D-STAG: A Formalism for Discourse Analysis Based on SDRT and Using Synchronous TAG

Laurence Danlos

Université Paris Diderot, IUF, Alpage
Laurence.Danlos@linguist.jussieu.fr

Abstract. D-STAG is a new formalism for the automatic analysis of discourse. It computes hierarchical discourse structures annotated with discourse relations. They are compatible with those computed in SDRT. A discursive STAG grammar pairs up trees anchored by discourse connectives with trees anchored by (functors associated with) discourse relations.

1 Introduction

The aim of this paper[1] is to propose a new formalism for the automatic analysis of texts, called D-STAG for Discourse Synchronous TAG. This formalism extends a sentential syntactic and semantic analyzer to the discursive level: a discursive analyzer computes the "discourse structure" of the input text. Discourse structures consist of "discourse relations" (also called "rhetorical relations") that link together discourse segments — or more accurately, the meanings these discourse segments convey. A discourse is coherent just in case every proposition that is introduced in the discourse is rhetorically connected to another bit of information, resulting in a connected structure for the whole discourse.

For the discursive part of our analyzer, we rely on SDRT — Segmented Discourse Representation Theory [1,2]. D-STAG computes discourse structures which are compatible with those produced in SDRT. Therefore, D-STAG can take advantage of the results brought by this discourse theory.

The research done in the framework of SDRT is theory-oriented, providing formally detailed accounts of various phenomena pertaining to discourse. Much less focus has been put on the issue of implementing a robust and efficient discourse analyzer. For this aspect of the work, we have designed a formalism based on TAG – Tree Adjoining Grammar [12]. After being used successfully for syntactic analysis in various languages, TAG has been extended in two directions: moving from sentential syntactic analysis to semantic analysis — with, among others, STAG [21,22,17] —, and moving from the sentence level to the discourse level, both for text generation — with, among others, G-TAG [5] —, and for discourse parsing — with, among others, D-LTAG [10]. The new formalism presented here

[1] A (longer) French version of this paper is published in *Revue TAL*, Volume 50, Numéro 1, 2009, pp 111-143.

P. de Groote, M. Egg, and L. Kallmeyer (Eds.): FG 2009, LNAI 5591, pp. 64–84, 2011.

relies on all this previous work. In particular, its architecture is inspired from that of D-LTAG, with three components:

1. a sentential analyzer, which provides the syntactic and semantic analyses of each sentence in the discourse given as input,
2. a sentence-discourse interface, which is a mandatory component if one wants not to make any change to the sentential analyzer,
3. a discursive analyzer, which computes the discourse structure.

This paper is organized as follows. Section 2 presents the main discursive linguistic data that motivate D-STAG. Section 3 gives an introduction to STAG. Section 4 briefly describes the sentence-discourse interface. Section 5 explains in detail the discursive analyzer. Section 6 compares D-STAG and D-LTAG.

2 Discursive Linguistic Data

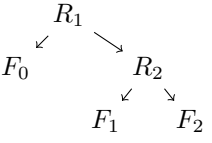
A discourse relation is often explicitly expressed by a "discourse connective". The set of discourse connectives includes subordinating and coordinating conjunctions (*because, or*) and discourse adverbials (*next, therefore*). Connectives can be ambiguous. For example, *then* lexicalizes the relation *Narration* in a narrative (*Fred went to the supermarket. Then, he went to the movies.*) whereas it lexicalizes *Continuation* in an enumeration (... *The second chapter presents a state of the art. Then, the third chapter explains the problematics.*). Discourse relations need not be explicitly marked. For example, the relation *Explanation* in the connective-free discourse *Fred fell. Max tripped him up.* of the form C_0. C_1. must be inferred from (extra)linguistic knowledge. For such a case, we assume the existence of an empty adverbial connective, noted ϵ, following a proposition made in [11]. So the previous discourse is assumed to be of the form C_0. ϵ C_1., and we say, although somewhat inaccurately, that ϵ lexicalizes *Explanation*. In a nutshell, a discourse relation can be considered as a semantic predicate with two arguments which is lexicalized by a discourse connective (possibly empty) with two arguments. The arguments of a discourse relation/connective are the discursive semantic/syntactic representations of the same (continuous) discourse segments. These are the basic principles on which our STAG discursive grammar relies.

One should wonder what discourse structures correspond to when represented as dependency graphs (in which a predicate dominates its arguments). The idea is widespread that dependency graphs representing discourse structures are tree-shaped: this is a basic principle in RST — Rhetorical Structure Theory [14,15] —, a theory on which many text generation or parsing systems have been based for the last twenty years. This is also a principle which guided the conception of D-LTAG. Yet this tree-shaped structure is more a myth than a reality, as shown in [24] and in some of our previous work [6,7]. SDRT discourse structures are not represented as dependency graphs, however, in our two aforementioned papers and in this one, we convert SDRT discourse structures into dependency graphs. These dependency graphs are DAGs — Directed Acyclic Graphs — which are not

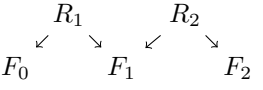
necessarily tree-shaped. However, these DAGs respect strong constraints which rule out a number of DAGs that don't correspond to any discourse structure. We shall justify this claim with discourses of the form C_0 *because* C_1. Adv_2 C_2., in which *because* lexicalizes *Explanation*; C_i symbolizes the ith clause, its logical form is noted F_i. These discourses yield four types of interpretation — but no more than four — which are illustrated in examples (1).[2]

(1) a. Fred is in a bad mood because he lost his keys. Moreover, he failed his exam.
b. Fred is in a bad mood because he didn't sleep well. He had nightmares.
c. Fred went to the supermarket because his fridge is empty. Then, he went to the movies.
d. Fred is upset because his wife is abroad for a week. This shows that he does love her.

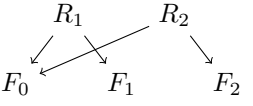
In (1a), Adv_2 = *moreover* lexicalizes the relation *Continuation*. The discourse segment C_1. Adv_1 C_2 forms a complex constituent whose logical form, $Continuation(F_1, F_2)$, is the second argument of *Explanation*. So the discourse structure is $Explanation(F_0, Continuation(F_1, F_2))$, which corresponds to a tree-shaped dependency DAG, see the adjacent figure with R_1 = *Explanation* and R_2 = *Continuation*. In this example, the second discursive argument of the conjunction *because* crosses a sentence boundary.



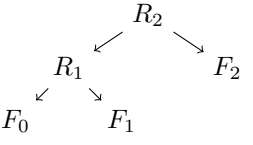
In (1b), Adv_2 = ϵ lexicalizes *Explanation*. The discourse structure is $Explanation(F_0, F_1) \wedge Explanation(F_1, F_2)$, which corresponds to a non tree-shaped dependency DAG.



In (1c), Adv_2 = *then* lexicalizes *Narration*. The discourse structure is $Explanation(F_0, F_1) \wedge Narration(F_0, F_2)$, which corresponds to a non tree-shaped dependency DAG.



In (1d), Adv_2 = ϵ lexicalizes *Commentary*. The discourse segment C_0 *because* C_1 forms a complex constituent whose logical form is the first argument of *Commentary*. So the discourse structure is $Commentary(Explanation(F_0, F_1), F_2)$, which corresponds to a tree-shaped dependency DAG.



