Argument and Computation
Vol. 3, No. 1, March 2012, 49–82

Taylor & Francis
Taylor & Francis Group

Processing natural language arguments with the <TextCoop> platform

Patrick Saint-Dizier*

IRIT-CNRS, 118 route de Narbonne 31062 Toulouse, France

(*Received 15 June 2011; final version received 31 January 2012*)

In this article, we first present the <TextCoop> platform and the Dislog language, designed for discourse analysis with a logic and linguistic perspective. The platform has now reached a certain level of maturity which allows the recognition of a large diversity of discourse structures including general-purpose rhetorical structures as well as domain-specific discourse structures. The Dislog language is based on linguistic considerations and includes knowledge access and inference capabilities. Functionalities of the language are presented together with a method for writing discourse analysis rules. Efficiency and portability of the system over domains and languages are investigated to conclude this first part. In a second part, we analyse the different types of arguments found in several document genres, most notably: procedures, didactic texts and requirements. Arguments form a large class of discourse relations. A generic and frequently encountered form emerges from our analysis: 'reasons for conclusion' which constitutes a homogeneous family of arguments from a language, functional and conceptual point of view. This family can be viewed as a kind of proto-argument. We then elaborate its linguistic structure and show how it is implemented in <TextCoop>. We then investigate the cooperation between explanation and arguments, in particular in didactic texts where they are particularly rich and elaborated. This article ends with a prospective section that develops current and potential uses of this work and how it can be extended to the recognition of other forms of arguments.

Keywords: natural language processing of arguments; discourse processing; logic programming

1. Introduction

Automatically identifying arguments in texts and capturing their conceptual dimensions is an extremely difficult challenge. There are multiple reasons for this, in particular the linguistic realisations of arguments are very diverse and very difficult to accurately characterise. Furthermore, domain or personal knowledge is often needed to identify that a given text span has an argumentative dimension. Linguistic marks may not be accurate and discriminatory enough to allow the identification of arguments in any situation. Furthermore, expressions denoting propositional attitudes or beliefs about certain facts may also confer the status of arguments to these facts which would not otherwise have been considered as arguments.

The profusion of expressions and forms which can be interpreted at varying degrees as arguments make it difficult to elaborate the kind of language representation formalism, knowledge representation and associated processing strategies which are required to process these forms. In an attempt to organise the different types of arguments, the logical roles and the communicative goals, argument targets have been developed and certainly contribute to this long-term challenge, e.g. Walton, Reed, and Macagno (2008), but robust language-processing formalisms integrating knowledge and inferences still remain to be developed. This is the main contribution of this paper.

*Corresponding author. Email: stdizier@irit.fr

ISSN 1946-2166 print/ISSN 1946-2174 online

http://dx.doi.org/10.1080/19462166.2012.663539
http://www.tandfonline.com

In this introductory section we first discuss our general aims in terms of argument analysis. We then introduce the current challenges and results in discourse analysis since arguments form a major class of discourse relations. Furthermore, arguments are often closely associated with various types of discourse structures, in particular those related to explanation, which contribute to their strength, communicative goals and persuasion effects (e.g. Reed and Long 1997; Masthoff, Reed, and Grasso 2008). This view is central in the pragma-dialectical theory (van Eemeren and Grootendorst 1984, 1992; van Eemeren 2002) which considers the entirety of argumentation as a discourse activity.

1.1. *Argument analysis: some methodological considerations*

Given the difficulty in developing a general model for argument analysis and extraction, our strategy is to focus on a relatively large and frequently encountered group of argument structures that form a homogeneous family from a linguistic, functional and conceptual point of view. Besides developing a precise linguistic analysis and a processing strategy for that group of structures, our aim is also to provide elements of a methodology so that other families of arguments can be investigated and developed following similar principles. In this paper, we focus on the generic form:

`Reasons supporting conclusion` as in:

(1) *I like this hotel because because it feels like home.*
(2) *Carefully plug-in the mother card vertically otherwise you risk damage to the connectors.*

In these two examples, the causal proposition expresses the reasons of the main proposition, called the conclusion. This type of construction is frequently encountered with a large diversity of language realisations in several textual genres, while sharing several conceptual features. This structure is found for example in opinion texts (expressing why consumers are happy or unhappy with a certain product), or in instructional texts such as didactic texts, procedures or requirements. In procedures, which basically have an injunctive style, reasons often have a form of advice or warning; in requirements, which are basically prescriptive, they explain or justify the constraints which are imposed:

(2a) *The system must respond in less than 2 seconds, otherwise customers will not buy it.*

Other classes of applications where arguments play a central role include question-answering (e.g. answering why, questions, or getting hints on how to realise a task, Canitrot, Roger, and Saint-Dizier 2011) and the planning aspect of natural language generation of texts (e.g. Rosner and Stede 1992).

Considering the large diversity of linguistic realisations of this generic form, our strategy, to guarantee a robust and stable linguistic analysis, was to focus on texts which contain proto-typical uses, in terms of forms and contents. Texts produced by professional authors certainly form the best sample for this study. We then investigated texts where style and language issues were less central, such as large public texts found on the web. The idea is to have a linguistically adequate formalism and a robust set of rules that accurately describe the structure of arguments found in texts with proto-typical argument structures while developing devices in the processing strategy that make rules somewhat flexible in order to be able to process more 'dirty' texts.

An important issue is the role played by various types of knowledge sources in the identification of arguments. Our investigations aim at developing a cooperation between language, knowledge and inference issues to get a system that can accommodate the different facets of argument analysis. For that purpose, a framework such as logic programming that integrates a logic-based view of language processing with inferential capabilities seems to be a good option.

1.2. *Discourse analysis challenges*

Discourse structure analysis is a very challenging task because of the large diversity of discourse structures, the various forms they take in language and the impact of knowledge and pragmatics in their identification (Longacre 1982; Keil and Wilson 2000). Recognising discourse structures cannot in general only be based on purely lexical or morphosyntactic considerations: subtle kinds of knowledge associated with reasoning schemas are often necessary. These latter capture the various facets of the influence of pragmatic factors in our understanding of texts (Kintsch 1988; Di Eugenio and Webber 1996). The importance of structural and pragmatic factors does depend on the type of relation investigated, on the textual genre and on the author and targeted audience. In our context, technical texts are obviously much easier to process than free-style texts.

Rhetorical structure theory (RST) (Mann and Thompson 1988, 1992) is a major attempt to organise investigations in discourse analysis, with the definition of 22 basic structures. Since then, almost 200 relations have been introduced which are more or less clearly defined. Background information about RST, annotation tools and corpora are accessible at http://www.sfu.ca/rst/. A recent overview is developed in Taboada and Mann (2006). Very briefly, RST poses that coherent texts consist of minimal units, which are all linked with each other, recursively, through rhetorical relations. No unit is left pending: all units are connected to others. Some text spans appear to be more central to the text purpose, these are called nuclei (or kernels), whereas others are somewhat more secondary, they are called satellites. Satellites must be associated with nuclei. Relations between nuclei and satellites are one-to-one or one-to-many. For example, an argument conclusion may have several supports, possibly with different orientations. Conversely, a given support can be associated with several distinct conclusions.

For example, in the sentence: (4) *To prepare such a tart, you need red fruits, for example strawberries or raspberries,...* the discourse relation 'illustration' is composed of a nucleus:

red fruits and a satellite, which is the list of such fruits: *strawberries, raspberries*.

Note that these two structures are not necessarily adjacent. Similarly, *prepare a tart* and *red fruits* are in a 'prerequisite' relation, where the latter is the satellite, the nucleus being expressed as a goal.

The literature on discourse analysis is particularly abundant from a linguistic point of view. Several approaches, based on corpus analysis with a strong linguistic basis are of much interest for our purpose. Relations are investigated together with their linguistic marks in works such as Delin, Hartley, Paris, Scott, and Vander Linden (1994), Marcu (1997), Marcu (2002), Kosseim and Lapalme (2000) with their usage in language generation in Rosner and Stede (1992), and in Saito, Yamamoto, and Sekine (2006) with an extensive study on how marks can be quite systematically acquired. A deeper approach is concerned with the cognitive meaning associated with these relations, how they can be interpreted in discourse and how they can trigger inferential patterns (von Wright (2004), Moeschler (2007) and Fiedler (2001) just to cite a few works).

Within computational linguistic circles, RST has been mainly developed in natural language generation for content planning purposes, e.g. Kosseim and Lapalme (2000) and Reed and Long (1998). Besides this area, Marcu (1997, 2000) developed a general framework and efficient strategies to recognise a number of major rhetorical structures in various kinds of texts. The main challenges are the recognition of textual units and the identification of relations that hold between them. The rhetorical parsing algorithm he introduced relies on a first-order formalisation of valid text structures which obey a number of structural assumptions. These, however, seem to be somewhat too restrictive w.r.t. our observations. In particular, our observations show that the following assumptions are too restrictive: relations occur between non-overlapping text spans, relations are either vertical or horizontal (they can involve non-parent nodes), text structure is a binary-branching tree in most cases (we have many situations with more than two nodes). His work is based on a number of psycholinguistic investigations (Grosz and Sidner 1986) which show that

discourse markers are used by human subjects both as cohesive links between adjacent clauses and as connectors between larger textual units. An important result is that discourse markers are used consistently with the semantics and pragmatics of the textual units; they connect and they are relatively frequent and unambiguous.

1.3. *Argumentation and discourse analysis*

We consider that the various types of arguments form a family of rhetorical relations. In our case study, a conclusion is a nucleus and a support is a satellite: this articulation is typical to discourse relations. From a syntactic point of view, a support is a right or left adjunct to a conclusion, which is its head. In other terms, a support must be connected to a conclusion to be syntactically acceptable.

For example, in the discourse relation 'argument' as illustrated in example (1) above, *I like this hotel* is the kernel of the relation, while *because I feel like being home* is the satellite. In general, kernels have an autonomous meaning, while satellites get their full meaning in context from their association with a kernel. This whole sentence can then be a kernel or a satellite for another statement. The sentence' (3a) *John said 'I like this hotel because it feels like home' however being home is not a source of pleasure for me* illustrates the 'contrast' relation which is composed of two rather opposite views. The connector 'however' connects the kernel with its satellite. In turn, this latter satellite could include the reasons of such an opposite view, e.g. *because this means home work, more noise and people around you, etc.* and be the kernel of a causal relation embedded into the contrast relation.

An argumentation framework is a complex graph structure which exhibits the various conceptual relations that hold between arguments (support and attack are the most well known, but there are many others, e.g. of a rhetorical nature). We will not develop this aspect here since there is an abundant, well-written literature on argument typology, argument schemes, etc. this is also beyond the scope of this article. Walton et al. (2008), among others, is a good analysis of the functional and logical aspects of argumentation. A synthesis of computational models for processing arguments is reported in Reed and Grasso (2007). Logical aspects of argumentation are particularly well developed in general, as in, e.g. Dung (1995), Dung, Kowalski, and Toni (2006), Amgoud, Parsons, and Maudet (2001), Amgoud, Bonnefon, and Prade (2005) and Caminada and Amgoud (2007).

The distinction between arguments and other discourse relations may not be straightforward. First, the schema 'reasons supporting conclusions' could be globally analysed as the 'evidence' relation, this latter being defined by: *an evidence relation is a logical relation in which a proposition(s) is intended to increase the addressee's assurance of another proposition(s)*. The definition of this relation is clearly very vague, as we understand it. We view it as a subtype of the elaboration relation, which is in our view a kind of proto-relation. The evidence relation may share similar language realisations with arguments in the surface, but its contents are substantially different, for example, it does not include any notion of warning or advice, support or attack, or even of logical consequence, which are at the heart of the family we are analysing.

Our goal is to define proto-typical language forms for a given set of arguments related to the family: `Reasons supporting conclusion`. In this article, we first consider procedural texts, which abound in warnings and advice, which are instances of that family. These occur in isolation: they do not attack or support each other. We then consider a special subgenre of procedures, didactic texts, which merges instructions with a large variety of arguments with various forms of explanation (Bourse and Saint-Dizier 2011).

