ROLES, EVENTS AND SAYING EVENTS
IN EXPOSITORY DISCOURSE

OLIVIER J. WINGHART*

AUGUST 1988       AI88-85

* Support for this research is provided by the ARO under grant number DAAG29-84-0060.
Roles, Events and Saying events in Expository Discourse\textsuperscript{1}

Olivier J. Winghart

Artificial Intelligence Laboratory
Taylor Hall 2.124, The University of Texas at Austin
Austin, TX 78712

Abstract

In this paper, we describe a computational representation of expository discourse, separating the elements of the described situation (Roles and Events) from those of the exposition itself (Saying events). The situation elements do not depend on the writer’s intentions; their interrelations are only due to world concept coherence. In particular, we propose a new notion of rhetorical relation as mapping of expansions between situation elements, rather than one single relation between linguistic units. This notion is clearer, and easier to recognize than the typical notion. On the other hand, it is less informative and doesn’t induce a structuring on the text.

Topic Discourse Analysis, Representation, Event Coherence

\textsuperscript{1}Support for this research is provided by the ARO under grant number DAAG29-84-0060.
Contents

1 Introduction 3
  1.1 AI work on text organization 3
  1.2 Analysis of expository text 4
  1.3 About this paper 5

2 Discourse elements 6
  2.1 Roles and events 7
  2.2 Saying events 7
  2.3 Clause representation 8
  2.4 Expansion relations 9

3 A detailed example 12

4 Activity in the model 14

5 The notion of rhetorical relation 18
  5.1 The typical notion 18
  5.2 Rhetorical relations as expansion mappings 21

6 Summary 21

7 Acknowledgements 22

A Text data and their discourse elements 25

B The small Prolog analyzer 28

C Execution trace 35
List of Figures

1. Partonomy of discourse elements ........................................... 6
2. Classes of saying events ..................................................... 8
3. Representation of "The old man ate fish." .............................. 9
4. Classes of expansions ....................................................... 10
5. Expansion examples and meanings ........................................ 11
6. Some expansion mappings .................................................. 13
7. Roles, events and saying events of the Hepatitis text ................. 15
8. The elements in graphic format ............................................. 16
9. The Lexical, Situation and Exposition experts ......................... 17
10. Activity of each expert .................................................... 17
11. An example RS ............................................................. 19
1 Introduction

1.1 AI work on text organization

Tasks faced in text analysis can conceptually be broken down into local ones and global ones. Local tasks include anaphor resolution, predicate argument congruence [Hob87], and word sense disambiguation. On the other hand, a typical global task is to outline a text, that is to show how it realizes a tree structure of exposition goals.

For outlining, the common assumption is that it is done and used during analysis, and not posterior to it. This is an influence of important theoretical works by Reichman [Rei85] and by Grosz and Sidner [GS86], using collaborative dialogue rather than written text. There, participants keep updating their current structure of utterances to be able to answer appropriately. A common default of these works, as noted by Sladky [Sla87] is that they rest on purposes the utterances fulfil and subgoal relations between them. What is left largely unspecified though, is how these purposes and subgoals are recognized. The decision to embed text segment B inside text segment A amounts to deciding whether or not the purpose of saying B is subservient to the purpose of saying A. In many cases, this decision is a matter of human judgment, and consistency may not obtain among different observers. For written text, complete outlining may not be effected in first reading, only derived if necessary, from the discourse elements of that reading.

Text organization can also be described by means of rhetorical relations between segments, with such relation classes as consequence-of, example-of, or elaboration-of. The emphasis there is on the continuation move from one text part to another, more than on a complete embedding structure. There are few research works on recognizing rhetorical relations in text mechanically. We can mention Dahlgren [Dah87], who presents a mechanical analysis of a small narrative, including relations like goal or cause. Nirenburg et. al. [TNR86] [Nir87] [NC87] also derive a representation with rhetorical relations, as a text interlingua for a machine translation system. Hobbs has pursued rhetorical analysis by computer until about 1985 [Hob76] [Hob82] [Hob85], but then switched to more local issues for the Tacitus project [Hob87].

