Shared Semantic Dependency Representations for LTAG

Kim Gerdes
TALANA
Université Paris 7
F-75251 Paris cedex 05
gerdes@linguist.jussieu.fr

Patrice Lopez
DFKI GmbH
Stuhlsatzenhausweg 3
D-66123 Saarbrücken
lopez@dfki.de

1 Introduction

This work addresses the problem of shared representation of semantic dependency ambiguities relevant for post-parsing process. We do not consider here probabilistic approximation but a compact representation of all ambiguous semantic dependency representations obtained on the basis of grammatical valid rule applications for a given sentence. Since the syntactic level does not provide enough information for the disambiguation of all parses, one solution is to maintain this ambiguity for the semantic and contextual analysis.

Considering such a choice, factorizing all possible parses with a shared representation is a necessity for two main reasons:

- A shared structure can be obtained more efficiently (in polynomial time) than an enumeration of all derivations (exponential time).
- The sharing of sub-results is relevant for post-parsing techniques because the semantic processing of shared sub-derivation could often be shared and consequently improve computational efficiency.

In addition, some basic Machine Translation tool for two close languages, for example, might not want to analyze any further and tempt to reproduce the ambiguities in the target language.

We propose first a new shared representation encoding all linguistic ambiguities that could be expected at the end of a parsing process, thus ensuring modularity of the analysis. The semantic module will not have to go back again to the surface. The particularity of this representation is to combine two kinds of structures. The first one, called *shared bubble*, is inspired by the bubble tree formalism presented in [Kahane, 1997]. Its role is to represent in a unique structure all scope ambiguities of modifiers. The second structure is a more classical shared semantic dependencies graph representing all argumental relation ambiguities and linking the shared bubbles. The combination of this two representations is called *shared bubble graph*.

We propose to associate the shared bubble graph representation to the parses obtained with Lexicalized Tree Grammars which localize semantic dependencies. We take here the example of the LTAG formalism. We present:

- How to obtain multiple dependency relations (encountered for example with equi-verbs see (5)) by the way of features.
- How to obtain the shared bubble graph representation from a shared item forest resulting from the classical tabular parsing algorithms.

Finally we will present how to obtain an enumeration of the different logical forms from a shared bubble graph.

2 Adequate Shared Semantic Representations

2.1 Various problems

2.1.1 Multiple modifiers

To warm up let us first consider a simple unambiguous example with two intersective adjectives:

(1) *a warm flat beer.*

The best corresponding semantic dependency representation, ignoring the determiner, is indisputably α_1

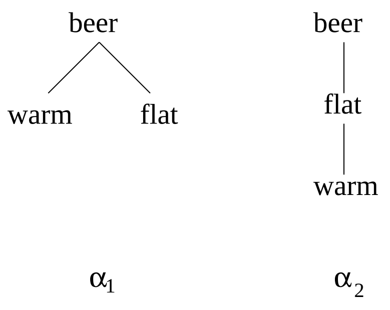

Figure 1: Dependency trees for multiple modification.

α_2, which is the derivation tree obtained in classical LTAG, seems less appropriate, though not totally wrong, since we don't know in which way *warm* influences *flat* (See also [Schabes and Shieber, 1994]). However, without a head, α_1 literally falls apart, as in the following example.

2.1.2 Head Ellipsis

(2) *Une seule de ces personnes.*
Only one of these people

(2) is an example of an every day life phrase with a head ellipsis and two modifiers. In German and French such ellipses are very frequent. Since the equivalent of such ellipses[1] in English are often one-anaphora, tree representations are often proposed as general representation for dependency representations, even for normalized reference representations (see for example [Carroll et al., 1999a] which focuses on heads for tree bank annotations). Still such ellipses in German and French raise a representation problem for strict tree representation because the semantic head of the phrases is not present in the sentence. One solution is to introduce "empty nodes" which cause well-known computational and linguistic problems. The second solution is to suppose transcategorization principles to retrieve a tree structure (the last determiner or adjective becomes a noun in the case of such ellipses[2]). At the step of the extraction of a logical form from these parsing, we have to re-consider again the modifier role of the transcategorized adjective. Both solutions are not satisfactory since they suppose extra computation due to the inadequacy of the representation structure.

[1]Ellipsis in a *head-driven* semantic sense, not in a sense that (2) abbreviates *une seule personne de ces personnes*

[2]For the German elliptic noun phrase *warmes schales von vorgetern (the warm and flat one from the day before yesterday)* a dependency structure usually puts *schales* in the head position without convincing linguistic reasons to do so.