If we consider rhetorical relations from a language point of view over various types of texts, we note that a number of these relations (e.g. illustration, reformulation, concession) have a

precise definition and scope and very recurrent language forms, with well-identified linguistic marks, while others have definitions with a larger scope and linguistic forms with rather shallow linguistic marks. This is the case in particular for the elaboration relation and for various types of arguments. These latter relations have a much larger conceptual scope than, e.g. illustration or reformulation, where their definition and realisations have several facets. Similar to Dowty (1991) for thematic roles, we consider that these complex relations are *proto-relations* which need to be defined according to their various facets via constitutive properties. Defining the facets of proto-relations associated with argument types is ongoing; it is beyond the scope of this article.

1.4. *Article organisation*

This article is organised as follows:

- from investigations of the structure of arguments in corpora, and discourse structures more generally, we first describe the features of the Dislog language and then the <TextCoop> engine and its environment. The approach being based on generative syntax principles, we introduce quite a powerful rule system, constrained by various types of restrictive principles.
- we then develop the treatment of complex argument realisations as found in rather free-style texts, and present principles on how to write rules, in order to enhance reusability and extensions to other structures,
- to end this first section, preliminary considerations on the use of knowledge to identify and interpret arguments and discourse relations in general are introduced. This is an open and complex problem with few contributions so far.
- A second part is devoted to the analysis of advice and warnings in procedural texts where large rule samples are given. The section ends by an accurate evaluation of the rule coverage and accuracy and the system performances. This allows us to develop a robust set of rules to process the family of arguments introduced above.
- In a third part we investigate the relations between arguments and various forms of discourse relations dedicated to explanation, and show how they interact. Developing an explanation theory is, again, a very open problem.
- The last section is devoted to the perspectives and applications under development. In particular, we show that the type of generic structure (proto-argument) we investigate in this paper occurs very frequently in a large diversity of types of texts, making this relation crucial for argument analysis and the development of argumentation networks.

2. The <TextCoop> platform and the Dislog language

Our investigations on the forms in language taken by arguments have motivated the definition of the <TextCoop> platform and the Dislog language, which allows us to renew the Logic Grammar tradition, applied to discourse. In this section, we first introduce the principles which are at the basis of <TextCoop>, taking into account the specificities of discourse structures, in contrast with sentence structures. We then introduce the Dislog language (Dislog stands for discourse in logic or discontinuities in logic since discourse structure analysis is based on marks which often are in long-distance dependency relations). The <TextCoop> platform is then presented, in particular the <TextCoop> engine and the linguistic architecture of the system. Dislog is further motivated and illustrated in the next sections devoted to argument analysis.

There are at the moment a few well-known and widely used language-processing environments. They are essentially used for sentence processing, not for discourse analysis. The reasons are

essentially that the sentence level and its substructures are the crucial level of analysis for a large number of applications such as information extraction, opinion analysis based on noun modifiers or machine translation. Discourse analysis turns out to be not so critical for these applications. However, applications such as summarisation (Marcu 2000) or question-answering do require an intensive discourse analysis level, as shown in Jin and Simmons (1987).

Dedicated to sentence processing, let us note the GATE platform (http://gate.ac.uk/) which is widely used and the Linguastream (http://www.linguastream.org) system which is based on a component architecture, making the system really flexible. Besides some specific features for simple aspects of discourse processing, none of these platforms allow the specifications of rules for an extensive discourse analysis nor the introduction of reasoning aspects, which is essential to introduce pragmatic considerations into discourse processing. GATE is used, e.g. for semantic annotation, corpus construction, knowledge acquisition and information extraction, summarisation and investigations around the semantic web. It also includes research on audio visual and language connections. Linguastream has components to mainly deal with part of speech and syntactic analysis. It also handles several types of semantic data with a convenient modular approach. It is widely used for corpus analysis. The GETARUNS system (http://project.cgm.unive.it/getaruns.html), based on the LFG grammar approach, has some capabilities to process simple forms of discourse structures and argumentation analysis. Finally, Marcu (2000) developed a discourse analyser for the purpose of automatic summarisation. This system is based on the RST assumptions which are not always met in texts, as developed in the section below.

On a very different perspective, and also inspired by sentence syntax, two approaches based on Tree Adjoining Grammars (TAGs) (Gardent 1997; Webber 2004) extend the formalism of TAGs to the processing of discourse structures via tree anchoring mechanisms. The approach remains essentially lexically based and is aimed at connecting propositions related by various discourse connectors or at relating text spans which are in a referential relation.

2.1. *Some linguistic considerations*

Most works dedicated to discourse analysis have to deal with the triad: discourse function identification, delimitation of its textual structure (boundaries of the discourse unit) and structure binding. By function, we mean a kernel or a satellite of a rhetorical relation, e.g. an illustration, an illustrated expression, an elaboration, or the elaborated expression, a conditional expression, a goal expression, etc. Functions are realised by textual structures which need to be accurately delimited. Functions are not stand alone: they must be bound based on the kernel–satellite or kernel–kernel principle.

<TextCoop> and Dislog are based on the following considerations:

- structure identification: requires the specification of the lexical, syntactic, morphological, punctuation and possibly typographic marks (Longacre 1982; Luc, Mojahid, Virbel, Garcia-Debanc, and Pery-Woodley, 1999) that allow the identification of discourse functions. In general, the recognition of satellite functions is easier than the recognition of their corresponding kernel(s) because they are more strongly marked. For example, it is quite straightforward to recognise an illustration (although we defined 20 generic rules that describe this structure), but identifying the exact text span which is its kernel (i.e. what is illustrated) is much more ambiguous. Similarly, the support of an argument is marked much more explicitly than its conclusion.
- structure delimitation: most of the literature on discourse analysis dedicated to the delimitation of discourse units refers either to elementary discourse units (EDUs) (Schauer 2006) or to the vague notion of text span. When identifying discourse structures in texts, and

in particular when attempting to identify arguments, finding their textual boundaries is very challenging. In addition, contrary to the assumptions of RST (e.g. Grosz and Sidner 1986; Marcu 2000) partly overlapping textual units can be involved in different discourse relations.

- binding discourse units: the last challenge is, given a set of discourse functions and their corresponding discourse units in a text, to identify relations between them. For example, argument conclusions and supports are not necessarily contiguous: discourse functions may be inserted between them. We also observed a number of one-to-many relations, besides the standard one-to-one relations: an argument conclusion may have several supports, possibly with different orientations. As a consequence, the principle of textual contiguity cannot be applied systematically. For that purpose, we have developed a principle called selective binding, which is also found in formal syntax to deal with long-distance dependencies or movement theory.

2.2. *Some foundational principles of <TextCoop>*

The previous subsection has outlined a number of challenging language-processing situations. The necessity of a modular approach, where each aspect of discourse analysis is dealt with accurately in a specific module, while keeping open all the possibilities of interaction (or concurrency) between modules has led us to consider some simple elements of the model of generative syntax (a good synthesis is given in Lasnik and Uriagereka 1988). As shall be discussed later, we introduce:

- productive principles, which have a high level of abstraction, which are linguistically sound, but which may be too powerful,
- restrictive principles, which limit the power of the first in particular on the basis of well-formed constraints.

Another foundational feature is an integrated view of marks used to identify discourse functions, merging lexical objects with morphological functions, typography and punctuation, syntactic constructs, semantic features and inferential patterns that capture various forms of knowledge (domain, lexical, textual). <TextCoop> is the first platform that offers this view within a logic-based approach. If machine learning is a possible approach for sentence processing, where interesting results have emerged, it seems not to be so successful for discourse analysis (e.g. Carlson, Marcu, and Okurowski 2001). This is due to two main factors: (1) the difficulty in annotating discourse functions in texts (Saaba and Sawamura 2008) and the high level of disagreement between annotators and (2) the large non-determinism of discourse structure recognition where identification marks are often immersed in long spans of text of no or little interest. For these reasons, we adopted a rule-based approach. Rules are hand coded, based on corpus analysis using bootstrapping tools.

Dislog rules basically implement the productive principles. They are composed of three main parts:

- a discourse function identification structure, which basically has the form of a rule or a pattern,
- a set of calls to inferential forms using various types of knowledge, these forms are part of the identification structure, they may contribute to solving ambiguities, they may also be involved in the computation of the resulting representation or they may lead to restrictions. This is developed in Section 2.5,
- a structure that represents the result of the analysis: it can be a simple XML structure, or any other structure a priori such as an element of a graph or a dependency structure. More complex representations, e.g. based on primitives, can be computed using a rich semantic

lexicon. This is of much interest since our analysis is oriented towards a conceptual analysis of discourse, and, in particular, semantic aspects of arguments.

Besides rules, Dislog allows the specification of a number of restrictive principles, e.g. dominance, precedence, exclusion, etc.

2.3. *The structure of Dislog rules*

Let us now introduce in more depth the structure of Dislog rules. Dislog follows the principles of logic-based grammars as implemented three decades ago in a series of formalisms, among which, most notably: Metamorphosis Grammars (Colmerauer 1978), Definite Clause Grammars (Pereira 1981) and Extraposition Grammars (Pereira 1981). These formalisms were all designed for sentence parsing with an implementation in Prolog via a meta-interpreter or a direct translation into Prolog (Saint-Dizier 1994). The last two formalisms include a simple device to deal with long distance dependencies. Various processing strategies have been investigated in particular bottom-up parsing, parallel parsing, constraint-based parsing and an implementation of the Earley algorithm that merges bottom-up analysis with top-down predictions. These systems have been used in applications, with reasonable efficiency and a real flexibility to updates, as, e.g. reported in Gazdar and Mellish (1989).

Dislog adapts and extends these grammar formalisms to discourse processing, it also extends the regular expression format which is often used as a basis in language-processing tools. The rule system of Dislog is viewed as a set of productive principles.

A rule in Dislog has the following general form, which is globally quite close to Definite Clause Grammars in its spirit:

L(Representation) $\rightarrow$ R, {P}. where:

- L is a non-terminal symbol.
- Representation is the representation resulting from the analysis, it is in general an XML structure with attributes that annotates the original text. It can also be a partial dependency structure or a more formal representation.
- R is a sequence of symbols as described below, and
- P is a set of predicates and functions implemented in Prolog that realise the various computations and controls, and that allow the inclusion of inference and knowledge into rules. These are included between curly brackets as in logic grammars to differentiate them from grammar symbols.

R is a finite sequence of the following elements:

- terminal symbols that represent words, expressions, punctuations, various existing html or XML tags. They are included between square brackets,
- preterminal symbols: are symbols that are derived directly into terminal elements. These are used to capture various forms of generalisations, facilitating rule authoring and update. Symbols can be associated with a type feature structure that encodes a variety of aspects of those symbols, from morphology to semantics,
- non-terminal symbols, which can also be associated with type feature structures. These symbols refer to 'local grammars', i.e. grammars that encode specific syntactic constructions such as temporal expressions or domain-specific constructs. Non-terminal symbols do not include discourse structure symbols: Dislog rules cannot call each other, this feature is dealt with by the selective-binding principle, which includes additional controls. A rule in Dislog thus basically encodes the recognition of a discourse function taken in isolation.

- optionality and iterativity marks over non-terminal and preterminal symbols, as in regular expressions,
- gaps, which are symbols that stand for a finite sequence of words of no present interest for the rule which must be skipped. A gap can appear only between terminal, preterminal or non-terminal symbols. Dislog offers the possibility to specify in a gap a list of elements which must not be skipped: when such an element is found before the termination of the gap, then the gap fails. The length of the skipped string can also be controlled.
- a few meta-predicates to facilitate rule authoring.

Similar to DCGs and to Prolog clause systems, it is possible and often necessary to have several rules to describe the different realisations of a given discourse function. These all have the same identifier *L*, as it is the case, e.g. for NPs or PPs. A set of rules with the same identifier is called a *cluster of rules*. Rule clusters are executed sequentially by the <TextCoop> engine following an order given in a *cascade*.

2.4. *Dislog advanced features*

In this section, we describe the features offered by the Dislog language that complement the grammar rule system. These mostly play the role of restrictive principles. At the moment we have three sets of devices: selective-binding rules to link discourse units identified by the rule system, correction rules to revise structures which may have, e.g. some erroneously placed tags, and concurrency statements that allow a correct management of clusters of rules. Concurrency statements are closely related to the cascade system. They are constrained by the notion of bounding node, which delimits the text portion in which discourse units can be bound. Similarities with sentence formal syntax are outlined when appropriate, however, phenomena are substantially different.

