Sentences with two subordinate clauses: syntactic and semantic analyses, underspecified semantic representation

Laurence Danlos
TALANA/ LATTICE
Université Paris 7
Laurence.Danlos@linguist.jussieu.fr

Abstract

I show that sentences with two subordinate clauses may receive two syntactic analyses, and that each syntactic analysis may receive two semantic interpretations. Hence, I put forward an underspecified semantic representation such that each syntactic analysis receives only one underspecified interpretation.

1 Introduction

Sentences with two subordinate clauses occur quite often in corpora. Theories and tools in Computational Linguistics are available now which allow us to study such sentences exhaustively, both at the syntactic and semantic level. It is what I intend to do in this paper, while using only well-known techniques.

Several sophisticated theories and discourse processing mechanisms have been designed which put forward a number of *principles*. This study on sentences with two subordinate clauses, which constitute one of the simplest cases of discourses, will question some of these principles (e.g., semantic dependency structures for discourses are tree shaped, discourse structure does not admit crossing structural dependencies). It therefore sheds light on discourse processing in general.

Section 2 focuses on the syntactic analysis of sentences with one or two subordinate clauses, including their linear order variants. The syntactic framework I use is LTAG. I show that sentences with two subordinate clauses may receive two syntactic analyses. Section 3 focuses on the semantic analysis of such sentences. The semantic framework I use is SDRT[1], although I translate the conditions of an SDRS into a dependency graph. I show that sentences with two subordinate clauses may receive four semantic dependency structures. Section 4 studies the mapping between syntax and semantics and shows that each syntactic analysis for sentences with two subordinate clauses receives two semantic interpretations. Hence the need of an underspecified semantic representation (henceforth USR). Section 5 presents this USR. Finally, Section 6 compares this work with D-LTAG (Webber et al., 2003).

[1] SDRT stands for Segmented Discourse Representation Theory (Asher and Lascarides, 2003). It is an extension of DRT, Discourse Representation Theory (Kamp and Reyle, 1993). (S)DRS stands for (Segmented) Discourse Representation Structure.

2 Syntax (in LTAG)

2.1 Sentences with one subordinate clause

Syntactically, subordinate clauses are adjuncts. Therefore in XTAG (XTAG Research Group, 2001) and FTAG (Abeillé et al., 2000), the English and French LTAG grammars, a subordinate conjunction (*Conj*) anchors an auxiliary tree, with two syntactic sentential (clausal) arguments, the foot node for the matrix clause and a substitution node for the subordinate clause.

Both in English and French, a subordinate clause may appear in three different positions relative to the matrix clause: (i) before the matrix clause separated by a punctuation mark (a comma), the linear order is then *Conj$_a$ S2, S1*, (ii) before the VP surrounded by two commas, and (iii) after the matrix clause optionally separated by a punctuation mark, the linear order is then *S1 (,) Conj$_a$ S2*. Therefore, in FTAG, a given subordinating conjunction anchors three auxiliary trees which correspond to these three positions. This is not the case in XTAG, where it anchors four auxiliary trees, two of them for the sentence final position: sentence final adjuncts without comma adjoin at a VP node, while those with a comma adjoin at the root S of the matrix clause. Let us quote (XTAG Research Group, 2001) p. 152 for the former ones. "One compelling argument is based on Binding Condition C effects. As can be seen from examples (1a-c) below, no Binding Condition violation occurs when the adjunct is sentence initial, but the subject of the matrix clause clearly governs the adjunct clause when it is in sentence final position and co-indexation of the pronoun with the subject of the adjunct clause is impossible."

TAG+7: Seventh International Workshop on Tree Adjoining Grammar and Related Formalisms.
May 20-22, 2004, Vancouver, BC, CA.
Pages 140-147.

(1) a. Unless she$_i$ hurries, Mary$_i$ will be late for the meeting.
b. *She$_i$ will be late for the meeting unless Mary$_i$ hurries.
c. Mary$_i$ will be late for the meeting unless she$_i$ hurries.

I agree with the data observed in (1a-c), however my point is that there would be no difference at all if the sentence final adjunct in (1b-c) were separated by a comma. Therefore, I see no reason to lay down two different trees - one adjoining at a VP node, the other one at a S node - for sentence final adjuncts with or without a comma. As in FTAG, I assume only one tree (with an optional comma) for sentence final adjuncts, which adjoins at the root S of the matrix clause. This solution presents the advantage not to rely heavily on the presence or absence of a comma, which is sometimes a matter of taste, as in (1c).

Because of lack of room, I leave aside subordinate clauses which appear before the VP of the matrix clause. To put it in a nutshell, I consider only two auxiliary trees for a subordinating conjunction *Conj*$_a$ according to the linear order: β1(Conj$_a$) when the subordinate clause is postposed to the matrix clause, and β2(Conj$_a$) when it is preposed. Figure 1 shows the auxiliary and derivation trees for postposed and preposed subordinate clauses[2].

