

Analyzing the Structure of Argument Compounds in Logic: the case of technical texts

Juyeon KANG (1), Patrick SAINT-DIZIER (2)

(1) PROMETIL, Toulouse France

(2) IRIT-CNRS, 31062 Toulouse, France

j.kang@prometil.com, stdizier@irit.fr

Abstract

In this contribution, we start from a concrete problem: the identification of argument compounds in technical texts and show how their structure can be modelled on a linguistic and logical basis. This investigation is, in a first stage, restricted to arguments found in technical texts because of the complexity of the problem. Then, we propose a preliminary type-based syntactic analysis and the premises of a conceptual model that accounts for the logical properties of such compounds. An implementation in Dislog is presented with an indicative evaluation of the results. Dislog runs on TextCoop which is a logic-based platform we developed for discourse analysis.

1 Introduction

1.1 Argument compounds

Arguments in texts or in dialogues seldom come in isolation, as independent statements. They are often embedded into a context that indicates e.g. circumstances, restrictions, purposes, and various forms of elaborations. Relations between an argument and its context may be conceptually complex. Furthermore, corpus analysis shows that, besides their contexts, arguments often appear in small and closely related groups or clusters where they share similar aims, where the first argument is complemented, supported, reformulated or elaborated by the subsequent ones. These small groups may also include statements in contrastive or concessive configurations. The elements in these units share strong relations: they must be considered as single conceptual argument unit.

In terms of language realization, clusters of arguments and their related contexts may be all included into a single sentence via coordination or subordination or may appear as separate sentences. In both cases, the relations between the different elements of a cluster are realized by means of conjunctions, connectors, various forms of references and punctuation. We call such a cluster an **argument compound**. The claim, behind this term, is that the elements in a compound form a single, possibly complex, unit, which must be considered as a whole from a conceptual and argumentative point of view. Such a compound is realized in a few sentences, so that its contents can be assimilated easily.

In this contribution, we start from a concrete problem: the identification of argument compounds in technical texts and show how their structure can be modelled on a linguistic and logic basis. For that purpose, an empirical analysis is combined with discourse analysis results. Then, we propose a type-based syntactic analysis and the premises of a conceptual model that accounts for the logical and inferential properties of such compounds. An implementation in Dislog (Discourse in Logic) is presented with an indicative evaluation of the results. Dislog runs on TextCoop which is a logic-based platform we developed for discourse analysis.

Our aim is to show how a model can be elaborated from a sample of texts. Obviously, the elaboration of a logic model for argument structure in technical texts requires a more extensive analysis than this first investigation.

1.2 Arguments in technical texts

Language expressions of arguments are often very diverse and complex, making their automatic identification in texts a very challenging task. Besides language complexity, a large number of

arguments are not clearly marked by specific linguistic cues, therefore, it is often necessary to have recourse to semantics and pragmatics to identify, delimit and understand them and then identify the relations within and between compounds.

Technical documents (e.g. procedures, product manuals, specifications) form a linguistic genre with restricted linguistic constraints in terms of lexical realizations, including business or domain dependent aspects, grammar, style and overall organization. Technical documents cover a large variety of types of documents: procedures, equipment and product manuals, various notices such as security notices, regulations of various types (security, management), requirements and product or process specifications. These documents are designed to be as efficient and unambiguous as possible. For that purpose, they tend to follow relatively precise authoring principles concerning both their form and contents. Technical documents abound in various classes of arguments, in particular recommendations, warnings, advice, requirements and regulations. These arguments appear in isolation, often with several supports, except for requirements. It is crucial to clearly identify and represent arguments and their contexts since they play important roles in technical documents, e.g. stating operational constraints or preventing risks.

Technical documents are in general not a mere list of instructions. They often start with general considerations such as purpose, scope, pre-requisites and context. Then follow definitions, examples, scenarios or schemas. Then come a set of sections that address the different facets of the problem at stake. Each section may include, for its own purpose, general elements followed by the relevant instructions, recommendations, etc. Each argument can be associated with several supports, possibly contradictory, and forms of explanation. Automatically identifying argument compounds in technical texts and producing a conceptual representation adequate for subsequent treatments is the major concern of this paper. For that purpose, we develop a discourse grammar that accounts for the conceptual structure of argument compounds. The modelling is based on logic, logic programming and constraint satisfaction, as implemented in the <TextCoop> platform via the Dislog language.