2.4.1. *Selective-binding rules*

Selective-binding rules is the means offered by Dislog to construct hierarchical discourse structures from the elementary ones identified by the rule system. Selective-binding rules allow to link two or more already identified discourse functions. The objective is, e.g. to bind a kernel with a satellite (e.g. an argument conclusion with its support) or with another kernel (e.g. concessive or parallel relations). Related to this latter situation is the binding of two argument conclusions, which, e.g. share a similar support, as in:

(5) *Do not use butter to cook vegetables because it contains too much cholesterol, similarly avoid palm oil.*

The *similarly* linguistic mark binds to argument conclusions related to the same topic. Selective-binding rules can be used for other purposes as well as implementing rhetorical relations.

From a syntactic point of view, selective-binding rules are expressed using the Dislog language formalism. Different situations occur that make binding rules more complex than any system of rules used for sentence processing, in particular:

- discourse structures may be embedded to a high degree, with partial overlaps,
- they may be chained (a satellite is a kernel for another relation), e.g. example (3),
- kernels and related satellites may be non-adjacent,
- kernels may be linked to several satellites of different types, e.g. example (6),
- some satellites may be embedded into their kernel.

Selective-binding rules allow the binding of:

- two adjacent structures, in general a kernel and a satellite, or another kernel. A standard case is an argument conclusion followed by its support.
- more than two adjacent structures, a kernel and several satellites. For example, it is quite common to have an argument conclusion associated with several supports, possibly with different orientations:
  (6) *The Maoists will win the elections because they have a large audience and because they threat Terrai tribe leaders*.
- two or more non-adjacent structures, which may be separated by various elements (e.g. causes and consequences, conclusion and supports may be separated by various elements). This is a frequent situation in standard language, where previous items are referred to via pronominal references or via various kinds of marks (*coming back to*...):
  (7) *avoid seeding by high winds. Avoid also frost periods. Besides the fact that the wind will disperse most of your seeds, your vegetables will not grow where you expect them to be.*
  However, limits must be imposed on the 'textual distance' between units. In discourse, such a constraint is not related to well-formed constraints as it is in sentence syntax, but it captures the fact that units which are very distant (e.g. several paragraphs) are very difficult to conceptually relate. One of the reasons is that focus or even topic shifts are frequent over paragraphs or sections and memorising the topic chain is somewhat difficult.

In terms of representation, the two first cases can be dealt with using a standard XML notation where the different structures are embedded into a parent XML structure that represents the whole structure. This is realised via the use of logic variables in logic programming which offer a very powerful declarative approach to structure building. The latter case requires a different kind of representation technique. We adopt a notation similar to the neo-Davidsonian notation used for events in sentence logical representations (Davidson 1963). An ID, which can be interpreted as a discourse event, is associated with each tagged discourse function.

A selective-binding rule for two adjacent structures can then be stated as follows:

```
argument(R) -->
[<conclusion>], gap(G1), [</conclusion>],
 connector(C,[type:cause]), [<support>], gap(G2), [</support>].
```

where the first gap covers the conclusion textual unit G1 and the second one covers the support textual unit G2, the connector language realisation is defined by C, with the constraint that it is of type cause.

To limit the textual distance between argument units, we introduce the notion of *bounding node*, which is also a notion used in sentence formal syntax to restrict the way long-distance dependencies can be established (Lasnik and Uriagereka 1988). Bounding nodes are also defined in terms of barriers in generative syntax (Chomsky 1986). In our case, the constraint is that a gap must not go over a bounding node. This allows to restrict the distance between the constituents which are bound. For example, we consider that an argument conclusion and support must be both in the same paragraph; therefore, the node 'paragraph' is a bounding node.

This declaration is taken into account by the <TextCoop> engine in a transparent way, and interpreted as an active constraint which must be valid throughout the whole parsing process. The situation is however more complex than in sentence syntax. Indeed, bounding nodes in discourse depend on the structure being processed. For example, in the case of procedural discourse, a warning can be bound in general to one or more instructions which are in the same subgoal

structure. Therefore, the bounding node will be the subgoal node, which may be much larger than a paragraph. Bounding nodes are declared as follows in Dislog:

```
boundingNode(paragraph, argument).
```

2.4.2. *Repair rules*

Although relatively unusual, annotation errors may occur. This is, in particular, the case when (1) a rule has a fuzzy or ambiguous ending condition w.r.t. the text being processed or (2) when rules of different discourse functions overlap, leading to closing tags that may not be correctly inserted. In argument recognition, we have indeed some forms of competition between a conclusion and its support which share common linguistic marks. For example, when there are several causal connectors in a sentence the beginning of a support is ambiguous since most supports are introduced by a connector. In addition to using concurrent processing strategies, repair rules can resolve errors efficiently, in the same spirit as those developed for TAGs in Lopez (1999).

Formally, the most frequent situation is the following:
<a>, ... <b> < /a>, ... < /b>
which must be rewritten into:
<a>, ... < /a>, ... <b>, ... < /b> .
This is realised by the following rule:

```
correction([<A> G1 </A> G2 <B> G3 </B>]) -->
[<A>], gap(G1),[<B>],gap(G2),  [</A>],  gap(G3), [</B>].
```

Our formalism in fact allows to specify any kind of correction rule. These rules have the same format as those used to identify discourse functions. In our example, A and B stand for any tag, possibly with attributes, not given here for the sake of readability.

2.4.3. *Rule concurrency management*

The current <TextCoop> engine is close to the Prolog execution schema. However, to properly manage rule execution and also the properties of discourse structures and the way they are usually organised, we introduce additional constraints, which are, for the most part, borrowed from sentence syntax.

Within a cluster of rules, the execution order is the rule reading order, from the first to the last one. Then, elementary discourse functions are first identified and then bound to others to form larger units, via selective-binding rules. Following the principle that a text unit has one and only one discourse function (but may be bound to several other structures via several rhetorical relations) and because rules can be ambiguous from one cluster to another, the order in which rule clusters are executed is a very crucial parameter. To handle this problem, Dislog requires that rule clusters are executed in a precise, predefined order, implemented in a *cascade of clusters of rules*. This notion was introduced by Stabler (1992) with the notion of layers and folding–unfolding mechanisms in an implementation of Generative Syntax theory in Logic programming.

For example, if, in a procedure, we want first titles, then prerequisites and then instructions to be identified, the following constraint must be specified:

```
title < prerequisite < instruction.
```

Since titles have almost the same structure as instructions, but with additional features (bold font, html-specific tags, etc.), this prevents titles from being erroneously identified as instructions.

Similarly, it is much preferable to process argument supports, which are easier to identify, before argument conclusions. Processing advice before warnings also limits risks of ambiguities:

```
advice-support < advice-conclusion < warning-support <
   warning-conclusion.
```

In our engine, there is no backtracking between clusters. When there is no *a priori* complete order between clusters, those not mentioned in the cascade are executed at the end of the process.

In relation with this notion of cascade, it is possible to declare '*closed zones*', e.g.:

```
closed_zone([title]).
```

indicates that the textual span recognised as a title must not be considered again to recognise other functions within or over it (via a gap).

2.4.4. *Structural constraints*

Let us now consider basic structural principles, which are very common in language syntax. This allows us to contrast the notion of constituent with the notion of relation in discourse. Constituency is basically a part-of relation applied to language structures (nouns are parts of NPs) while discourse is basically relational. Let us introduce here dominance and precedence constraints. Discourse abound in various types of constraints, which may be domain dependent (Barenfanger and Hilbert 2006). Dislog is open to the specification of a number of such structural constraints. For argument recognition, these are not so important.

Dominance constraints can be stated as follows:

```
dom(instruction, condition).
```

This constraint states that a conditional expression is always dominated by an instruction. This means that a condition must always be part of an instruction, not in a discourse relation with an instruction. In that case, there is no discourse link between a condition and an instruction, the implicit structure being constituent a condition is a constituent or, a part of, an instruction.

Similarly, *non-dominance constraints* can be stated to ensure that two discourse functions appear in different branches of a discourse representation, e.g.:

```
not_dom(instruction, warning).
```

states that an instruction cannot dominate a warning. However, a warning may be associated with an instruction via a rhetorical relation if its scope is that instruction. This is implemented by a selective-binding rule.

Finally, *precedence constraints* may be introduced. We only consider here the case of immediate linear precedence, for example:

```
prec(elaborated, elaboration).
```

indicates that an elaboration must follow what is elaborated. This is a useful constraint for the cases where a nucleus must necessarily precede its satellite: it contributes to the efficiency of the selective-binding mechanism and resolves some recognition ambiguities. Concerning arguments, supports and conclusions may appear *a priori* in any order; however, if the support appears first, then it is the focus of the arguments rather than the conclusion.

2.5. *Introducing reasoning aspects into discourse analysis*

Discourse relation identification often requires some forms of knowledge and reasoning. This is in particular the case to resolve ambiguities in relation identification when there are several candidates or to clearly identify the text span at stake. While some situations are extremely difficult to resolve, others can be processed, e.g. via lexical inference or reasoning over ontological knowledge. Dislog allows the introduction of reasoning, and the <TextCoop> platform allows the integration of knowledge and functions to access it and reason about it.

This problem is very vast and largely open, with exploratory studies, e.g. reported in Van Dijk (1980), Kintsch (1988), and more recently some debates reported in http://www.discourses.org/UnpublishedArticles/SpecDis&Know.htm.

Within our perspective, let us give here a simple motivational example. The utterance (found in our corpus):

(8) . . .*red fruit tart (strawberries, raspberries) are made*. . .

contains a structure: (*strawberries, raspberries*) which is ambiguous in terms of discourse functions: it can be an elaboration or an illustration; furthermore, the identification of its kernel is ambiguous: *red fruit tart, red fruit*?

A straightforward access to an ontology of fruits tells us that those berries are red fruits, therefore:

- the unit *strawberries, raspberries* is interpreted as an illustration, since no new information is given (otherwise it would have been an elaboration)
- its kernel is the '*red fruit*' unit only,
- and it should be noted that these two constituents, which must be bound, are not adjacent.

The relation between an argument conclusion and its support may not necessarily be straightforward to identify and may involve various types of domain and common-sense knowledge:

(9) *do not park your car at night near this bar: it may cost you fortunes.*

(10) *Women's living standards have progressed in Nepal: we now see long lines of young girls early morning with their school bags.* (*Nepali Times*).

In this latter example, *school bag* means going to school, then *school* means education, which in turn means better living conditions.

2.6. *Processing complex constructions: the case of dislocation*

As in any language situation, there are complex situations where discourse segments that contribute to form larger units, which are not clearly delimited, may overlap, be shared by several discourse relations, etc. Similar to syntax, we identified in relatively 'free-style' texts (i.e. not as controlled as technical procedures) phenomena similar to quasi-scrambling situations, free-structure ordering or cleft constructions. This is in particular the case for arguments which are semantically complex constructs, subject to syntactic variations due to pragmatic considerations such as focus or foregrounding. These issues are 'deep' syntactic discourse constructions that need to be explained and modelled from a language point of view.

As an illustration, let us consider a relatively frequent situation that we call *dislocation*, which is very close to cleft constructions in syntax (Lasnik and Uriagereka 1988), which occurs when in a two segment construction, one segment is embedded into the other, as in: (8i) *strawberries and raspberries are red fruits, for example*.

'red fruits' is the kernel of the relation, while the illustration is split into two parts: 'strawberries and raspberries' and 'for example'. Here, the kernel is included into the satellite.

In the following example:

(11) *products X and Y, because of their toxicity, are not allowed in this building.*

the support of the argument is embedded into the conclusion, probably to add some stress on the toxicity of the products.