2.2 Sentences with two subordinate clauses

A sentence *S1 Conj*$_a$ *S2 Conj*$_b$ *S3* receives two syntactic analyses. In the first one, obtained by recursivity and noted SA1, *Conj*$_a$ *S2* is an adjunct to *S1* and *Conj*$_b$ *S3* an adjunct to *S2*. In the second one, obtained by adjunct iteration and noted SA2, both *Conj*$_a$ *S2* and *Conj*$_b$ *S3* are adjuncts to *S1*. Figure 2 shows the derived and derivation trees for these two analyses. For SA2, the derivation tree is based on multiple adjunctions to the same node, as proposed in (Schabes and Shieber, 1994). These multiple adjunctions are ordered (from left to right). The syntactic ambiguity of sentences *S1 Conj*$_a$ *S2 Conj*$_b$ *S3* is systematic without any comma. On the other hand, sentences *S1 Conj*$_a$ *S2, Conj*$_b$ *S3* with a comma before the second conjunction are preferably analyzed as SA2.

Let us examine the possible variants of the canonical linear order *S1 Conj*$_a$ *S2 Conj*$_b$ *S3*, which corresponds to the case where both *Conj*$_a$ and *Conj*$_b$ anchor a postposed tree. For each analysis, it must be examined what happens (a) when *Conj*$_a$ anchors the preposed tree β2(Conj$_a$) and *Conj*$_b$ the postposed one β1(Conj$_b$), (b) symmetrically, when *Conj*$_a$ anchors β1(Conj$_a$) and *Conj*$_b$ β2(Conj$_b$), (c) and finally when both *Conj*$_a$ and *Conj*$_b$ anchor a preposed tree. Figure 3 shows the linear orders for SA1 and SA2 other than the canonical one.

Consider the syntactic ambiguity issue for these variants. (a2) and (b2) in Figure 3 are both of the form *Conj S, S Conj S.*, with a preposed and a postposed adjunct. Therefore any sentence of this form receives two syntactic analyses and corresponds to two sentences in the canonical order (Section 3.2). The variants (a1) and (c2) are both of the form *Conj S (,) Conj S, S.* The comma before the second conjunction is obligatory in (c2) and nearly forbidden in (a1). Therefore, these forms are nearly unambiguous. The variants (b1) and (c1) correspond to sentences which are syntactically unambiguous.

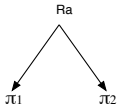
3 Semantics

3.1 Sentences with one subordinate clause

Following works in SDRT, I use an intermediate level of representation to determine the logical form of a discourse (*what is said*). This "semantic" level reflects the discourse structure (*how things are said, how the discourse is rhetorically organized*). This structure plays an important role, e.g., it constrains both anaphora resolution and the attachment of incoming propositions in understanding.

A nice tool for the semantic level is dependency graphs. This is what is adopted in RST[3], but not in SDRT: discourse structures, called SDRSs, are represented as boxes with a Universe and a set of conditions. Nevertheless, it is easy to translate the conditions of an SDRS into a dependency graph (Danlos, 2004). Therefore, while adopting SDRT as a discourse framework, I can use a conventional semantic dependency representation for sentences of the type *S1(,) Conj*$_a$ *S2*. Namely, *Conj*$_a$ denotes a discourse relation R$_a$. R$_a$ is a predicate with two arguments π1 and π2, which correspond to the semantic representations of *S1* and *S2* respectively. These arguments are ordered: π1 precedes π2.

This semantic representation is graphically represented in the DAG besides, also simply written as R$_a(\pi 1, \pi 2)$.



3.2 Semantics for linear order variants

We have seen in Section 2.1 that a subordinate clause can be postposed or preposed. Following works in MTT[4], a trace of the linear order should be recorded in a semantic dependency representation (giving so a piece of information on the communicative structure), however it should

[2] In a derivation tree, a dashed line indicates adjunction, a solid line substitution; each line is labeled by the Gorn address of the argument at which the operation occurs; αi stands for the LTAG tree for *Si*.

[3] RST stands for Rhetorical Structure Theory (Mann and Thompson, 1987). Rhetorical structures correspond roughly to dependency structures.

[4] MTT stands for Meaning to Text Theory, a dependency formalism for sentences (Mel'cuk, 2001).

not affect its dependency structure. From this principle, the position of subordinate clauses should not affect dependency structures: *S1(,) Conj$_a$ S2* and *Conj$_a$ S2, S1* are both represented as $R_a(\pi_1, \pi_2)$ in which π_1 precedes π_2.

What happens for a sentence with two subordinate clauses? Establishing the canonical order with only postposed subordinate clauses may generate ambiguities: for example, a sentence X of the type *Conj$_a$ S1, S2 Conj$_b$ S3*, with a preposed and a postposed adjunct, corresponds in the canonical order either to Y_1 = *S2 Conj$_a$ S1 Conj$_b$ S3* or to Y_2 = *S2 Conj$_b$ S3 Conj$_a$ S1*. X receives two syntactic analyses: either (a2) - Figure 3 - from Y_1 (the first adjunct is preposed), or (b2) from Y_2 (the second adjunct is preposed). These analyses allow us to compute Y_1 and Y_2. From the above principle that the position of subordinate clauses does not affect dependency structures, X does not yield any other DAGs than Y_1 and Y_2. As a consequence, our study on the dependency structures of sentences with two subordinate clauses can be limited to the study of such sentences in the canonical order.