This paper further elaborates on results presented in (1) (Saint-Dizier 2012) where processing isolated warnings and advice are presented together with their implementation in Dislog, (2) (Villalba et al. 2012) where we show that discourse structures, for which a detailed semantic analysis is developed, can be interpreted as argument supports in opinion analysis and (3) (Kang et al 2013) dedicated to requirement mining. The main contribution of this paper is the analysis of the structure of argument compounds and how discourse structures interact with arguments.

2 Linguistic and Conceptual Analysis

The linguistic structure of arguments as isolated utterances or as networks of arguments has been investigated in a number of works in linguistics and cognitive semantics, e.g. (Eemeren et al. 1992), (Walton et al. 2008), (Walton 2011). Much less has been developed from a technical perspective in computational linguistics. Difficulties come from the large diversity of language forms arguments may have in language, the need of contextual information to identify them and the difficulty to relate arguments with their supports or with other arguments, in particular when they are not adjacent in a text or a dialogue. In addition argument supports or attacks may be related to several arguments. Identifying the relatedness between an argument conclusion and its supports or attacks is also an open and challenging problem.

In terms of discourse, the RST (Man et al. 1988), (Taboada, 2006) has been very influential. Several approaches, based on corpus analysis with a strong linguistic basis, are of much interest for our approach. Besides the Penn Discourse Treebank ¹, relations have been investigated together with their linguistic markers in e.g. (Delin et al. 1994), (Marcu 1997), (Miltasaki et al. 2004). (Rossner et al. 1992) and (Saito et al. 2006) developed an extensive study on

¹available at <http://www.seas.upenn.edu/pdtb>

how markers can be quite systematically acquired. (Stede 2012) develops a useful typology of markers.

Our approach merges argument and discourse structure analysis. In this context, the typical configuration of an argument compound can be summarized as follows:

```
CIRCUMSTANCE(S)/CONDITION(S), PURPOSE(S) -->
  [ARGUMENT CONCLUSION + SUPPORT(S)]*
  <-- PURPOSE(S), ELABORATION(S), CONCESSION(S)/CONTRAST(S)
```

The kernel of this structure is the organized set (noted with the *) of arguments and their supports. The main argument occurs in general first, it is then followed by secondary arguments; their functions are developed below. The compound starts with circumstances or conditions, possibly purposes, when they have a wide scope. Then follows the set of arguments and their supports. The compound ends by purposes, concessions or contrasts and elaborations.

At the language realization level, this logical organization is often not realized straightforwardly. The language data remains often raw and unpolished. In particular, we observed that:

- the initial group, that should logically precede the set of arguments, may be inserted between arguments,
- the last group, that should also logically follow the set of arguments, may be inserted between these arguments,
- purposes may be realized as supports or vice versa,
- an argument may have several supports, possibly with different orientations, supports may not be adjacent to their related conclusion,
- supports may be inserted within their conclusion, instead of following or preceding it.

Let us illustrate argument compounds, where a few tags have been inserted to anticipate on our analysis:

Ex. 1. <ArgCompound> <purpose> Cleaning your leathers.</purpose> <advice> <conclusion> Prefer natural products. </conclusion> <support polarity="-"> They are more expensive </support> but <support polarity="+"> they will have a longer effect and make minor repairs. </support> </advice> </ArgCompound>

Ex. 2. <ArgCompound> <definition> Inventory of qualifications refers to norm YY. </definition> <mainArg> Periodically, an inventory of supplier's qualifications shall be produced. </mainArg> <secondaryArg> In addition, the supplier's quality department shall periodically conduct a monitoring audit program. </secondaryArg> <elaboration> At any time, the supplier should be able to provide evidences that EC qualification is maintained. </elaboration> </ArgCompound>

