Hierarchical XML Layers Representation for Heavily Annotated Corpora

Dan Cristea and Cristina Butnariu
University “Al. I. Cuza” of Iași
Faculty of Computer Science
and
Institute for Theoretical Computer Science
Romanian Academy – the Iași Branch
Cristea, cris}@infoiasi.ro

Abstract

The paper proposes a scheme for hierarchical representation of XML annotation standards. The representation allows individual work on documents displaying partial fitness in markings, mixing of annotated documents observing or not the same standard, as well as concurrent annotation. The approach allows access to different annotations of a corpus, with minimal representation overhead, which also facilitates accommodation of different, even incompatible, annotations of the same data. Two methods to build a hierarchical representation of annotation standards are shown, one allowing explicit declarations and the other inferring the hierarchy from a set of consistently annotated documents. Merging and extraction operations, which produce derived documents from existing ones are described. A system that implements the formal declarations of the hierarchy and the operations over it is presented.

1. Introduction

The more deeply the linguistic research, the more sophisticated the annotation required. Recently, since XML has become a de facto standard for the representation of annotated corpus resources (Ide, Bonhomme, and Romary, 2000), the sophistication of types of processing over texts, speech or multi-media documents resulted in the production of over-crowded marked documents. Annotation in corpora is not only used to record experts’ view on specific linguistic phenomena, but also to store intermediate results in pipe-line NLP architectures and to post NLP results on the Web (Cunningham et al., 2002). But not always and not for each step in a processing chain are all layers of annotation useful. Usually an NLP step uses as input a document conforming to a certain annotation standard to which it adds another layer of annotation. Also, a human expert uses a tool to annotate a certain document. During the annotation process the expert can make use of some previous annotation layers that, through an interactive tool, can help the task at hand. In all cases, there are reasons to consider that, for a specific task (automatic processing or manual annotation), some existing markings in the input document are useful while others are not and, therefore, could be obscured. Examples of corpora use of this kind are corpus annotation in teamwork and re-usage of annotated corpora. A certain research task in teamwork could require individual experts or software modules, each exhibiting specific knowledge, to annotate at different layers the same original document. Individual results should be merged together in an attempt to compose a document that includes all involved layers. Another example of mixed annotation is given by research tasks that employ existing corpora, to which supplementary annotation layers are added. Heavily annotated corpora obtained in these ways could then be used to draw inter-layer correlations.

The paper reconsiders and enhances a hierarchical scheme to represent annotation standards, proposed in (Cristea et al., 1998), to which a processing machinery is added. Annotation standards are represented in a hierarchy, which enables multiple views over a document. Navigation within the hierarchy observes inheritance criteria. The approach allows access to different annotations of a corpus, with minimal representation overhead, which also facilitates accommodation of different (and sometimes incompatible) annotations of the same data. The approach prefers a standoff encoding scheme to an embedded one (Thompson and McKeVie, 1997). Potentially, the original hub (empty annotation) document resides in an URL that could be different from the one on which the annotation is added. Then, any annotation brackets around a piece of text can be recorded separate from the flesh data through their beginning and end character offsets onto the original text. As such, the hub string, identical in all documents, serves as the absolute system of reference.

We show how relations between different markings can be described in the hierarchy and how the directed acyclic graph representation can accommodate circular dependencies between annotations standards. Two methods to build such a graph are shown, one allowing explicit declarations and the other inferring the hierarchy from a set of consistently annotated documents. The case of concurrent annotations over the same hub document is discussed and a solution for contradictory (overlapping) representations is proposed. Finally, we introduce a set of operations that simplify an existent annotated document and combine two different annotations over the same hub document into a unique one.

2. The Hierarchy – a Lattice Representation

In our approach, different layers of annotation over a corpus are codified as a hierarchy of annotation standards (directed acyclic graph, or DAG). A node in the hierarchy is described according to the following syntax:

```xml
<standard name="standard-name" parents="list-of-parents">
  <tag name="tag-name" attributes="list-of-attributes"/>

  <ref source-tag="tag-name" source-attribute="attribute-name" target-tag="tag-name" target-attribute="attribute-name"/>

  ...
</standard>
```
A standard (node) name is a unique symbol in the hierarchy. A standard inherits all features of its parents. To avoid conflicts, in the present implementation no preference inheritance criteria are given, which means that the features belonging to the parents of a node are supposed to be orthogonal. Features which are new to a standard, vis-à-vis of those inherited, are defined in between <standard></standard> brackets by any number of <tag> and <ref> labels. A <tag> label records a new XML element tag. It has a name (label) and a list of attributes. A <ref> label records a semantic relation (dependency) between two annotation standards. It describes a reference between an attribute, called source-attribute, belonging to an XML tag, called source-tag, of the current standard (the one that contains the ref description) and another attribute, called target-attribute, belonging to another XML tag, called target-tag of a superior standard. A standard is superior to a standard if and only if there is a path from A to the root of the hierarchy that passes through A.