3.3 Sentences with two subordinate clauses

We are going to show that sentences with two subordinate clauses may be interpreted in four different ways. Two interpretations are found in which one conjunction has wide scope over the other one, two other ones without wide scope. The former are represented in tree shaped DAGs, the latter in non tree shaped DAGs.

This semantics study is based on the following compositionality principle. Let $\mathcal{D}_n$ be a DAG with n leaves representing the dependency structure of a discourse D_n. If $\mathcal{D}_p$ is a sub-DAG of $\mathcal{D}_n$ with p leaves, $1 < p < n$, then the discourse D_p corresponding to $\mathcal{D}_p$ should be inferable from D_n[5].

A) Wide scope of *Conj$_a$*: The wide scope of *Conn$_a$* = *because* in (2a) can be seen in the dialogue in (2b-c) in which the answer is *Because S2 Conn$_b$ S3*[6]. The semantic dependency structure of (2a) is DAG (A) in Figure 4. In this DAG, which is tree shaped, the dependency relations must be interpreted in the standard way used in mathematics or computer science: the second argument of R_a is its right daughter, i.e. the tree rooted at R_b which is the semantic representation of *S2 Conj$_b$ S3*. This reflects the fact that *Conj$_a$* has wide scope and is in conformity with our compositionality principle: (A) includes the sub-DAG $R_b(\pi_2, \pi_3)$ and *S2 Conn$_b$ S3* can be inferred, i.e. (2a) is true, then it is true that *Fred played tuba while Mary was taking a nap*.

(2) a. Mary is in a bad mood because Fred played tuba WHILE she was taking a nap.
b. - Why is Mary in a bad mood?
c. - Because Fred played tuba WHILE she was taking a nap.

When *while* is not stressed, the question in (2b) may be given as answer only *Because S2*. The interpretation of (2a) corresponds then to DAG (C) presented below. (2a) when written could be considered as ambiguous with a scope ambiguity of *because*. The scope of *because* is underspecified in the USR proposed in Section 5 for (2a).

B) Wide scope of *Conj$_b$*: The wide scope of *Conn$_b$* = *in order that/to* in (3a) can be seen in the dialogue in (3b-c) in which the question is *Why S1 Conj$_b$ S2?* The semantic dependency structure of (3a) is DAG (B) in Figure 4. This tree shaped DAG must be interpreted in a way similar to (A), which reflects that *Conj$_b$* has wide scope.

(3) a. Fred played tuba WHILE Mary was taking a nap in order to bother her.
b. - Why did Fred play tuba WHILE Mary was taking a nap?
c. - In order to bother her.

As for (2a), (3a) when written could be considered as ambiguous. The scope of *in order that/to* is underspecified in the USR proposed in Section 5 for (3a).

C) S2 factorized: The clause S2 in (4a) is said to be factorized since both *S1 Conj$_a$ S2 = Fred played tuba while Mary was washing her hair* and *S2 Conj$_b$ S3 = Mary was washing her hair before getting dressed for her party* can be inferred from (4a). A similar situation is observed in (4b).

(4) a. Fred played tuba while Mary was washing her hair before getting dressed for her party.
b. Fred was in a foul humor because he hadn't slept well that night because his electric blanket hadn't worked.[7]

In (4a), no conjunction has wide scope over the other one. Its semantic structure is DAG (C) in Figure 4. This DAG is not tree shaped: $\pi 2$ has two parents.

One could argue that tree shaped DAGs (A) and (B) should not be interpreted in the standard way. This is argued in RST in which dependency relations in trees are interpreted with the "nuclearity principle" (Marcu, 1996). With this principle, the arguments of a discourse relation can only be *leaves* of the tree, for example, the second argument of R_a in (A) is $\pi 2$, and the first argument of R_b in (B) is $\pi 1$. This amounts to interpreting (A) as (C), and

[5] On the other hand, the converse principle is not always true (Danlos, 2004): if a sub-discourse D_p can be inferred from D_n, it does not always mean that the DAG $\mathcal{D}_p$ is a sub-DAG of $\mathcal{D}_n$.

[6] To indicate that it is stressed when spoken, the word *while* is written in capital letters in (2).

[7] This discourse is a modified version (including *because*) of an example taken in (Blackburn and Gardent, 1998), who acknowledged that its structure is a "re-entrant graph".

(B) as (D) presented below. But then, cases with wide scope are not represented at all: they are not taken into account, which is unacceptable. As a consequence, tree shaped DAGs must be interpreted in the standard way, in which the arguments of a discourse relation may be either intermediary nodes or leaves.