In conclusion, these empirical data show that a formalism for the automatic analysis of discourse must be able to compute dependency structures which are not

[2] In [13], these four types of interpretation are finally brought to light, thanks to the Penn Discourse Tree Bank (PDTB), which is an English corpus manually annotated for discourse relations and their arguments [18].

tree-shaped.[3] This principle guided the conception of D-STAG. More precisely, from this (and other) data, we laid down the constraints below which govern the arguments of a discourse connective/relation using the following terminology. The clause in which a connective appears is called its "host clause". An adverbial connective appears in front of its host clause or within its VP. A subordinating conjunction always appears in front of its host clause, which is called an "adverbial clause." At the sentence level, an adverbial clause modifies a "matrix clause." It is located on its right, on its left, or inside it before its VP. When it is located on its right, the subordinating conjunction is said to be "postposed," otherwise it is said to be "preposed." A discourse connective/relation has two arguments which are the syntactic/semantic representations of two discourse segments that we call the "host segment" and the "mate segment". These segments are governed by the following constraints.

Constraint 1. *The host segment of a connective is identical to or starts at its host clause (possibly crossing a sentence boundary).*

Constraint 2. *The mate segment of an adverbial is anywhere on the left of its host segment (generally crossing a sentence boundary).*[4]

Constraint 3. *The mate segment of a postposed conjunction is on the left of its host segment without crossing a sentence boundary.*

Constraint 4. *The mate segment of a preposed conjunction is identical to or starts at the matrix clause (possibly crossing a sentence boundary).*

3 Introduction to TAG and STAG

This section is reproduced except where noted from [17] with permission of the authors. It begins with a brief introduction to the use of TAG in syntax.

"A tree-adjoining grammar (TAG) consists of a set of elementary tree structures and two operations, substitution and adjunction, used to combine these structures. The elementary trees can be of arbitrary depth. Each internal node is labeled with a nonterminal symbol. Frontier nodes may be labeled with either terminal symbols or nonterminal symbols and one of the diacritics $\downarrow$ or $*$. Use of the diacritic $\downarrow$ on a frontier node indicates that it is a *substitution node*. The *substitution* operation occurs when an elementary tree rooted in the nonterminal symbol A is substituted for a substitution node labeled with the nonterminal symbol A. Auxiliary trees are elementary trees in which the root and a frontier node, called the *foot node* and distinguished by the diacritic $*$, are labeled with the same nonterminal. The *adjunction* operation involves splicing an auxiliary

[3] In RST, (1b) and (1c) are represented as trees that must be interpreted with the "Nuclearity Principle" [16]. However, as explained in [7], the Nuclearity Principle leads to a wrong interpretation for (1d), namely $Explanation(F_0, F_1) \wedge Commentary(F_0, F_2)$.

[4] However, the mate segment must conform to the Right Frontier Constraint, which has been postulated in SDRT, see Sect. 5.1.

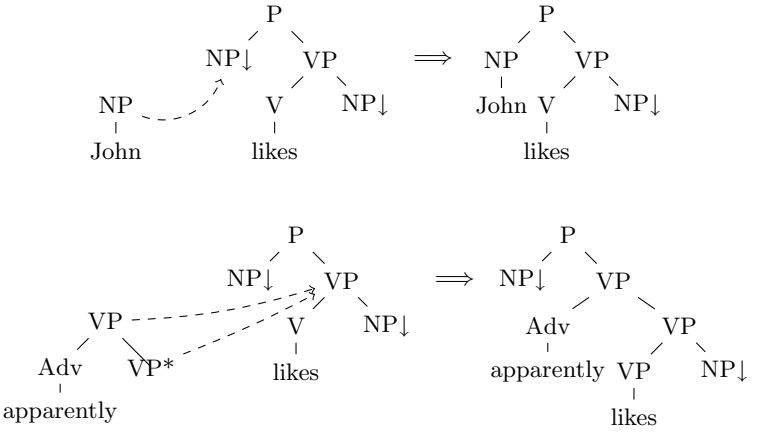


Fig. 1. Example TAG substitution and adjunction operations (reproduced from [17])

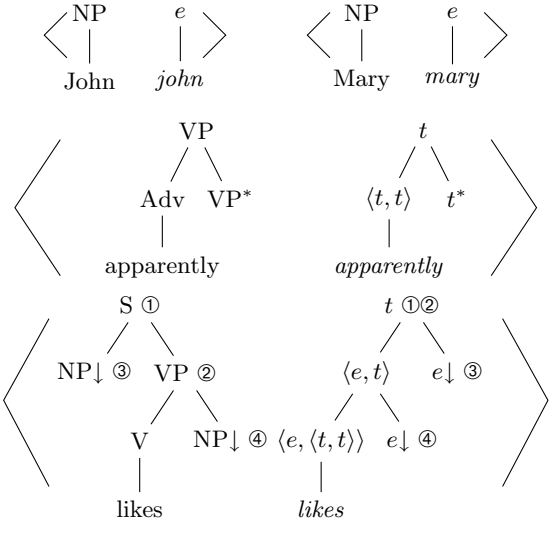


Fig. 2. An English syntax/semantics STAG fragment (reproduced from [17])

tree with root and designated foot node labeled with a nonterminal A at a node in an elementary tree also labeled with nonterminal A. Examples of the substitution and adjunction operations on sample elementary trees are shown in Figure 1."

"Synchronous TAG (STAG) extends TAG by taking the elementary structures to be pairs of TAG trees with links between particular nodes in those trees. An STAG is a set of triples, $\langle t_L, t_R, \frown \rangle$ where t_L and t_R are elementary TAG trees and $\frown$ is a linking relation between nodes in t_L and nodes in t_R [21,22]. Derivation

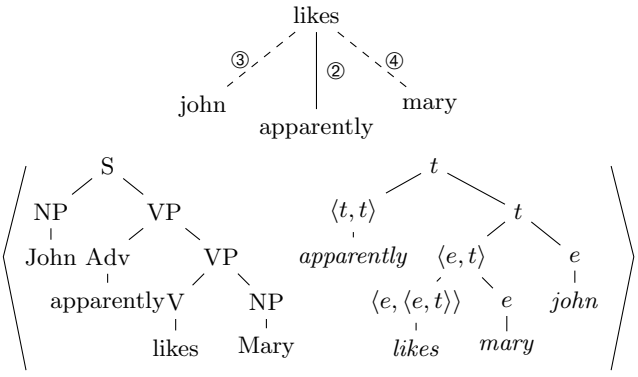


Fig. 3. Derivation tree and derived tree pair for *John apparently likes Mary* (reproduced from [17])

proceeds as in TAG except that all operations must be paired. That is, a tree can only be substituted or adjoined at a node if its pair is simultaneously substituted or adjoined at a linked node." We notate the links by using circled indices (e.g. ①) marking linked nodes.

STAG has been successfully used in an English sentential syntax/semantics interface [17]. For the sentence *John apparently likes Mary*, Fig. 2 gives the STAG fragment, Fig. 3 the derivation tree and the derived tree pair. In derivation trees, "substitutions are notated with a solid line and adjunctions are notated with a dashed line. Note that each link in the derivation tree specifies a link number in the elementary tree pair. The links provide the location of the operations in the syntax tree and in the semantics tree. These operations must occur at linked nodes in the target elementary tree pair. In this case, the noun phrases *John* and *Mary* substitute into *likes* at links ③ and ④ respectively. The word *apparently* adjoins at link ②. The resulting semantic representation can be read off the derived tree by treating the leftmost child of a node as a functor and its siblings as its arguments. Our sample sentence thus results in the semantic representation $apparently(likes(john, mary))$."

4 Sentence-Discourse Interface

We first explain why this interface is necessary. The idea in D-STAG is to extend a sentential analyzer to the discourse level **without making any change to it**. Yet, one cannot directly pass from sentence to discourse because there are mismatches between the arguments of a connective at the discourse level and its arguments at the sentence level. First, an adverbial connective has compulsorily two arguments at the discourse level, whereas it has only one argument at the sentence level. Second, a subordinating conjunction can have an argument at the discourse level which crosses a sentence boundary (see (1a) and (3) below), whereas this is out of the question at the sentence level.