Ex. 3. <ArgCompound> <warning> <conclusion> Products X and Y, <support> because of their toxicity, </support> are not allowed in this building. </conclusion> </warning> <concession> In case of emergency, a special permission is needed to use them in buildings. </concession> </ArgCompound>

Ex. 1 comes from a do-it-yourself text, it illustrates the case where an argument of type advice has several supports with different orientations. The contrastive connector 'but' introduces the inversion of the polarity. The first support is not really an attack, but a kind of contrast (see section 3). Ex. 2 is a requirement compound. It shows how a definition makes the requirements more accurate. A secondary requirement complements the main one. This secondary requirement is further elaborated in the last sentence. The elaboration is not a requirement because of the modal 'should be able to' which is not sufficiently injunctive. Ex. 3 come from a security notice, it illustrates the case where a support is inserted into the middle of a conclusion. The second sentence is a concession that allows exceptional situations.

2.1 Corpus Elaboration

Constructing a corpus is a very challenging task: it is first necessary to identify the parameters to measure, while keeping the others constant. Then, the scope of each parameter needs to be defined (the values they can take) and the value distributions (the percentage of texts for each parameter value which is needed for an adequate observation). The last step is the corpus construction, and its validation w.r.t. these parameters. Our investigations require the taking into account of the following features:

- **domains** (energy, chemistry, transportation, food processing),
- **authors** (technical writers, engineers, administrative staff),
- **style** (basic, normal, elaborated with a rich typography),
- **target audience** (technicians from beginner to confirmed, engineers) and
- **difficulty** to realize the procedure (i.e. complex temporal sequences of actions, use of complex equipment, large number of warnings).

154 texts have been selected for the development corpus, while 74 texts have been selected to make an indicative evaluation of the results to improve our working method. This set of texts does not constitute a large corpus. We rather view it a initial set designed to measure the language complexity and to develop a preliminary investigation and analysis. Further corpus will allow us to deepen constructions and semantic structures.

2.2 A typology of argumentation in technical documents

Let us first develop an analysis of the types of arguments, usually found in technical texts. In a second stage, let us explore how argument compounds are constructed and analyzed. This analysis is illustrated by (1) typical patterns that identify arguments and (2) related lexical resources with their semantic contents for which we have developed specific linguistic categorizations. The main structures are presented here. There are in general no authoring recommendations in technical texts to produce arguments. Nevertheless, technical writers tend to use relatively standard forms of expressions, explicit and understandable for their users.

This analysis is manual and relates corpus observations. It is not elaborated enough to allow the development of an in-depth conceptual analysis. Similarly, the predictive ability of our analysis remains to be established and evaluated; it probably needs a larger corpus to be settled. This analysis however shows the complex points and the main structures that one can find in technical texts and how a computer analysis can be derived out of these observations.

Requirements and regulations, requirements (Hull et al. 2011) and regulations form a special class of arguments, with specific linguistic forms and a very injunctive orientation. Their support(s) must not be confused with purpose clauses: their role is to justify the requirement, its importance, and the potential risks and difficulties that may be encountered. Their identification is based on precise patterns in a sentence (Kang et al. 2013) such as:

[modal(*shall, must, have to*) + infinitive verb].

Supports are introduced by a purpose connector, e.g. *to, for, in order to*, A comprehensive requirement is e.g. *an inspection shall be carried out monthly for a correct cleaning of the universal joint shafts*.

Prevention arguments or warnings basically explain and justify an instruction or a group of instructions. These are very frequent in most types of technical documents. Formulations with a negative polarity are frequent, their structure is given in (Saint-Dizier 2012) and summarized here. The structure of a conclusion is:

(w1) prevention verbs like *avoid*’ NP / to VP (*avoid hot water*)

(w2) *do not / never / ...* VP(infinitive) ... (*never expose this product to the sun*)

(w3) *it is essential, vital, ...* (to never) VP(infinitive).

it is essential that you switch off electricity before starting any operation.

Supports are realized by one of the following main syntactic schemas (ws= warning support):

- (ws1) negative causal connector + risk verb,
- (ws2) positive causal connector + VP(negative form),
- (ws3) positive causal connector + prevention verb.