We say that a node A subsumes a node B in the hierarchy (therefore B is a descendant of A) if and only if:
- any tag-name of A is also in B;
- any attribute in the list of attributes of a tag-name in A is also in the list of attributes of the same tag-name of B;
- any semantic relation which holds in A also holds in B;
- either B has at least one tag-name which is not in A, and/or there is at least one tag-name in B such that at least one attribute in its list of attributes is not in the list of attributes of the homonymous tag-name in A, and/or there is at least one semantic relation which holds in B and which doesn’t hold in A.

As such, a hierarchical relation between a node A and one descendant B describes B as an annotation standard which is more informative than A and/or defines more semantic constrains.

Figure 1 displays an example of a declaration of a hierarchy of linguistic annotations. The definition builds a lattice, as that in Figure 2, which intends to describe different layers of annotation useful in many NLP applications. ST-ROOT represents the “empty” annotation (no tags), therefore describing the hub document of free text. Immediately under this trivial standard, three standards, ST-TOK, ST-SEG and ST-PAR are placed. ST-TOK is intended to identify tokens, as words and punctuation, and to mark words’ lemmas, ST-SEG marks borders between elementary discourse units (edus), like in (Marcu, 2000), and ST-PAR simply marks paragraphs. ST-POS is placed under ST-TOK. This standard does not contribute with new tags to the TOK labels inherited but adds the part-of-speech information through its attribute pos. The standard ST-POS is a parent for both ST-NP and ST-VP, which are supposed to mark noun phrases (NPs) and verb phrases (VPs), respectively. Tags of these kinds indicate also the heads of the corresponding compounds, as ids of TOK tags corresponding to the headwords. The ref definitions specify that the head-id attribute of the NP and VP tags should be filled with values of the id attribute of the TOK tags. Then, ST-COREF, placed under ST-NP, is a standard, which intends to mark anaphoric links between co-referential NPs. It supplements the NP tag with a coref attribute. The ref definition evidences the constraint that a coref attribute of an anaphoric NP indicates the id attribute of the antecedent NP. ST-SEG-NP-VP is an standard of an annotation, which marks simultaneously noun phrases, verb phrases and discourse units boundaries. It adds no new markings to those inherited from its three parents. Finally, ST-COREF-IN-SEG is a standard in which the coreferences and segment boundaries are marked, while ST-PAR-SEG-NP-VP adds the paragraph layer annotation to the markings for NPs, VPs and edus.