In conclusion, it is necessary to pass through a sentence-discourse interface which gives sentence boundaries the simple role of punctuation signs and which allows us to re-compute the (two) arguments of a connective. Such an interface is also used in D-LTAG, by which we were inspired. From the sentential syntactic analysis, this interface deterministically produces a "Discourse Normalized Form" (henceforth DNF), which is a sequence of "discourse words:" a discourse word is mainly a connective, an identifier C_i for a clause (without any connective) or a punctuation sign. The syntactic and semantic analyses for C_is are those obtained by the sentential analyzer by removing connectives. An adverbial connective is moved in front of its host clause if not already there, while keeping a trace of its original position. If a normalized sentence (except the very first one) doesn't start with an adverbial connective, the empty connective ϵ is introduced. As an illustration, for (2), the DNF is C_0. ϵ *as* C_1, C_2. $then^{internal}$ C_3 *because* C_4.

(2) Fred went to the movies. As he was in a bad mood, he didn't enjoy it. He then went to a bar because he was dead thirsty.

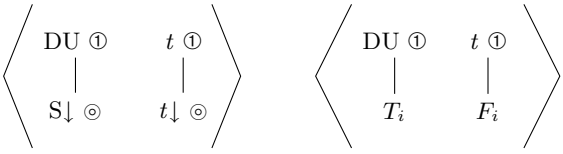
The sequence of discourse words making up a DNF follows a regular grammar. In Section 5.1 dedicated to adverbial connectives and postposed conjunctions, a DNF follows the regular expression C $(Punct\ Conn\ C)^*$, in which the sequence *Punct Conn* is either . *Adv* or (,) *Conj* where the comma is optional. Disregarding punctuation signs, the DNF is: $C_0\ Conn_1\ C_1 \ldots Conn_n\ C_n$, with $Conn_i = Adv_i$ ou $Conj_i$. A DNF with a preposed conjunction (Sect. 5.2) includes one element C which is preceded by the expressions $Conj\ C((,)\ Conj\ C)^*$. Connectives can be optionally followed or preceded by a modifier (Sect. 5.3). Coordinating conjunctions are studied in Section 5.4. Cases of "multiple connectives" in which two connectives share the same host clause are studied in Section 6.

This regular grammar doesn't decompose clauses into sub-clauses: it takes into account neither clausal complements nor incident clauses nor relative clauses, while these sub-clauses may play a role at the discourse level. We plan to complete the regular grammar for DNFs in future research and to extend the discursive component of D-STAG accordingly (Sect 7).

5 Discursive Component of D-STAG

For a clause C_i (without any connective), the sentential-discourse interface provides its syntactic tree rooted in S and noted T_i, its semantic tree rooted in t and noted F_i, and its derivation tree noted τ_i. To plug the clausal analyses into the discourse ones, we use the pair $\alpha S\text{-}to\text{-}D$ given in Fig. 4-a, in which the symbol DU represents the category "discourse unit". In the rest of this paper, we note η_i the derivation tree made of $\alpha S\text{-}to\text{-}D$ in which τ_i is substituted at link $\circledcirc$; η_i corresponds to the pair given in Fig. 4-b. We also use the following convention: as any tree of our grammar includes at the most one substitution node, this one (when it exists) is systematically marked with link $\circledcirc$.

When a given connective $Conn_i$ lexicalizes a single discourse relation R_i, the basic principle of the discursive STAG grammar consists in designing a tree pair,



(a) α*S-to-D* (b) η_i

Fig. 4. Tree pairs α*S-to-D* and η_i

noted $Conn_i \div R_i$, whose syntactic tree is anchored by $Conn_i$ and whose semantic tree is anchored by a lambda-term associated with R_i. We say, although somewhat inaccurately, that the semantic tree is anchored by R_i. When a connective is ambiguous, i.e. it lexicalizes several discourse relations, it anchors as many syntactic trees as discourse relations it lexicalizes (this is in particular the case for the empty connective ϵ). However, ambiguity issues are not in the scope of this paper.

We start with the presentation of the STAG discursive grammar for adverbials and postposed conjunctions, which are connectives with a similar behavior.

5.1 Adverbial Connectives and Postposed Conjunctions

Syntactic trees. The syntactic trees anchored by an adverbial connective or by a postposed conjunction are given in Fig. 5, in which a discourse connective is of category DC. Disregarding the features for now, these trees differ only in the co-anchors which are punctuation signs of category Punct.

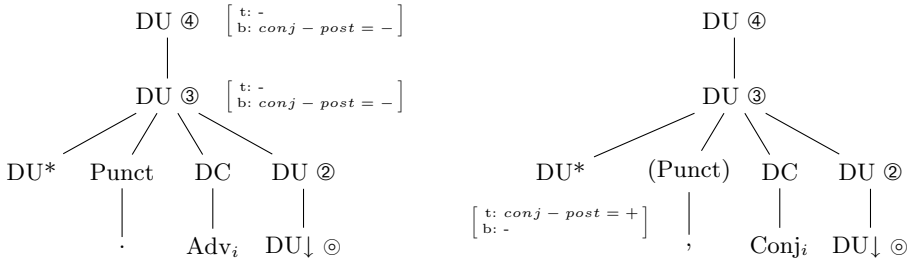


Fig. 5. Syntactic trees for adverbial connectives and postposed conjunctions

These trees observe the following principles: they are **auxiliary** trees with two arguments given by a substitution node DU↓ and a foot node DU*. The substitution node is for the host argument of the connective, i.e. the DU for the host segment substitutes at DU↓. If the DU for the host clause has undergone an adjunction, then the host segment starts at — but is not identical to — the host clause, see Constraint 1 in Sect. 2. The foot node corresponds to the mate argument of the connective. It is located on its left, which conforms to Constraints 2 and 3. The fact that the mate segment of a postposed conjunction

cannot cross a sentence boundary, contrarily to that of an adverbial, is handled with features $[conj - post = \pm]$ explained later.

We postulate an incremental analysis procedure in which the sequence of the discourse words of a DNF of the form $C_0\ Conn_1\ C_1\ \ldots\ Conn_n\ C_n$ (disregarding punctuation signs) is analyzed from left to right. After analyzing $C_0 \ldots\ Conn_n\ C_n$, the attachment of the new connective $Conn_{n+1}$ is realized by **adjunction** of the tree anchored by $Conn_{n+1}$ at a node DU on the **right frontier** of the syntactic tree representing $C_0 \ldots\ Conn_n\ C_n$. The attachment of the new clause C_{n+1} is realized by substituting the syntactic tree of the pair η_{n+1} at the substitution node DU$\downarrow$ of the tree anchored by $Conn_{n+1}$. Let us underline that this incremental procedure doesn't take into account the segmentation of the discourse into sentences.

Trees anchored by an adverbial or a postposed conjunction include three nodes labelled DU, with link ②, ③ or ④, on their right frontier. These nodes are marked with different links, which allows us to get various semantic interpretations, as shown below. There exist three nodes DU with different links, and not a single node DU with three different links, so as to allow several adjunctions to different nodes, for example an adjunction at DU③ to attach $Conn_n$ and an adjunction at DU④ to attach $Conn_{n+1}$. It should be noted that if an adjunction is done at DU③ to attach $Conn_n$, DU② is no longer on the right frontier of the syntactic tree. Therefore, DU② can no longer be an adjunction site to attach $Conn_{n+1}$. This constraint will be generalized in Constraint 5 below.

