Short Scientific Report

COST-STSM-IC1207-35297

Jakub Waszczuk

Host institution: Abteilung für Computerlinguistik, Institut für Sprache und Information, Heinrich-Heine-Universität Düsseldorf

Period: 2016-11-16 to 2016-11-23

1. Purpose

The purpose of the visit was two-fold: (i) to obtain from the local TAG and parsing experts feedback on the parsing strategy of promoting MWEs and the corresponding evaluation experiments (Waszczuk *et al.* 2016b), and (ii) to work on the specification of the requirements that ParTAGe, a parser for TAGs working on compressed, finite-state grammar representations (Waszczuk *et al.* 2016a), would need to satisfy in order to be compatible with XMG-generated TAG grammars (Petitjean *et al.* 2016). A peripheral goal of the mission was to elaborate the workplan related to the publication on the lexical encoding formats for MWEs, planned as a chapter for the WG2 volume.

2. Work carried out

The first day of the STSM was dedicated to the parsing strategy of promoting MWEs and the publication on the lexical encoding formats for multiword expressions. Together with Timm Lichte, Simon Petitjean, and Agata Savary, we worked in the morning on the workplan related to the planned publication. At the moment of writing this report, the pre-final version of our chapter proposal has been submitted to the PARSEME WG2 volume.

The strategy of promoting MWEs was presented, jointly with Agata Savary, during the Computational Linguistics Research Colloquium in the afternoon. Fruitful discussions with the members of the local team followed after the presentation. During the second day of the STSM we met again with Laura Kallmeyer and Andreas van Cranenburgh to continue the discussions related to the proposed MWE-promoting strategy. A more detailed description of the feedback we obtained from the local team can be found in Sec. 3.1.

The remaining part of the STSM was dedicated to the ParTAGe/XMG compatibility issues. As a result of several meetings with Simon Petitjean we identified, as the main issue, the fact that ParTAGe does not support feature structures, which often play an important role in concrete (in particular, XMG-generated) TAG grammars. We proposed a modified set of inference rules supporting unification-like computations, which were subsequently (partly during the mission, partly afterwards) implemented in ParTAGe. More details about this topic can be found in Sec. 3.2.

3. Main results

This section contains the main results of the scientific mission: a report summarizing the valuable feedback we obtained from the team in Düsseldorf regarding the proposed MWE promoting strategy (see Sec 3.1.), as well as a short description of the work carried out to increase the compatibility between ParTAGe and XMG (see Sec. 3.2.).

3.1. Promoting MWEs in A* TAG parsing

The method of promoting MWEs based on A* parsing is designed to work both in symbolic and probabilistic setting. However, the experiments we performed – evaluation of parsing speed gains and the influence on accuracy – are more relevant in the probabilistic setting where disambiguation of syntactic analyses is performed, which is also the case of our experiments. Even if what we propose is only a partial disambiguation based on MWEs, it can still influence the accuracy of the parser, thus the influence of the strategy on parsing accuracy in a fully probabilistic setting (where only the most probable analysis is proposed by the parser per sentence) should be eventually considered.

Another issue touched on during discussions was related to the fact that our experiments were carried out on a grammar which, being automatically extracted from the treebank, is not necessarily of the highest quality. Besides, the underlying assumption of our experiments was that the grammar has a 100% coverage, since MWE promotion can only positively influence parsing results for sentences which are covered by the grammar. While this seems to be a valid argument, it is clear that the outcomes of the experiments would be more reliable and informative if performed on a real-sized and high-quality grammar.

We discussed several different ideas which would allow, to varying degrees, to overcome the two issues mentioned above. One method would be to use probabilistic supertagging in order to obtain probability estimations for the individual elementary trees (ETs). Probably the easiest way to perform such supertagging would be to use relative frequencies of ETs (with what frequency a given ET is attached to its terminal anchor amongst all the ETs with the same terminal anchor). Alternatively, a sequential model (HMM, CRF) could be used to obtain marginal probabilities of using different supertags (ETs) on the individual positions of the input sentence. Apart from their relative simplicity, the supertagging methods guarantee that the weights assigned to the individual elementary trees (ETs) are non-negative.