The most notable project on rhetorical organization of text is the Rhetorical Structure Theory, or RST, by Mann, Thompson and Matthiesen [Man84] [MT86] [MT87]. The theory is oriented at generation rather than analysis. It is large in coverage (the authors claim some 25 relation classes cover hundreds of texts reviewed by hand from various genres), and counts many man-years of research. RST is still essentially descriptive as opposed to constructive, although Hovy shows a generation planner which implements a
small part of the theory [Hov88]. Other works in generation exist which use rhetorical relations strategically, e.g. Paris and McKeown to describe physical artifacts [PM87].

In sum, we find on the analysis side that most work is in oral rather than written discourse, and only one or two projects aim at recognizing rhetorical relations in text. None is concerned with expository text. On the generation side, where a main purpose is usually an input datum, there exist working systems which decide on a rhetorical strategy. The most developed rhetorical theory is RST.

1.2 Analysis of expository text

We are concerned with mechanical analysis of expository text, and as we saw above, there is no AI project we can draw from directly. Yet the potential applications are important in two domains. The first domain is instruction, where students have to read expository material from textbooks. An analyzer could be used on the same data to monitor the students' comprehension of the new concepts and terminology. The second domain is technical documentation, where the expository text serves to give useful information on a complex artifact. There an analyzer is a necessary part of a machine translation system.

Our text genre is that of basic encyclopedia or science textbooks, such as [BFMK86]. Paragraphs in that genre are carefully edited and reviewed. They use fairly simple syntax, and fairly standard paragraph organizations.

A general question to address is what analysis should do. One common answer is that it has no overall goal, other than producing a particular output. It consists of going through specific steps like parsing, establishing coreference, or recovering implicit conceptual relations. This answer is inadequate because each step is likely to encounter ambiguities that only knowledge derived in some other step can solve. It is also inadequate because it suggests that the output is determined only by the text. If \( I \) is the interpretation output, \( S \) the initial state (knowledge and goals) of the analyzer, and \( T \) the text data, it suggests \( I = f(T) \) while \( I = f(S, T) \) would be more correct.

The alternative answer is that analysis has an overall goal, as a model of reading. Following this answer, we must state what goal we choose, and how it leads to specific operations on text.

The reader's goal for our text genre is assumed to be unique and fixed, namely to take the maximum instruction (new conceptual knowledge) from the text. This is the underlying intention. It gives rise to a different analysis than if it were e.g. to examine the literary style of the author, or to skim for particular topics of interest.

The underlying goal entails more specific operations, such as identifying the discourse
referents mentioned in the text, both objects and events, and deciding what sayings the
writer does with them (here we call saying a communicative intent directly reflected in
the text, such as giving a title, making a suggestion to the reader, or describing an evènt).
These operations are interdependent, and derive interpretation elements with various levels
of confidence. There are three sources of plausibility for a given interpretation:

1. Plausibility of the writer’s exposition, i.e. how typical the sequence of sayings is in
   the paragraph, using communicative patterns we know,

2. Plausibility of the described situation, using world knowledge patterns,

3. Plausibility of the linguistic use, from syntax, lexical features and collocation.

The three sources are listed from stronger to weaker; in case plausibilities give conflicting
favorites, the stronger supersedes the weaker.

Speaking of confidence levels and plausibility shouldn’t surprise. According to Alter-
man [Alt82], the reader is a sort of detective who cumulatively makes use of text clues.
It is also the case that the writer implements his strategies with no one single item but
rather “in a number of little clues which cumulatively convey a certain impression to the
reader” [Hov87]. Yet confidence levels, if common for treatment of any other expert task,
are not used much in text analysis research (see [RL88] for an exception).