Semantic trees. At first sight, one could consider that a discourse relation R_i is associated with the functor $\mathcal{R}_i = \lambda xy.R_i(x, y)$ with $x, y : t$, $R_i(x, y) : t$, and $\mathcal{R}_i : \langle t, \langle t, t\rangle\rangle$, $\mathcal{R}_i$ anchoring a tree with a foot node $t*$ and a substitution node $t\downarrow$. Yet this is appropriate only to analyze a simple DNF with two clauses, for example a DNF of the form C_0 *because* C_1 as shown in Fig. 6 in which $\beta_1 = because_{post} \div Explanation$.

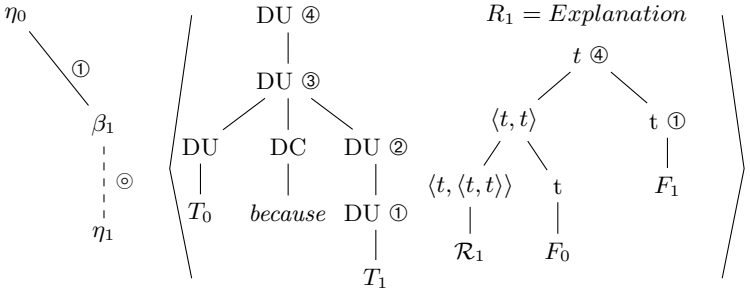


Fig. 6. Derivation tree and derived tree pair for a DNF of the form C_0 *because* C_1 (using the functor $\mathcal{R}_1$ in the semantic tree)

However, with this simple functor $\mathcal{R}_i$, it is impossible to obtain four interpretations (of which two are a conjunction of formulae) for DNFs with three clauses (see Sect. 2). Therefore, we define two type-shifting operators Φ' and Φ'': they

take $\mathcal{R}_i$ as argument and return two new functors $\mathcal{R}'_i$ and $\mathcal{R}''_i$ associated with the discourse relation R_i. Below the definitions.

Definition 1. $\Phi' = \lambda\mathcal{R}_i XY.X(\lambda x.Y(\lambda y.\mathcal{R}_i(x,y)))$
$\Phi'(\mathcal{R}_i) = \mathcal{R}'_i = \lambda XY.X(\lambda x.Y(\lambda y.\mathcal{R}_i(x,y)))$
with $X, Y : ttt = \langle\langle t,t\rangle,t\rangle$ and $x, y : t$

Φ' triggers a type raising on the arguments. The resulting functor $\mathcal{R}'_i$ is of type $\langle ttt, \langle ttt, t\rangle\rangle$ in which ttt symbolizes the type $\langle\langle t,t\rangle,t\rangle$. It co-anchors tree (A), given in Fig. 7-a, whose foot node is of type t. (A) is used for adjunctions at links ① and ④. If the first argument of $\mathcal{R}'_i$ is $\lambda P.P(F_0)$ of type ttt, the second one $\lambda Q.Q(F_1)$ of type ttt, then the result is $R_i(F_0, F_1)$ of type t. So, for a DNF with two clauses, $\mathcal{R}'_i$ leads to the same result as $\mathcal{R}_i$. Yet, the type raising is necessary to introduce nodes ttt ② and ttt ③ at which (B) can adjoin.

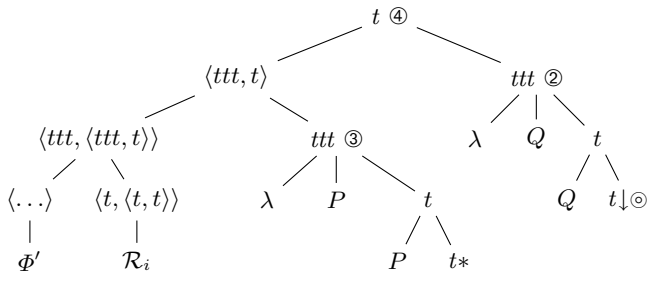
Definition 2. $\Phi'' = \lambda\mathcal{R}_i XYP.X(\lambda x.Y(\lambda y.\mathcal{R}_i(x,y) \wedge P(x)))$
$\Phi''(\mathcal{R}_i) = \mathcal{R}''_i = \lambda XYP.X(\lambda x.Y(\lambda y.\mathcal{R}_i(x,y) \wedge P(x)))$
with $X, Y : ttt = \langle\langle t,t\rangle,t\rangle$, $P : \langle t,t\rangle$ and $x, y : t$

Φ'' introduces a conjunction of terms. The resulting functor $\mathcal{R}''_i$ is of type $\langle ttt, \langle ttt, ttt\rangle\rangle$. It anchors tree (B), given in Fig. 7-b, whose foot node is of type ttt. (B) is used for adjunctions at links ② and ③. If the first argument of $\mathcal{R}''_i$ is $\lambda P.P(F_0)$, the second one $\lambda Q.Q(F_1)$, then the result is $\lambda P.(R_i(F_0, F_1) \wedge P(F_0))$ of type ttt.

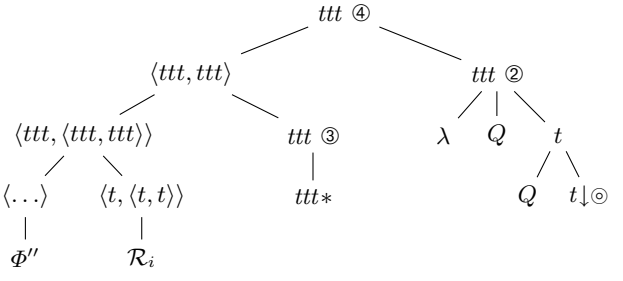
Analysis of DNF*s with three clauses.* For DNFs with three clauses, four types of interpretation must be computed. These were illustrated in examples (1) of the form C_0 *because* C_1. Adv_2 C_2 in Sect. 2, and we are going to explain the analysis of these examples. We call β_1 the tree pair $because_{post} \div Explanation$ and β_2 the pair $Adv_2 \div R_2$. After analyzing C_0 *because* C_1, the syntactic tree is that shown in Fig. 6. The right frontier of this tree includes four nodes labelled DU which can receive the adjunction of the syntactic tree of β_2. These nodes are marked with link ① coming from the syntactic tree of η_1 or link ②, ③ or ④ coming from the syntactic tree anchored by *because*. The analyses of the four examples in (1) are obtained by adjoining β_2 at one of these links.

We start with (1a) with $\beta_2 = moreover \div Continuation$, for which the discourse structure is $Explanation(F_0, Continuation(F_1, F_2))$. This is obtained by adjoining β_2 at link ① of η_1. The node with link ① in the semantic tree of η_1 is of type t. Therefore, one must use tree (A) anchored by $\mathcal{R}'_2$, whose foot node is of type t. The semantic derived tree for (1a) is given in Fig. 8. The sub-tree rooted at Gorn address 2 results in $\lambda P.P(Continuation(F_1, F_2))$ with $P : \langle t,t\rangle$. So $Continuation(F_1, F_2)$ is the second argument of $R_1 = Explanation$, whose first argument is F_0, hence the formula $Explanation(F_0, Continuation(F_1, F_2))$.

We go on with (1b) with $\beta_2 = \epsilon \div Explanation$, for which the discourse structure is $Explanation(F_0, F_1) \wedge Explanation(F_1, F_2)$ with a conjunction of formulae. This is obtained by adjoining β_2 at link ② of β_1. The node with link ② in the semantic tree of β_1 is of type ttt. Therefore, one must use tree



(a) Tree (A)



(b) Tree (B)

Fig. 7. Semantic trees (A) and (B) anchored by $\mathcal{R}'_i = \phi'(\mathcal{R}_i)$ and $\mathcal{R}''_i = \phi''(\mathcal{R}_i)$

(B) anchored by $\mathcal{R}''_2$, whose foot node is of type *ttt*. The semantic derived tree for (1b) is given in Fig. 9-a. The sub-tree rooted at Gorn address 2 results in $\lambda P.(Explanation(F_1, F_2) \wedge P(F_1))$ with $P : \langle t, t\rangle$. As only F_1 is under P, it is the second argument of $R_1 = Explanation$, whose first argument is F_0, hence the formula $Explanation(F_0, F_1) \wedge Explanation(F_1, F_2)$.