2.1.3 Modifier scope ambiguity

(3) *a former professor from Tübingen.*

(3) is an example of an ambiguous scope of the modifier *old*. Depending on the scope of *old*, this example can mean that (a) we refer to a former professor which is still in Tübingen or (b) to an active professor which was in the past in Tübingen. One can represent these two ambiguities by an underspecified dependency tree where only dominance restrictions of the semantemes are given: $profesor \prec$ *from Tübingen*, $professor \prec former$. The risk with this kind of representation is to obtain three possibles dependency trees as shown in figure 2 (the determiner is ignored here), where α_2 and α_3 are equivalent for the scope semantic interpretation (a).

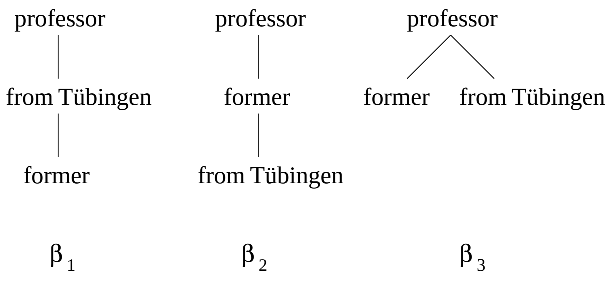

Figure 2: Dependency trees for an ambiguous structure.

This example illustrates that pure underspecification, which is a very classical solution for scope semantics, can result in overgeneration. This article focuses on a more adequate alternative to underspecified dependency tree representation that could avoid such overgeneration.

2.1.4 Sharing structure in case of mutual exclusion scopes

(4) *Le joueur de football américain de Forbach.*
the American football player from Forbach
(the French phrase has an additional third reading where the football comes from Forbach)

(4) supposes that if we speak about an American player, the football does not come from Forbach. Note that in the LTAG compositional processing, the two different ambiguities are correctly produced and avoid the attachment of *Forbach* on *football* if *American* has been attached to *joueur*. We obtain the three derivation trees presented in figure 3 ignoring the determiner.

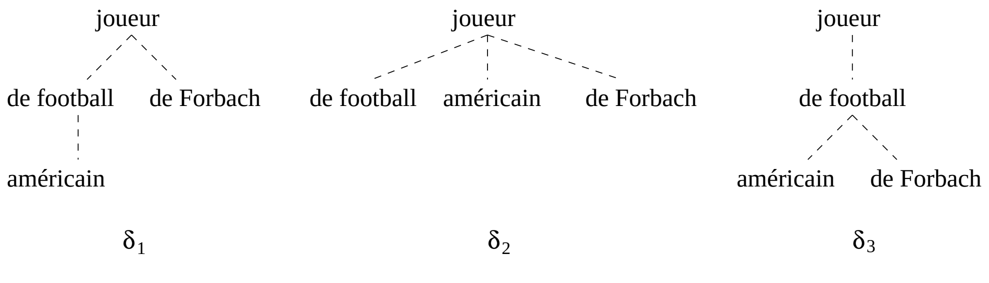

Figure 3: Derivation trees

We can not easily for this example represent all ambiguities in a single sharing structure. There are two solutions to deal with such phenomena :

- Consider two different shared dependency trees (one for the trees δ_1 and δ_2 one for tree δ_3) but it is clear that some sharing is lost.
- Add a special constraint by the way of features for example.

Again both solutions, eventually based on underspecified dependency trees, are not satisfactory since they suppose additional mechanisms. Additionally we would like to draw attention to the fact that, contrary to the similar structure β_1 in figure 2, the adjective *américain* does not modify *joueur*; here, only the football is American. This limited *scope* has to be encoded somewhere else.

2.1.5 Multiple dependencies

Independently of the problems with modifiers, derivation trees in LTAG have the inconvenience that each semanteme introduces only one link when it is added to the derivation, excluding double dependencies. For control verb, derivation are thus always lacking one of their Θ-relations, in LTAG usually the relation between the actor and the infinitive. For

(5) *John asked to leave yesterday*

we get the derivation trees in figure 4, with the missing link represented with a bold line.

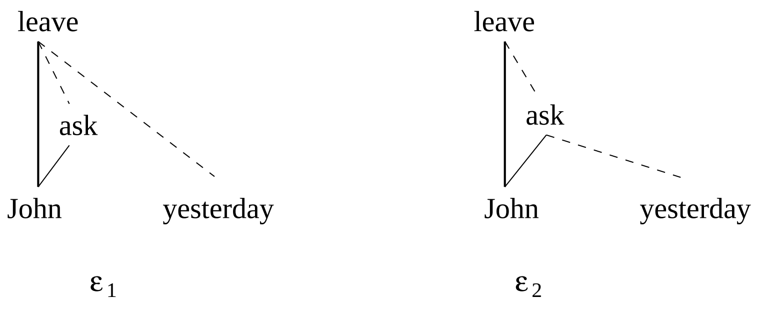

Figure 4: Ambiguous derivation trees.