1.3 About this paper

In this paper, we voluntarily restrict our attention to the discourse elements, i.e. the sorts
of objects our model includes. This is a complex enough issue to be discussed by itself.
Two other issues, no less important for a global analysis system are:

1. Syntactic parsing (see for example [SS87])

2. A global confidence level mechanism to handle ambiguity

Both issues are sidestepped in our demonstration program. About the first issue, we
assume as input a verb complex (i.e. a verb and its case arguments) for each clause.
About the second issue, we assume that verb complexes are certain (no syntactic ambigu-
ity), and other discourse elements are either certain or suggested. The fate of a suggested
element is to be either confirmed as certain, or discarded.

Our motivation is to present a computational representation of discourse, one which is
clear and supports a better notion of rhetorical relation than the typical, most common
one. The typical notion is not satisfactory for computation, because the decision that a rhetorical relation holds in a particular case is not reliable enough to allow consistency between different observers.

The remainder of the paper is as follows. First we present the discourse elements in our model, and a small demonstration analysis (the corresponding Prolog program can be found in Appendix B, and the execution trace in Appendix C). We describe activity in the model in terms of experts applying patterns to a working base of elements. We then discuss our notion of rhetorical relation taking RST as reference, and summarize. Appendix A shows discourse elements for additional expository texts from [BFMK86], derived by hand.

2 Discourse elements

The partonomy of our discourse elements is given in Figure 1. For a given text, there are some elements which represent the described situation (the said). Works such as [Alt82] and [Nor87] are entirely devoted to this level. They try to interrelate the conceptual tokens realized in the text, using concepts in a knowledge dictionary, and a delimited set of arcs. Other elements represent the exposition itself (the saying) in the form of saying events. This other level of elements is usually not present in other researchers' work, Nirenburg being an exception [NC87]. We present roles, events, and saying events, as well as expansion relations, in the following subsections. Expansion relations are particular relations between role and event tokens (that is, between situation tokens), which hold across verb complexes.
2.1 Roles and events

Roles are the discourse entities, either mentioned explicitly, or necessary for expansions. As an example for this last case, in the text "I was driving on the freeway. The engine started to make a noise.", a vehicle role is necessary, of which the engine role is a part. Roles are so called because they are always presented as filling a role in some event or saying event.

World events, or just events, are the happenings in the described situation. We use the word event in a more global sense than usual, for events as well as states. Events have their own coherence, independent of discourse exposition, as demonstrated by Alterman [Al85]. There are direct relations by which events mutually define each other. Using Alterman's examples, gorging on something is a kind of eating (taxonomic relation), soaking is a subsequence of washing (partonomic relation), and falling is a consequent of dropping (temporal relation). This coherence only depends on a concept dictionary, not on the exposition chosen for the text. It is natural to distinguish roles from events, since events are markedly temporal and roles are not. We note though that roles and events do not correspond one on one to nouns and verbs. They do most of the time, but abstract temporal concepts and nominalizations present events as nouns.

Both roles and events can have conceptual features, contributed by adjectives, adverbs and other modifiers. They can also carry marks such as tense, determiner, or force. Marks are lexical or syntactic indications, but play an important part in the propositional content of a text. We do not claim to handle them in any complete way. Tense and determiner abstract away complex issues of tense, aspect and quantification. A possible value for tense might be simple-past, and one for determiner might be def-sing (for definite singular). Force indicates whether, according to clause syntax, a role or event is focus or point of assertion, in the sense of the following definitions. The point of assertion is generally the tensed verb in the clause, and the focus is its functional argument appearing first in surface order. This notion of focus, solely determined by clause syntax, is distinct from the notion of focus as most foregrounded referent in the writer's current point.

2.2 Saying events

Saying events are the happenings in the communication itself, as opposed to what happens in the described situation. In our text genre, there are only a few classes of saying events the writer can bring about, shown in Figure 2. They all have one necessary argument, called obj, for the concept token at which the saying event happens.
Figure 2: Classes of saying events

*Describe* is the default saying event for narratives; it just adheres to the corresponding world event. This may be why the two are usually not separated. For the other saying events, we choose not to have any corresponding world event. For instance, titling on a concept, classifying a concept, or giving a new terminology for it, only happens in the communication, not in the outside world. Equivalently, the reader may assume there is a world event if he so wishes, but one which only duplicates the information of the saying event.