The grammatical and lexical elements in these constructions are in particular:

- negative causal connectors: otherwise, under the risk of, (e.g. *otherwise you may damage the connectors*).
- risk verb class: risk, damage, etc. (e.g. *in order not to risk your fingers*) or verbs of a "conservative" type : *preserve, maintain*, etc. (e.g. *so that the axis is maintained vertical*)
- prevention verbs: *avoid, prevent*, etc. (e.g. *in order to prevent the card from skipping off its rack*).
- positive causal mark and negative verb form: *in order not to*, (e.g. *in order not to make it too bright*).
- modal SV: *may, could*, (e.g. *because it may be prematurely stop due to the failure of another component*).

Performing or advice arguments are less imperative than the previous ones, they express advices or task evaluations. These are also very frequent, in particular in documents designed for novices. Conclusions have in general the following structure:

- (a1) 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., prefer professional products*
- (a2) expression of optionality or of preference followed by an instruction: *our suggestions: ...,* or expression of optionality within the instruction: *use preferably an J78 Key*.

Their supports follow the following schemas:

- (as1) causal connector + performing NP,
- (as2) causal connector + performing verb,
- (as3) causal connector + modal-performing verb,
- (as4) performing proposition.

Resources or structures are, for example:

performing verbs: *allow, improve*,

performing NP: (e.g. *for a better performance, for more competitive fares*),

performing proposition. A comprehensive example is e.g. *<conclusion> we advise you to have small bills </conclusion>, it is easier to tip and to pay your fare that way*.

Threatening arguments are less frequent. The reader is directly involved in the consequences. These arguments have a strong impact on the users attention when he realizes the instruction. These arguments follow one of the following syntactic schemas:

- (t1) otherwise connectors + consequence proposition,
- (t2) otherwise negative expression + consequence proposition, with, e.g.:

otherwise connectors: e.g. *otherwise*,

otherwise negative expression: *if ... do not ...* (e.g. *if you do not pay your registration fees within the next two days, we will cancel your application*).

3 Processing argument compounds

3.1 Identification and delimitation of argument compounds

In general, the development of relatedness criteria between an argument conclusion and its support(s) or related discourse structures is complex, in particular when there is no strict adjacency, which is frequent.

The principle is that all the statements in an argument compound must be clearly related either by the reference to a precise theme or via specific marks that define **cohesion links**. Such links must be made explicit in technical texts to avoid ambiguities. They can be identified and

categorized in a relatively clear way. The theme of a compound is a nominal construction (object or event, non human, e.g. *inventory of qualifications* in Ex. 2)). Links between arguments and related discourse structures are defined on the following basis:

- the use of the theme in the sentences that follow or precede the main requirement (e.g. *inventory of qualifications* in Ex. 2). The theme can possibly undergo morphological variations, a different determination (e.g. *safety test*, *all safety tests*) and simple syntactic variations. However, the variation threshold for this latter point is not easy to evaluate. This first situation occurs in about 70% of the cases. The theme is the subject of the clause (about 65% of the cases) when the subject does not denote a human actor or the direct object in the other cases (35%).
- the use of a pronoun referring to the theme, when the theme is in subject position in the sentence that precedes (about 25% of the cases),
- the use of a more generic term than the theme, used as a simple form of reference, e.g. *this process*, *this constraint*, in the utterances that follow the main requirement (about 20% of the cases). We identified 42 such general purpose terms.
- the use of discourse connectors to introduce a sentence, e.g. *however*, *for that purpose*, *if*, etc., found in about 35%
- the use of sentence binders such as: *for information*, *in this case*, *at any time*, *next*, *also*, etc. found in 20% of the cases.

Obviously these criteria may overlap, in particular in utterances other than requirements where the theme and a connector can be found.

For example, in Ex. 2 above, the theme is *inventory of qualifications* that undergoes a few variations such as *inventory of supplier's qualifications*. Connectors are used in the last two sentences: *in addition*, *at any time*. In Ex. 1, links are realized via pronouns for very short utterances where they are not ambiguous.