The question where to adjoin the modifier does not interfere with the dependency structure of the verbs and their actants.

2.2 Shared bubble representation for modifier scope

In order to deal with the previous phenomena, we propose a shared bubble representation to encode in a single structure all ambiguous modifier scopes. For this specific task, we argue that a shared bubble representation is more relevant than shared trees or graphs (unspecified or not).

If we reconsider the example (1), we propose to represent *warm*, *flat*, and *beer* on the same level without hierarchical relation among them as in structure ϕ_1(figure 5)[3].

In exactly the same manner, we can represent a "headless" phrase as example (2) (see γ_2 in figure 6).

The intuitive interpretation of such representations is very simple: Ambiguous phrases can be represented with the help of dotted lines which stand for optional bubbles. A bubble representation of example (3) is presented in γ_1 of the figure 6. The intersective reading of *former* corresponds to the possibility to delete the dotted bubble; the non-intersective reading puts *former* on the same level with the complex bubble *professor from Tübingen.*

[3]To leave the determiner outside is a purely linguistic choice, we could also describe it as a further information on the referential function of the noun phrase as in ϕ_2.

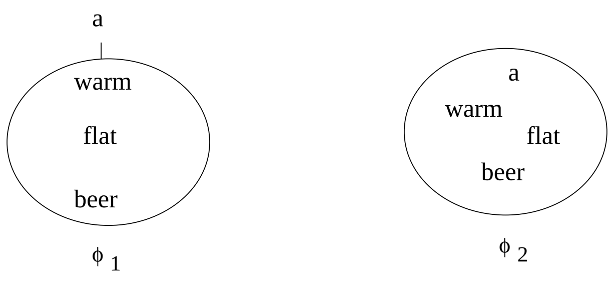

Figure 5: A simple bubble structure

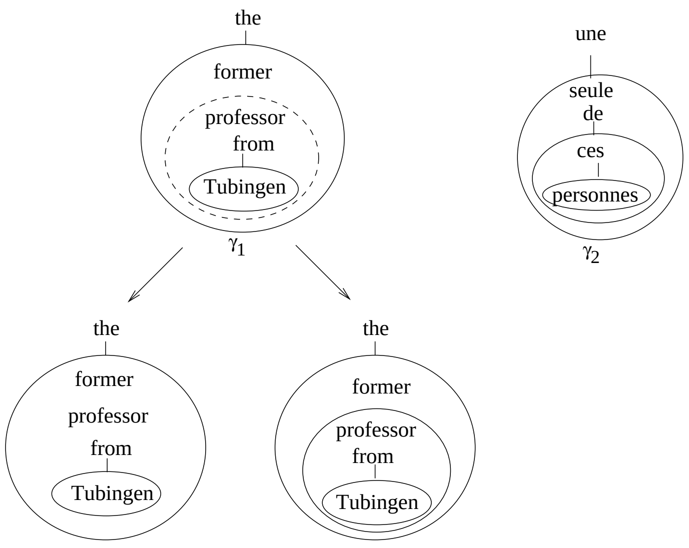

Figure 6: Shared bubble representations.

Some semantemes introduce a new unambiguous scope level (a new bubble in our representation), such a semanteme is called *bubble handle*. Usually, bubble handles are determiners and modifier prepositions.

2.3 Formal definition

Let $S = \{s_1, \ldots, s_n\}$ be the set of semantemes of phrase P on which we have classical shared semantic dependency graph for the predicate argument structure[4] (leaving aside the modification structure of the phrase; the graph is thus not necessarily connected). $L = \{dotted, continous\}$ and $\mathfrak{P}(M)$ is the usual power set of M (the set of all subsets of M). Then we define a *shared bubble* $\mathcal{R}$ of P to be the pair of mappings

$$l : \mathfrak{P}(S) \to L$$

$$h : \mathfrak{P}(S) \to S$$

such that

- $\mathcal{B} := l^{-1}(L) \supseteq h^{-1}(L)$ (if a subset of S has a handle it is a bubble)
- $\forall B, C \in \mathcal{B} :\ h(B) \in C \Rightarrow h(B) \notin B \subsetneq C$ (handle is outside and not separated from its bubble)

[4]Ambiguous argument structure is represented, as usual, by co-indexation of the concerned edges.

we call $h(B)$ the *bubble handle* of B.

We call a shared bubble *simple* if the following conditions hold:

- $l(\mathfrak{P}(S)) = \{continuous\}$ (only continuous lines)
- h is invertible (a handle handles only one bubble)
- for $B, C \in l^{-1}(L) : B \cap C \neq \emptyset \Rightarrow B \subseteq C$ or $C \subseteq B$ (nonempty intersection implies inclusion)

We also call a simple bubble of P together with an unambiguous argument structure a *reading* of P.

