Linguistic ways for expressing a discourse relation in a lexicalized text generation system

Laurence DANLOS
TALANA, Université Paris 7
laurence.danlos@linguist.jussieu.fr

Introduction

In many cases, RST (Mann & Thomson 1988) is interpreted or used in such a way that discourse relations hold only between sentences or clauses, i.e. the leaves of the rhetorical tree structure of a text correspond to sentences or clauses. This interpretation of RST has already been criticized (Smedt et al. 1996) and this paper goes in this line. In general, a discourse relation can be realized in many different ways. In addition to traditional ways such as a text made up of two sentences with or without a cue phrase *(Ted hammered the metal. (Therefore), it is flat)*, there exist other ways such as a single sentence, e.g. a "resultative construction" *(Ted hammered the metal flat)* or an "operator verb construction" *Ted's hammering the metal caused it to be flat)*. The question arises: given the numerous and heterogeneous set of possible linguistic realizations, how can these possible realizations be expressed in a linguistically motivated manner so that a natural language processing system can make use of this knowledge? This paper presents a uniform framework for expressing the linguistic knowledge needed to relate discourse relations to linguistic realizations. In addition, it sketches how such a representation can be used during the linguistic planning stage of text generation and presents a lexicalized text generation system designed in the lines of a lexicalized grammar for sentence analysis. In such a system, lexical entries give semantic and syntactic information in a well defined structure. Therefore, the generation process for linguistic choices relies mainly upon a unique operation: lexicalization, i.e. choice of a lexical item with its semantic and syntactic structure to express a concept.

1 Preliminaries

The domain model or conceptual level is a collection of concepts hierarchically organised. The concept THING (a concept is written in upper cases) groups together a set of objects (HUMAN, CONCRETE, etc.). The concept RELATION is divided into ATOMIC-RELATION (i.e. mainly relations between objects) and NON-ATOMIC-RELATION (i.e. relations between relations or "discourse relations"). A concept has a structure, namely a set of arguments which are written in small upper cases (BUYER, SELLER and OBJECT for TRANSACTION). The value of each argument is conceptually restricted, e.g. the BUYER of TRANSACTION must referred to an HUMAN. Below the representations of TRANSACTION and RESULT.

```
TRANSACTION < ATOMIC-RELATION [BUYER => HUMAN, SELLER => HUMAN, OBJECT => CONCRETE]
RESULT < NON-ATOMIC-RELATION [CAUSE => RELATION, EFFECT => RELATION]
```

In an instance of a concept, called a token (e.g. E1 as an instance of TRANSACTION, which means that TRANSACTION is the "class" of E1), the values of the arguments are given: they are instances of concepts. For example, H1 for the buyer of E1, where H1 is an instance of HUMAN as illustrated below.

```
E1 =: TRANSACTION [buyer => H1, seller => H2, object => O1]
```

An instance of an atomic (resp. non atomic) relation is called an atomic (resp. non atomic) event. I consider that the module in charge of linguistic choices in a generation system takes as input an event (enriched with pragmatic information) and produces as output a text made up of one or several sentences.

I first present (Section 2) the generation of an atomic event in a lexicalized system. This well known case allows me to describe the linguistic data bases and to give a rapid overview of the generation process. Next, I examine how to express a discourse relation: Section 3 presents the cases where it is lexicalized, Section 4 when it is not lexicalized. Section 5 concludes and presents briefly a formalism to write a lexicalized generation system.

2 Lexicalizing an atomic relation with a verb, an adjective or a noun

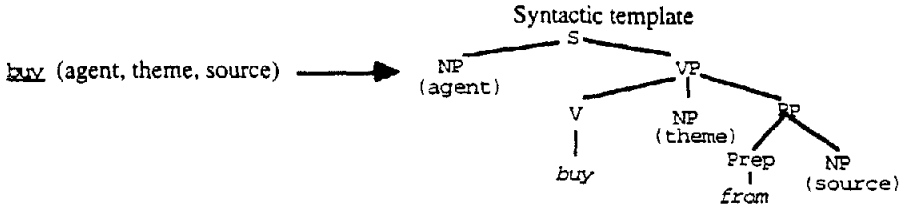
2.1 Linguistic data bases

The linguistic data are divided up in three levels: conceptual, semantic and syntactic. The conceptual level (supposedly language independent) have been described above. A concept is lexicalized in a target language as one or several predicates. For example, TRANSACTION can be lexicalized in English as sell or buy (a predicate is underlined). At the (lexical) semantic level, a predicate has an argument structure, i.e. a list of arguments which are understood as thematic roles: buy(agent, theme, source) or sell(agent, theme, goal). There exist correspondence rules between the arguments of a concept and the arguments of a predicate lexicalizing it, e.g. the BUYER of TRANSACTION corresponds to

the goal of sell. The interface between the conceptual and semantic levels is made through Lexical data Bases (noted as LBs). A concept C0 is associated with a LB as illustrated below with LB (TRANSACTION).