To model this construction, as a first experimentation, in particular to evaluate over-recognition problems and the non-determinism introduced in the parsing, constructions subject to dislocation must be declared as follows:

```
dislocation(argument_conclusion, argument_support, argument).
```

where the first two arguments of this predicate are the two structures subject to dislocation, the second being the embedded one, while the third argument refers to the discourse structure that should bind these two structures. The constraints are the following:

- the embedded construction is not further dislocated, i.e. it is in a single text segment,
- the construction that embeds the other is required to be in only two parts, and each segment, the right and the left one, can be recognised by at least one terminal or non-terminal symbol.
- gap symbols cannot range over the embedded structure: they must be fully processed on one sub-segment only,
- the selective-binding operation is directly realised from these two segments: these are bound to the type of the third argument of the dislocation predicate without any further control.

From a processing point of view, the <TextCoop> engine attempts to recognise the embedded structure first, then, if no unique text segment can be found for the embedding structure (standard case), it non-deterministically decomposes the rules describing the embedding structure one after the other, following the above constraints, and attempts to recognise it 'around' the embedded one.

Finally, we observed in our corpora quasi-scrambling situations, a simple case being the illustration relation. Consider again the example above, which can also be written as follows:
(8ii) *strawberries are red fruits similarly to raspberries, for example.*
where the enumeration itself is subject to dislocation.

2.7. *The <TextCoop> engine*

Let us now give some details about the way the <TextCoop> engine runs. The engine and its environment are implemented in SWI Prolog, using the standard syntax without referring to any libraries to guarantee readability, ease of update and portability. It is therefore a stand-alone application. Since this is quite a complex implementation, we simply survey here the elements which are crucial for our current purpose. The principle is that the declarative character of constraints and structure processing and building are preserved in the system. The engine, implemented in Prolog, interprets them at the appropriate control points. The <TextCoop> engine code will be shortly available under GPL license together with a programming environment for rules and linguistic resources (for French and English).

The engine first realises a part of speech tagging. This greatly improves efficiency by limiting backtracking at the rule execution level. Then the engine follows the cascade specification for the execution of rule clusters. Within each cluster, rules are activated in their reading order, one after the other. Backtracking manages rule failures. If a rule in a rule cluster succeeds on a given text span, then the other possibilities for that cluster are not considered (but rules of other clusters may be considered in a later stage of the cascade).

A priori, the text is processed via a left to right strategy. However, <TextCoop> offers a right to left strategy for rules where the most relevant marks are to the right of the rule, in order to limit

backtracking. For the two types of readings, the system is tuned to recognise the smallest text span that satisfies the rule structure.

The engine can work on different textual units: sentences, paragraphs, sections, etc. depending on the kind of structure or phenomenon to recognise (some have a very large scope such as the 'frame' relation that constrains a whole paragraph or even more, while others such as the goal of an instruction or an illustration usually appear in a single sentence. 'Title' relations also range over a large text fragment). These can be specified in the cascade for each cluster of rules. The system is more efficient and generates less ambiguities with smaller units. It processes raw text, html or XML texts. *A priori*, the initial XML structure of the processed document is preserved.

2.8. *System performances and discussion*

Let us now analyse the performances of <TextCoop> with respect to relevant linguistic dimensions, and contrast these with performances of parsers dedicated to sentence processing.

2.8.1. *General results*

The <TextCoop> engine and related data are implemented in SWI Prolog which runs on a number of environments (Windows, Linux, Apple). Our implementation can support a multi-threaded approach, which has been tested with the <TextCoop> engine embedded into a Java environment in collaboration with the Prometil company. This is useful for example for 'parallel' processing on several machines or to distribute, e.g. lexical data, grammars and domain knowledge on various machines for large scale or real-time applications.

The <TextCoop> engine has been relatively optimised and some recommendations for writing rules have been produced (see Section 2.10) in order to allow for a reasonable efficiency. Our basis is a test application with a lexicon of 1300 words and a set of 78 rules related to procedure processing. The test system recognises eight different structures (or a subset of them): instructions, warnings, advice, prerequisites, goals, conditions, illustrations and circumstances. In Section 4, more relations are considered. This test application is purely linguistic, it does not use any external source of knowledge to solve ambiguities. The system runs on a standard PC with Windows XP, an average volume of 18 Megabytes (Mb) of text is processed per hour.

Results are discussed in more depth below: they give a better view on the current performances of the system, using our test application. An important feature is the type of text which is processed: indeed, professional texts, because of their very regular form, possibly following authoring recommendations, produce much better results than dirty texts, such as those coming from the web. However, we also observed large differences in quality and homogeneity in professional texts which make evaluation results somewhat relative.

2.8.2. *Lexical issues*

An important feature of argument recognition is that the lexical resources which are needed are generic, for most of them. This means that the system can be deployed almost on any application domain without any major lexical changes. This result is not proper to argument analysis but to a number of discourse relations whose recognition is based on a small set of predefined linguistic marks and structures.

More precisely, in terms of lexical resources, the following main categories are used for argument recognition: (1) closed classes: discourse connectors, negation, pronouns, prepositions and some basic icons and forms of punctuation and typography, (2) open classes: common communication and change of state verbs (about 600 verbs for French), and nouns, verbs and some adjectives with a strong positive or negative polarity (currently 360 terms for French, slightly less

for English). In our test application, instructions also require the recognition of action verbs, which is quite large a set in general (about 10,000 verbs for French, more for English). To overcome this difficulty, we specialise the action verb lexicon to a given application domain: this is the main lexical tuning which is needed to deploy a system on a specific domain. It should be noted that even for a small domain like cooking, the number of verbs is about 350. For gardening and do-it-yourself, this number is about 700 verbs. In professional procedures, this number is much lower, between 50 and 150 for a given activity. This is essentially due to the use of authoring recommendations. The same remark holds for most types of open lexical resources.

In total, the average size of the required lexical resources (number of rules being fixed) for discourse processing for an application such as procedural text parsing on a given domain is around 900 words, which is very small compared to what is necessary to process the structure of sentences for the same domain. Results below are given for French. Results for English are *a priori* comparable.

The following figures give the system performances depending on the lexicon size. These sizes correspond to real and comprehensive lexicons for a given domain (e.g. 400 corresponds to the cooking domain, the case with 180 lexical entries is a toy system).

Lexicon size (in no. of words)	Mb of text/h
180	39
400	27
900	20
1400	18
2000	17

These results are somewhat difficult to precisely analyse, since they depend on the number of words by syntactic category, the way they are coded and the order in which they are listed in the lexicon (in relation with the Prolog strategy). In order to limit the complexity related to morphological analysis, a crucial aspect for Romance languages, a preliminary tagging process has been carried out to limit backtracking. The way lexical resources are used in rules is also a parameter which is difficult to precisely analyse.

Globally, reducing the size of the lexicon to those elements which are really needed for the application allows for a certain increase in the system performances. This is particularly true for small size lexicons, which are those required for industrial applications. This means some lexical tuning, but on a limited scale.

2.8.3. *Issues related to the rule system size and complexity*

Two parameters related to the rule system are investigated here: how much the number of rules and the rule size impact the efficiency.

The results obtained concerning the number of rules are the following:

Number of rules	Mb of text/h
20	29
40	23
70	19
90	18

As can be noted, increasing the number of rules has a moderate impact on performances, one of the reasons is that the most proto-typical rules are in general executed first. Rules have here an average complexity: four symbols and a gap in average, and an average of eight rules per cluster. Lexical size here is fixed (500 entries). Twenty rules is a very small system, while 80–120 rules is a standard size for an application. The results we obtain are difficult to accurately analyse: besides rule ordering considerations, results depend on the distribution of rules per cluster and on the form of the rules. For example, the presence of non-ambiguous linguistic marks at the beginning of a rule enhances rule selection, and therefore improves efficiency. Constraints such as those presented in 2.4.3 and 2.4.4 are also very costly since they are checked at each step of the parsing process for the structures at stake. Selective-binding rules have little impact on efficiency: their first symbol being an XML tag, backtracking occurs at an early stage of the rule inspection.

Let us now consider rule size, which is obviously an important feature. In particular the number of gaps is crucial:

Rule complexity (symbols per rule)	Mb of text/h
3	30
4	23
5	20
7	18

With the number of rules and the size of the lexicon being kept fixed, we note also that the rule size has a moderate impact on performances, slightly higher than the number of rules. This may be explained by the fact that the symbols starting the rules are in a number of cases sufficiently well differentiated to provoke early backtracking if the rule is not the one that must be selected. However, the number of lexical entries associated with these symbols may have an important impact. If the symbol is a specific type of connector or, conversely, if it is a noun or a verb, this may entail efficiency differences. This is difficult however to evaluate at this stage. Finally, note that rules have in general between four and six symbols, including gaps.

2.8.4. *A comparison with sentence processing*

Globally, we can conclude that there is an impact on efficiency in what concerns the size of the lexicon, the number of rules and their complexity. However, from a toy system to a real size application the impact is about a factor of 5– 8, which is moderate. For the reasons advocated above, the system is not very sensitive to the size of the rule system.

Although we do not have precise figures for comparable treatments, performances substantially contrast with sentence parsers where complexity does increase very much with the number of rules, and to a lesser extent with the size of the lexicon. This being said, there are major general differences between sentence and discourse processing which justifies these differences:

- although this depends on the syntactic theory adopted, however, in general, sentence parsers based on rules have a larger number of rules (a few hundred), and these rules are often recursive,
- in contrast, discourse processing requires rules which are not recursive, structures being constructed by selective-binding rules (about three rules per discourse structure), which form an autonomous system,
- sentence processing requires in general much more lexical resources and an extensive morphological analysis, which is more limited in the case of discourse,

- sentences in real documents being complex, most parsers are shallow parsers, which can process substructures which are either left as such or bound by means of various relations including dependencies,
- discourse-processing rules are based on a few, recurrent, linguistic marks, what is in between (gaps) is of little interest for discourse rules: this allows a comprehensive bottom-up parsing where complete structures can be recognised.

2.9. *The <TextCoop> environment*

The <TextCoop> environment is in a very early stage of development: many more experiments are needed before reaching a stable analysis of the needs. Accessing already defined and formatted resources is of much interest for authors. We have already designed the following sets of resources, for French and English

- lists of connectors, organised by general types: temporal, causal, concession, etc. Miltasaki, Prasad, Joshi, and Webber (2004) developed an original learning method to classify connectors and related marks,
- list of specific terms which can appear in a number of discourse functions, e.g.: terms specific of illustration, summarisation, reformulation, etc.
- lists of verbs organised by semantic classes, close to those found in WordNet, that we have adapted or refined for discourse analysis, with a focus, e.g. on propositional attitude verbs, report verbs (Wierzbicka 1987), etc.
- list of terms with positive or negative polarity, essentially adjectives, but also some nouns and verbs, this is useful in particular to evaluate the strength of arguments,
- local grammars for, e.g.: temporal expressions, expression of quantity, etc.
- some already defined modules of discourse function rules to recognise general-purpose discourse functions such as illustration, definition, reformulation, goal and condition.
- some predefined functions and predicates to access knowledge and control features (e.g. subsumption),
- morphosyntactic tagging functions,
- some basic utilities for integrating knowledge (e.g. ontologies) into the environment.

This environment is compatible with sentence parsers which can operate on the text independently of the tags, or within tag fields.

2.10. *The art of writing Dislog rules and constraints*

The ease of writing rules and the 'natural' character of those rules with respect to language and corpus observations are major properties that any rule system must offer. This, however, needs experiments over a large number of domains and applications on the way to identify rules, generalise them, reach a certain linguistic adequacy and predictability and elaborate a comprehensive set of linguistic marks, etc. Authoring tools are also needed for various kinds of operations, including checking duplicates and overlaps among large sets of rules. While some tools are available for sentence processing (e.g. Sierra, Alarcon, Aguilar, and Bach 2008), there is no such tool customised for discourse. We develop in this section some considerations about a methodology for writing rules and what the services an authoring tool should offer.

Some investigations have been realised to identify linguistic marks on subsets of discourse relations (Redeker 1990; Rosner and Stede 1992; Marcu 1997; Takechi, Tokunaga, Matsumoto, and Tanaka 2003; Stede 2012). These mostly establish general principles and methods to extract terms characterising these relations, rules are then also written by hand (i.e. rules do not result

from automatic learning procedures). The linguistic and pragmatic forms and principles that have emerged seem to be compatible with our perspective. Some of our discourse patterns are due to these previous works.