It is generally assumed that semantic dependency structures for discourses should be tree shaped. As a consequence, to avoid DAGs, some authors use trees in which some predicate-argument relations are given by the nuclearity principle, while others are given by the standard interpretation. Nevertheless, one should not feel free to use trees relying on a mixed interpretation (the standard and nuclearity ones), except if the conditions governing the use of one or the other interpretation can be formally defined. In (Danlos, 2004), I show that no rule can be laid down to choose one of these two interpretations. A mixed interpretation for trees must thus be discarded. Since the standard interpretation is needed for wide scope cases, the nuclearity principle should be discarded. As another consequence, one has to admit that discourse structures for discourses are DAGs.

D) S1 factorized: The clause S1 in (5) is said to be factorized since both *S1 $Conj_a$ S2 = Fred prepared a pizza while it was raining* and *S1 $Conj_b$ S3 = Fred prepared a pizza before taking a walk* can be inferred. The semantic structure of (5) is DAG (D) in Figure 4, which is in conformity with our compositionality principle. This DAG is not tree shaped.

(5) Fred prepared a pizza, while it was raining, before taking a walk.

In discourses analyzed as (D), *S3* is linked to *S1* (which is not adjacent) and not to *S2* (which is adjacent). Therefore, these discourses are counter-examples to the adjacency principle advocated in RST.

DAG (D) exhibits crossing dependencies. It is thus a counter-example to the stipulation made by (Webber et al., 2003), namely "*discourse structure itself does not admit crossing structural dependencies*"[8].

Summary: A sentence with two subordinate clauses may receive one of the four interpretations represented in DAGs (A), (B), (C) and (D). In the next section, we will see that (A) and (C) are the interpretations of the syntactic analysis SA1, while (B) and (D) are those of SA2.

These four interpretations are the only possible ones. In particular, I cannot find any example in which *S3* would be factorized, although I wrote all possible examples I could think of and Laurence Delort, who works on (French) corpus, could not find anyone neither. The factorization of *S3* is represented as DAG (E) in Figure 5. Note that no compositional syntax-semantics rule could lead to (E) from the syntactic analyses SA1 and SA2, which are the only possible ones. More generally, in (Danlos, 2004), I show that any DAG with three ordered leaves other than (A)-(D) is excluded, i.e. does not correspond to coherent discourses with three clauses. For example, DAG (K) in Figure 5 is excluded. This comes from the "letf1-right2 principle", which is a weaker version of the adjacency principle[9].

4 Mapping between syntax and semantics

We are going to examine the interpretation(s) of the syntactic analyses put forward in Section 2. The criterion to be used is that of linear order. So, we are going to examine the linear order(s) for each interpretation (A)-(D).

A) Wide scope of *$Conj_a$*: The linear order variants of (2a), repeated in (6a), are shown in (6b-d).

(6) a. Mary is in a bad mood because Fred played tuba while she was taking a nap.
b. Because Fred played tuba while she was taking a nap, Mary is in a bad mood.
c. Mary is in a bad mood because, while she was taking a nap, Fred played tuba.
d. Because, while she was taking a nap, Fred played tuba, Mary is in a bad mood.

These linear order variants correspond to the variants which are allowed with the first analysis SA1 (see Figure 3). On the other hand, the variants which are allowed with SA2 are forbidden: the discourses in (7) do not make sense (hence the sign #).

(7) a. # Because Fred played tuba, Mary is in a bad mood while she was taking a nap.
b. # While she was taking a nap, Mary is in a bad mood because Fred played tuba.
c. # While she was taking a nap, because Fred played tuba, Mary is in a bad mood.

To conclude, interpretation (A) corresponds to SA1, or conversely SA1 can be interpreted as (A).

B) Wide scope of *$Conj_b$*: The linear order variants of (3a), repeated in (8a), are shown in (8b-d). They correspond to the variants of SA2. I leave it to the reader to

[8] Among discourse connectives, (Webber et al., 2003) distinguish "structural connectives" (e.g. subordinating conjunctions) from discourse adverbials including *then, also, otherwise*. They argue that discourse adverbials do admit crossing of predicate-argument dependencies, while structural connectives do not. I emphasize that (5) comprises only structural connectives (subordinating conjunctions) while its structure exhibits crossing structural dependencies.

[9] Recall that the adjacency principle does not hold because of examples such as (5). In sentences of the type *S1 $Conj_a$ S2 $Conj_b$ S3*, the letf1-right2 principle states that the first (resp. second) argument of a subordinating conjunction is given by a text span which occurs on its left (resp. right). This principle excludes (K) since *S1* is the only text unit on the left of *$Conj_a$*, while π_1 is not the first argument of R_a.

check that the variants of SA1 are forbidden. To conclude, interpretation (B) corresponds to SA2, or conversely SA2 can be interpreted as (B).

(8) a. Fred played tuba while Mary was taking a nap in order to bother her.
b. While Mary was taking a nap, Fred played tuba in order to bother her.
c. In order to bother Mary, Fred played tuba while she was taking a nap.
d. In order to bother Mary, while she was taking a nap, Fred played tuba.