For a given phrase P, we say that the shared bubble $\mathcal{R}$ *has a reading* $\mathcal{R}'$ if, with $\mathcal{B} := l^{-1}(L)$ and $\mathcal{B}' := l'^{-1}(L)$

- $l' \subseteq l, l(\mathcal{B} \setminus \mathcal{B}') \subseteq \{dotted\}$ (dotted bubbles can disappear or become continuous)
- $h(\mathcal{B}) = h'(\mathcal{B}')$ (handles don't change)

To explain why these conditions are necessary, let us consider the example (4). We have seen that three derivations and scopes ambiguities are valid in this example (figure 3). Considering the shared bubble representation presented in figure 7, we can obtain only three valid simple bubble representations. The simple bubble representation β_4 does not satisfy the first condition since there are two bubbles open by the same bubble handler *de* (the preposition which introduces *football*). The simple bubble representation β_5 isn't a valid reading of the shared bubble representation either since it introduces a bubble non-existing in the shared representation.
In the following, we will distinguish:

- *Heavy modifier* which always fall in the current bubble (*from Tübingen* for example, corresponding to intersective modifiers).
- *Light modifier* which can create a higher bubble (including the current bubble with a dotted line) because of its corresponding scope ambiguity (*former* for example, corresponding to non-intersective modifiers).

We have shown that this shared bubble representation for modifier relations:

- Is adapted to head ellipses which are frequent in German and French while a tree structure representation which supposes the occurrence of a head in each syntagmatic constituents is not.
- Gives precisely possible modifier scopes while underspecified trees can result in redundant descriptions.

This bubble representation is associated to a shared argumental semantic dependency graph in order to represent all possible ambiguities.

2.4 Shared argumental semantic dependencies graphs

The shared bubble representation only aims to encode modifier dependencies. To represent the argumental semantic dependencies we propose to use a classical dependency graph. The nodes are the shared bubbles (possibly only one bubble, equivalent to noting it without any bubble at all). Each edge is labeled with a Θ-role. The resulting graph is non directed.
To illustrate this representation, we consider the example of an (object-raising) equi-verb from (6) in figure 8. We do not discuss here the list of relevant Θ-roles.

(6) *John persuades the former professor from Tübingen to examine Sandy sincerely*

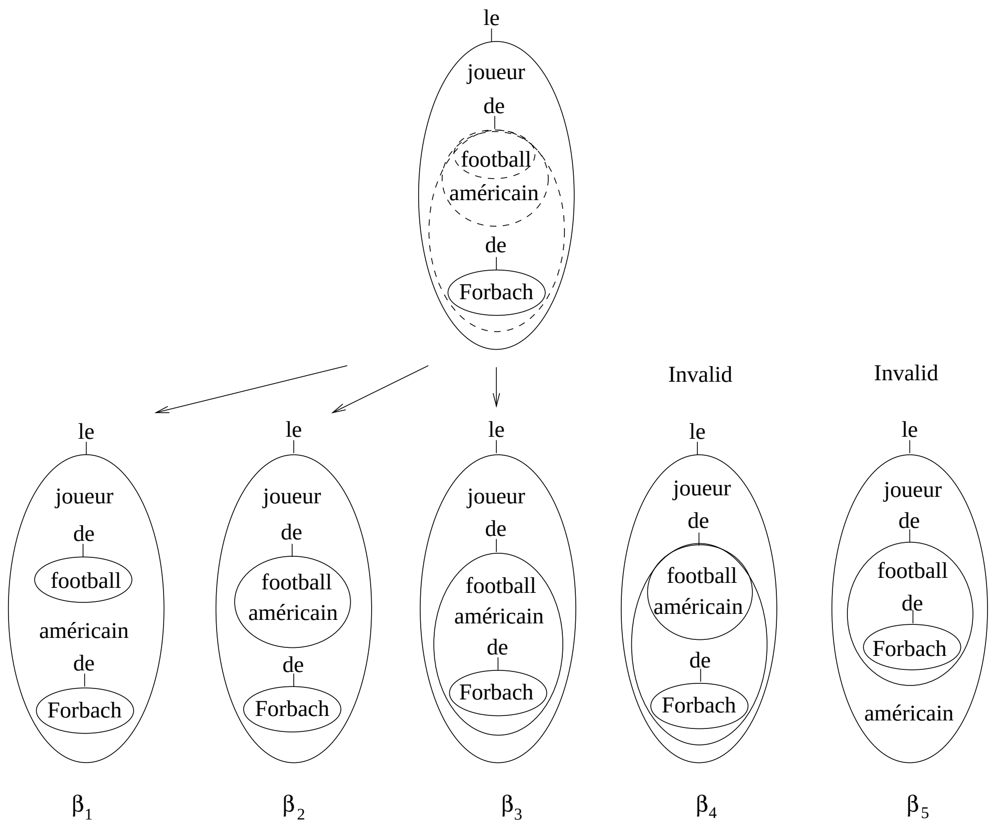

Figure 7: Shared bubble representations.