At the present stage, rules are basically written by hand. Although this is not the main trend nowadays, we feel this is the most reliable approach given the complexity and variability of discourse structures and the need to elaborate semantic representations. Let us briefly review here how rules are produced.

The first step is, given a discourse function one wants to investigate, to produce a clear definition of what it exactly represents and what is its scope, possibly in contrast with other functions. This is realised via a first corpus construction where a number of realisations over several domains are collected, analysed and sorted by decreasing proto-typicality order. This must be realised preferably by a few people and in connection with the literature, in order to reach the best consensus.

Then a larger corpus must be elaborated possibly via bootstrapping tools. Morphosyntactic tagging contributes to identifying regularities and frequencies.

From this corpus, a categorisation must be first elaborated of the different lexical resources which are needed. Then rules can be written. Rules should be expressed at the right level of abstraction to account for a certain level of predictability and linguistic adequacy. This means avoiding low-level rules (one rule per exceptional case) or too high-level rules which would be difficult to be constrained. Rules must be well delimited, starting and ending by non-terminal or terminal symbols which are as specific as possible of the construct. Each rule should implement a particular form of a discourse function. In general, the number of rules for a discourse function (which form a cluster of rules) ranges from 5 to about 25 rules. About 10 are really generic, while the others relate to much more restricted situations. This means that managing such a set of rules and evaluating them for a given function on a test corpus is feasible. An example for warnings is developed in Section 3.

The next step is to order rules in the cluster, starting by the most constrained ones considering the processing strategy implemented in <TextCoop>. In general, the most constrained rules correspond to less frequent constructions than generic ones, which could be viewed as the by-default ones. In this case, this means going through a number of rules with little chances of success, involving useless computations. As an alternative, it is possible to start by generic rules if (1) they correspond to frequently encountered structures and (2) they start by specific symbols not present in he beginning of other rules. In this case, backtracking would occur immediately. This is a compromise that needs to be evaluated by the rule writer. An alternative would be to transform these rules into a deterministic network, as proposed by Javacup tools. We experimented this approach, but the cost of designing an artificial deterministic network was so high that we never concluded this line of research.

Overlap with already existing rules must be investigated since this will generate ambiguities. This is essentially a syntactic task that requires rule contents inspection. This task could certainly be automated in an authoring tool. Ambiguities may be resolved by using knowledge. If it turns out that this is not possible, then preferences must be stated: a certain function must be preferred over another one. Preferences can then be coded in the cascade, starting with the preferred rule clusters, the recognition of the competing rules being then excluded.

The last stage for rule writing is the development of selective-binding rules and possibly correction rules for anomalous situations. Selective-binding rules are relatively easy to produce since they are based on the binding of two already identified structures. Structure variability, long-distance or dislocations are automatically managed by the <TextCoop> engine, in a transparent way. Finally, the rule writer must add the cluster name at the right place in the cascade and possibly state constraints as given in Section 2.4.4.

Although there are important variations, the total investigations for encoding from scratch a discourse function of a standard complexity, including corpus collection, readings and testing should take a maximum of about one month full time. This is a very reasonable amount of time considering, e.g. the workload devoted to corpus annotation in the case of a machine learning approach. As a comparison, we estimate that annotating about 800 occurrences of a given discourse relation and making the necessary controls take about 2– 5 weeks depending on how easy it is to find an adequate diversity of occurrences for that relation. Then, when implemented, the relation must be tested and evaluated on a test corpus, which takes one more week. Possibly, lexical improvements and tunings need then to be carried out.

We feel the quality of manual encoding is also better, in particular rule authors are aware of the potential weaknesses of their descriptions. If a rule or a small set of rules are already available in an informal way, then encoding this small set in Dislog is much faster: checking for needed lexical resources, writing the rules, checking overlaps and testing the system on a toy text should not take more than a day or two for a somewhat trained person. Our current environment contains about 280 rules describing 16 discourse structures associated with argumentation and explanation (Bourse and Saint-Dizier in press). These rules are essentially the core rules for these 16 discourse structures: it is clear that they can be used as a kernel for developing variants or more specific rules for these structures or for structures that share some similarities in form. This should greatly facilitate the development of new rules for trained authors as well as for new ones.

Coming back to an authoring tool, it is necessary at a certain stage to have a clear policy to develop the lexical architecture associated with the rule system. Redundancies (e.g. developing marks for each function even if functions share a lot of them) should be eliminated as much as possible via a higher level of lexical description. This would also help update, reusability and extensions.

3. Arguments in procedural texts

In the second part of this article, let us focus on the family of arguments `Reasons for conclusion` as they are realised in procedural texts. These are in general realised as advice or warnings. We show how they are expressed in Dislog and evaluate the performances of the system in terms of accuracy and portability. Our strategy, to guarantee a robust and stable linguistic analysis, is to focus on texts which contain proto-typical uses, in terms of form and contents, of this form of argument. Procedures, in spite of their diversity, is probably the best type of text for our analysis, in terms of regularity, quality and productivity. They guarantee the development of rules with a large coverage.

Procedural texts have received some attention, in particular from a psycholinguistic and ergonomics point of view, e.g. Bieger and Glock (1985), Lemarie, Lorch, Eyrolle, and Virbel (2008), Aouladomar (2005), Adam (1987). Van der Linden (1993) and Webber and Di Eugenio (1990) provide partial analyses of procedural texts, focussing on the instructional component.

Arguments in procedures are expressed in isolation, in conjunction with an instruction or a group of instructions, they never attack or support each other. Their form is relatively standard, in particular in professional texts where they often follow various authoring principles. Their role is to help the user in the execution of the instructions by providing him advice (ways to improve the task he is doing and the ensuing consequences according to several dimensions: comfort, security, overall quality, cost, etc.) or by providing him warnings that indicate the potential risks if he does not follow the instructions very strictly.

Procedural texts often exhibit a rational structure, the instructions, and an ‘irrational’ structure which is mainly composed of advice, warnings, preferences, evaluations, user stimulations, etc. (Reed 1998; Walton et al. 2008). The latter constitutes what we call the explanation structure,

which develops, motivates and justifies the goal-instructions structure, viewed as the backbone of procedural texts (Bourse and Saint-Dizier 2011). A number of these elements are forms of argumentation, they appear to be very useful, sometimes as important as instructions, since they provide a strong and essential internal cohesion and coherence to procedural texts. Forms of explanations besides arguments are developed in Section 4.

One of the main challenges of the argumentation community is to be able to automatically construct graphs of arguments which are related by a number of relations, in particular attack and support. The system presented here can identify text spans as arguments (conclusions and supports). This is certainly the first step towards constructing such graphs. However, we feel there is still a very long way before being able to automatically construct such graphs with an acceptable accuracy. The first stage would be to provide an accurate semantic representation of the contents of arguments taken in isolation (i.e. coming from various text portions), keeping in mind that arguments may convey very subtle nuances in meaning. Then it is necessary to identify how, or how much, they are related to each other (e.g. are they talking about the same thing?), then their polarity and strength. Some examples in this paper show (e.g. (10)) that knowledge and various forms of inferences are needed to realise such a task, besides accurate language processing. Finally, arguments seldom come in isolation: they are associated with various forms of explanation (illustrations, elaborations, comments, etc.) which must also be taken into account since their meaning is not neutral to construct graphs of arguments.

3.1. *A few methodological considerations*

We have investigated the different forms advice and warnings may take and how they are realised in French and in English from several corpora of procedural texts. We noted that, in a very large number of cases, these arguments can be identified by means of specific terms, without making complex parses or inferences. For most of them, they are embedded into instructions or instructional compounds (a group of related instructions realised in a single sentence, e.g. using coordination), it is therefore quite easy to delimit them. Their scope is in general the compound. Finally, these arguments turn out to have regular forms over several domains, making their corresponding rules relatively portable modulo some lexical adaptations.

We have defined a set of rules that recognise warnings and advice conclusions and their related supports. We defined those rules from a development corpus of about 1700 texts from various domains (cooking, do it yourself, gardening, video games, maintenance, production, social behavior recommendations, etc.). About a third of the corpus is related to professional activities. The relevance of the domain and corpus is analysed in Delpech and Saint-Dizier (2008) and Fontan and Saint-Dizier (2008).

3.2. *Processing warnings*

Procedures have essentially an injunctive character. Warnings are basically organised around a unique structure composed of an 'avoid or make sure to' expression combined with a proposition (Fontan et al. 1998). The strength of the terms used characterise the illocutionary force of the argument, ordered here *a priori* by increasing force:

- 'prevention verbs like avoid' NP / to VP
  (12) (*avoid hot water*)
- do not / never / . . . VP(infinitive) . . .
  (13) (*never put this cloth in the sun*)
- it is essential, vital, . . . (to never) VP(infinitive).
  (14) *it is essential that you switch off electricity before starting any operation*

In the cases where the conclusion is relatively weak in terms of marks, then its recognition is based on the observation that it is the instruction that immediately precedes an identified support.

In the third item above, *never* is preferably used with positively oriented formulations, e.g.:

(14a) *it is essential to never leave the electricity on*.

In addition, it may also reinforce the overall strength of the statement when combined with other marks of the rule. Finally, as pointed out by a reviewer, the injunctive strength induced by forms such as *avoid, it is essential, never, etc.* is subject to personal interpretation and may depend on the text style. The strength of the argument is encoded in an attribute of the 'warning' tag and is elaborated within the rules for warnings, possibly on the basis of lexical descriptions which may also be specified at will. Therefore, it is possible to implement different views and levels of granularity than the classification proposed above.

Finally, the notation '. . .' stands for a gap. This gap must be sufficiently constrained so that it does not skip any symbols, such as negation or, e.g. *unless, apart from* which would lead to incorrect interpretations of the statement. This is part of the rule tuning process.

Supports for warnings are propositions identified from the following marks:

- connectors such as: *otherwise, under the risk of, because* , etc., in French: *sinon, car, sous peine de, au risque de* or via verbs expressing consequence,
  (15) *otherwise it will shrink dramatically* is a support for (12).
- negative expressions of the form: *in order not to, in order to avoid, etc.*
- specific verbs such as risk verbs introducing an event (*you risk to break*). In general, this verb has a negative polarity.
- the presence of very negative terms, such as: nouns: *death, disease, etc.*, adjectives, and some verbs and adverbs.

(16) *Electricity shocks are a major source of injuries and death* is a support for (14).
We have a lexicon of about 300 negative terms elaborated from our corpora.

An important result is that the set of marks used is relatively limited, standard and almost domain independent (some positive or negative terms may be specific to certain technical domains). Examples are given in Section 3.4. The style is very direct (e.g. almost no metaphors to express negative statements). One of the reasons is that authors want their warnings to be easy to understand, without any ambiguities.

In the current implementation, we have:

- nine rules for warning supports and seven for conclusions,
- two repair rules,
- five selective-binding rules.

We carried out an indicative evaluation (e.g. to get improvement directions) on a corpus of 66 texts over various domains, containing 262 arguments. We get the following results for warnings:

Conclusion recognition	Support recognition	(3)	(4)
88%	91%	95%	95%

(3) conclusions well delimited (4) supports well delimited, with respect to warnings correctly identified.

These results are really good, but it should be kept in mind that procedures are a specific textual genre with strong stylistic constraints, in particular in professional documents. Rules are relatively

generic with a large coverage of the phenomenon, the structures they describe are found in most types of texts we had investigated.

In other textual genres, warnings may have different linguistic forms. For example, in prescriptive texts such as requirements, forms using modals such as *must*, or *shall* are frequent, e.g.
(2a) *the system must process the text in less than 2 seconds*.
Supports keep the same linguistic structure. Requirement analysis is under investigation in our LELIE project (Section 5.4). Requirements are very rich in arguments.

3.3. *Processing advice*

Conclusions of type advice essentially describe optional actions, whose aim is to improve the quality of the results or offer suggestions in 'softer' domains such as social behavior recommendations or beauty. They are identified essentially by means of two types of structures:

- advice or preference expression followed by an instruction. The preference expression may be a verb or a more complex expression: *is advised to, prefer, it is better, preferable to, etc.*,
  (17) *prefer professional products*
- expression of optionality or of preference followed by an instruction: *our suggestions:. . .,* or expression of optionality within the instruction
  (18) *use preferably a sharp knife*.