C) S2 factorized and **D) S1 factorized**: When S2 is factorized, the linear order variants correspond to SA1, when S1 is factorized, they correspond to SA2, as the reader can check it.

Summary: A sentence with two subordinate clauses may receive the syntactic analyses SA1 and SA2, SA1 can be interpreted as (A) or (C), SA2 as (B) or (D). In the next section, I put forward underspecified semantic representations (USRs) such that the syntactic analysis SA1 receives only one underspecified semantic representation, USR1, which is specified in (A) or (C), and that SA2 receives only USR2 which is specified in (B) or (D).

5 Underspecified semantic representation

It is now classical to use USRs for quantifier scope ambiguities (among other ambiguities). Following works by (Duchier and Gardent, 2001), I adopt a scope underspecification formalism based on dominance constraints. Let us illustrate the overall idea briefly. The clause *every yogi has a guru* is represented (in a simplified way) as the "tree description" in Figure 6, in which a solid line represents immediate dominance, a dotted line dominance. Quantifier scopes are underspecified in this tree description. The dominance constraints are solved in the trees (a) and (b) in Figure 6 (in both (a) and (b), the root dominates `has(x, y)`). Quantifier scopes are specified: in (a) `forall(x)` has wide scope, in (b) it is `exists(y)`. The USR I propose for subordinate conjunctions follows this overall idea. However, it differs in two ways: (i) "left-dominance" is used instead of dominance, (ii) constraints are solved in DAGs which may be not tree shaped.

Left-dominance: It has been seen in Section 3 that the nuclearity principle is too restrictive (wide scope cases are not taken into account). It will be seen below that dominance relations are not restrictive enough. Therefore, I introduce a new relation, called "left-dominance", which is intermediary and defined as follows.

A node X in a tree left-dominates a node Y iff Y is a daughter of X (immediate dominance) or there exists a daughter Z of X such that Y belongs to the left-frontier of the tree rooted at Z.

As an illustration, R_a left-dominates $\pi 1$, R_b and $\pi 2$ in (A), while R_b left-dominates R_a, $\pi 1$ and $\pi 3$ in (B). Left-dominance is more restrictive than (strict) dominance (e.g. R_a strictly dominates $\pi 1$, R_b, $\pi 2$ and also $\pi 3$ in (A)) and less restrictive than the nuclearity principle (e.g. by this principle, R_a dominates only the leaves $\pi 1$ and $\pi 2$ in (A))[10].

Syntax to semantics: Following works in semantics with LTAG (Candito and Kahane, 1998) (Kallmeyer and Joshi, 2003), I assume that (i) each elementary tree is linked to an (underspecified) semantic representation, (ii) the way the semantic representations combine with each other depends on the derivation tree. I propose the following rule to link the elementary trees of a subordinate conjunction to an USR.

Rule (R1): The USR for $\beta 1(\text{Conj}_a)$ and $\beta 2(\text{Conj}_a)$[11] in which $Conj_a$ denotes a discourse relation R_a is the description of a DAG in which R_a left-dominates $\pi 1$ and $\pi 2$, the semantic representations of the arguments of $Conj_a$. This rule is graphically represented in Figure 7, in which a dashed-dotted line represents left-dominance.

Let us show how rule (R1) allows us to compute the right interpretations for the syntactic analyses SA1 and SA2 depending on their derivation trees.

Interpretations for SA1: From the derivation tree of SA1 given in Figure 2, rule (R1) leads to USR1 given in Figure 8. The constraints on left-dominance and order in USR1 are solved in DAGs (A) and (C). (C) is identical to USR1 except that immediate dominance replaces left-dominance. In (A), R_a left-dominates $\pi 2$. USR1 cannot be solved in (B) since, in (B), R_b dominates $\pi 2$ but does not left-dominate it. On the other hand, in (Duchier and Gardent, 2001) who use dominance, USR1 can be solved in (B), which is not in accordance with the data. This is why I have introduced the notion of left-dominance.

Interpretations for SA2: From the derivation tree of SA2 given in Figure 2, rule (R1) leads to USR2 given in Figure 8. The order of the multiple adjunctions to the same node in SA2 is echoed by the order of the leaves in USR2: $\pi 1$ precedes $\pi 2$ which precedes $\pi 3$. The constraints on left-dominance and order in USR2 are solved in DAGs (B) and (D) - which is correct - but also in (K) given in Figure 5[12]. However, (K) is excluded because it does not follow the left1-right2 principle, see note 9.

To conclude, with (R1), SA1 can be interpreted only as (A) or (C) and SA2 only as (B) or (D), which is correct.

[10] More formally, the nuclearity principle states that, in a (binary) tree rooted at R, the arguments of R are the *leaves* of the tree which are *left-dominated* by R.

[11] Recall (Section 3.2) that linear order does not affect dependency structures. So, $\beta 1(\text{Conj}_a)$ and $\beta 2(\text{Conj}_a)$ are both linked to the same (underspecified) semantic representation.

[12] I thank Laura Kallmeyer for drawing my attention on this point.