Since the scope of *sincerely* is ambiguous (on *persuade* or *examine*), we introduce a dotted bubble in order to retrieve correctly the two possible readings. The ambiguous dependency for the *agent* relation of *examine* (the one who is supposed to examine Sandy can be either *John* or *the doctor*) is also represented. The disjunction between the two possible *agent* arguments can obtained by the way of variables which is a classical solution in shared graphs.

3 Extracting shared bubbles and shared graph from LTAG parses

The derivation forest is usually obtained with a partial projection of the shared chart items representation given by a tabular parsing algorithm. We present here how to obtain the two representations introduced before from the shared parse forest resulting from a LTAG grammar which localizes semantic dependencies (for instance a LTAG grammar designed following the principles presented in [Abeillé, 1991] and [Candito, 1999]).

We suppose that the following informations are present in lexicalized elementary trees:

- In the case of auxiliary tree, if the corresponding modifier is heavy or light.
- In each substitution and foot node a specific Θ-role feature indicates the semantic relation which is realized when the corresponding operation is performed.

3.1 How to build Shared bubbles

We suppose here that we use the definition of LTAG derivation proposed by [Vijay-Shanker, 1987] (no multiple-adjunction on the same node). As explained in [Kallmeyer and Joshi, 1999], the

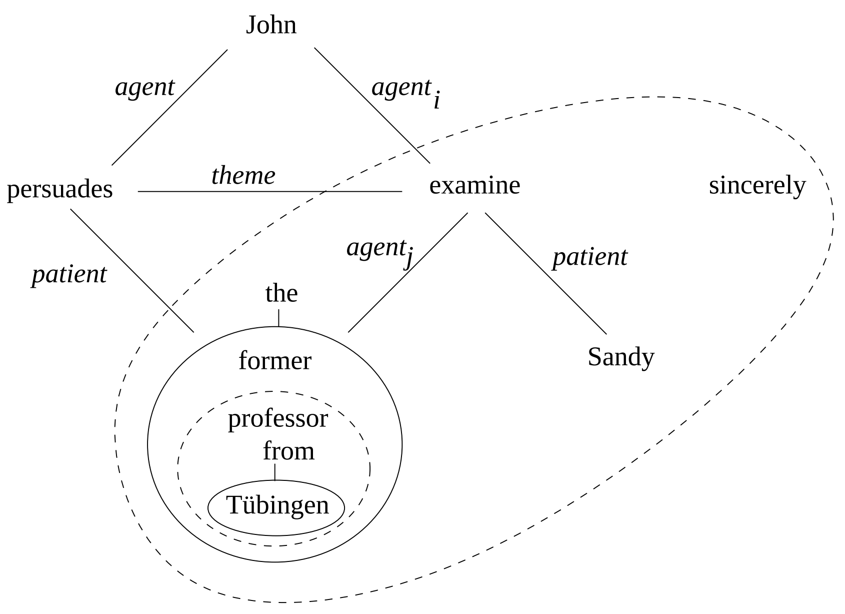

Figure 8: Shared bubble graph representation for the example 6.

definition presented in [Schabes and Shieber, 1994] is not able to capture the scope ambiguity of modifiers. Our solution does not suppose multi-adjunction on the same node for particular modifiers [Kallmeyer and Joshi, 1999] which could lead to extra-processing during the parsing process and additional complexity to design the LTAG parser. Here we just add a lexical information (heavy or light modifier) which will be exploited at the step of the shared dependency structure extraction.

We propose the following sketch, considering a standard bottom-up processing:

- For an initial tree, we create a bubble containing only one element: The semanteme corresponding to this lexical unit.
- If an adjunction of an elementary tree β occurs on the spine of an elementary tree γ, then:
  - if β is *heavy* and γ is not *light* (*heavy* auxiliary tree or initial tree) then add the semanteme corresponding to β in the same bubble as the semanteme for γ.
  - if β is *heavy* and γ is *light* then add the semanteme corresponding to β in the bubble contained by the bubble corresponding to γ (if any, else in the same bubble as the one corresponding to γ.)
  - if β is *light*, then create a new bubble containing the semanteme corresponding to β and the bubble corresponding to γ with a dotted frontier.
- If an adjunction of an elementary tree β occurs on an elementary tree γ but not on its spine, then there is no ambiguous scope: Create a new bubble containing the semanteme corresponding to γ with the semanteme corresponding to β as handler, then replace γ by the β and the new bubble.