3.2 Discourse Relations in a compound

In an argument compound, as shown in section 1 above, arguments and sentences other than arguments are linked by means of discourse relations. This defines a kind of network of relations. The relations between arguments are essentially contrasts, concessions and specializations. The structure and the markers and connectors typical of discourse relations found in technical texts are developed in e.g. (Rossner et al. 1992) and (Saint-Dizier 2014). These have been enhanced and adapted to the compound context via several sequences of tests on our corpus. The main relations found are the following:

- **contrast**, (Wolf and Gibson 2005) and (Spenader and Lobanova 2007), is a relation between two arguments that introduce one or more related but alternative views, which refer to a unique situation. Formally, the apparent contradiction that results motivates the use of a defeasible inference logic and semantics to preserve the coherence of the whole structure. Contrast is introduced by *however*, *although*, *but* combined, in the utterance, with e.g. adverbs such as *also*, modals or specific verbs expressing choice.
- **concession** states a general requirement followed by an apparently contradictory argument that could be admitted as an exception (e.g. Ex. 3.). The contradiction with the implicit conclusion which can be drawn from the first argument is partial (e.g. (Couper-Kuhlen et al. 2000)). Concessions are often categorized as denied phenomenal causes or motivational causes. Typical marks are, e.g.: *however*, *although*, *even though*, *despite*, or modal constructions such as *may be*, *could be*. We observe a kind of continuum between contrast and concession. The ambiguity is represented by the *polymorphic relation* 'contrast-concession'. Ambiguities may then be resolved via knowledge and inferences.
- **specializations, and subsequent constraints.**

- **information and definitions** mainly occur before the main argument. They anticipate and develop notions given in the main argument which may be complex or insufficiently clear to the reader or possibly contradict his beliefs,
- **elaborations** follow an argument, they develop some of its facets to facilitate its understanding. Elaborations may play the role of supports. Since this relation is very under-specified, we consider it as the by-default relation in the compound. A categorization of the main functions covered by elaboration are in particular: localization, precision, focus, future actions, application domains, constraints, prerequisites. An automatic identification of these functions is ongoing and beyond the scope of this paper.
- **illustration** provides related examples. It is characterized by simple marks such as: *this includes, for example, an example, examples* associated with an enumeration. Illustration can also be analyzed as a form of support.
- **result** specifies the outcome of an action. Its linguistic structure is basically the active-inchoative alternation that describes the expected result, implemented via the use of the theme combined with the main verb past participle or with an aspectual verb denoting completion or quasi-completion.
- **circumstance** introduces a kind of local frame under which the argument compound is valid or relevant. Circumstances often appear before the argument(s) they apply to. Circumstances introduce temporal, spatial or factual contexts or particular events or occasions. The structure of circumstance is illustrated in 4.2.
- **purpose** which expresses the underlying motivations of the argument compound. It must not be confused with argument supports. These are introduced by purpose connectors, causal verbs, purpose verbs (e.g. *demonstrate*) or expressions such as *the objective is*.

4 Implementation in Dislog

4.1 TextCoop: a platform for discourse analysis

The TextCoop platform and the Dislog language (for Discourse in Logic) have been primarily designed for argumentation and discourse processing (Saint-Dizier 2012). TextCoop is based on Logic Programming. It uses the properties of logic programming that allows declarative specifications of language processing. This is particularly convenient for discourse processing where phenomena are complex and involve several levels of description (structures, constraints, etc.).

TextCoop is a platform that includes:

- (1) **Dislog**, which is a logic-based language designed to describe in a declarative way discourse structures and the way they can be bound via selective binding rules,
- (2) **an engine** associated with a set of processing strategies. Dislog rules are processed according to a cascade that specifies their execution order. This engine offers several mechanisms to deal with ambiguity and **concurrency** when different discourse structures can be recognized on a given text fragment,
- (3) **a set of active constraints**, in the sense of Constraint Logic Programming, that state well-formedness typical language and of discourse structures (e.g. precedence, dominance, bounding nodes); these can be parameterized by the grammar writer,
- (4) **input-output facilities** (XML, MS Word), and interfaces with other environments
- (5) a set of **lexical resources** which are frequently used in discourse analysis (e.g. connectors),
- (6) a set of about 180 **generic rules** that describe 12 frequently encountered discourse structures such as reformulation, illustration, cause, contrast, concession, etc.