For (1c) with $\beta_2 = then \div Narration$, the structure is $Explanation(F_0, F_1) \wedge Narration(F_0, F_2)$ with also a conjunction of formulae. This is obtained by adjoining β_2 at link ③ of β_1. This case is similar to the previous one, so we simply give the semantic derived tree in Fig. 9-b.

Let us finish with (1d) with $\beta_2 = \epsilon \div Commentary$, for which the structure is $Commentary(Explanation(F_0, F_1), F_2)$. This is obtained by adjoining β_2 at link ④ of β_1. The node with link ④ in the semantic tree of β_1 is of type t. Therefore, one must use tree (A) anchored by $\mathcal{R}'_2$. The semantic derived tree for (1d) is given in Fig. 10. The sub-tree rooted at Gorn address 1.2 results in $\lambda P.P(Explanation(F_0, F_1))$ with $P : \langle t, t\rangle$.

In conclusion, the four types of interpretation of DNFs of the form $C_0\ Conj_1\ C_1.\ Adv_2\ C_2$ are computed thanks to the four adjunction sites on the right frontier of the syntactic tree for $C_0\ Conj_1\ C_1$ and to semantic trees (A) and (B) whose foot nodes are respectively of type t et *ttt* and which are anchored by $\mathcal{R}'_i$ and $\mathcal{R}''_i$.

For DNFs with three clauses of the form $C_0\ Conn_1\ C_1\ Conn_2\ C_2$, we have just examined the case $C_0\ Conj_1\ C_1.\ Adv_2\ C_2$ where $Conn_1$ is a postposed conjunction and $Conn_2$ an adverbial. Three cases are left: $Conn_1$ is a postposed

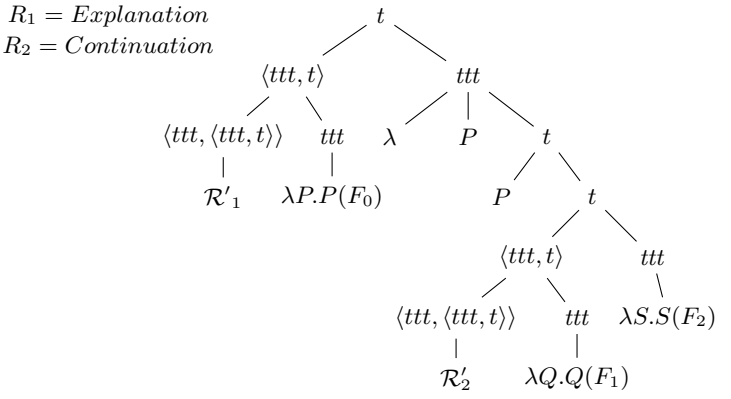


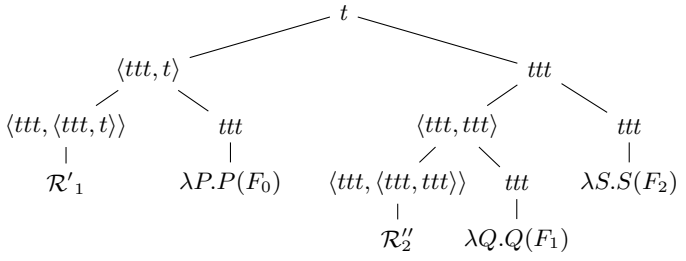
Fig. 8. Semantic derived tree for (1a) with interpretation $R_1(F_0, R_2(F_1, F_2))$

conjunction and $Conn_2$ also, $Conn_1$ is an adverbial and $Conn_2$ also, $Conn_1$ is an adverbial and $Conn_2$ a postposed conjunction. The first two cases raise no new issues. The third case, which concerns DNFs of the form C_0. Adv_1 C_1 $Conj_2$ C_2, raises the issue of fully implementing Constraint 3 from Sect. 2, which states that the mate segment of a postposed conjunction cannot cross a sentence boundary. This constraint is implemented thanks to features $[conj - post = \pm]$ which decorate some nodes in the syntactic trees anchored by an adverbial or a postposed conjunction, which are given in Fig. 5. More precisely:

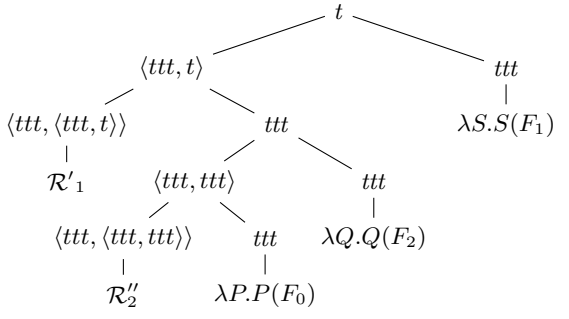
- the foot node of a tree anchored by a postposed conjunction is decorated with the top feature $conj - post = +$,
- the nodes with links ③ and ④ in a tree anchored by an adverbial are decorated with the bottom feature $conj - post = -$.

These features block the adjunction of $Conj_2 \div R_2$ at links ③ and ④ of $Adv_1 \div R_1$ thanks to unification failure $(conj - post = +) \cup (conj - post = -)$. Therefore, $Conj_2 \div R_2$ can only adjoin at link ② of $Adv_1 \div R_1$ and at link ① of η_1, which results respectively in interpretations $R_1(F_0, F_1) \wedge R_2(F_1, F_2)$ and $R_1(F_0, R_2(F_1, F_2))$. These interpretations conform to Constraint 3: the mate argument of R_2 is F_1, so the mate segment of $Conj_2$ is C_1, which doesn't cross a sentence boundary.

Analysis of DNF*s with n clauses* ($n > 3$) *of the form* C_0 $Conn_0$ C_1 ... C_n. For DNFs with n clauses, no new mechanism is involved. Attaching $Conn_{n+1}$ next C_{n+1} consists, in the derivation tree representing $C_0 \ldots C_n$, in adjoining $Conn_{n+1} \div R_{n+1}$ — in which η_{n+1} is substituted — at link ① of η_n or at link (i) with $i \in \{2, 3, 4\}$ of a node $\beta_k = Conn_k \div R_k$, the node at link (i) coming from the syntactic tree of β_k being on the right frontier of the syntactic tree for $C_0 \ldots C_n$ (to keep to the linear order of the DNF). As it is long and tedious



(a) (1b) with $R_1 = R_2 = Explanation$



(b) (1c) with $R_1 = Explanation$ and $R_2 = Narration$

Fig. 9. Semantic derived trees for (1b) and (1c) with interpretations $R_1(F_0, F_1) \wedge R_2(F_1, F_2)$ and $R_1(F_0, F_1) \wedge R_2(F_0, F_2)$