LB (TRANSACTION) ={ buy (agent ↔ BUYER, theme ↔ OBJECT, source ↔ SELLER)
sell (agent ↔ SELLER, theme ↔ OBJECT, goal ↔ BUYER) }

A predicate is realized in syntax in a lexicalized syntactic template which consists of a head with its subcategorization frame, i.e. its arguments considered as XPs. The interface between the semantic and syntactic levels is made through projection rules as illustrated below for buy.



As a projection rule associates a predicate with a unique syntactic template, a direct interface between the conceptual and syntactic levels seems possible, as schematically resumed below, see (Stone & Doran 1997).

TRANSACTION –> { NP(BUYER) *buy* NP(OBJECT) *from* NP(SELLER)
NP(SELLER) *sell* NP(OBJECT) *to* NP(BUYER) }

However, alternations must be taken into account. One can decide to use a verb in the passive (see when in the next section). Such information can be featurized at the semantic level: a predicate can be marked with a set of features like [Passive = +]. Such alternation features are taken into account in projection rules, as illustrated below.

buy [Passive = +] (agent, theme, source) --> NP (theme) *be bought from* NP (source) *by* NP (agent)

Considering that a predicate can be marked with alternation features means that an LB associated with a concept takes into account only lexicalization and not syntactic alternations. Therefore, an LB does not have to record all the predicates lexicalizing the concept, and for each predicate, all its syntactic constructions.

In the LB associated with an atomic relation, the predicates are either verbal, adjectival or nominal. For example, the LB of LEAVE[LEAVER] includes leave(agent) and departure(agent). Thereby, an atomic event can be expressed either in a sentence or an NP. This is required to produce either *Ted wants Mary to leave* or *Ted wants Mary's departure*, from an instance of LEAVE embedded in an instance of WANT. XP disjunctions (noted as "/") are licensed in syntactic templates. For example, as the theme of *want* can be realized as an S or NP, its syntactic template is the following (in a flat structure): *NP* (experiencer) *want S/NP* (theme).

2.2 Rapid overview of the generation process

Let us examine briefly how to generate an atomic event (i.e. how to generate a token such as E1= TRANSACTION[H1, H2, O1]) into an S or NP given the linguistic data bases presented in 2.1. The first step consists in lexicalizing the class C0 (a concept) of the token, i.e. in selecting a predicate in LB (C0). For this selection, the predicates are equipped with tests which take into account conceptual and pragmatic factors. These factors may add one or several alternation feature(s). For example, a SHOOTING in which the target is missed can be expressed by shoot in the conative alternation (Levin 1993): *Ted shot at the rabbit (but he missed it).* Or, for E1 in the case the person referred to by H1 must be the focus, either buy without alternation *(Mary bought a book from John)* or sell with the dative and passive alternations *(Mary was sold a book by John)* can be selected[1]. When a predicate has been selected[2] it is instantiated as illustrated in the following structure supposed to be selected for E1 when H1 is in focus: sell [Dative = +] [Passive = +] (agent ↔ H2, theme ↔ O1, goal ↔ H1). Next the generation process is based on a recursivity principle. The global lexicalization of E1 is the previous structure in which the tokens are lexicalized recursively. The recursion stops roughly because things are generated typically into "constants", i.e. predicates without argu-

[1] This requires the syntactic functions to be known, therefore that the lexicalization process can access syntactic information. Syntactic functions must also be known for optional arguments. For example, if the BUYER in an instance of TRANSACTION is not specified, sell [without-goal = +] is selected *(John sold a book)*, or possibly buy [passive = +] [without-agent = +] *(A book was bought from John).*

[2] In fact, it is preferable that the lexicalization process leads to a list of predicates in order of preference, so as to reduce backtracking in case of incompatibility with future decisions, but the data flow is not within the scope of this paper.

ments (corresponding to nouns without arguments). At the end of this recursive process, a structure is produced in which all the lexical items with a semantic content have been chosen and possibly marked with alternation features. The projection of this lexicalized structure into syntax is then achieved by means of the semantic-syntax interface, i.e. projection rules. This leads to a lexicalized syntactic tree from which a text is derived thanks to the application of syntactic rules and to low level operations.