3.2 Building shared semantic graphs

Multi-dependent semantemes can not be obtained by a classical LTAG: Only one dependency per combination of trees can be given because of the tree structure of the derivation tree. For instance, all dependency relations can not be given for raising verbs (8) and equi-verbs (7) (here object control verbs).

(7) *John persuades the doctor to examine Sandy.*

(8) *John believes the doctor to examine Sandy.*

Semantic dependency relations which are not localized in the tree structure of elementary trees are obtained with the classical two steps unification processing [Vijay-Shanker, 1987]. The cost of this mechanism is important since it supposes percolation and consequently values of variables which depend on the current derivation (which is possibly exponential in n being the length of the string to parse). But we argue that all phenomena that could not be localized in the elementary tree structure need such a mechanism. For instance, since syntactic dependencies are not localized in classical LTAG [Carroll et al., 1999b], the subject-verb agreement need this two-step unification. On the contrary since semantic dependencies are localized in the tree structure of LTAG elementary trees, the predicate-argument constraints are fulfilled straightforwardly. Consequently the retrieval of semantic dependency relations which are not localized in the tree structure of elementary tree with a two-step mechanism is justified.

We introduce special features called Θ-features. Each substitution and foot node contains a specific Θ-role feature indicating the semantic relation which is realized when the corresponding operation is performed. In order to retrieve double dependencies, we also introduce δ-features (noted δ-struct) which role is to co-index lexical units for the dependencies which are not localized in tree structures of elementary trees. Θ-features and δ-features are propagated by the way of percolation and the two-step unification mechanism of LTAG as other classical syntactic features.

Raising verbs and equi-verbs indicate by the way of a δ-struct feature with a disjunctive value that two of their arguments can be used in the clausal argument (see tree β in the figure 9).

Elementary trees for the main verbs used in such a context have a special node for the subject argument in order to respect the argument-predicate co-occurrence principles defined in [Abeillé, 1991] (see tree β in the figure 9). The empty category ϵ is used to show the extraction of the argument. A δ-struct feature is introduced at this node. The value of this feature will be percolated to the δ-struct feature at the root node of the elementary tree corresponding to the dependent semanteme. The dependency indicated by this co-indexation realizes the semantic relation indicated by the Θ-feature.

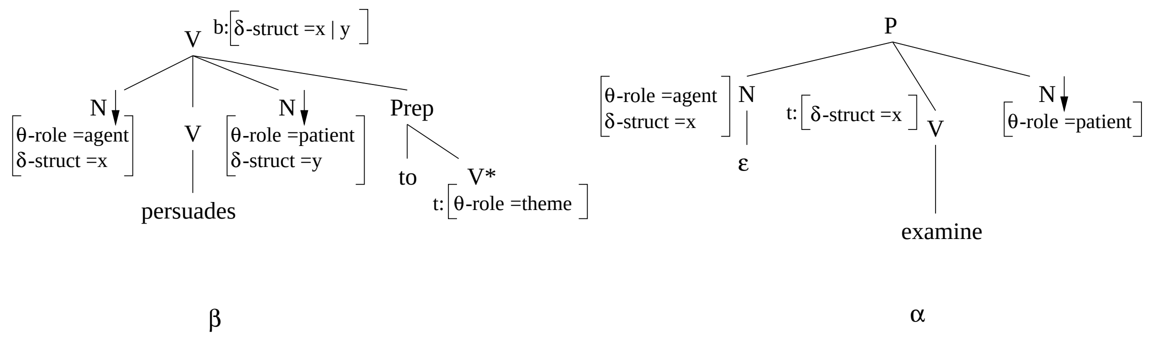

Figure 9: Features for double dependencies.

On the example figure 9 with the sentence 7, we obtain at the end of the combination process a co-indexation between the node for the *agent* relation of the main verb (tree α) and the root node of the initial tree for either *John* or *doctor*. This disjunctive co-indexation reflects the two possible dependencies for the *agent* relation of the main verb.

4 Recovering logical forms

4.1 Principle

The proposed shared representation does not imply a particular semantic post-processing formalism. One can map easily this representation to conceptual graph representations. We will present here how to obtain a logical form in second order logic from a shared bubble graph $\mathcal{R}$ for each reading of $\mathcal{R}$. Depending on the needs of the semantic or pragmatic modules, this recovery can be done only partly, leaving some parts of the graph in its compact shared state.