In addition, as for warnings, any instruction preceding a support of type advice can be inferred to be a conclusion if it cannot be recognised by one of these criteria.

Then, supports of type advice are identified on the basis of three distinct types of structures:

- Goal exp + (adverb) + positively oriented term. Goal expressions are, e.g.: *in order to, for* , whereas adverb includes: *better, more* (in French: *mieux, plus, davantage*), and positively oriented term includes: nouns (*savings, perfection, gain* , etc.), adjectives (*efficient, easy, useful*, etc.) or adverbs (*well, simply*, etc.). We constructed a lexicon that contains about 150 terms for this class of positively oriented terms.
- goal expression with a positive consequence verb (*favor, encourage, save,* etc.), or a facilitation verb (*improve, optimize, facilitate, embellish, help, contribute,* etc.),
  (17a) *they will greatly improve the quality of your leathers and make minor repairs* are two supports for (17).
- the goal expression above can be replaced by the verb 'to be' in the future:
  (18a) *it will be easier to cut your pie into pieces* is a support for (18).

In the current implementation, we have:

- eight rules for advice supports and six for their conclusions.
- two repair rules,
- five selective-binding rules.

In the cascade, warnings and advice are executed in a relatively early stage. Since rules exhibit clear marks, with a reasonable level of lexical variation, efficiency is really high, compared, e.g. to instruction recognition. Since supports have structures which are close to some types of goal expressions, ambiguities may arise. It is necessary to recognise supports first provided then related conclusions are found. The remaining structures are then identified as goals in a later stage of the cascade.

[$_{procedure}$
[$_{title}$ How to embellish your balcony
[$_{Prerequisites}$ One lattice, window boxes, etc.]
....
[$_{instructional-compound}$ In order to train a plant to grow up a wall, select first a sunny area, clean the floor and make sure it is flat.....
[$_{Argument}$ [$_{Conclusion:Advice}$ You should leave a 10 cm interval between the wall and the lattice.]
[$_{Support:Advice}$ This space will allow the air to move around, which is beneficial for the health of your plant.]
.....]]]]

Figure 1. An extract of an annotated procedure.

Similar to as described above, we carried out an indicative evaluation on the same corpus with 68 texts containing 240 manually identified advice. We obtained the following results for advice:

Conclusion recognition	Support recognition	(3)	(4)	(5)
79%	84%	92%	91%	91%

(3) conclusions well delimited, (4) supports well delimited, both with respect to advices correctly identified and (5) support and conclusion correctly related.

A short example of an annotated text is given in Figure 1. For ease of reading, we use a conventional bracketed notation instead of XML tags.

As the reader may note, results are less satisfactory for advice than for warnings. This is mainly due to the fact that advice is expressed in a much 'softer' way than warnings, with less emphasis and strength; therefore, terms typical of advice are not necessarily strongly marked. We feel that it is difficult to go much beyond the current results with the approach adopted in Dislog. This is due to the variability of advice constructions and the difficulty to accurately identify them without a heavy use of domain knowledge. If we expand or make rules more flexible, we risk introducing noise, lowering precision, which would not be a real overall gain. We doubt that learning methods can provide better results for the same reasons, there are no results available.

3.4. *A few illustrative rules*

Let us now illustrate the above descriptions by means of a few proto-typical rules and examples for warnings. We first give five rules for warning conclusions, among the most frequent ones. Their order is not relevant. Illustrations are provided for the sake of readability. Then, we give four rules for warning supports, for case (f) we show how this support can be related to the conclusions that precede. We then give a simple example of a selective-binding rule designed to bind a conclusion with a support that immediately follows it. Binding is realised on the basis of the tags which have already been produced for conclusions and supports, respectively. A full example is then given to summarise this section. Rules are given in an 'external' format since their implementation in Dislog is slightly less readable.

WarningsConclusions →

(a) [it, is], {adverb(intensifier)}, imperativeMark, complementizer, gap, verb(action), gap, endMark.

(20) (*It is very important that you open the door carefully.*)

complementizer : to, that, etc.
endMark: a dot, an html tag, a connector, etc.,
adverb(intensifier): very, really, also, etc.,
imperativeMark: essential, fundamental, required, vital, etc.
(b) exp(ensure), [to], negation, verb(action), gap, endMark.
(21) (*Make sure to never touch this part of the cd.*)
exp(ensure): ensure, make sure, be sure, etc.
(c) exp(ensure), negation, [to], verb(action), gap, endMark.
(22) (*Be sure not to press the button before the procedure ends.*).

(d) pronoun, modal(should), negation, gap, endMark.
(23) (*You should not touch this part of the cd.*)

(e) [always], verb(action), gap, endMark.
(24) (*Always put on gloves before tackling any garden task.)*.
(24) is tagged as follows:
< *warning_concl* > Always put on gloves before starting any garden task < */warning_concl* >

WarningsSupports →
(f) comp(avoid), verb(avoid), gap, endMark.
(25) (*so as to avoid injuries.*)
comp(avoid): to, in order to, this, this will, so that you, etc.
(26) (*in order to avoid small injuries and insect bites.)*.
(25) and (26) are supports for (24).

(g) comp(avoid), exp(ensure), [to], negation, verb(action, infinitive), gap, endMark.
(27) (*to ensure to never use it again.*)

(h) connector(cause), gap, verb(negative consequence), gap, endmark.
(27b) (*because otherwise the liquid may leak.*)
connector: because, otherwise, because otherwise, etc.

(j) connector(cause), pronoun, gap, modal, gap, exp(negative), gap, endMark.
(28) (*otherwise your dish would be ruined.*).

(26) is tagged as follows:
< *warning_supp* > in order to avoid small injuries and insect bites < */warning_supp* >

Selective-binding rules simply bind supports with conclusions:
SelectiveBinding(warnings) →
(k) [< *warning_concl* >], gap, [< */warning_concl* >], [< *warning_supp* >], gap, [< */warning_supp* >].

An example fully tagged is then:
(29) < *warning* >< *warning_concl* > avoid using too much salt < */warning_concl* > < *warning_supp* > so as not to ruin your dish < */warning_supp* >< */warning* >.

3.5. *Dealing with empty supports*

Considering do-it-yourself and gardening texts, we noted that about two-thirds of the arguments are not supported. This very large number of unsupported arguments can be explained by several

factors: (1) procedural texts are more oriented towards action than control, (2) some supports could in fact introduce useless doubts or confusions instead of being really useful, (3) some explanation types (supports) may be too complex to be understood by a casual user and (4) supports are sometimes sufficiently explicit in the conclusions:
(30) *do not scatter seeds by high winds ! = they won't go where you want them to go*.
Reconstructing or inferring empty supports would be a very challenging task. However, in socially oriented procedural texts supports are often much more explicit.

4. Arguments in Didactic texts

4.1. *Global situation*

Didactic texts can be viewed as a special kind of procedure, where the goal is to teach something to an audience, often with recommendations. While the style remains basically instructional, the language is much more elaborated. We observed a number of differences with procedural texts. First, texts are relatively short, of a size comparable to large public procedures, i.e. less than a page of text. Next, instructions are more difficult to clearly identify and, for each instruction, there is a profusion of explanations of various kinds (reformulations, elaborations, illustrations, etc.) in which arguments are immerged (von Wright 2004; Bourse and Saint-Dizier 2011). In this respect, the pragma-dialectical theory (van Eemeren and Grootendorst 1984) analyses argumentation as a speech act which has specific communicative goals.

Arguments are, in general, either warnings or advice with forms very close to those observed in procedures; they often have the form of a principle to follow. We observed more advice than warnings, this is not surprising since didactic texts are meant to help, not to prevent. Finally, the speech act structure (Wierzbicka 1987) is often rich and elaborated, it allows, for example, to make explicit various view points on a topic. The links between speech acts and argumentation is a vast area of investigation which will not be investigated here, e.g. Reed and Long (1997), and from a more pragmatic point of view (Pollock 1974; Moeschler 2002, 2007).

Here again, arguments appear mostly in isolation; conclusions are often associated with two or more contrasting supports in order to discuss the facets of the idea being presented. Since it is supposed to be an example to follow, the language in didactic texts is in general quite rich and accurately follows grammatical rules. Another feature is that discourse marks, and in particular connectors, are made as explicit as possible to make sure understanding is clear and unambiguous. Didactic texts are therefore a very useful corpus for our investigations.

The following example illustrates some typical discourse structures and their interactions:

[$_{procedure}$ [$_{purpose}$ Writing a paper: [$_{advice}$ Read light sources, then thorough]]

[$_{assumption/circumstance}$ Assuming you've been given a topic,]

[$_{circumstance}$ When you conduct research], [[$_{advice_c oncl}$ move from light to thorough resources [$_{advice_s upport}$ to make sure you're moving in the right direction]].

Begin by doing searches on the Internet about your topic [$_{purpose}$ to familiarise yourself with the basic issues;]

[$_{temporal-sequence}$ then] move to more thorough research on the Academic Databases;

[$_{temporal-sequence}$ finally], probe the depths of the issue by burying yourself in the library.

[$_{warning_c oncl}$ Make sure that despite beginning on the Internet, you don't simply end there.

[$_{warning_s upport}$ A research paper using only Internet sources is a weak paper, [$_{consequence}$ which puts you at a disadvantage. . .]]]

While the internet should never be your only source of information, [$_{contrast}$ it would be ridiculous not to utilize its vast sources of information. [$_{advice_c oncl}$ You should use the internet to acquaint yourself with the topic more before you dig into more academic texts.]]]

Rules for recognising arguments are relatively similar to those presented for procedures, a few adaptations are needed for the lexical items used as marks, which are richer and less injunctive. We noted in particular:

- richer and more elaborated injunctive marks, with politeness forms (*ensure, please consider*, etc.),
- richer morphology: since the style is more personal, verbs often have a richer morphology: subjunctive forms (especially in French and Spanish), forms using the second person plural to make it more personal, etc. While this does not affect rules, it means a more important access to lexical and morphological data,
- richer causal marks, with a few metaphorical formulations (Talmy 1993), as in the example above,
- more subtle negative expressions (*disadvantage, pressure, depressing, boring*) which often have a psycho-social dimension. An organised lexicon of these forms would be interesting to construct with an analysis of the different categorisations which would need to be made to make the rule system accurate and efficient,
- Manner adverbs are also used to qualify some statements, e.g. to soften them.
- Typographic marks tend to disappear, except for very low-level didactic texts.

These differences give an indication of the kind of lexical tuning which is needed when considering a new textual genre for the same types of arguments.

4.2. *A cooperation between explanation and argumentation*

Explanation analysis remains a largely open research area. A number of analysis in pragmatics have been carried out (Pollock 1974; Fiedler and Horacek 2001; von Wright 2004) and in cognitive science, e.g. Schank (1986). This problem is still at the formulation and exploratory stage in more formal and computational circles (Fiedler and Horacek 2001; Ashley, Desai, and Levine 2002), with some considerations concerning language generation (Zukermann, McConachy, and Korb 2000).

The major interest in studying didactic texts w.r.t. procedures is that it makes it possible to investigate the cooperation and the strong links that discourse relations dedicated to explanation have with arguments. In fact, we consider arguments in this type of text as a very central form of explanation to which are adjoined discourse structures such as illustration, concession or reformulation. Besides arguments, elaboration is also a very central form.

As in a procedure, an explanation is also composed of a sequence of informational elements in general structured with the intent of reaching a goal. This goal may be practical, more interpersonal or epistemic (e.g. convince someone to do something in a certain way, negotiate with someone while providing explanations about a certain point of view).

At the moment, in addition to arguments, the following sets of rules have been defined for French and English (Bourse and Saint-Dizier 2011) and implemented in Dislog, these recognise (the number of rules in each cluster is given between parenthesis): instructions (19), titles (14) and subtitles, illustration (20), restatement (12), purpose (9), condition (13), circumstance (15), concession (12), contrast (14), elaboration (19), frame (17), definition (9), goal (14), analogy (5) and some forms of causes (11). More details can be found in Bourse and Saint-Dizier (in press). Similar to arguments, the recognition of these structures is based on a limited number of linguistic marks.