### 2.3 Clause representation

At this point, it may be best to show the representation of a simple clause in terms of roles, events, and saying events. Figure 3 shows the tokens for “The old man ate fish.” The representation is standard, except the saying events part, typically omitted in other works. It is similar to the representation used in the MCC system Lucy [RL88], as well as the one used by Allen in his textbook [All87b].

The representation readily translates to first order logic, but with the advantage that

---

2This is the case in particular for two-sentence narratives given by researchers to exemplify their rhetorical relations.
Figure 3: Representation of “The old man ate fish.”

relations are indexed by tokens. Relation \([A, R, B]\) stands for the binary predicate \(R(A, B)\), indexed under token \(A\). Tokens are existential variables\(^3\). Concepts and feature values are constants. The equivalent formula would thus start as:

\[\exists \text{man} \cdots \text{tok(\text{man}, \text{man})} \land \text{age(\text{man}, \text{old})} \cdots\]

For comparison, the representation of the entire clause in the first order logic system Tacitus would be:

\[\exists e_1, x_1, x_2 \text{ past}([e_1|\text{eat}'(e_1, [x_1|\text{man}(x_1) \land \text{old}(x_1) \land \text{the}(x_1)], [x_2|\text{fish}(x_2)])])\]

2.4 Expansion relations

We saw above a representation in which the situation elements (roles and events) are tokens, and each token is associated with a list of binary relations in which it appears as first argument. Among these binary relations are expansion relations, in short expansions, linking a token to another token in a different verb complex. Given two verb complexes,

\(^3\)Tokens have short names for convenience, combined with the verb complex number, such as \(hepa-1\) for a hepatitis token mentioned in verb complex 1, instead of an atom like \(x36\) (see \([\text{Hob82}]\) for a similar convention).
Figure 4: Classes of expansions

the set of all expansions between one token in the first complex and one token in the second one expresses how the two complexes are relevant to each other. We call such a set an expansion mapping, and this is our notion of rhetorical relation (more on rhetorical relations is given in Section 5 below).

As Figure 4 shows, there are four categories of expansions, which decompose into ten classes. The four categories are Categorial (c), Causal (gc), Temporal (t), and Parallel (p).

- An expansion is Categorial if one token is taxonomic parent of the other, or is a part of, or plays a role for the other. We know for instance that chairs are part of offices, or that cars play a role in accidents.

- An expansion is Causal if one token is known to bring about the other with some degree of causality, with or without human intention. Examples are that earning money is done with the intent of buying goods, or that a physical object when hit may break.

- An expansion is Temporal if there is a time overlap, sequence, or subsequence between
<table>
<thead>
<tr>
<th>Expansion</th>
<th>Example</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Categorial</td>
<td></td>
<td></td>
</tr>
<tr>
<td>coext(A,B)</td>
<td>coext(dog1,animal2)</td>
<td>A is a B or B is a A</td>
</tr>
<tr>
<td>part(A,B)</td>
<td>part(office1,chair2)</td>
<td>A has part B</td>
</tr>
<tr>
<td>role(A,B)</td>
<td>role(accident1,car2)</td>
<td>A has role B</td>
</tr>
<tr>
<td>Causal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>goal(A,B)</td>
<td>goal(earn1,buy2)</td>
<td>A has goal B</td>
</tr>
<tr>
<td>cond(A,B)</td>
<td>cond(hit1,break2)</td>
<td>A conditions B (with some cause)</td>
</tr>
<tr>
<td>Temporal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>temp(A,B)</td>
<td>temp(enter1,sit2)</td>
<td>A time-precedes B</td>
</tr>
<tr>
<td>cotemp(A,B)</td>
<td>cotemp(eat1,drink2)</td>
<td>A is cotemporal with B</td>
</tr>
<tr>
<td>subtemp(A,B)</td>
<td>subtemp(eat1,pay2)</td>
<td>A is a subsequence of B</td>
</tr>
<tr>
<td>Parallel</td>
<td></td>
<td></td>
</tr>
<tr>
<td>similar(A,B)</td>
<td>simil(run1,walk2)</td>
<td>A is similar to B</td>
</tr>
<tr>
<td>disimil(A,B)</td>
<td>disimil(open,closed)</td>
<td>A is opposed to B</td>
</tr>
</tbody>
</table>