Another strategy would be to use discriminative methods to model probabilities of derivations. While a discriminative method would allow us to assign (sentence-independent) weights to individual grammar ETs, it is not clear how such weights could be used in A* parsing, given that they are not guaranteed to be non-negative.

The methods described above could be used to estimate probabilities assigned to simplex ETs. It is not clear to what extent they would be able to handle MWEs. The main issue here is the relative sparcity of MWEs – while in overall MWEs are quite frequent, most of the particular MWEs are rare. Thus, the easiest solution would be to stick to the current strategy and promote MWE-based derivations over the compositional ones systematically, with weights assigned to simplex ETs allowing to choose between the different derivations with the highest number of MWE ETs. However, we also discussed methods which would allow to obtain probability estimations for MWE ETs. One such method, which would allow to bypass the issue of sparcity of MWEs in a discriminative setting, would be to group MWE ETs in classes and then to estimate one parameter per class. For example, each MWE ET could be represented by a sequence of its POS tags, and then the weight assigned to an ET would be the same for all MWE ETs with the same sequence of POS tags.

Yet another solution, which would allow us to bypass the problem of lacking resources and data sparcity, would be to bootstrap a treebank based on a PCFG extracted from Składnica, a Polish treebank with manually disambiguated constituency trees (Świdziński and Woliński 2010). Such a PCFG grammar could be then used to syntactically annotate a large portion of data (e.g., the 1-million part of the National Corpus of Polish) over which similar experiments to those described by Waszczuk *et al.* (2016b) could be carried out. This method would allow us to work on a significantly bigger treebank and thus test the method of MWE promotion on a larger scale. On the other hand, the errors in the syntactic annotation of the treebank would probably cause errors in the mapping procedure and many MWEs would not be identified. Moreover, the quality of the extracted TAG grammar would diminish significantly, thus decreasing the linguistic relevance of the MWE promotion experiment.

Finally, we discussed another idea which would potentially allow us to calculate the idiomaticity rate of MWEs and the validity of our MWE-promoting heuristic on a large scale. Based on the MWE resources we have, we could annotate the potential occurrences of the individual MWEs in a large corpus, in accordance with the assumptions adopted by the heuristic itself. Namely, the heuristic asserts that, if all the words belonging to the given MWE occur in the given sentence, then this MWE should be used in the syntactic analysis of this sentence. By annotating MWEs in a large corpus this way and subsequently verifying the resulting annotations, we should be able get some estimations and ideas about which MWEs should be actually promoted and to what extent.

3.2. XMG/ParTAGe compatibility issues

As already mentioned before, we identified the fact that ParTAGe does not provide any support for feature structures (FSs) as the main issue hindering the usage of ParTAGe with XMG-generated grammars. In particular, typed feature structures can be used within the context of XMG to implement semantic frames (Lichte and Petitjean 2015).

One possible way to add a support for FSs would be to handle them in the post-processing phase of syntactic parsing, a solution implemented in TuLiPa, a parsing environment which employs Range Concatenation Grammar as a pivot formalism and which can be used to parse with several mild context-sensitive formalisms, in particular with TAGs (Parmentier *et al.* 2008). On the other end of the spectrum would be a solution where FSs are not only handled during parsing, but also common parts of different FSs are shared between the individual trees of the grammar. Such a solution would extend the mechanism of sharing common parts of ETs implemented in ParTAGe.

We have chosen a solution which is somewhere in the middle between the two solutions mentioned above. Namely, FSs are processed during parsing, but no sharing of common FSs parts is performed. Structurally, FSs can be seen as graphs, and thus implementing the sharing functionality would be certainly more difficult for them than for elementary trees. Moreover, this decision allows to abstract away from the particularities and low-level details of FSs and to focus on what they represent from the computational point of view – that is, unification-like computations over derivation trees, computations which can possibly fail.