6 Comparison with D-LTAG

This study on sentences with two subordinates clauses is extended to other discourses. As requested by the reviewers, let me compare my approach to D-LTAG (Webber et al., 2003), which extends a sentence level grammar, namely XTAG, for discourse processing.

Let us first look at subordinating conjunctions. They anchor *auxiliary* trees in XTAG (or FTAG) and *initial* trees in D-LTAG. Why do they anchor *initial* trees in D-LTAG? The authors give the following answer: "One reason for taking something to be an *initial* tree is that its local dependencies can be stretched long-distance". That is a wrong argument. One major advantage of TAG is that adjunction is possible both in initial and auxiliary trees (and iteratively). So local dependencies in any tree can be stretched long-distance. Moreover, as any other adjuncts, several subordinate clauses can iteratively modify the same matrix clause (Section 2.2). One may wonder how iterativity is taken into account when subordinate conjunctions anchor initial trees.

Secondly, let us examine the distinction made in D-LTAG between structural and anaphoric connectives. The status of some connectives (e.g., *however*) is admittedly not clear and so is determined on empirical grounds, using crossed structural dependencies as a test. In note 8, I have emphasized that (5) comprises only structural connectives - subordinating conjunctions are unquestionably structural connectives in D-LTAG - while its structure exhibits crossing structural dependencies. So the main test to distinguish structural and anaphoric connectives is not valid.

D-LTAG defends the idea that there is no gap between sentence and discourse processing. There exist discrepancies, e.g., discourse adverbials have one argument at the (syntactic) sentence level and two at the (semantic) discourse level[13]. Such discrepancies are handled in the D-LTAG parsing system (Forbes et al., 2002) by the use of two passes: one based on XTAG syntactic trees, the other one on D-LTAG semantic trees. This amounts in positing two levels as in my approach however without a well-defined syntax-semantics interface.

Finally, in D-LTAG, the logical form of a discourse is computed from its derivation tree, a level of representation which is poor compared to SDRSs, e.g., there is no notion of Universe which groups the discourse referents. As said in Section 3.1, a rich semanctic level as the one proposed in SDRT is necessary for text understanding and also for text generation (Danlos et al., 2001). Moreover, the "anaphoric" behavior of discourse adverbials is seriously taken into account in SDRT. Unfortunately, I have no room left to discuss this issue.

[13]There is no such discrepancy for subordinating conjunctions.

References

A. Abeillé, M.-H. Candito, and A. Kinyon. 2000. The current status of FTAG. In *Proceedings of TAG+5*, pages 11–18, Paris.

N. Asher and A. Lascarides. 2003. *Logics of Conversation*. Cambridge University Press, Cambridge.

P. Blackburn and C. Gardent. 1998. A specification language for discourse semantics. In *Proceedings of LACL'98*, pages 61–67, Grenoble.

M.H. Candito and S. Kahane. 1998. Can the TAG Derivation Tree represent a Semantic Graph? In *Proceedings of TAG+4*, pages 25–28, Philadelphia.

L. Danlos, B. Gaiffe, and L. Roussarie. 2001. Document structuring à la SDRT. In *International workshop on text generation - ACL*, pages 94–102, Toulouse.

L. Danlos. 2004. Discourse dependency structures as constrained DAGs. In *Proceedings of SIGDIAL'04*, Boston.

D. Duchier and C. Gardent. 2001. Tree descriptions, contraints and incrementality. In R. Muskens H. Bunt and E. Thijsse, editors, *Computing Meaning*, pages 205–227. Kluwer Academic Publishers, Dordrecht.

K. Forbes, E. Miltsakaki, R. Prasad, A. Sarkar, A. Joshi, and B. Webber. 2002. D-LTAG system: Discourse parsing with a lexicalized tree adjoining grammar. *Journal of Logic, Language and Information*, 12(3):261–279.

L. Kallmeyer and A. Joshi. 2003. Factoring Predicate Argument and Scope Semantics: Underspecified Semantics with LTAG. *Research on Language and Computation*, 1(1–2):3–58.

H. Kamp and U. Reyle. 1993. *From Discourse to Logic*. Kluwer Academic Publishers, Dordrecht.

W. Mann and S. Thompson. 1987. Rhetorical structure theory. In G. Kempen, editor, *Natural Language Generation*, pages 85–95. Martinus Nijhoff Publisher, Dordrecht.

D. Marcu. 1996. Building up rhetorical structure trees. In *The Proceedings of the 13th National Conference on Artificial Intelligence*, pages 1069–1074, Portland.

I. Mel'cuk. 2001. *Communicative Organization in Natural Language: The Semantic-Communicative Structure of Sentences.* John Benjamins Publishing Company, Amsterdam.

Y. Schabes and S. Shieber. 1994. An Alternative Conception of Tree-Adjoining Derivation. *Computational Linguistics*, 20(1):91–124.

B. Webber, A. Joshi, M. Stone, and A. Knott. 2003. Anaphora and discourse structure. *Computational Linguistics*, 44:1–45.