In summary, the generation of a token relies on recursive lexicalization of its class and arguments. Let us now examine the linguistic knowledge needed for realizing a discourse relation, first to the cases where a discourse relation is lexicalized, second when it is not so.

3 Expressing a discourse relation by a lexical item

Subordinating conjunctions are cue phrases frequently used to link two sentences: *because* lexicalizes EXPLANATION (1a), *before* and *after* lexicalizes SEQUENCE (1b)-(1d).

(1)a The metal is flat because Ted hammered it.
b Ted hammered the metal before melting it.
c Ted cried after Mary's departure.
d After hammering the metal, John melted it.

I consider subordinating conjunctions as predicates with two arguments (called simply arg1 and arg2). Therefore, the LB of SEQUENCE[1ST-EVENT, 2ND-EVENT] includes the two following elements: before (arg1 ↔ 1ST-EVENT, arg2 ↔ 2ND-EVENT) and after (arg1 ↔ 2ND-EVENT, arg2 ↔ 1ST-EVENT). At the syntactic level, a subordinating conjunction generally subcategorizes either for an S or an NP, (1c) and (1d) with *after*. This means that the syntactic template associated with after includes a category disjunction. The anteposition of a subordinate clause, (1d), is considered as an alternation of the conjunction and is represented with the alternation feature [Anteposition = +] added to the predicate.

Adverbials such as *therefore* or *afterwards* are other cue phrases frequently used to link two sentences. They lexicalize discourse relations, e.g. *therefore* lexicalize RESULT, (2a), and *afterwards* SEQUENCE, (2b) and (2c).

(2)a Ted hammered the metal. Therefore, it is flat.
b Ted hammered the metal. Afterwards, he melted it.
c Ted hammered the metal, and afterwards he melted it.

I consider also those adverbials as predicates with two arguments. So the LB of SEQUENCE includes afterwards (arg1 ↔ 1ST-EVENT, arg2 ↔ 2ND-EVENT). The predicate afterwards is associated with a syntactic template whose root is the category T(as TEXT) and whose leaves are: S(arg1). *Afterwards* S(arg2). The use of *afterwards* in a sentence, (2c), is considered as an alternation.

The claim that those adverbials are predicates is linguistically motivated. It extends the lexical approach advocated for S or NP to T. It bridges the (artificial) gap between sentences and texts. This gap is artificial for several reasons, among them, the fact that the same discourse relation can be expressed in a T or an S, as shown in (2b) and (2c) and in the examples below.

Another way to lexicalize a discourse relation is to use an "operator verb" such as *cause* for RESULT (3a), or *follow* or *succeed* for SEQUENCE (3b). To generate (3a) or (3b), it is enough to include cause (arg1 ↔ CAUSE, arg2 ↔ EFFECT) in the LB of RESULT and follow (arg1 ↔ 2ND-EVENT, arg2 ↔ 1ST-EVENT) in the LB of SEQUENCE. Moreover, a discourse relation can also be expressed in a nominalization of an operator verb, (3c).

(3)a Ted's hammering the metal caused it to be flat.
b Mary's arrival followed / succeeded Ted's departure.
c the succession of Ted's departure and Mary's arrival (totally upset Fred)

In summary, a discourse relation can be lexicalized by a subordinating conjunction, an adverbial, an operator verb, or the nominalization of an operator verb. A text, sentence or nominal phrase is then produced. For all these cases, the generation process is based on recursive lexicalization. Let us examine cases where a discourse relation is not lexicalized.

4 Expressing a discourse relation without lexicalizing it

In a *S1. S2* discourse, there is no lexical item that indicates which discourse relation is involved. The fact that (4a) expresses a RESULT while (4b) expresses an EXPLANATION is based on a) the core meanings of *S1* and *S2*, b) the tenses and aspectual properties of each sentence, and c) extra-linguistic knowledge such as the "Push Causal Law" (Lascarides & Asher 1991) for (4b).

(4)a Ted hammered the metal. It is flat
b Ted fell. John pushed him.

At the semantic level, I propose for a *S1. S2* discourse the use of a ⊕ predicate. This predicate has the particularity to have no lexical head. It can be used in several LBs, e.g. in LB(RESULT) with ⊕ (arg1 ↔ CAUSE, arg2 ↔ EFFECT) or in LB(EXPLANATION) with ⊕ (arg1 ↔ EFFECT, arg2 ↔ CAUSE) It is associated with a syntactic template whose root is T and whose leaves are: S(arg1). S(arg2).