From a test corpus (31,500 words) and for some of the above relations, we have the following coverage and accuracy rates, expressed in terms of recall and precision. Our strategy was to favour precision over recall since some discourse structures may be somewhat ambiguous or close to

each other. The following figures are based on a comparison of the system performances w.r.t. a manual annotation. A structure is correct if it is correctly identified and well delimited:

Structure	Number manually annotated	Precision (%)	Recall (%)
Instruction	554	98	96
Illustration	38	92	87
Restatement	47	86	79
Purpose	101	89	86
Condition	168	93	82
Circumstance	121	95	92

It is quite challenging to identify the role played *a priori* by some of these forms of explanations when combined with argument as a whole or combined more precisely with conclusions or supports. Restricting our observations to didactic texts, we can formulate the following preliminary conclusions, based on the uses of explanation by authors, and a global analysis of their communicative intentions, while the contents of these structures is not considered:

- A reformulation is neutral w.r.t. an argument, it simply rephrases the argument in a different way to make it as clear as possible:
  Make sure to never touch this part of the cd, do not put your fingers on it.
- An illustration contributes to the understanding of an argument, but it may also add low-level information to make the argument more convincing: it therefore can add a certain strength to an argument that depends on the nature of the illustration:
  (26i) *in order to avoid small injuries and insect bites, such as mosquitoes or fleas*.
  The same remark can be made for the analogy relation, which is relatively frequent:
  (32) *carefully clean the mother card with a spray as you would for a jewel*. . .
- A concession will most certainly weaken an argument:
  (33) *Make sure that despite beginning on the Internet, you don't simply end there, however, a number of students get good results by using only the web.*
  Similarly, a frame will limit the scope of an argument to a certain context.
- A circumstance or an assumption limits the scope of an argument, without affecting its strength:
  (21ii) *When it is new, make sure to never touch this part of the cd.*

1. An elaboration or a contrast will add strength to the argument: (33i) *Make sure that despite beginning on the Internet, you don't simply end there, indeed, most web contents are obsolete, somewhat incorrect and superficial.*
   Most probably, general rules conveyed by elaborations have a stronger impact than illustrations, but this is really dependent on the contents and on the audience.

At this stage, we can simply state general principles on how explanation has an impact on the argument. It is however of much interest to see how explanation operates on arguments from the above considerations. So far, we have made a clear-cut distinction between arguments and the various forms of explanation. However, it may be possible to consider that the notions of argument and explanation may overlap in a certain way and that a certain form of explanation, by virtue of its use in a given utterance, also gets the status of an argument. A model for discourse function merge or coercion could then be developed. In the same range of ideas, other features, in particular speech acts and communication structures may induce that a fact becomes an argument.

5. Perspectives

5.1. *Main results*

In this article, we have first presented the <TextCoop> platform and the Dislog language, designed for discourse analysis. <TextCoop> is based on a cooperation between grammar theory and knowledge and reasoning, which allows the introduction of knowledge and pragmatic factors to identify and properly bound discourse structures. From that point of view, this platform offers several original features.

Our vision of the rule and constraints architecture borrows a few aspects from generative syntax: a productive system of rules, filtering constraints and parameters. Furthermore, the grammatical level is based on a number of modules: a rule system, a lexical architecture, synchronisation mechanisms and the expression of constraints (exclusion zones, precedence, dominance, etc.). The <TextCoop> engine is tuned to process the different components of this system.

Some elements of an environment for rule authoring and management have been designed. A number of rules associated with the family of arguments `Reasons supporting Conclusion` and dedicated to warning, advice and explanation structures are now available for French and English with a method for developing other types of rules in a variety of languages.

From a software point of view, we plan to distribute the kernel of <TextCoop> as freeware (GPL opensource) when sufficiently tested. Probably, and more conveniently, the system will also be available as a web service with an adequate interface and input–output facilities, where users can upload texts and see the tagged texts. Remote control, protected by a login, could be offered so that users can update or add rules, which will then become available to the community (see also Ferrari 1988). We also plan to make available a large corpus of arguments tagged by the system.

From a more foundational point of view, this work raises several aspects of interest, in particular:

- what are the required techniques and forms of linguistic analyses and knowledge that allow an accurate extraction and interpretation of other forms of arguments, following the principles and method initiated in this paper?
- How portable is our approach to other languages? While English and French seem to share a lot of common properties, it is not fully clear how our formalism can be used and possibly extended to other languages such as German or Spanish. For example, in Spanish, the style is much more direct: inflected verb forms are often used instead of infinities. This may seem a detail, but cultural differences would need to be also investigated, for example, languages from Asia where honorifics certainly play a role in argumentation: a standard argument formulation in French could possibly be viewed as a very rude formulation in such a language.
- given the complexity of the task, and given the proximity of the different forms of arguments in language, is it possible to organise arguments (and discourse relations more generally) into classes, allowing for the definition of proto-arguments and proto-discourse relations? These would be based on constitutive properties whose conceptual and functional status is still to be determined. This would allow the development of a conceptual semantics of arguments in a more principled and rational way.
- what are the interactions between arguments and explanation? What purpose do they serve? We feel several types of functions (reinforce, weaken, develop, etc.) could be investigated and integrated into an argumentation network.

5.2. *Towards argument conceptual representation*

The above rules allow to identify and tag the different components of warnings and advice in a number of documents, more generally, 'Reasons supporting Conclusion', with quite a large diversity of linguistic realisations. It is then of much interest to consider the conceptual dimensions of the argument, in particular its strength, its orientation and possibly the conceptual dimension(s) it addresses. We have developed a relatively simple shallow parser that recognises the main elements of the structure of an argument: the verb, the different proto-typical expressions as reported in the rules above, the verb object and relevant nominal and adverbial modifiers.

Strength and orientation can then be determined on the basis of the verb global semantics and the strength of the marks. Numerical values can *a priori* be associated with positive and negative expressions, and then a metrics can be elaborated that combines the different values coming from the various constituents of the argument. This is a very simple approach but which is somewhat arbitrary and too global, since, e.g. the style and tone of the text are not taken into account. A different approach consists in organising marks and relevant verbs along scales. These scales are common in lexical semantics (Cruse 1986), they introduce partial orders. Then comparisons can be made between arguments w.r.t. their strength, to develop, e.g. an argumentation graph.

It is also of much interest to identify the conceptual domain addressed by supports. We noted that supports correspond to two main trends: (1) supports that address one or more general-purpose aspects such as: efficiency of actions, safety and security, ease of execution, adequate execution, speed, aesthetics, costs, space, quality, psychological status, etc., where subtypes can be defined, and (2) supports that cover more precise, contingent domain dependent situations, e.g.:
(34) *avoid pruning trees when temperature drops below zero*.
This observation can help to categorise and organise arguments, showing their benefits, types of risks, etc. For example, support (26) addresses safety, (30) and (31) adequate execution, (28) psychological status and adequate execution and (19) ease of execution.

5.3. *Extending the work: towards processing other kinds of arguments*

The set of rules we have developed for processing arguments in procedures is obviously somewhat limited considering the different forms arguments may take in language. We basically consider the schema 'Reasons supporting conclusion', besides warnings and advice, rewards and threats can be processed: they have about the same linguistic realisations with the exception that the source of the arguments is included in the argument:
(35) *Please pay your subscription within a week otherwise we will stop your internet connection.*

Our feeling is that extracting arguments in natural language is an extremely difficult task in general. Most probably, only those which have a clear linguistic realisation and requiring only standard domain knowledge involved (e.g. via ontologies), can be relatively accurately recognised and semantically characterised.

We conducted two small experiments to validate our working method and to explore the possibilities offered by our rule system in other contexts where argumentation plays an important role. Both experiments start from a controversial issue, the goal being then is to extract in texts related arguments for or against that statement. The general form of these arguments is in general the following:
paraphrase of the statement Connector(cause) support.
where the support is very close to the supports developed for procedures, respectively, introducing a negative and a positive orientation. The first experiment we carried out is related to opinion extraction. It is reported in Bal and Saint-Dizier (2010). The goal is to extract in news editorials from Nepali journals facts or opinions under the form of arguments for or against a statement, e.g. *the Maoist will win the next elections; Women's living standards in Nepal have improved.*

The place of emotion in argumentation (Walton 1992) is obviously a central issue to be taken into account.

The second experiment is related to getting pros and cons for a given attitude in health care. For example, an open question a year ago was: *should I get an injection against virus H1N1*? For example, we got statements such as:
(36) *I will not get the injection because the adjuvant may be dangerous.*,
(37) The injection is safe: millions of people have been treated without any side effect.
In these experiments, we simply extracted statements (facts or opinions), we did not construct any argumentation structure, which is another kind of task. One of our difficulties was to identify that the first part of a statement is indeed in relation with the controversial statement. For both experiments, the extraction rate was quite good, with more than 60% of correct statements extracted without changing any data in our rules.

5.4. *Ongoing applications*

At the moment, we developed two main application frameworks: (1) argument extraction in opinion analysis and (2) risk analysis and prevention as these can be detected from procedures.

Argument extraction in opinion analysis, applied to customer opinions, aims at identifying the reasons why customers are happy or unhappy with a certain product or brand. Arguments may be explicit, introduced by a causal mark, or they may be incorporated into an evaluative expression such as an adjective or an adverbial construct. This enlarges very much the argument extraction perspective and language-processing technology.

We also initiated the LELIE project, aimed at analysing and preventing risks (e.g. health, ecology) from procedure analysis, in production and maintenance situations. This project requires an extensive discourse processing, in particular warnings and advice recognition. The goal is to make sure that a set of procedures dedicated to a given task contain all the necessary safety advice and warnings as stipulated by norms, regulations or business rules. Related to this project, we are investigating from a language point of view a number of aspects of requirement technology. In particular, we are investigating a model that pairs requirements with arguments so that the reasons or the motivations for a requirement are made more explicit, in addition to the information which may be inherited from the parent requirements. Besides safety prevention, the goal is to strongly improve traceability, coherence analysis and requirement update techniques, which is a major challenge at the moment.

Acknowledgements

This project is supported by the French ANR project LELIE and partly by an IFCPAR Indo-French project. We are also very grateful to a number of colleagues for discussions about this work, including Leila Amgoud, David Roussel, Lionel Fontan, Estelle Delpech and Sarah Bourse and two anonymous reviewers.

References

Adam, J.M. (1987), *Types de Sequences Textuelles Elementaires* (Vol. 56), Pratiques, Metz, 1987.

Amgoud, L., Bonnefon, J.F., and Prade, H. (2005), 'An Argumentation-Based Approach to Multiple Criteria Decision', in *8th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, ECSQARU'2005, Barcelona.

Amgoud, L., Parsons, S., and Maudet, N. (2001), 'Arguments, Dialogue, and Negotiation', in *14th European Conference on Artificial Intelligence*, Berlin.

Aouladomar, F. (2005), 'Towards Answering Procedural Questions. Dans: KRAQ'05– IJCAI Workshop', eds. F. Benamara and P. Saint-Dizier, Edinburgh, Juillet, pp. 24–36.

Ashley, K.D., Desai, R., and Levine, J.M. (2002), 'Teaching Case-Based Argumentation Concepts Using Dialectic Arguments vs. Didactic Explanations', in *ITS 2002*, Lecture Notes in Computer Science, Springer Verlag, Heidelberg.

Bal, B.K., and Saint-Dizier, P. (2010), 'Towards Building Annotated Resources for Analyzing Opinions and Argumentation in News Editorial', in *LREC*, Malta.

Barenfanger, M., and Hilbert, M. (2006), 'Cues and constraints for the relational discourse analysis of complex text types – the role of logical and generic document structure', *Journal of language technology and computational linguistics*, 23(2): 49–73.

Bieger, G.R., and Glock, M.D., (1984–85), 'The Information Content of Picture-Text Instructions', *Journal of Experimental Education*, 53, 68–76.

Bourse, S., and Saint-Dizier, P. (2011), *The Language of Explanation Dedicated to Technical Documents* (Vol. 27), Syntagma.