We are aware of no work which would tackle the problem of handling FSs on-the-fly within the scope of tree-rewriting systems similar to TAGs. As already mentioned, TuLiPa handles them in post-processing, but there are also advantages of handling them during parsing. Notably, if we consider probabilistic A* parsing, then handling FSs after parsing may lead to the rejection of the most-probable parse(s) found by the A* algorithm due to potential unification failures over the corresponding FSs. This undesirable situation is avoided when unification of FSs is resolved on-the-fly.

As mentioned above, we decided to abstract over the particularities of FSs and to adopt a point of view where unification of a fully recognized elementary tree (together with the FSs attached to its individual nodes) is represented by a function ω[1] which takes the FS-like values computed for the dependent elementary trees, attached through the operations of substitution and adjunction to its individual nodes, and returns the FS-like value computed for the entire tree, or fails. This function can be represented by the following type:

$$\omega : T(V_\perp) \to V_\perp,$$

where:

- V is the set of FS-like values,
- $X_\perp = X \cup \{\perp\}$, and
- $T(X)$ is the set of (rose) trees with nodes labeled by the values from set X.

Using $V_\perp$ means that the corresponding value may be undefined, which in case of the argument $T(V_\perp)$ of ω means that to some internal nodes no values need to be attached (which is natural since adjunction is not obligatory by default)[2], while in case of the result it means that the unification operation may fail.

We also assume that a plain unification function $\uplus : V \to V \to V_\perp$ is available, which is motivated by the fact that to an internal node several

[1] Or rather, a family of functions ω_t, a separate function assigned to each ET t in the grammar.

[2] An alternative solution would be to assume that a neutral element $e \in V$ exists such that $e \uplus v = v \uplus e = v$ for each $v \in V$, and assign it to all non-modified internal nodes. Then the type of ω would be $T(V) \to V_\perp$. An empty FS provides such a neutral element in case of FSs. An advantage of this alternative solution is that non-modified nodes are not distinguished from nodes modified by empty FSs, which seems like a reasonable choice.

other trees can adjoin[3]. In this case we unify the values corresponding to the individual trees adjoined at the same site before the computation ω corresponding to the ET is performed.

ParTAGe requires a total order to be defined over chart items. This allows to speed up the search of the corresponding chart items when the individual inference rules of the parser are considered. In order to satisfy this requirement, we assume that a total order is also defined over the set V. Thanks to this assumption, we can add information about the computed values to the corresponding chart items without worrying about the resulting total order for chart items, which is derived by the compiler automatically.

It is worth noting that even when such an abstract view of unification over TAG derivations is adopted, it could be feasible to adapt a deductive parsing algorithm to handle unifications without important modifications and, notably, without heavy interference with the inference rules of the parser. However, we were not able to think of any efficient and clear way of doing this within the context of A* parsing, thus we decided to adapt the inference rules themselves to handle FS-like values. While computationally costly, this way of handling FSs on-the-fly is quite transparent and should facilitate reasoning about the properties (correctness, completeness) of the parser.

Appendix A presents the full set of inference rules adapted to handle FSs. They have been implemented and integrated with ParTAGe in a dedicated development branch available at `https://github.com/kawu/partage/tree/simple`. The main modifications introduced in the FS-aware version of the parser are:

- Three types of chart items are distinguished – active, passive and (new) top items. The top items represent fully recognized ETs over which the unification computation has been performed (and, obviously, did not fail). Passive items, on the other hand, can also represent fully recognized ETs, but their root nodes can still undergo adjunction and thus their unification computation has not yet been performed.
- A new inference rule (FI, standing for *finalize*), related to the distinction between the passive and the top chart items, has been added. It models the transition from a passive to the corresponding top item and applies the unification computation ω assigned to the corresponding ET. If the computation fails, the corresponding top chart item is not added to the chart.
- A *trace* is added to the individual chart items. It keeps track of the FS-like values computed for the ETs inserted (substituted, adjoined) in place of the already processed non-terminals of the ET represented by the given chart item.[4] This allows to perform the unification computation once a full ET has been matched. The values computed for the dependent trees are all known and represented explicitly in the trace of the corresponding passive item, which is then transformed into a top item by the FI inference rule.