1 There is no a-priory motivation for which to call one attribute source and another target, apart from the fact that, usually, the target attribute is the id attribute of the target tag. Moreover all target attributes belong to nodes placed upper in the hierarchy.
Figure 3 presents an example of a portion of George Orwell’s novel “Nineteen Eighty-Four” annotated conforming to the ST-COREF-IN-SEG standard.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<R solitude id="0">
  <NP head-id="2" id="0">
    <TOK id="2" pos="N" lemma="Winston">Winston</TOK>
    <TOK id="3" pos="V" lemma="be">was</TOK>
    <TOK id="4" pos="ING" lemma="dream">dreaming</TOK>
    <TOK id="5" pos="PREP" lemma="of">of</TOK>
    <NP head-id="7" id="1" coref="0">
      <TOK id="6" pos="PRON" lemma="he">his</TOK>
      <TOK id="7" pos="N" lemma="mother">mother</TOK>
    </NP>
    <TOK id="8" pos="PUNCT">.</TOK>
  </NP>
  <SEG id="1">
    <NP head-id="9" id="3" coref="0">
      <TOK id="9" pos="PRON" lemma="he">He</TOK>
      <TOK id="10" pos="V" lemma="must">must</TOK>
      <TOK id="11" pos="PUNCT">.</TOK>
    </NP>
  </SEG>
  <SEG id="2">
    <NP head-id="12" id="4" coref="0">
      <TOK id="12" pos="PRON" lemma="he">her</TOK>
      <TOK id="13" pos="V" lemma="think">thought</TOK>
    </NP>
    <SEG id="3">
      <TOK id="14" pos="PUNCT">.</TOK>
    </SEG>
  </SEG>
</ROOT>
```

Figure 3: Example of annotation
3. Representing Circular References

From the above subsumption definition it follows that a standard $B$ that includes references to tags belonging to another standard $A$ should be placed under $A$ in the hierarchy. But what if $A$ refers $B$ as well? Imagine, for instance, an annotation standard in which we have VPs (verb phrases) and SEGs (elementary discourse units) and we want to record for each VP the unit it belongs to, and in each SEG the head VP. Following the above observation, this would raise circularities, which are not acceptable in a DAG structure. Suppose ST-SEG and ST-VP are the standards corresponding to the initial markings that contain the SEG tags, respectively the VP tags, and neither of the two includes references to the other. Because we do not want to modify the existing standards, we can place a new standard, say ST-SEG-TO-VP, under both ST-SEG and ST-VP, in which the SEG tags contain an attribute head pointing the id of the head VP. Similarly, a standard ST-VP-TO-SEG, placed also under ST-SEG and ST-VP, will enrich the VP tags with an attribute, say belongs-to, pointing the id of the surrounding SEG. Finally, a standard ST-SEG-VT, child of both ST-SEG-TO-VP and ST-VP-TO-SEG, would inherit both attributes head and belongs-to from its parents without adding anything else. The result is a hierarchy as that in Figure 4a, whose corresponding description is given in Figure 5a. However, if the intermediate standards ST-SEG-TO-VP and ST-VP-TO-SEG are not useful by themselves, they can be deleted without any loss, such that only the final ST-SEG-VT be kept, child of both ST-SEG and ST-VP, as in Figure 4b, and the description given in Figure 5b. The circular-like constraints appear in the two ref declarations of the ST-SEG-VT standard.

![Diagram of hierarchical representations](a.) and with (b.) circular patterns of ref constraints

It follows that the representation of XML standards that we propose is not in contradiction with some constrains which can have circular patterns.

```
<ROOT>
<standard name="ST-ROOT"/>
<standard name="ST-SEG" parents="ST-ROOT">
  <tag name="SEG" attributes="id"/>
</standard>
<standard name="ST-VP" parents="ST-POS">
  <tag name="VP" attributes="id"/>
</standard>
<standard name="ST-SEG-TO-VP" parents="ST-SEG ST-VP">
  <tag name="SEG" attributes="head"/>
  <ref source-tag="SEG" source-attribute="head" target-tag="VP" target-attribute="id"/>
</standard>
<standard name="ST-VP-TO-SEG" parents="ST-SEG ST-VP">
  <tag name="VP" attributes="belongs-to"/>
  <ref source-tag="VP" source-attribute="belongs-to" target-tag="SEG" target-attribute="id"/>
</standard>
<standard name="ST-SEG-VP" parents="ST-SEG-TO-VP ST-VP-TO-SEG"/>
</ROOT>
```

b.

```
<ROOT>
<standard name="ST-ROOT"/>
<standard name="ST-SEG" parents="ST-ROOT">
  <tag name="SEG" attributes="id"/>
</standard>
<standard name="ST-VP" parents="ST-POS">
  <tag name="VP" attributes="id"/>
</standard>
<standard name="ST-SEG-VP" parents="ST-SEG ST-VP">
  <tag name="SEG" attributes="head"/>
  <tag name="VP" attributes="belongs-to"/>
  <ref source-tag="SEG" source-attribute="head" target-tag="VP" target-attribute="id"/>
  <ref source-tag="VP" source-attribute="belongs-to" target-tag="SEG" target-attribute="id"/>
</standard>
</ROOT>
```

4. Automatic Classification

In order to interact with an existing hierarchy, one should be able to automatically place a new document within it. Two things are important here: compatibility of names and detection of semantic relations.

The first problem deals with name-spaces: in order for a document to be compared against a hierarchy it should be compatible with the tag and attribute names populating the hierarchy. If the annotations in the new document are semantically identical with those in the hierarchy but there exist name mismatches, compatibility can be achieved by...
a translation mechanism. More complex compatibility adjustments can be obtained by working on values, as for instance exploding a range of values of an attribute into new attribute-value pairs.2

The semantic-relations problem deals with recognizing domain values intersection. The identity of values of two attributes or their intersection cannot be certified otherwise but by explicit declaration. Automatic detection is always prone to errors, which can be generated by fortuitous value fitness.

In our system, the classification module takes a hierarchy and an XML document and classifies the document within the hierarchy. The header of the document should declare the list of the semantic relations as a collection of <ref> records, enclosed in a pair of brackets <semantic-relations> ... </semantic-relations>, each having the same syntax as in a hierarchy declaration:

```
<semantic-relations>
  <ref source-tag="tag-name" source-attribute="attribute-name" target-tag="tag-name" target-attribute="attribute-name"/>
  ...
</semantic-relations>
```

The classification process proceeds in two steps. First the document to be classified is parsed and a collection of <tag> and <ref> declarations, having the same syntax as in the hierarchy declaration, is compiled. The <tag> records are computed by collecting all tags and their corresponding attributes of the XML elements, and the <ref> records — by simply reading the <semantic-relations> declarations in the header. Let’s call this computed collection of <tag> and <ref>, the witness collection. Also, let’s call the proper and inherited features of a node — the node collection.

The witness collection is matched against the node collections of the hierarchy, from top to down, starting in the root node. The classification of the witness collection down the hierarchy, generally follows the programming by classification paradigm (Mellish&Reiter, 1993). We say that the witness collection satisfies the restrictions of a node collection of the hierarchy (or is classified under that node) if the features of the node collections represent a subset of the features of the witness collection, therefore if all (name, attributes) pairs of the <tag> declarations of the node, and all (source-tag, source-attribute, target-tag, target-attribute) quadruples of the <ref> declarations of the node, proper and inherited, are part of the witness collection as well. In this way the witness collection “falls” down the hierarchy reaching certain levels, possibly more than just one. Below those levels, the witness collection cannot be classified any more under none of the nodes found there. The set of all down-most nodes the witness collection is classified under forms a superior borderline. In order to fulfill the process, an inferior borderline must also be determined. Two cases are possible: a). there is a set of nodes in the hierarchy which all have as parents the set of nodes on the superior borderline and only these nodes and all the corresponding node collections satisfy the witness collection. Then, the inferior borderline is given by the set of these nodes, and a new node should be included between the superior borderline and the inferior borderline (see figure 6a). b). either there is a set of nodes in the hierarchy which have as parents the set of nodes on the superior borderline and only these nodes, but none of these node collections satisfy the witness collection, or no common child of the superior borderline can be found. Then, the inferior borderline is not defined in the hierarchy and a leaf node is created and placed as child of the nodes belonging to the superior borderline (see figure 6b).

When the classification is completed, the search should continue beyond the nodes placed above the classified node in order to find other nodes that could be in a subsumption relation with the classified node. If such nodes are found, they should be added to the list of parents of the classified node.

5. Concurrent Annotations

Two annotations are called concurrent if they intend to represent the same linguistic phenomenon from different perspectives, therefore possibly resulting in different solutions.

Below we give two examples where concurrent annotations are needed:

a). Same standard, different target documents. Often, in order to validate an annotated corpus, different teams receive the same task and their work is compared. In case of agreement, the common solution is adopted with a high trust. In case of mismatch, either the controversial versions are given to a third judge, who is asked to decide in favour of one of the two solutions, or the subjects are persuaded to negotiate for an agreement. In these cases one would like to compile a unique document that keeps the common annotations, while indicating also the concurrent parts and the corresponding individual markings. In annotation tasks of these kinds it is likely that the agreed parts be significantly larger than the concurrent parts.

b). Different standards and documents. Suppose a corpus has do be syntactically annotated with respect to two distinct linguistic theories. In this case, two standards have to be considered. It is not impossible to imagine a certain research task, for instance comparing a phrase structure and a dependency structure, tempting to check whether a certain clausal constituent is in a given constituency relation within the sentence and in a certain functional dependency with respect to the main verb. It is likely that a pair of two such documents have no identical parts. However, it is also likely that a granularity border exists, up from where the two documents have the same structure and down from where the solutions are different. Then, in order to demonstrate the two approaches over the

---

2 The morpho-syntactic descriptions, for example, use complex attribute-value pairs (as msd="Nomsc"), which can be expanded into a set of elementary features (pos="noun" type="common" gender="masculine" number="singular" case="oblique").
same text, a common layer of annotation should indicate at least this granularity limits (sentence, clause, etc.).

Viewed from the perspective of the hierarchical graph representation, documents conforming to concurrent annotation cannot be combined in a common standard, at least for the reason that the target XML documents would contain crossing markings. However, supposing a unique document is preferable to two different versions (for the reasons exposed above), one should allow for both common and concurrent annotations in the same document. Solutions to accommodate concurrent annotations have been previously proposed (Wit, 2002), (Sperberg-McQueen&Huitfeldt, 1999). In our system we represent concurrency in annotation as follows:

```xml
<concurrent>
  <version id="1" author= etc..> 
    <marking1>text1</marking1>...
    <marking1>text2</marking1>
  </version>
  <version id="2" author= etc..> 
    <marking2>text3</marking2>...
    <marking2>text4</marking2>
  </version>
</concurrent>
```

where text1+text2=text3+text4.

### 6. Operations within the hierarchy

Following the discussion above, our system implements a set of operations, as described in this section.

The **initialize-hierarchy** operation takes a document, headed by a semantic-relations statement, and builds a trivial hierarchy formed by the ROOT node (the empty annotation) and one standard corresponding to the annotation in the document.

The **classify** operation takes an existing hierarchy and a document, headed by a semantic-relations statement, and classifies the document with respect to the hierarchy, as described in section 4. It will end either by naming an existing standard in the hierarchy to which the document fully observes, or by placing a new standard in a certain place within the hierarchy. As such, building of a hierarchy can be done two ways: ad-hoc, by manually declaring it, when there is sufficient a-priori knowledge over a full range of corpus annotations, already existent or to-be-created, as shown in section 2; or corpus-driven, by an initialize-hierarchy command followed by any number of classify commands, when a range of annotated documents are used to inseminate a hierarchy. To note that in this case it is not compulsory for all annotated documents from which the hierarchy is triggered be replica of the same hub document. When annotation conventions are consistent within the collection of documents, different hub documents can be used to incrementally build a hierarchy of annotation standards.

Given a graph of annotation standards and documents annotated corresponding to these standards, all having the same hub document, **merge** and **extract** operations can be defined. A merge combines two documents having identical hubs and corresponding to two distinct nodes of the hierarchy, which are not in a subsumption relation, and produces, on one hand, another node in the hierarchy, descendant of the two input nodes, and, on the other, the

Figure 6: Examples of final classification
corresponding document which contains the union of the annotation tags of the two originating documents. An extract applies the reverse operation, extracting from a document, corresponding to a certain node, a document conforming to one of the node’s ascendants in the hierarchy.

Finally, there are two types of concurrent-checks. One receives a standard name and two XML files, annotated versions of the same hub document, both supposed to observe the standard, and produces a file in which the annotation differences are put in evidence, as described in section 5, example a. The second receives two standard names and two files corresponding to these standards, and produces a difference file, as described in section 5, example b.

7. Conclusions

We described a data structure and a system aimed at facilitating the definition and exploitation of annotation standards over corpora. The system, interpreting the hierarchy definition declarations and implementing the described operations, has been built in Java. It is freely available and can be downloaded from the address: http://consir.info.uaic.ro/~pic/lc.

As further developments, we intend to supplement the described operations with others, which will finally configure a complex environment, provided with a graphical interface, for working with annotated corpora. This environment could include, for instance, visualization of the hierarchy and interactive operations over it, including the deletion of nodes under some restrictions, unification of two hierarchies, cutting of a sub-hierarchy from an existing one, etc. To unify different annotation with identical or close semantics, we also intend to complement the tag and attribute names with a declarative semantic description. The final goal is to provide automatic conversion from an annotation name space to another, when the associated tags are semantically equivalent. This will aim at keeping a strict control over annotation standards, avoiding the proliferation of tag and attribute names.

We have acquired a collection of corpora, all based on George Orwell’s “Nineteen Eighty Four” novel as hub document, in both English and Romanian, on which the program was tested. In particular, a discourse parsing application, at present under development, makes heavy use of the merging operations on a rich hierarchy of standards. Also all resources used for the development of the Romanian WordNet, under the Balkanet and Balkanet-MEC projects (Tufiş, Cristea, Stamou, 2004), have been classified in a unique hierarchy with the described system.

The described system can help efforts oriented towards the standardisation of language resources. To give an example, we intend to describe all resources which have been created or will be created for the Romanian language, and which are deposited on the site of the Consortium for the Romanian Language Technology, in an NLP-dedicated unique hierarchy. Using this hierarchy, each document will be assigned to a node, whose corresponding standard it observes.

Acknowledgements

The research presented in this paper has been partly supported by the EC funded IST-2000-29388 Balkanet project, and the Balkanet-MEC project funded by the Romanian Ministry of Education and Research.

References