The approach adopted in TextCoop and Dislog is rather traditional in the sense that it is rule based. In Dislog, rules and constraints are specified in an abstract and declarative way which is well adapted to deal with the complexity of discourse processing.

As an illustration of discourse analysis rule, let us consider the following two simple rules for the 'circumstance' structure (that form a *rule cluster*):

circumstance $\rightarrow$ *conn(circ)*, *gap(G)*, *punctuation*. / *gap(G)*, *verb(aspectual)*, *eos*.

Lexical resources are quite diverse:

conn(circ): *when, once, as soon as, after, before, in case of*,

verb(aspectual): *start, resume, stop*

The circumstance ranges here between the marker (*conn(circ)*) and a punctuation, the gap allows to skip a finite number of words of no present interest. 'eos' stands for end of sentence. The result is the text annotated with XML tags or a dependency structure. More rules are described and evaluated in (Saint-Dizier 2014).

Warnings as described above can be implemented as follows:

Warning_Conclusions $\rightarrow$

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

e.g.: (*It is very important that you open the door carefully.*)

adverb(intensifier): *very, really, also, etc.*,

imperativeMark: *essential, fundamental, required, vital, etc.*

connector is not in the rule but delimits its end. Alternatively it can also be the end of a sentence.

exp(ensure), [to], negation, verb(action), gap, connector.

e.g. (*Make sure to never touch this part of the cd.*)

exp(ensure): *ensure, make sure, be sure, etc.*

exp(ensure), negation, [to], verb(action), gap, endMark.

e.g.: (*Be sure not to press the button before the procedure ends.*).

Warning_Supports $\rightarrow$

introducer(support), verb(avoid), gap, endMark.

e.g. (*so as to avoid injuries.*)

comp(avoid): *to, in order to, this, this will, so that you, etc.*

In this example, the identification of the type of the cue term that introduces the support clause is not yet semantically stable.

4.2 Higher-order programming for compound delimitation

An argument compound is identified by its theme and by its boundaries (where it starts and ends) following the criteria given in 2.1. Let T_0 be the theme identified in the main argument. Let $variant(T_0)$ be the finite set of the n variants which can be generated from T_0 via functions producing morphological variants, generalizations, etc.:

$variant(T_0) = \{T_i, i \in [1, n]\}$.

Then all the sentences S_j , $j \in [1, k]$, $k < 6$ (assuming that a compound has a maximum of 5 sentences, but this is a parameter) in the compound must meet the following constraints, expressed by the following two Dislog rules:

$\forall S_j j \in [1, k] \exists i, i \in [1, n]; sentence \rightarrow T_i, gap(G), eos. / gap(G1), verb, T_i, gap(G2), eos.$

These rules introduce a kind of higher-order programming with a quantification on the form of the theme. The rules state that the theme or its variants must appear in the set of related sentences either in subject or object position.

In more concrete terms, given a theme T and T_i one of its expressions (word or expression), identified from a main argument, related sentences are identified by the following simple pattern, where P is a compound identifier:

`compound_sentence(P) → beginSentence, gap(G1), word(Ti), gap(G2), endSentence.`

A compound sentence is tagged; the theme appears as an attribute. In the processing strategy, arguments and related themes are first identified. Then, related sentences occurring either before or after it are identified. This is managed via the TextCoop cascade.

Selective binding rules: Selective binding rules are essentially based on already tagged structures. Binding rules allow, under constraints, to bind arguments together or with discourse structures, under constraints. Binding rules are abstract, higher-order schemas. For example, binding the main argument $R1$ with the structure $G1$ (e.g. $G1$ = definition) that precedes it is represented as follows:

`<X>, gap(G1), </X> <req status="main"> gap(R) </req> →`

`<Argcompound> <X>, G1, </X> <req status="main">, R, </req> </Argcompound>.`

This rule inserts $G1$ and R into the `<Argcompound>` tag. X is a variable that stand for any discourse structure provided that the precedence constraints (see next subsection) are met. Besides binding structures, these rules double-check that the two sentences share the same cohesion link, as described in the previous subsection.