Another difference is that the flat production rules (compressed in a form of a FSA) contain references to nodes in a directed-acyclic-graph (DAG) representation of the grammar, rather than terminals or complex non-terminals adorned with additional indices (Waszczuk *et al.* 2016a). The DAG representation naturally accounts for subtree sharing and simplifies the implementation of the parser. However, this change was introduced before the STSM and was motivated by issues not related to FSs.

4. Confirmation of the successful execution of the STSM

Laura Kallmeyer: I hereby confirm that the work which Jakub Waszczuk describes in the

[3] The constraint that to an internal node at most one ET can adjoin could be also represented by an appropriate set V with the corresponding unification-like function which does not allow multiple adjunctions to a single node.

[4] Each item determines a grammar subtree or a sequence thereof already matched against the item's span, provided that only prefixes are shared in the FSA representation of the grammar. In case of a minimized FSA grammar, this simple correspondence is lost – there may be several FSA paths leading to the state referred from a given active chart item.

report concerning his PARSEME STSM at the University Duesseldorf in November 2016 was indeed completed during his research visit in Duesseldorf. The exchange of ideas and the collaboration concerning TAG-based parsing and TAG-based (meta)grammar implementation with a focus on Multiword Expressions was very fruitful and inspiring and will lead, among others, to a publication on MWE encoding using XMG. The computational linguistics group in Duesseldorf is looking forward to continue working with Jakub on these topics. My thanks to the PARSEME consortium for making his visit possible.

References

Lichte, T. and Petitjean, S. (2015). Implementing semantic frames as typed feature structures with XMG. *Journal of Language Modelling*, **3**(1), 185–228.

Parmentier, Y., Kallmeyer, L., Lichte, T., Maier, W., and Dellert, J. (2008). TuLiPA: A Syntax-Semantics Parsing Environment for Mildly Context-Sensitive Formalisms. In *9th International Workshop on Tree-Adjoining Grammar and Related Formalisms (TAG+9)*, pp. 121–128, Tübingen, Germany.

Petitjean, S., Duchier, D., and Parmentier, Y. (2016). XMG: Describing Description Languages. In *Logical Aspects of Computational Linguistics (LACL) 2016*, Nancy, France.

Waszczuk, J., Savary, A., and Parmentier, Y. (2016a). Enhancing practical TAG parsing efficiency by capturing redundancy. In *Proceedings of the 21st International Conference on Implementation and Application of Automata (CIAA 2016)*, Seoul, South Korea.

Waszczuk, J., Savary, A., and Parmentier, Y. (2016b). Promoting multiword expressions in A* TAG parsing. In *Proceedings of the 26th International Conference on Computational Linguistics (COLING 2016)*. To appear.

Świdziński, M. and Woliński, M. (2010). Towards a bank of constituent parse trees for Polish. In P. Sojka, A. Horák, I. Kopeček, and K. Pala, eds., *Text, Speech and Dialogue: 13th International Conference, TSD 2010, Brno, Czech Republic*, pp. 197–204, Heidelberg. Springer-Verlag.

A Inference Rules

We first introduce the symbols and functions on which the inference rules are based. Their meaning is, in vast majority, the same as in (Waszczuk *et al.* 2016a), which provides much more detailed explanations.

