Towards the Cross-Roads of MWE Identification and Tree Correction [WG 4-2-3]

Agata Savary
Université François Rabelais Tours
Laboratoire d'informatique
3 place Jean-Jaurès, 41000 Blois, France
agata.savary@univ-tours.fr

1 Variability of MWEs

Linguistic variability is among the major properties of MWEs. It can appear on different levels: (i) orthographic (*to see the col**o**r of sb's money* → *to see the col**ou**r of sb's money*), (ii) morphological (*image converter**s**, image conver**sion*** → *image converter*), (iii) syntactic (*the beens have been spilled* → *to spill the beans*), (iv) lexical semantic ((FR) *se **fourrer** le doigt dans l'oeil* → *se **mettre** le doigt dans l'oeil* 'to put one's finger in one's eye').

2 Identifying MWEs in syntax tree – problem statement

When lexical resources of MWEs are available, one of the challenges is to be able to identify their occurrences in syntax trees despite their variability with respect to the base forms. For contiguous MWEs, a possible solution is to generate extensional lexicons enumerating all possible variants, as in Multiflex (Savary, 2009), and search for these variants straightforwardly on the leaf level of a corpus. However, when a non contiguous, especially verbal, MWE is concerned, all its grammatically correct instantiations correspond to a possibly infinite set of syntactic subtrees (due to admitting unconstrained nominal group complements, adverbial modifiers, etc.).

Our intuition is that the problem of identifying MWEs in syntax trees could be modeled as an instance of the tree-to-language correction problem, briefly introduced in the following section.

3 Tree-to-tree and tree-to-language correction

Several applications in computer science use the idea of *proximity between trees*. First, an application-dependent set of elementary edit operations on tree nodes or subtrees is defined, and a real non-negative cost is assigned to each operation. Then, for two trees t_1 and t_2 to be compared,

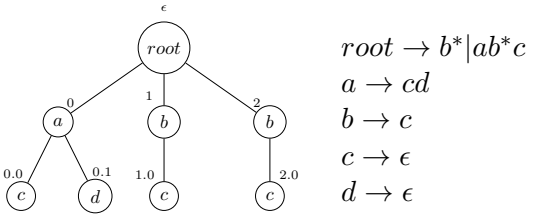

Figure 1: A tree t and a structure description (equivalent to a DTD) defining a set of XML trees

one looks for sequences of edit operations allowing to transform t_1 into t_2. The distance between t_1 and t_2 is said to be the minimal cost of such edit sequences, i.e. $dist(t, t') = min_{t \xrightarrow{seq} t'} cost(seq)$

For instance, suppose that the elementary edit operations on trees are relabeling a tree node, inserting a leaf or deleting a leaf, and that each of these operations has cost 1. Then, the minimal edit sequence transforming the tree t in Fig. 1 into the first tree in Fig. 2 consists of renaming the node a at position 0 into b and deleting the node d at position 0.1. The cost of this sequence, i.e. the distance between the two trees is equal to 2.

Consequently, one can also express the *proximity between a tree and a tree language*, i.e. a (possibly infinite) set of trees. Namely, the distance between a tree t and a tree language L is the minimal distance between t and any tree in L:
$DIST(t, L) = \min_{t' \in L}\{dist(t, t')\}$

Given these notions, one of the possible definitions of the tree-to-language correction problem (Amavi et al., 2013) is, for a tree t, a tree language L and a non-negative threshold th, to find all trees in L whose distance from t is no higher than th. For instance, when considering the domain of XML trees, if t is the tree in Fig. 1, L is the set of trees defined by the structure description (equivalent to a DTD) in Fig. 1, and th is 2, then the set of all trees in L whose distance from t is no higher than th is given in Fig. 2, and $DIST(t, L) = 1$ (because the distance from t to

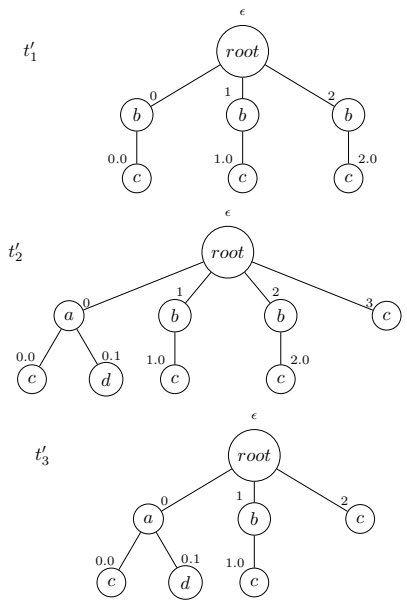

Figure 2: Three correction trees

t'_2 is 1).