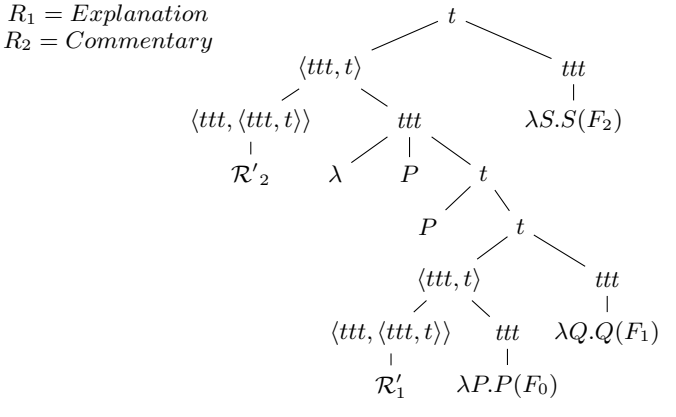


Fig. 10. Semantic derived trees for (1d) with interpretation $R_2(R_1(F_0, F_1), F_2)$

to determine (the right frontier of) the derived syntactic tree, it is convenient to define a notion of right frontier on the derivation tree. Since derivation trees are intrinsically not ordered, a graphical convention which represents derivation trees as ordered must be called upon. The following convention is satisfactory:

the nodes labeled η_k projected onto a line are ordered by following the linear order of the DNF. With this order relation noted $\prec$, the nodes β_k in the derivation tree which are possible sites of adjunction for a new segment are those on the right frontier of the derivation tree, while observing Constraint 5 which governs two adjunctions at links $\textcircled{n}$ and $\textcircled{m}$ of the same node:[5]

Constraint 5. *If β_j — in which η_j is substituted — adjoins at link $\textcircled{n}$ of a node β_i, then β_k — in which η_k is substituted — can adjoin at link $\textcircled{m}$ of the same node β_i only if the following rule is observed: $\eta_j \prec \eta_k \Rightarrow n < m$ (with n and m belonging to $\{2, 3, 4\}$).*

This constraint generalizes the one we formulated before, namely, if an adjunction is performed at node DU③ of a syntactic tree anchored by $Conn_i$, then a new adjunction in this tree can be performed at DU④ but not at DU②.

Implementation of RFC. The Right Frontier Constraint (RFC) postulated in SDRT relies on a distinction between two types of discourse relations: coordinating (*Narration*, *Continuation*) versus subordinating (*Explanation*, *Commentary*) ones. RFC says that it is forbidden to attach new information to the first argument of a coordinating relation [2]. As a consequence, for example, the interpretation $R_1(F_0, F_1) \wedge R_2(F_0, F_2)$ is excluded when R_1 is coordinating.

Implementing RFC in D-STAG requires first to distinguish the semantic trees anchored by a coordinating versus subordinating relation. This is achieved by creating two copies of semantic trees (A) and (B), which differ in a top feature $[coord = \pm]$ decorating their foot node. Then RFC is implemented by forbidding any adjunction at link ③ of the copies of (A) and (B) whose foot node is decorated with the feature $[coord = +]$: it is link ③ which is used to obtain interpretations like $R_1(F_0, F_1) \wedge R_2(F_0, F_2)$ which must be excluded when R_1 is coordinating.

SDRT postulates other semantic constraints based on the distinction between coordinating versus subordinating relations, for example the "Continuous Discourse Pattern" [3]. There is no room to describe them, however we can say that they are easily implemented in D-STAG thanks to the addition of a set of features in the semantic trees.

5.2 Preposed Conjunctions

The syntactic tree anchored by a preposed conjunction is given in Fig. 11. It is designed so as to respect Constraints 1 and 4 given in Sect. 2. It differs from the syntactic trees anchored by an adverbial or a postposed conjunction, among other things, by the fact that the foot node DU$*$ is dominated by a node DU with link ⑤. To take into account this new link, a node t⑤ dominating the foot node $t*$ is added into the two copies of semantic tree (A).

An adjunction to link ⑤ is used when the preposed conjunction introduces a "framing adverbial" [4] as illustrated in (3) of the form $When\ C_0, C_1.\ Next\ C_2$. In

[5] This constraint is valid because we assigned links ②, ③ and ④ in a well-thought-out manner.

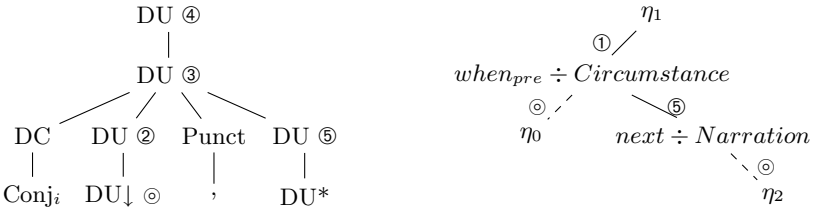


Fig. 11. Tree anchored by a preposed conjunction and derivation tree for (3)

this discourse, the preposed conjunction *when* has its mate segment which crosses a sentence boundary. Its interpretation is $Circumstance(Narration(F_1, F_2), F_0)$, assuming that *when* conveys *Circumstance* and *next* conveys *Narration*.

(3) When he was in Paris, Fred went to the Eiffel Tower. Next, he visited The Louvre.

The derivation tree for (3) is given in Fig. 11. The semantic trees anchored by *Circumstance* and *Narration* are both (A). The arguments of the functor associated to *Circumstance* are $\lambda P.P(Narration(F_1, F_2))$ and $\lambda Q.Q(F_0)$. Therefore, $Narration(F_1, F_2)$ is the first argument of *Circumstance*, F_0 the second one.[6]

5.3 Modifiers of Discourse Connectives/Relations

As far as we are aware, modification of discourse connectives/relations is a phenomenon which has been neglected. However, it is a common phenomenon as illustrated in (4).

(4) a. Fred is in a bad mood *only/even/except* when it is sunny.
b. You shouldn't trust John because, *for example*, he never returns what he borrows. [23]
c. John just broke his arm. So, *for example*, he can't cycle to work.[23].

In [23], *for example* is not considered as a connective modifier but as a connective whose interpretation is "parasitic" on the relation conveyed by the connective on its left. This position, which is not linguistically justified, leads to laborious computations in D-LTAG [10, pp 31-35] to obtain the interpretation of (4b). On the contrary, in D-STAG, we propose that *for example* in (4b) or (4c) is a modifier of the connective on its left in the same way that *only, even* and *except* in (4a) are modifiers of the connective on their right. This position sounds more justified on linguistic grounds and it allows us to obtain the interpretation of a discourse such as (4b) in a very simple way, as we are going to show.

[6] Examples such as (3) make it that, despite appearances, the syntactic discursive grammar of D-STAG is not a TIG (Tree Insertion Grammar) [20], since the right tree anchored by *next* is adjoined on the spine of the left tree anchored by the preposed conjunction *when*.

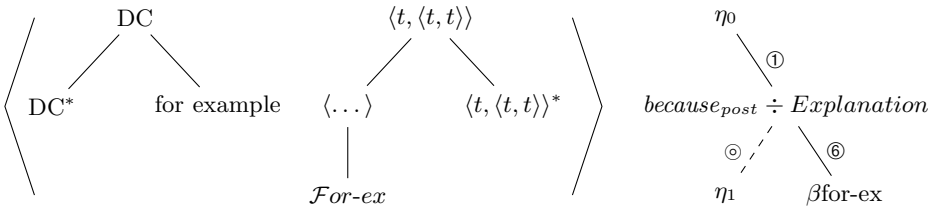


Fig. 12. Pair $\beta$for-ex and derivation tree for (4b)

In D-STAG, connective modifiers anchor (syntactic) auxiliary trees whose foot node is DC (left tree for *only, even* and *except*, right tree for *for example*). To adjoin these trees, we mark the node labelled DC with link ⑥ in the syntactic trees for connectives (Fig. 5 and 11). At the semantic level, we assume that the contribution of a discourse relation modifier consists in transforming a functor $\mathcal{R}_i$ of type $\langle t, \langle t, t\rangle\rangle$ into another functor of the same type. Therefore, the nodes dominating $\mathcal{R}_i$ are marked with link ⑥ in the two copies of (A) and (B). Let us illustrate adjunctions to link ⑥ with (4b) of the form C_0 *because for example* C_1. As explained in [23], the interpretation of (4b) is $Exemplification(F_1, \lambda r.Explanation(F_0, r))$ with $r : t$. To get this interpretation, we define the functor $\mathcal{F}$*or-ex* as below. The pair named $\beta$for-ex is given in Fig. 12, which also shows the derivation tree for (4b). The functor $\Phi'(\mathcal{F}or\text{-}ex\ (\mathcal{R}_i))$ with $R_i = Explanation$ results in the right interpretation.