The above example is simple and binds two adjacent structures. Selective binding rules can related adjacent as well as non-adjacent structures. Binding can occur between groups of structures: a set of structures (discourse and arguments) can be bound, and then bound to another group of structures. This allows the construction of subtrees, which is a frequent situation. This requires some form or priority among binding rules. This is a relatively complex and generic investigation that is on-going. In our approach we basically proceed by aggregating structures around the main argument.

Constraints: Constraints mainly encode in this investigation precedence constraints (`<` encodes precedence). In Dislog, they are specified in a very declarative manner, as described below. The TextCoop engine checks that these constraints are met at each step of the processing. The following constraints induce a partial ordering of arguments and discourse structures in an argument compound:

information, definition `< argument.` : Information and definition occur before any argument.

Argument `< result.` : result always follows the relevant arguments.

Argument `< concession, contrast.` : Contrasts and Concessions always follow the arguments on which they operate.

Argument `< elaboration.` : An elaboration always follows the argument it elaborates.

An additional constraint states that, in an argument compound, concessions and contrasts cannot co-occur:

`<>(contrast,concession).`

Finally, an argument compound must be fully realized within a paragraph, therefore, the node paragraph is a bounding node:

`bounding_node([paragraph])`

for binding rules.

The system designed for compound analysis is composed of a set of rule clusters, associated lexical entries, and constraints. To deal with 'scrambling' situations as illustrated in Ex. 3, rules are non deterministically decomposed under constraints (Saint-Dizier 2012). Rule clusters are activated one after the other with an order specified in a cascade. This cascade allows, among others, to specify priorities (a cluster must be fully processed before another one is activated) and to avoid ambiguities.

4.3 Indicative evaluation

The indicative evaluation has been realized on our test corpus of 74 texts. Considering (Saint-Dizier 2014) for the recognition of discourse structures, the identification of discourse structures in a compound produces the following results:

relation	nb of rules	nb of annotated structures	precision	recall
contrast	14	24	84	88
concession	11	44	89	88
specialization	5	37	72	71
information	6	23	86	80
definition	9	69	87	78
elaboration	13	107	84	82
illustration	20	42	91	83
result	14	97	86	80
circumstance	15	102	89	83
purpose	17	93	91	83

Some relations have more elaborated sets of rules because they have been reused and improved from previous experiments. This explains the differences in number of rules. Some sets of rules may need further expansion to produce more accurate results, this is the case for 'specialization' which remains somewhat vague. Information and definition are not necessarily identified on the basis of marks but on their position in the compound, which is also a vague criterion. In general, however, results are good for discourse analysis.

5 Perspectives

In this paper, we have developed a linguistic model for the analysis and the representation of argument compounds with a logic-based approach. This contribution illustrates and is a first investigation the complexity of argument constructions. Our results form a kind of simple logic-based discourse grammar dedicated to argument compounds. The specific discourse relations we have identified are conceptually characterized, with the functions they play, so that inferences can be drawn within and between argument compounds.

Besides going on improving the recognition of argument compounds, which is still at an early stage, linguistically and computationally speaking, we aim at investigating other forms of arguments in texts which have relatively controlled language forms (e.g. didactic texts). Another important direction is the development of a conceptual model that allows various forms of inferences so that sets of argument compounds can be analyzed for example w.r.t. their coherence or overlap in a text. This is a different perspective than the usual one which consists in constructing argumentation graphs, i.e. chains of arguments.

The implementation is carried out in Dislog on the <TextCoop> platform. Dislog allows high level specifications in logic that allow fast and easy prototyping. Elements of an indicative evaluation are developed: results are good for a discourse processing task. Most of the code of this project is freely available under a Creative Commons BY License and can be obtained from the authors.

Acknowledgements

We thank the French ANR for their funding, under the TextCoop and Lelie projects. We also thank the three anonymous reviewers that made useful comments to improve the initial text.