XTAG Research Group. 2001. A lexicalized tree adjoining grammar for english. Technical Report IRCS-01-03, IRCS, University of Pennsylvania.

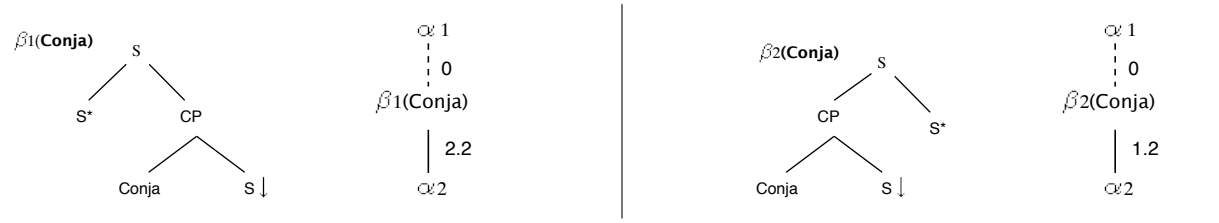


Figure 1: Auxiliary and derivation trees for postposed and preposed subordinate clauses

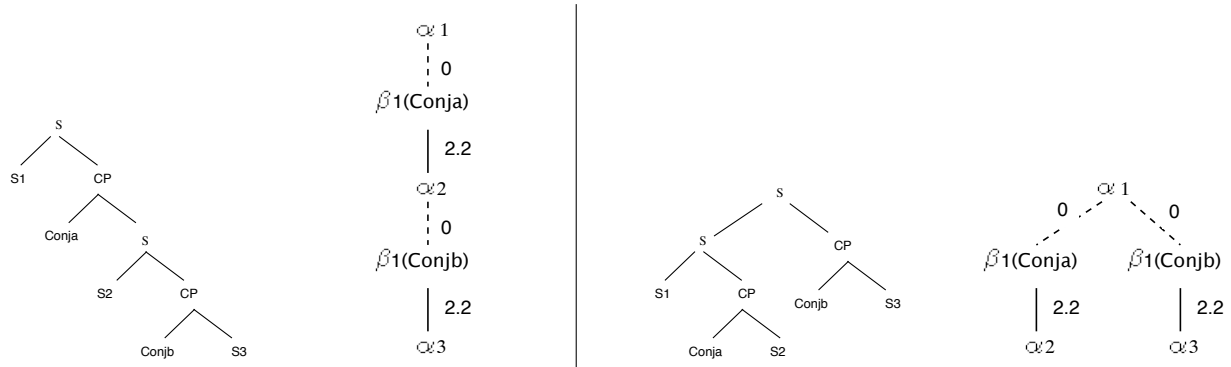


Figure 2: Derived and derivation trees for SA1 and SA2

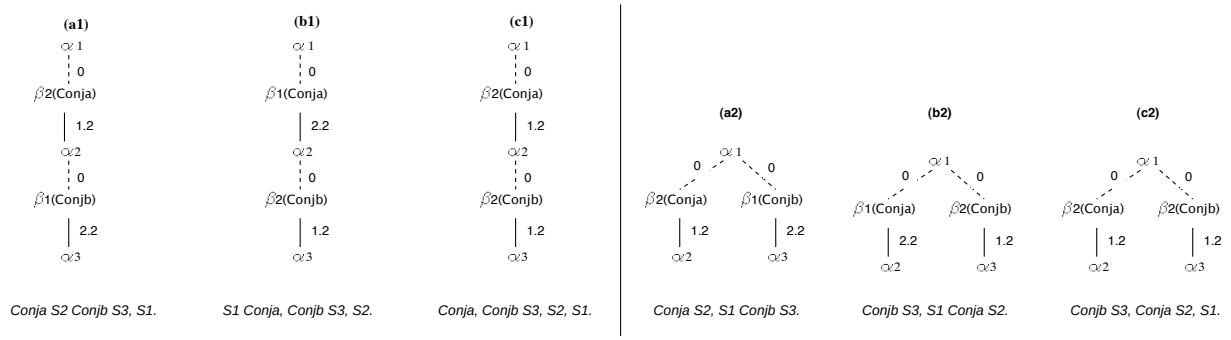


Figure 3: Other linear orders for SA1 and SA2

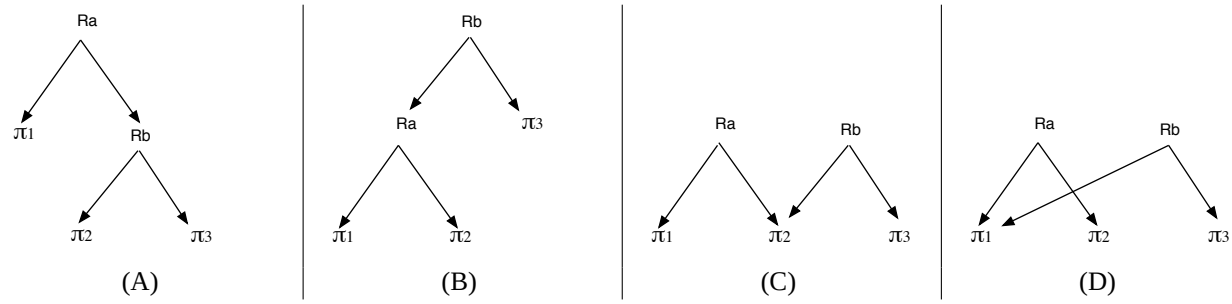


Figure 4: DAGs (A), (B), (C) and (D)