Definition 3. $\mathcal{F}or\text{-}ex = \lambda\mathcal{R}_i pq.Exemplification(q, \lambda r.\mathcal{R}_i(p, r))$
with $\mathcal{R}_i : \langle t, \langle t, t\rangle\rangle$ and $p, q : t$.

5.4 Coordinating Conjunctions

For reasons that will be obvious later, we start with correlative coordinations of adverbial clauses, which are illustrated in (5). These correlative coordinations can easily be handled in D-STAG by considering e.g. *neither* and *nor* as (adverbial) modifiers of the subordinating conjunction on their right. The correlative part of these constructions, e.g. the fact that *neither* cannot modify a subordinating conjunction without the presence of *either* modifying another subordinating conjunction, is taken into account by a set of features in the syntactic trees anchored by a subordinating conjunction (there is no room to explain these features). This set of features also forces the adjunction of the tree anchored by the second (modified) subordinating conjunction to link ③ of the tree anchored by the first (modified) subordinating conjunction. As a result, the interpretation is a conjunction of formulae in which the logical form of the matrix clause is shared. More precisely, for (5a) of the form C_0 *neither when* C_1 *nor when* C_2, the interpretation $\neg Condition(F_0, F_1) \wedge \neg Condition(F_0, F_2)$ — assuming that *when* lexicalizes the relation *Condition* — is obtained (through an adjunction of the second *when* at link ③ of the first *when*) by giving *neither* and *nor* the

semantics of negation. For (5b) of the form C_0 *either if* C_1 *or if* C_2, the interpretation $Condition(F_0, F_1) \vee Condition(F_0, F_2) = \neg(\neg Condition(F_0, F_1) \wedge \neg Condition(F_0, F_2))$ is obtained by giving to *or* the semantics of negation and by associating to *either* a semantic tree with two parts, one for the local scope of negation, the other one for the global scope of negation over the conjunction of formulae.[7] For (5c) of the form C_0 *both when* C_1 *and when* C_2, the interpretation $Condition(F_0, F_1) \wedge Condition(F_0, F_2)$ is obtained by giving to *both* and *and* the semantics of identity.

(5) a. Fred is pleased *neither* when it is sunny *nor* when it is rainy.
b. Fred will come *either* if it is sunny *or* if it is rainy.
c. Fred is pleased *both* when it is sunny *and* when it is rainy.

We now examine non-correlative coordinations. Coordinations of two matrix clauses with a DNF of the form C_0 $Coord_1$ C_1 are handled with a syntactic tree anchored by the coordinating conjunction $Coord_1$ similar to the syntactic tree anchored by a subordinating conjunction (Fig. 5). Coordination of n clauses with $n > 2$ with a DNF of the form C_0, $C_1(,)$ $Coord_2$ C_2 are handled with the standard mechanisms for multiple coordinations. Coordinations of adverbial clauses with a DNF of the form C_0 $Conj_1$ C_1 *or/and* $Conj_2$ C_2 raise drastic problems, in particular to get their interpretation. However, it must be noted that the interpretation of (5b)/(5c) doesn't change if *either/both* is omitted. In other words, a non-correlative coordination of adverbial clauses is semantically equivalent to a correlative coordination. Therefore, we propose that the sentence-discourse interface automatically converts a non-correlative coordination of adverbial clauses into a correlative one, e.g. converts the DNF C_0 $Conj_1$ C_1 *or* $Conj_2$ C_2 into the DNF C_0 *either* $Conj_1$ C_1 *or* $Conj_2$ C_2 that we can handle by considering *either* and *or* as modifiers of the subordinating conjunction on their right.

6 Comparison between D-STAG and D-LTAG

D-STAG and D-LTAG [10] roughly share the same goal and the same architecture. The discrepancies between these two formalisms mainly lie in the discursive component.[8] First, D-LTAG makes little use of discourse relations and ignores the distinction between coordinating versus subordinating relations. In short, it doesn't build on discourse theories. This is even a principle, as shown in the quotation [10, p 1] " D-LTAG presents a model of low-level discourse structure and interpretation that exploits the *same* mechanisms used at the sentence level and builds them directly on top of clause structure and interpretation". This prevents D-LTAG from taking advantage of the insights provided by discourse theories, which supply rhetorical knowledge, among other things.

[7] This is the only case which requires a multi-component semantic tree.

[8] The sentential components of D-STAG and D-LTAG are also crucially different, since D-STAG relies on STAG, which is not the case for D-LTAG. However, the sentential level is out of the scope of this paper, see [17] for a discussion on the various approaches for a sentential syntax-semantics interface.

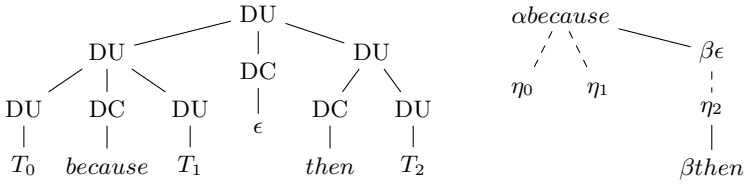


Fig. 13. D-LTAG syntactic tree and derivation tree for (1c) = *Fred went to the supermarket because his fridge is empty. Then, he went to the movies.*

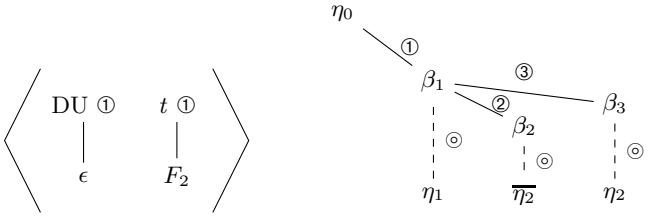
Next, discourse connectives in D-STAG all anchor elementary trees with **two** arguments, whereas, in D-LTAG, most adverbial connectives (but not the empty one ϵ) anchor trees with only **one** argument (this is, for example, the case for *then* whose host segment is "structurally" provided, the mate segment being "anaphorically" provided [23]). Moreover, subordinating conjunctions in D-LTAG anchor trees with two arguments, but these trees are **initial** trees whereas they are **auxiliary** in D-STAG (subordinating conjunctions are considered in D-LTAG as "structural" connectives, neglecting the fact that one of their segments can cross a sentence boundary, (1a) and (3)). These discrepancies between the trees anchored by a connective lead to crucial differences in the discursive analyses, especially in the semantic ones.

As an illustration, the syntactic tree and derivation tree produced by D-LTAG for (1c) are given in Fig. 13. The syntactic tree includes three nodes labelled DC, while it includes only two nodes labelled DC in D-STAG since the empty connective ϵ is introduced only in the absence of any other adverbial. The derivation tree results in the discourse structure $Narration(Explanation(F_0, F_1), F_2)$, which is wrong: the explanation given for Fred's visit to the supermarket (*his fridge was empty*) shouldn't be under the scope of $Narration$, which is a relation linking together two events and not a causal relation and an event. As explained in Sect.2, the discourse structure for (1c) is $Explanation(F_0, F_1) \wedge Narration(F_0, F_2)$. This structure, which corresponds to a non tree-shaped dependency graph, cannot be obtained in D-LTAG.