Recursive recovery of the logical form(s) corresponding to a reading, with e an arbitrary natural number (counter to introduce new variables):

- Let $\{s_1, s_2, \ldots, s_k\} = B \in \mathcal{B}$ be an interior bubble (i.e. $\forall C \in \mathcal{B} \wedge C \subseteq B \Rightarrow C = B$; this can also be a simple semanteme for which we usually don't draw the bubble around it).
- If the bubble consists only of logical forms connected by the argument dependency, combine the logical forms as usually done for dependency structures. If the order between the outside quantifiers matters, one obtains correspondingly many forms.
- If B has no handle, replace B by $t_e : s_1(x_e) \wedge s_2(x_e) \wedge \ldots \wedge s_k(x_e)$
- If B has a determiner handle h_e with the corresponding quantifier Q_e (e.g.$\exists$), replace $B \cup h_e$ by $Q_e x_e : s_1(x_e) \wedge s_2(x_e) \wedge \ldots \wedge s_k(x)$. Increase e by 1.
- For other handles, one proceeds equivalently, corresponding to the sense of the handle-semanteme. (This can be a hard bit of calculus, depending on the chosen logical representation. See the examples.)
- Repeat with the modified bubble representation until the reading has disappeared ($\mathcal{B} = \emptyset$)

4.2 Examples

For our simple example ϕ_1 of figure 1, with only one step, we get $\exists x : warm(x) \wedge flat(x) \wedge beer(x)$. For the right hand reading of γ_1 in figure 2 (the professor is no longer in Tübingen), much depends on the difficult logical representation of *from*: We start with the inner bubble *Tübingen* with its handle *from*. Let's say that the desired logical form for this PP is $from(Tübingen)(x)$. In the next bubble we have the semanteme *professor* together with the logical form just obtained. We put them together to $y : professor(x) \wedge from(Tübingen)(x)$. The next bubble contains *former* and it is handled by *the*. We put all together to $\exists z : former(z) \wedge (y : professor(x) \wedge from(Tübingen)(x))(z)$. This might not be satisfying as logical form already for the reason that we don't distinguish between defined and undefined determiners, but it resembles the structure of Minimal Recursion Semantics. The form has to depend on the semantic calculus used. As an example of multiple logical forms obtained lets briefly look at the classical example for ambiguous quantifier scope, which gives figure 10 as a shared bubble graph:

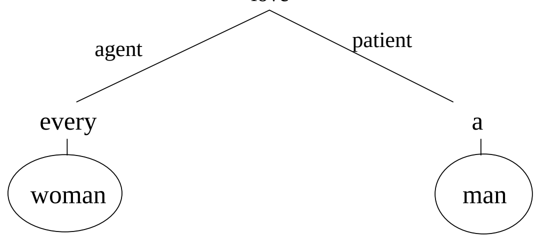

Figure 10: Every woman loves a man.

Since at the last step, the order of the two quantifiers $\forall x : woman(x)$ and $\exists y : man(y)$ is not specified, they can be put in two orders: $\forall x \exists y : woman(x) \wedge man(y) \wedge love(x,y)$ and $\exists y \forall x : woman(x) \wedge man(y) \wedge love(x,y)$. We're aware of the fact that this is not true for all languages; the direct translation of this phrase into German for example is arguably non-ambiguous, and the order of the involved quantifiers is an information which has to be attached to the *love*-tree.
Last but not least, we want to draw attention to the fact that island constraints are well observed: The bubble graph for

(9) *Every woman who loves a man hates a linguist.*

(figure 11) excludes the undefined determiner *a* de *man* from competing with the other quantifiers, since it is inside the bubble handled by *every*.

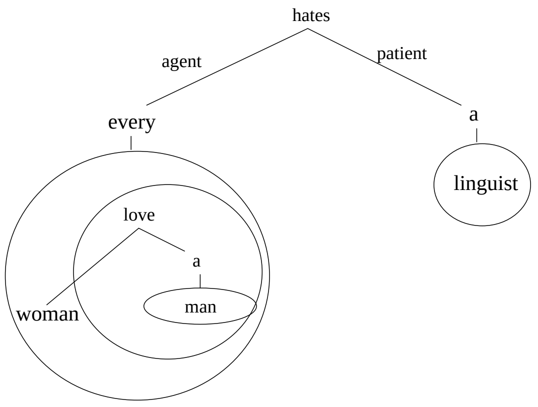

Figure 11: Island constraint

This recovery of the logical form from the shared bubble graph would have to be developed in more detail for a given usage. What counts here is that this is possible without going back to the surface, and that it can be done only partially, which preserves the compact representation as long as possible.

5 Comparison with existing works

[Joshi and Vijay-Shanker, 1999] presents a different approach based on a MRS-like semantic representation associated to elementary trees. This semantic representations are combined on the basis of the derivation tree and deals with scope and quantifier ambiguities.