4 MWE identification as a tree-to-tree correction problem

In order to model the MWE identification as an instance of the tree-to-tree correction problem, we propose to view each MWE as a tree or a family of trees (one per variant), following e.g. the LTAG formalism (Abeillé and Schabes, 1989).

For instance, the multi-word adverb *at once* could be represented as the (auxiliary) tree $[[[at]_{Prep}[once]_{Adv}]_{AdvP}[]_S]_S$. A MWE that admits no lexical or syntactic transformations (in particular has a fixed order) but admits inflexion could be represented as a tree decorated by feature structures (e.g. to impose agreement). For instance (FR) *mémoire(s) vive(s)* (lit. live memory = random access memory) could be represented as its (initial) tree $[[]_{Det}[memoire]_N[vive]_{Adj}]_{NP}$ with feature structures imposing agreement in number and gender among the 3 components. Finally, a syntactically-flexible MWE, could be assigned a tree family, with one tree per syntactic variant.

A particular occurrence of a MWE in a syntax tree would also be seen as a syntactic subtree. In this context, identifying MWEs could be modeled as tree-to-tree correction in that each syntax tree fragment whose leaves satisfy the minimal lexical constraints for a particular MWE would be corrected with respect to the tree of this MWE.

In order for this modeling to be feasible, it is necessary to select which operations on syntax trees are seen as elementary and which costs they are assigned. When the LTAG point of view is adopted, substitution and adjunction could be seen as elementary operations with cost 0. Then, each other operation, which makes sense in terms of tree rewriting but not necessarily in terms of valid syntactic transformations, would have a non-negative cost related. For instance, relabeling a node would have cost 1 (or less in case of synonyms or orthographic variants in terminal nodes), inserting or deleting a subtree at syntactically non-allowed positions would have the cost equal to the size of the subtree, etc.

5 MWE identification as a tree-to-language correction problem

Alternatively, the MWE identification could be seen as an instance of the tree-to-language correction problem. Each MWE e would then be represented as a tree language L_e. Thus, *at once* could correspond to the set of all trees that result from its auxiliary tree (see Section 4) by its adjunction to any other tree. Similarly, *mémoire(s) vive(s)* would be assigned the set of trees resulting from any substitution at node S in its initial tree (additionally to any valid instantiation of its feature structures). Finally, for a syntactically-flexible MWE, its language would contain its tree family together with all transformations of each tree belonging to this family obtained through a non-prohibited substitution or adjunction.

In this second model, all elementary operations on trees underlying the tree distance definition would carry a positive cost depending on their nature and on the size of the involved subtrees. This model would probably also be more precise as it could integrate various variability restrictions to be expressed at the level of a particular MWE (e.g. allowing no internal modifiers). This model, however, may require highly expressive tree description formalisms, which may lead to very costly tree-to-language correction algorithms.

6 Applications

If the above definition of the MWE identification proves correct and operational, then it could be applied to several MWE-related tasks, notably post-annotating MWEs in treebanks, and detecting MWEs in a post-parsing stage. With a simi-

larity threshold equal to 0, fully grammatical occurrences only would be recognized. With a small positive threshold, partial (but not too huge) ungrammaticality would be allowed, which may help process noisy data, e.g. in spontaneous speech, social networks, etc. Admitting a small threshold might also help detect errors in corpus annotation, in the grammar underlying a parsing process, or in the MWE lexicon itself.

References

Anne Abeillé and Yves Schabes. 1989. Parsing idioms in lexicalized tags. In Harold L. Somers and Mary McGee Wood, editors, *Proceedings of the 4th Conference of the European Chapter of the ACL, EACL'89, Manchester*, pages 1–9. The Association for Computer Linguistics.

Joshua Amavi, Béatrice Bouchou, and Agata Savary. 2013. On Correcting XML Documents with Respect to a Schema. *The Computer Journal*. preprint: `http://www.info.univ-tours.fr/~savary/English/papersASgb.html#TheComputerJournal`.

Agata Savary. 2009. Multiflex: A Multilingual Finite-State Tool for Multi-Word Units. In Sebastian Maneth, editor, *Implementation and Application of Automata*, volume 5642 of *Lecture Notes in Computer Science*, pages 237–240. Springer Berlin Heidelberg. preprint: `http://www.info.univ-tours.fr/~savary/English/papersASgb.html#CIAA09`.