"Multiple connectives" is a phenomenon which guided the conception of D-LTAG, as shown in the quotation [23, p 552]: "The distinction between structural and anaphoric connectives in D-LTAG is based on considerations of computational economy and behavioral evidence from cases of multiple connectives". The discourse (6) gives an illustration of multiple connectives. Its DNF is of the form C_0 *but* C_1 *because then* C_2, in which two connectives share the same host clause C_2, the second one being an adverbial.

(6) John ordered three cases of Barolo. But he had to cancel the order *because then* he discovered he was broke. [23]

In D-STAG, we propose the following solution to handle multiple connectives. The DNF for (6) is automatically converted into C_0 *but* C_1 *because* $\overline{C_2}$ *then* C_2 which conforms to the regular pattern for DNFs (without any preposed conjunction).



(a) $\overline{\eta_2}$ (b) *Derivation tree for* (6)

Fig. 14. Tree pair $\overline{\eta_2}$ and derivation tree for (6)

The tree pair $\overline{\eta_2}$ associated to $\overline{C_2}$ is given in Fig. 14-a: the syntactic leaf is the empty string ϵ, the semantic leaf is F_2 (the semantic formula for C_2). So we can say that $\overline{C_2}$ is the (fake) host clause of *because*, which has only a semantic contribution. The interpretation of (6), i.e. $Contrast(F_0, F_1) \wedge Explanation(F_1, F_2) \wedge Narration(F_0, F_2)$ assuming that *then* conveys $Narration$, is obtained by the mechanisms described in Sect. 5.1.1 which build the derivation tree given in Fig. 14-b (with $\beta_1 = but \div Contrast$, $\beta_2 = because \div Explanation$ and $\beta_3 = then \div Narration$).[9]

In conclusion, cases of multiple connectives can be boiled down to the general pattern in which an host clause receives only a single connective, thanks to the sentence-discouse interface (which in any case cannot be bypassed (Sect.4)). Therefore, there is no need to postulate a distinction between anaphoric and structural connectives, the former with a single argument, the latter with two arguments.

7 Conclusion

We focused on an STAG discursive grammar for connectives/discourse relations. It aims at handling any type of connectives (adverbials, postposed and preposed subordinating conjunctions, coordinating conjunctions (including coordinating conjunctions in correlative constructions), any modifier of a connective, and finally cases of "multiple connectives". The syntactic discursive grammar for connectives shows that it is justified to postulate that any connective anchors an auxiliary tree with two arguments: a foot node DU* for the mate argument , a substitution node DU$\downarrow$ for the host argument which is identical to or starts at the host clause of the connective. The semantic discursive grammar is made up from trees anchored by functors associated to discourse relations, which are considered as predicates lexicalized by discourse connectives. It allow us to compute discourse structures which are not compulsorily represented as tree-shaped dependency graphs.

[9] The interpretation of (6) looks as a counter-example to RFC since the discourse relation *Contrast* is coordinating. However, we explain in [8] how to handle this apparent counter-example.

The discursive component takes as input a DNF that is computed by a sentence-discourse interface. This interface is necessary if one doesn't want to make any change to the sentential analyzer. A DNF is a sequence of discourse words which follows a regular grammar. The regular grammar we have presented in this paper focuses on connectives but is not yet completed. In particular, it doesn't take into account sub-clauses (clausal complements, incident clauses, relative clauses) which may play a role at the discursive level. This is ongoing work [9] and we don't foresee any crucial difficulty for the resulting extended discursive grammar, thanks to STAG's richness of expressivity.

The implementation of D-STAG in a French discourse analyzer is work in progress [8]. The analyzer will produce a forest of derivation trees which represents the set of possible analyses. The extraction of the best analysis (or the n best analyses) will require to build probabilistic disambiguation models based on the French annotated corpus Annodis [19].

References

1. Asher, N.: Reference to Abstract Objects in Discourse. Kluwer, Dordrecht (1993)
2. Asher, N., Lascarides, A.: Logics of Conversation. Cambridge University Press, Cambridge (2003)
3. Asher, N., Vieu, L.: Subordinating and coordinating discourse relations. Lingua 115(4), 591–610 (2005)
4. Charolles, M.: Framing adverbials and their role in discourse cohesion, from connection to foward labelling. In: Proceedings of the First Symposium on the Exploration and Modelling of Meaning, SEM 2005, Biarritz, pp. 194–201 (2005)
5. Danlos, L.: A lexicalized formalism for text generation inspired from TAG. In: Abeillé, A., Rambow, O. (eds.) TAG Grammar. CSLI, Stanford (2001)
6. Danlos, L.: Discourse dependency structures as constrained DAGs. In: Proceedings of SIGDIAL 2004, Boston, pp. 127–135 (2004)
7. Danlos, L.: Strong generative capacity of RST, SDRT and discourse dependency DAGs. In: Benz, A., Kühnlein, P. (eds.) Constraints in Discourse, Benjamins (2007)
8. Danlos, L.: D-STAG: un formalisme d'analyse automatique de discours basé sur les TAG synchrones. Revue TAL 50(1), 111–143 (2009)
9. Danlos, L., Sagot, B., Stern, R.: Analyse discursive des incises de citations. In: Actes du Second Colloque Mondial de Linguistique Française New-Orleans, USA (submitted)
10. Forbes-Riley, K., Webber, B., Joshi, A.: Computing discourse semantics: The predicate-argument semantics of discourse connectives in D-LTAG. Journal of Semantics 23(1) (2006)
11. Harris, Z.: Mathematical Structures of Language. Krieger Pub. Co., New York (1986)
12. Joshi, A.: Tree-adjoining grammars. In: Dowty, D., Karttunen, L., Zwicky, A. (eds.) Natural Language Parsing, pp. 206–250. Cambridge University Press, Cambridge (1985)
13. Lee, A., Prasad, R., Joshi, A., Webber, B.: Departures from tree structures in discourse: Shared arguments in the penn discourse tree bank. In: Proceedings of the Constraints in Discourse Workshop (CID 2008). Postdam, Germany (2008)

14. Mann, W.C., Thompson, S.A.: Rhetorical structure theory: Toward a functional theory of text organization. Text 8(3), 243–281 (1988)
15. Marcu, D.: The rhetorical parsing of unrestricted texts: A surface-based approach. Computational Linguistics 26(3), 395–448 (2000)
16. Marcu, D.: The Theory and Practice of Discourse Parsing and Summarization. The MIT Press, Cambridge (2000)
17. Nesson, R., Shieber, S.: Simpler TAG semantics through synchronization. In: Formal Grammars, Malaga (2006)
18. PDTB Group: The penn discourse treebank 2.0 annotation manual. Tech. Rep. IRCS-08-01, Institute for Research in Cognitive Science, University of Philadelphia (2008)
19. Péry-Woodley, M.P., Asher, N., Enjalbert, P.: ANNODIS: une approche outillée de l'annotation de structures discursives. In: Proceedings of TALN 2009, pp. 190–196. Senlis, France (2009)
20. Schabes, Y., Waters, R.: Tree insertion grammar. Computational Intelligence 21, 479–514 (1995)
21. Shieber, S.: Restricting the weak-generative capacity of synchronous tree-adjoining grammars. Computational Intelligence 10(4), 371–385 (1994)
22. Shieber, S., Schabes, Y.: Synchronous tree-adjoining grammars. In: Proceedings of the 13th International Conference on Computational Linguistics, vol. 3, pp. 253–258. Helsinki, Finland (1990)
23. Webber, B.L., Joshi, A., Stone, M., Knott, A.: Anaphora and discourse structure. Computational Linguistics 29(4), 545–587 (2003)
24. Wolf, F., Gibson, E.: Coherence in Natural Language: Data Structures and Applications. The MIT Press, London (2006)