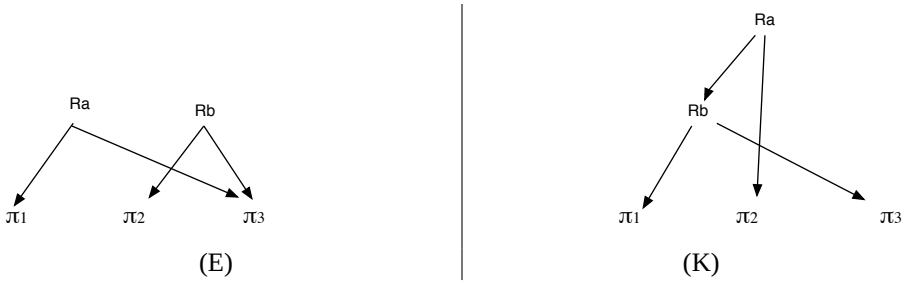


Figure 5: DAGs (E) and (K)

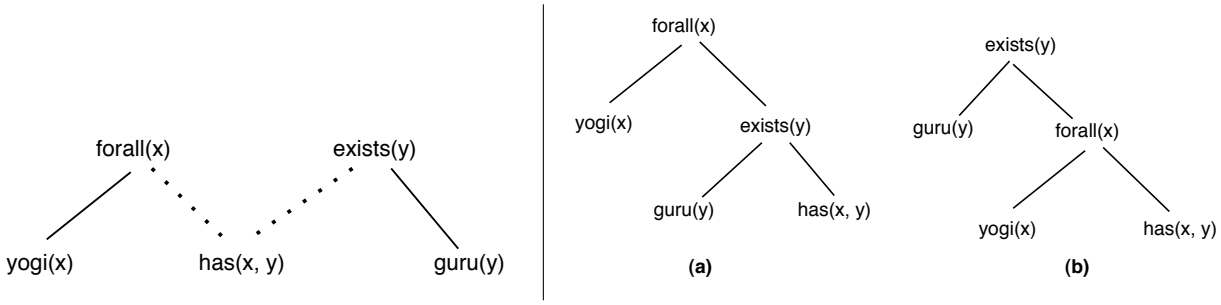


Figure 6: USR for *every yogi has a guru*, and representations with quantifier scope specified

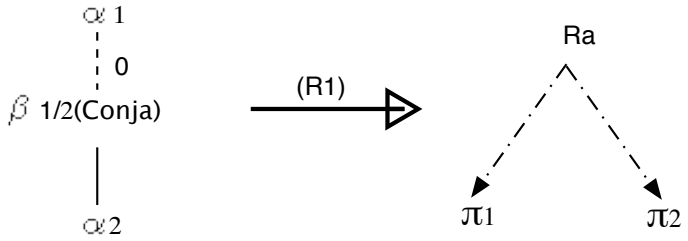


Figure 7: Rule (R1)

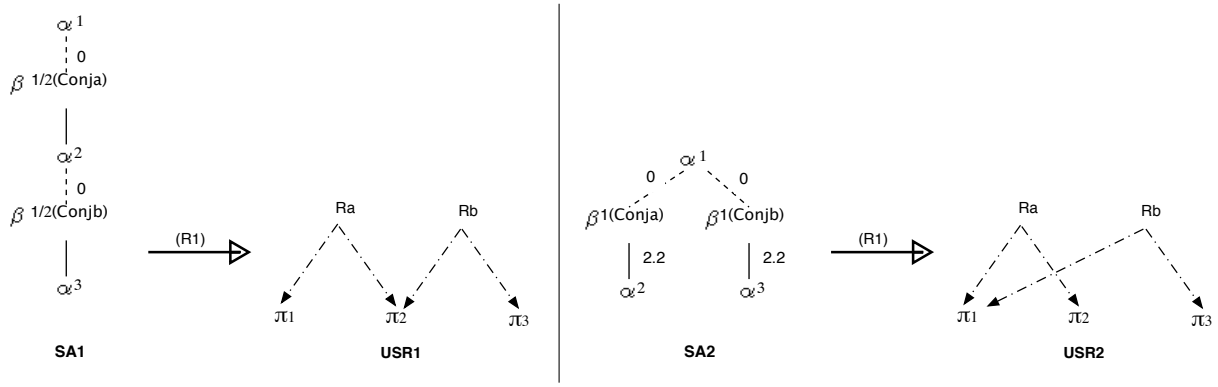


Figure 8: USR1 and USR2 (Underspecified Semantic Representations for SA1 and SA2),