Bourse, S., and Saint-Dizier, P. (in press), *The Structure of Explanation: the case of Didactic Texts*, under submission, LREC, May 2012, Istanbul, ed. ELRA Paris.

Caminada, M., and Amgoud, L. (2007), 'On the Evaluation of Argumentation Formalisms', *Artificial Intelligence Journal*, V.171 (5–6), 286–310.

Canitrot, M., Roger, P.Y., and Saint-Dizier, P. (2011), 'How-To Question-Answering: Hints Extraction', in *LTC Conference*, Poznan.

Carlson, L., Marcu, D., and Okurowski, M.E. (2001), 'Building a Discourse-Tagged Corpus in the Framework of Rhetorical Structure Theory', in *Proceedings of the 2nd SIGdial Workshop on Discourse and Dialog*, Aalborg.

Chomsky, N. (1986), 'Barriers', in *Linguistic Inquiry Monograph* (Vol. 13), Cambridge, MA: MIT Press.

Chomsky, N. (1990), 'Some Concepts and Consequences of the Theory of Government and Binding', in *Linguistic Inquiry Monograph* (Vol. 6), Cambridge, MA: MIT Press.

Colmerauer, A. (1978), 'Metamorphosis Grammars, in Natural Language Understanding by Computers', ed. L. Bolc, Lecture Notes in Computer Science (Vol. 63), Springer Verlag.

Cruse, A. (1986), *Lexical Semantics*, Cambridge: Cambridge University press.

Davidson, D. (1980), *Essays on Actions and Events*, Oxford: Oxford University Press.

Delin, J., Hartley, A., Paris, C., Scott, D., and Vander Linden, K. (1994), 'Expressing Procedural Relationships in Multilingual Instructions', in *Proceedings of the Seventh International Workshop on Natural Language Generation*, Maine, USA, pp. 61–70.

Delpech, E., and Saint-Dizier, P. (2008), 'Investigating the Structure of Procedural Texts for Answering How-To Questions', in *Language Resources and Evaluation Conference* (*LREC 2008*), Marrakech: European Language Resources Association (ELRA).

Di Eugenio, B. and Webber, B.L. (1996), 'Pragmatic Overloading in Natural Language Instructions', *International Journal of Expert Systems*, 23, 58–87.

Dowty, D. (1991), 'Thematic Proto-Roles and Argument Selection', *Language*, 67-3.

Dung, P.M. (1995), 'On the Acceptability of Arguments and its Fundamental Role in Nonmonotonic Reasoning, Logic Programming and n-Person Games', *Artificial Intelligence*, 77, 321–357.

Dung, P.M., Kowalski, R., and Toni, F. (2006), 'Dialectic Proof Procedures for Assumption-Based, Admissible Argumentation', *Artificial Intelligence*, 170(2), 114–159.

van Eemeren, F.H. (ed.) (2002), *Advances in Pragma-Dialectics*, Newport News: Vale Press.

van Eemeren, F.H. and Grootendorst, R. (1984), *Speech Acts in Argumentative Discussions: A Theoretical Model for the Analysis of Discussions Directed Towards Solving Conflicts of Opinion*, Floris Publications.

van Eemeren, F.H. and Grootendorst, R. (1992), *Argumentation, Communication, and Fallacies: A Pragma-Dialectical Perspective*, Lawrence Erlbaum Associates.

Fontan, L., and Saint-Dizier, P. (2008), 'Analyzing the Explanation Structure of Procedural Texts: Dealing With Advices and Warnings', *International Symposium on Text Semantics* (*STEP 2008*), ed. J. Bos, Venice: Association for Computational Linguistics (ACL).

Ferrari, G. (1988), 'Preliminary Steps Toward the Creation of a Discourse and Text Resource', in *Proceedings of the First International Conference on Language Resources and Evaluation* (*LREC*), Granada.

Fiedler, A., and Horacek, H. (2001), 'Argumentation in Explanations to Logical Problems', in *Proceedings of ICCS 2001*, Springer Lecture Notes in Computer Science (Vol. 2073), Springer, pp. 969–978.

Gardent, C. (1997), 'Discourse Tree Adjoining Grammars', Report nb. 89, Univ. Saarlandes, Saarbrucken.

Gazdar, G., and Mellish, C. (1989), *Natural Language Processing in Prolog: An Introduction to Computational Linguistics*, Addison Wesley.

Grosz, B., and Sidner, C. (1986), 'Attention, Intention and the Structure of Discourse', *Computational Linguistics*, 12(3).

Jin, W., and Simmons, R. (1987), 'Question Answering with Rhetorical Relations', in *IEEE Conference on AI*.

Helbig., H. (2005), 'Knowledge Representation and the Semantics of Natural Language', in *Cognitive Technologies*, Springer-Verlag.

Keil, F.C., and Wilson, R.A. (2000), *Explanation and Cognition*, Bradford Book.

Kintsch, W. (1988), 'The Role of Knowledge in Discourse Comprehension: A Construction-Integration Model', *Psychological Review*, 95(2), 121–132.

Kosseim, L., and Lapalme, G. (2000), 'Choosing Rhetorical Structures to Plan Instructional Texts', *Computational Intelligence*, Boston, MA: Blackwell.

Lasnik, H., and Uriagereka, J. (1988), *A Course in GB Syntax*, Cambridge, MA: MIT Press, 1988.

Lemarie, J., Lorch, R. F., Eyrolle, H., and Virbel, J. (2008), 'SARA: A Text-Based and Reader-Based Theory of Text Signaling', *Educational Psychologist*, 43, 27.

Longacre, R. (1982), 'Discourse Typology in Relation to Language Typology', in *Text Processing, Proceeding of Nobel Symposium 51*, ed. S. Allen, Stockholm: Almquist and Wiksell, pp. 457–486.

Lopez, P. (1999), *Repair Strategies for Lexicalized Tree Grammars*, *EACL'99*.

Luc, C., Mojahid, M., Virbel, J., Garcia-Debanc, C., and Pery-Woodley, M-P. (1999), 'A Linguistic Approach to Some Parameters of Layout: A Study of Enumerations', in *Using Layout for the Generation, Understanding or Retrieval of Documents, AAAI 1999 Fall Symposium*, eds. R. Power and D. Scott, pp. 20–29.

Mann, W., and Thompson, S. (1988), 'Rhetorical Structure Theory: Towards a Functional Theory of Text Organisation', *TEXT*, 8(3), 243–281.

Mann, W., and Thompson, S.A. (eds.) (1992), *Discourse Description: Diverses Linguistic Analyses of a Fund Raising Text*, John Benjamins.

Marcu, D. (1997), *The Rhetorical Parsing of Natural Language Texts*, ACL.

Marcu, D. (2000), *The Theory and Practice of Discourse Parsing and Summaruization*, Cambridge, MA: MIT Press.

Marcu, D. (2002), *Au Unsupervised Approach to Recognizing Discourse Relations*, ACL.

Masthoff, J., Reed, C., and Grasso, F. (eds.) (2008), 'Symposium on Persuasive Technology', in *Conjunction with the AISB (UK) 2008 full proceedings: Convention Communication, Interaction and Social Intelligence*, Aberdeen.

Miltasaki, E., Prasad, R., Joshi, A., and Webber, B. (2004), *Annotating Discourse Connectives and Their Arguments*, New Frontiers in NLP.

Moeschler, J. (2002), 'Speech Act Theory and the Analysis of Conversation', in *Essays in Speech Act Theory* eds. Vandervecken, D. & Kubo, S., Amsterdam: John Benjamins.

Moeschler, J. (2007), 'The Role of Explicature in Communication and in Intercultural Communication', in *Exporations in Pragmatics, Linguistic, Cognitive and Intercultural Aspects*, eds. I. Kecskes et al. Berlin: Mouton de Gruyter.

Pereira, F. (1981), 'Extraposition Grammars', *Computational Linguistics*, 9-4, 7, 243–256.

Pereira, F., and Warren, D. (1980), 'Definite Clause Grammars for Language Analysis', *Artificial Intelligence*, 13(3), 123–145.

Pollock, J.L. (1974), *Knowledge and Justification*, Princeton: Princeton University Press.

Redeker, G. (1990), 'Ideational and Pragmatic Markers of Discourse Structure', *Journal of Pragmatics*, 14, 56–77.

Reed, C. (1998), 'Generating Arguments in Natural Language', Ph.D. dissertation, University College, London.

Reed, C., and Grasso, F. (2007), 'Recent Advances in Computational Models of Natural Argument', *International Journal of Intelligent Systems*, 22(1), 7–23.

Reed, C.A., and Long, D.P. (1997), 'Content Ordering in the Generation of Persuasive Discourse', in *Proceedings of 15th International Joint Conference on Artificial Intelligence*, Nagoya, Japan.

Reed, C.A., and Long, D.P. (1998), 'Generating the Structure of Argument', in *Proceedings of the 17th International Conference on Computational Linguistics and 36th Annual Meeting of the Association for Computational Linguistics (COLING-ACL98)*, ed. W. Hoeppner, Montreal, Canada.

Rosner, D., and Stede, M. (1992), 'Customizing RST for the Automatic Production of Technical Manuals', in *Aspects of Automated Natural Language Generation*, eds. R. Dale, E. Hovy, D. Rosner, and O. Stock, Lecture Notes in Artificial Intelligence, Springler-Verlag, pp. 199–214.

Saaba, A., and Sawamura, H. (2008), 'Argument Mining Using Highly Structured Argument Repertoire', in *Proceedings EDM08*, Niigata.

Saito, M., Yamamoto, K., and Sekine, S. (2006), *Using Phrasal Patterns to Identify Discourse Relations*, ACL.

Saint-Dizier, P. (1994), *Advanced Logic Programming for Language Processing*, Academic Press.

Schank, R. (1986), *Explanation Patterns: Understanding Mechanically and Creatively*, Laurence Erlbaum.

Schauer, H. (2006), *From Elementary Discourse Units to Complex Ones*, Sigdial Workshop, ACL.

Sierra, G., Alarcon, R., Aguilar, C., and Bach, C. (2008), Definitional Verbal Patterns for Semantic Relation Extraction', *Journal of Terminology*, 14-1, 22–39.

Stabler, E. (1992), *The Logical Approach to Syntax*, Cambridge, MA: MIT Press.

Stede, M. (2012), *Discourse Processing*, Morgan and Claypool Publishers, 2012.

Takechi, M., Tokunaga, T., Matsumoto, Y., and Tanaka, H. (2003), 'Feature Selection in Categorizing Procedural Expressions', in *The Sixth International Workshop on Information Retrieval with Asian Languages* (*IRAL2003*), pp. 49–56.

Talmy, L. (2001), *Towards a Cognitive Semantics* (Vols. 1 and 2), Cambridge, MA: MIT Press.

Taboada, M., and Mann, W.C. (2006), 'Rhetorical Structure Theory: Looking Back and Moving Ahead', *Discourse Studies*, 8(3), 423–459.

Van der Linden, K. (1993), 'Speaking of Actions Choosing Rhetorical Status and Grammatical Form in Instructional Text Generation', Thesis, University of Colorado.

Van Dijk, T.A. (1980), *Macrostructures*, Hillsdale, NJ: Lawrence Erlbaum Associates.

Walton, D. (1992), *The Place of Emotion in Argument*, Pennsylvania: Pennsylvania State University Press.

Walton, D., Reed, C., and Macagno, F. (2008), *Argumentation Schemes*, Cambridge: Cambridge University Press.

Webber, B. (2004), 'D-LTAG: Extending Lexicalized TAGs to Discourse', in *Cognitive Science* (Vol. 28), Amsterdam: Elsevier, pp. 751–779.

Webber, B.L. and Di Eugenio, B. (1990), 'Free Adjuncts in Natural Language Instructions', *Proceedings COLING-90*, Helsinki, Finland.

Wierzbicka, A. (1987), *English Speech Act Verbs*, New York: Academic Press.

von Wright, G.H. (2004), *Explanation and Understanding*, Cornell, Ithaca: Cornell University Press.

Zuckerman, I., McConachy, R., and Korb, K. (2000), *Using Argumentation Startegies in Automatic Argument Generation*, INLG.