Let $s = s_0 s_1 \dots s_{n-1}$ be the input sentence and $Pos(s) = \{0, \dots, n\}$ the set of positions between the words in s, before s_0 and after s_{n-1}. Let each word of the sentence be a pair (t, x) where t is a terminal and x is the corresponding FS-like value. Let also $[]$ be an empty list, $x : xs$ be a list with head x and tail xs, (x, ts) be a tree with non-terminal x in its root (possibly $x = \bot$) and subtrees ts, $rev(xs)$ be a function which reverses the given list xs, $tree(r)$ be the ET represented by rule r, $leaf(x)$ be a function which determines the DAG non-terminal leaf containing label x, $foot(x)$ be the DAG non-terminal foot node with non-terminal x, $root(x)$ be a predicate which tells whether x is a root of the underlying DAG or not, and $label(r)$ be the non-terminal assigned to DAG node r. Let $\delta(q, x)$ be a transition function of the underlying FSA representation of the grammar.

We define three types of items:

- A **top item** is a tuple $(v, x, i, j, k, l)_T$ where v is the non-terminal assigned to a root of the corresponding ET, x is the FS-like value computed for the corresponding derivation, and the remaining values i, j, k, l represent its span.

- A **passive item** is a tuple $(r, t, i, j, k, l)_P$ where r represents an internal node of the DAG representation of the grammar, t is a tree of FS-like values computed for the dependented ETs, and values i, j, k, l represent the span.

- An **active item** is a tuple $(q, ts, i, j, k, l)_A$ where q is a state in the FSA representation of the grammar (such a state represents one or more dotted production rules), ts is a list of trees of FS-like values computed for the dependent ETs attached to the already matched part of the tree represented by state q, and values i, j, k, l represent the span.

Table 1 presents the full set of inference rules of the parser adapted to handle unification of FS-like structures.

AX:	$\dfrac{}{(q_0,[],i,-,-,i)_A}$	$i{\in}Pos(s)\backslash\{n\}$	PS:	$\dfrac{(q,ts,i,j,k,l)_A \quad (r,t,l,-,-,l')_P}{(\delta(q,r),t{:}ts,i,j,k,l')_A}$	$\delta(q,r)$ defined
SC:	$\dfrac{(q,ts,i,j,k,l)_A}{(\delta(q,s_l),(x,[]){:}ts,i,j,k,l{+}1)_A}$	$(t,x){=}s_l$ $\delta(q,t)$ defined	SU:	$\dfrac{(q,ts,i,j,k,l)_A \quad (x,v,l,-,-,l')_T}{(\delta(q,r),(v,[]){:}ts,i,j,k,l')_A}$	$r{=}leaf(x)$ $\delta(q,r)$ defined
DE:	$\dfrac{(q,ts,i,j,k,l)_A}{(r,(\bot,rev(ts)),i,j,k,l)_P}$	$r{\in}heads(q)$	FA:	$\dfrac{(q,ts,i,-,-,l)_A \quad (r,t,l,j,k,l')_P}{(\delta(q,r'),(\bot,[]),i,l,l',l')_A}$	$r'{=}foot(label(r))$ $\delta(q,r')$ defined $root(r) \Longrightarrow (j,k){\neq}(-,-)$
FI:	$\dfrac{(r,t,i,j,k,l)_P}{(label(r),x,i,j,k,l)_T}$	$root(r)$ $x{=}\omega_{tree(r)}(t)$ $x{\neq}\bot$	IA:	$\dfrac{(q,ts,i,-,-,l)_A \quad (r,t,l,j,k,l')_P}{(\delta(q,r),t{:}ts,i,j,k,l')_A}$	$\delta(q,r)$ defined $(j,k){\neq}(-,-)$
RA:	$\dfrac{(x,v,i,j,k,l)_T \quad (r,t,j,j',k',k)_P}{(r,(u,ts),i,j',k',l)_P}$			$label(r){=}x, \quad root(r) \Longrightarrow (j,k){\neq}(-,-),$ $(w,ts){=}t, \quad u{=}$if $w{\neq}\bot$ then $v{\uplus}w$ else $v, \quad u{\neq}\bot$	

Table 1: Inference rules of the parser adapted to handle feature structures