References

- [1] D. Blakemore, *Relevance and linguistic meaning : the semantics and pragmatics of discourse markers*, CUP, Cambridge, (2002).
- [2] E. Couper-Kuhlen, B. Kortmann, *Cause, Condition, Concession, Contrast: Cognitive and Discourse Perspectives*, Topics in English Linguistics, No 33, de Gruyter (2000).
- [3] J. Delin, A. Hartley, C. Paris, D. Scott, K Vander Linden, *Expressing Procedural Relationships in Multilingual Instructions*, Proceedings of the Seventh IWNLG, (1994), 61-70.
- [4] F.H. Eemeren, R. van Grootendorst, *Argumentation, communication, and fallacies: A pragma-dialectical perspective*, Lawrence Erlbaum Associates, 1992.
- [5] B. Grosz, C. Sidner, *Attention, intention and the structure of discourse*, Computational Linguistics 12(3), (1986).
- [6] E. Hull, K. Jackson, J. Dick, *Requirements Engineering*, Springer, (2011).
- [7] J. Kang, P. Saint-Dizier, *Discourse Structure Analysis for Requirement Mining*, International Journal of Knowledge Content Development and Technology, Vol 3(2), (2013).
- [8] W. Mann, S. Thompson, *Rhetorical Structure Theory: Towards a Functional Theory of Text Organisation*, TEXT 8(3), (1988), 243-281.
- [9] D. Marcu, *The Theory and Practice of Discourse Parsing and Summarization*, MIT Press (2000).
- [10] E. Miltasaki, R. Prasad, A. Joshi, B. Webber, *Annotating Discourse Connectives and Their Arguments*, new frontiers in NLP, (2004).
- [11] I. Mirbel, S. Villata, *Enhancing Goal-based Requirements Consistency: an Argumentation-based Approach*, Computational Logic in Multi-Agent Systems, LNCS Vol. 7486, (2012).
- [12] D. Rosner, M. Stede, *Customizing RST for the Automatic Production of Technical Manuals*, in R. Dale, E. Hovy, D. Rosner and O. Stock eds., *Aspects of Automated Natural Language Generation*, LNAI, Springer-Verlag, (1992), 199-214.
- [13] P. Saint-Dizier, *Processing natural language arguments with the TextCoop platform*, journal of Argumentation and Computation, vol 3(1), (2012).
- [14] P. Saint-Dizier, *Challenges of Discourse processing: the case of technical documents*, April 2014, Cambridge Scholars Publishing.
- [15] M. Saito, K. Yamamoto, S. Sekine, *Using Phrasal Patterns to Identify Discourse Relations*, Proceedings ACL06, (2006).
- [16] J. Spenader, A. Lobanova, *Reliable Discourse Markers for Contrast*, Eighth International Workshop on Computational Semantics, Tilburg (2007).
- [17] M. Stede, *Discourse Processing*, Morgan and Claypool Publishers, (2012).
- [18] M. Taboada, W. C. Mann, *Rhetorical Structure Theory: Looking back and moving ahead*, Discourse Studies, 8(3), (2006), 423-459.
- [19] M. G. Villalba, P. Saint-Dizier, *Some Facets of Argument Mining for Opinion Analysis*, COMMA, IOS Publishing, Vienna, (2012).

- [20] Walton, D., Reed, C., Macagno, F., *Argumentation Schemes*, Cambridge University Press, 2008.
- [21] D. Walton, *Argument Mining by Applying Argumentation Schemes*, *Studies in Logic*, Vol. 4(1), 2011.
- [22] B. Webber, A. Joshi, *Discourse Structure: Past, Present and Future*, *Proceedings of the ACL 2012 Workshop on Rediscovering 50 Years of Discoveries*, (2012), 42-54. Jeju, Korea.
- [23] G. Winterstein, *La dimension probabiliste des marqueurs de discours. Nouvelles perspectives sur l'argumentation dans la langue*. PhD, Paris Diderot-Paris 7, (2010).
- [24] G. Winterstein, *What but-sentences argue for: An argumentative analysis of but*, in *Lingua* 122 (2012), 1864-1885.
- [25] F. Wolf, E. Gibson, *Representing Discourse Coherence: A Corpus-Based Study*, *Computational Linguistics* 31(2) (2005), 249-288.