Czech Case. In V. Matoušek et al., editor, *Proceedings of the Third Workshop on Text, Speech, Dialogue*, pages 21–30, Železná Ruda, Czech Republic.