We have tried to exploit here that a LTAG localizes semantic dependencies and that consequently an elementary tree can be view as a basic unit for some computational semantics processing. Rather than introducing a new level of semantic representation as in [Joshi and Vijay-Shanker, 1999] or [Kallmeyer and Joshi, 1999],we are directly enriching elementary tree in order to produce efficiently a shared representation for all semantic ambiguities that could then be exploited in a straightforward way in a particular semantics formalism. We don't try to remodel the usual TAG-derivation tree into a useful semantic representation as for example [Kallmeyer and Joshi, 1999], but we create directly a different, more precise graph during the derivation, relying on additional, though very simple features that have to be added to the elementary trees. As we hope to have shown, the usual derivation tree can't encode all relevant information and at the latest when treating more difficult, in particular non-English phenomena, all approaches have to add information somewhere. We believe that all relevant information can be encoded in a shared bubble graph, and we demonstrate this with several problems (head ellipsis, mutual exclusion of dependencies, overgeneration of underspecified tree representation) ignored by previous similar works.

6 Conclusion

This new representation presents significant characteristics compared with existing shared representations:

- This representation gives in a single structure every dependencies that could be expected from a deep syntactic analysis. All semantic relations and all possible modifier scopes (which are often neglected) are given.
- The shared bubble representation is more precise than a representation based on underspecified dependency trees. For instance with (3), the predicative role of the adjective *old* is ambiguous. In the kind of underspecified tree representation proposed by [Kallmeyer and Joshi, 1999] for this example, two possible dependency trees would be redundant. Moreover, in the case of mutual exclusion dependencies as in (4), it is useless to add additional constraints or to consider two different underspecified trees.
- Our proposal is also well adapted to constituents with empty head (see (2)) which are frequent in French or German and problematic with a tree representation.
- Our proposal does not suppose a particular formalism of representation for the semantic processing (both [Kallmeyer and Joshi, 1999] and[Joshi and Vijay-Shanker, 1999] introduce predicative logic mechanisms).

We have proven the practical tractability of this representation by:

- Describing how to extract shared bubble graphs from classical chart items forest obtained at the end of the parsing process.
- Describing how to obtain easily a logical form from this representation.
- Factorizing in the graph all elementary trees that correspond to the same semanteme (which corresponds to taking into account the semantic consistency principle in the derivation tree).

Future work should explore the linguistic pertinence of the separation of predicate argument structure and modifying structure, which is partly inherited from the distinction between initial and auxiliary trees in LTAG. Furthermore we hope to apply this proposal concretely:

- The proposed representation should be tested on more complex representations for a specific LTAG grammar, since everything depends on the occurring elementary trees.
- The recovery of a logical form stays quite vague for the moment since we didn't want to decide on a specific semantic formalism. This should be tried out.
- In this proposal we are mainly concerned with parsing. Since shared bubble graphs are structures a bit more complicated than pure derivation trees (which can be seen as TAG-building-instructions), we want to find out whether the generation starting with a shared bubble graph involves other difficulties.
- The usefulness of the shared bubble graph by itself, before extracting other representations, should be explored further, for example for Machine Translation by linking shared bubble graphs of two languages.

References

[Abeillé, 1991] Abeillé, A. (1991). *Une grammaire lexicalisée d'arbres adjoints pour le français.* PhD thesis, Université Paris 7.

[Candito, 1999] Candito, M.-H. (1999). *Structuration d'une grammaire LTAG : application au français et à l'italien.* PhD thesis, University of Paris 7.

[Carroll et al., 1999a] Carroll, J., Minnen, G., and Briscoe, T. (1999a). Corpus annotation for parser evaluation. In *EACL'99 Workshop on Linguistically Interpreted corpora (LINC-99), Bergen, Norway.*

[Carroll et al., 1999b] Carroll, J., Nicolov, N., Shaumyan, O., Smets, M., and Weir, D. (1999b). Parsing with an extended domain of locality. In *Eighth Conference of the European chapter of the Association for Computational Linguistics (EACL'99), Bergen, Norway.*

[Joshi and Vijay-Shanker, 1999] Joshi, A. K. and Vijay-Shanker, K. (1999). Compositional Semantics with Lexicalized Tree-Adjoining Grammar (LTAG): How much Underspecification is Necessary? In *Proceedings of the third International Workshop on Computational Semantics (IWCS-3), Tilburg.*

[Kahane, 1997] Kahane, S. (1997). Bubble trees and syntactic representations. In *Proceedings of the 5th Meeting of the Mathematics of the Language, DFKI, Saarbrücken.*

[Kallmeyer and Joshi, 1999] Kallmeyer, L. and Joshi, A. (1999). Factoring predicate argument and scope semantics: Underspecified semantics with ltag. In *Proceedings of the 12th Amsterdam Colloquium, December.*

[Schabes and Shieber, 1994] Schabes, Y. and Shieber, S. (1994). An alternative conception of tree-adjoining derivation. *Computational Linguistics*, 20(1):91–124.

[Vijay-Shanker, 1987] Vijay-Shanker, K. (1987). *A Study of Tree Adjoining Grammar.* PhD thesis, University of Pennsylvania, Philadelphia.