The *S1, V-ing* ... . sentences in (5) express a non atomic event without any lexical item to express the underlying discourse relation: (5a) expresses a CIRCUMSTANCE while (5b) expresses a RESULT. So, to generate (5), I propose a ⊗ predicate which is similar to ⊕ except that ⊗ builds a sentence by concatenating two clauses, the second one being in the gerundive form.

(5)a Ted went out of the restaurant, moaning.
b Ted hammered the metal, flattening it.

There is no room left to explain how to generate a resultative construction *(Ted hammered the metal flat)* which expresses a RESULT without item lexicalizing it. Let us just say that this can be achieved by a "function" inspired from (Jackendoff 1993). In summary, a non atomic event can be expressed in a S or T without any item lexicalizing the underlying non atomic relation. It seems that this situation has no equivalent for an atomic event: the underlying atomic relation is always lexicalized (even if it leads to VP ellipsis or gapping).

5 Conclusion

This paper has shown that an atomic event is expressed in a NP or S (Section 2), a non atomic event in a NP, S or T, with an item lexicalizing it (Section 3) or not (Section 4). The consequences for text generation are twofold:
• The conceptual representation of a text should be a tree structure whose non terminal nodes are non atomic relations, and whose leaves are conceptual representations of atomic events (based on atomic relations). These leaves do not correspond to the conceptual representations of the sentences or clauses of the text.
• From such a conceptual representation (enriched with pragmatic information), the generation process should not be modularized into "text planning" and "sentence planning", as generally admitted (Reiter 1994). The only possible modularization is a component for non atomic events and another one for atomic events. In a lexicalized system, the generation of atomic and non atomic events can be based on the same process, i.e. recursive lexicalization.

Formalism and implementation. A formalism for a lexicalized generation system must obviously be inspired from a formalism designed for lexicalized grammar in analysis. Among other advantages, this make it possible to use an already existing grammar for the syntactic level. Among the existing lexicalized grammars, TAG has long been seen as especially well suited for text generation (Joshi 1987). Hence my choice of designing a generation formalism inspired by TAG and called G-TAG (Danlos 1995, 1998). G-TAG has been first implemented in ADA (Meunier 1997) and used in three technical domains (chemical, software, and aeronautic). The TAG grammar used for French is the one written by (Abeillé 1991). The elementary tree families are automatically generated out of a hierarchical representation (Candito 1996). G-TAG is currently re-implemented in Java in a multi-agent structure (Meunier & Reyes 1998).

References

Abeillé,. A. (1991) *Une grammaire lexicalisée d'arbres adjoints pour le français,* PhD Thesis, Université Paris 7.
Candito, M.H. (1996) "A principle-based hierarchical representation of LTAGs", in *Proceedings of "COLING'96".*
Danlos, L. (1995) "Présentation de G-TAG, un formalisme pour la génération de textes", *Actes de* TALN-95, Marseille.
Danlos, L. (1998) "G-TAG: a Formalism for Text Generation inspired from Tree Adjoining Grammar: TAG issues", *in* A. Abeillé and O. Rambow (eds.), *Tree-adjoining Grammars,* CSLI, Stanford.
Jackendoff, R. (1993) *Semantic Structures,* MIT Press, Cambridge MA.
Joshi, A. (1987) "The relevance of tree adjoining grammar to generation" , in G. Kempen (ed), *Natural Language Generation: New Results in Artificial Intelligence, Psychology and Linguistics,* Martinus Nijhoff Publishers.
Lascarides A. and Asher N. (1991) "Discourse Relations and Defeasible Knowledge", in *Proceedings of ACL'91,* Berkeley.
Levin, B. (1993), *English Verb Classes and Alternations,* The University of Chicago Press, Chicago.
Mann W. and Thomson S. (1988) "Rhetorical Structure Theory: Toward a functional theory of text organization."in *Text: An Interdisciplinary Journal for the Study of Text,* vol. 8 n° 2.
Meunier, F. (1997) *Implémentation d'un formalisme de génération inspiré de TAG,* PhD Thesis, Université Paris 7.
Meunier, F., Reyes, R. (1998) "CLEF: Computed Lexical-Choice Extend Formalism", Rapport n° 4, Thomson CSF, Paris.
Reiter , E. (1994) "Has a consensus NL generation architecture appeared, is it psycholinguistically plausible?", *INLG'94.*
de Smedt K., Horacek H., Zock M., (1996) "Architectures for Natural Language Generation: Problems and Perspectives", in G. Adorni and M. Zock (eds) *Trends in Natural Language Generation,* Springer-Verlag.
Stone, M., Doran, C. (1997) "Sentence Planning Using TAG", in *Proceedings ACL/EACL '97.*