Semgrex-Plus: a tool for automatic dependency-graph rewriting

Fabio Tamburini
FICLIT - University of Bologna
Email: fabio.tamburini@unibo.it

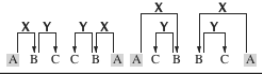
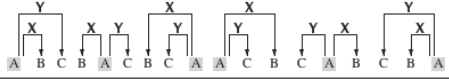
1. Introduction

- We developed the Semgrex-Plus tool an automatic procedure to convert dependency treebanks into different formats.
- It allows for the definition of formal rules for rewriting dependencies and token tags as well as an algorithm for treebank rewriting able to avoid rule interference during the conversion process.
- We extended the behaviour of the Stanford Semgrex tool adding some new functionalities for automatic dependency-graph rewriting useful for treebank maintenance, revision and conversion, producing a new publicly available tool.

2. The Stanford Semgrex Search Language

- it represents nodes as a (non-recursive) attribute-value matrix;
- it uses regular expressions for subsets of attribute values;
- it specifies a complex set of relations between nodes:

Symbol	Meaning
{}=1	Generic node without any attribute with ID='1'
{tag:W}=2	Generic node with attribute tag='W' and with ID='2'
A <reln=X B	A is the dep. of a rel. reln (with ID='X') with B
A >reln=X B	A is the gov. of a rel. reln (with ID='X') with B
...	

Semgrex search pattern	Retrieved subgraphs
{A} >X ({B} >Y {C})	
{A} >X {B} >Y {C}	
{D} >Z ({A} >X {B} >Y {C})	 ... See previous example to build all retrieved subgraphs.

3. Semgrex-Plus

- We defined pairs of patterns: the first is used to search into the treebank for the required subgraphs, and the second is used to specify how the retrieved subsgraphs have to be rewritten.
- For example the pattern pair
  "{tag:det}=1 >arg=A {tag:noun}=2" →
  "{tag:ART}=1 <DET=A {tag:NN}=2",
  what we called a '*Semgrex-Plus rule*', changes the direction of the dependency and, at the same time, changes the words tags and relation label.
- The starting '*search*' pattern and final '*rewrite*' pattern have to contain the same number of nodes and dependency edges. Node and relation naming in Semgrex has been the fundamental trick to introduce such extension.

4. Semgrex-Plus: Rule Application Procedure

We decouple the search and rewrite operations for the rule application defining a set of new rewriting operations on a general dependency treebank:

- DEL_REL(graphID, depID, headID): deletes a dependency edge;
- INS_REL(graphID, depID, headID, label): inserts a new labelled edge;
- REN_TAG(graphID, nodeID, tag): replace the tag of a specific graph node.

The conversion task has been implemented as a three-steps process:

1. each Semgrex-Plus rule is always applied to the original treebank producing a set of matching subgraphs that have to be rewritten;
2. for each match, a set of specific operations for rewriting the subgraph corresponding to the processed matching are generated and stored;
3. this set is sorted by graphID, duplicates removed and every operation is applied graph by graph: first DEL_REL, then INS_REL and REN_TAG.

5. Semgrex-Plus: Rule Overlap/Interference Checking

The tool first identifies which edges in the search pattern are modified by the rewrite pattern of each rule. An edge is modified if:

- the relation is modified;
- one of its nodes is modified by an attribute change.

Then, if the intersection of modified edges is not empty and

a) the two search patterns completely match, then we have a *full overlap* between rules and this mark a problem.

b) the two search patterns do not completely match, then we got a *partial overlap* between rules and this is a potential problem.