Figure 5: Expansion examples and meanings
two tokens, without any known causality. Almost any couple of events can occur in the same time frame, but we have scriptal knowledge of typical subsequences, such as eating in a restaurant.

- A Parallel expansion is between two tokens known to be similar or opposed, such as open and closed.

Meanings and examples are given for the ten expansion classes in Figure 5. Depending on the knowledge base used, the expansion relations are either primitive arcs between concepts, or result from combination of such arcs into paths. In our application, expansions are simple combinations of primitive conceptual arcs. They can also be suggested by some explicit signal, especially Temporal and Parallel expansions ('but' can suggest a Disimil expansion with what would be a Simil expansion otherwise).

These expansion classes are very broad, more so than typical rhetorical relation categories. They are meant to expand on Alterman's relations [Alt85], which only cover our Causal and Temporal categories, and the Categorial relations between two events. They maintain Alterman's argument in favor of abstraction: abstract classes allow us to avoid complex judgmental decisions which would be required for more specific classes. The Cond expansion for instance encompasses the occasion, enablement, and consequence relations. The classification decisions are easier to compute, and more consistent among observers.

There are minor design decisions about expansion mappings. First, we assume that given two tokens in distinct verb complexes, there can be at most one expansion. Second, we represent text order by conventionally representing all expansions from the point of view of the most recent token. For the Coext, Cotemp, Simil and Disimil expansions, this doesn't make any difference. These expansions are symmetric, i.e. if \( R(A, B) \) then \( R(B, A) \). All the other expansions are antisymmetric, i.e. if \( R(A, B) \) then \( \neg R(B, A) \). For those, we indicate by + or − the direction of the expansion. Suppose \( R(A, B) \) holds. If \( A \) is the most recent, the relation will be \( [A, +R, B] \); if \( B \) is the most recent, the relation will be \( [B, −R, A] \). With this convention, we can represent both the cause and consequence relations using +Cond and −Cond. Figure 6 shows examples of expansion mappings using this convention, for typical two-clause narratives.

3 A detailed example

The small Prolog program in Appendix B is made to analyze the following text:
The inspector checked for bugs. The secretaries had reported roaches.

There was an accident. Two vehicles collided.

I entered the room. The chandelier was large.

John can open Bill's safe. He knows the combination.

I was driving on the freeway. The engine made a noise.

---

**Legend**

- Exp: Expansion
- X: Token
- Role relation
- Verb complex

Figure 6: Some expansion mappings
Hepatitis

Hepatitis is a disorder of the liver. In hepatitis, the liver becomes inflamed.
The infected person loses appetite.

After the program is run, the working base of roles, events and saying events is shown in Figure 7. Figure 8 shows these same elements, but in graphic format. We comment here on the main decisions to reach that final working base (a complete execution trace is given in Appendix C).

With the first verb complex, a Classify saying event (class-1) is abstracted with a lexical pattern. The mentioned disorder (diso-1) is specialized as an organic disorder, since it is of the liver and liver is an organ; the relation between diso-1 and liver-1 is specialized to affected organ. The same sort of specialization by context (or concretion as Norvig calls it [Nor87]) is found in other clauses, and we won't mention it again.

For the second verb complex, two coextension are suggested, one to hepa-1 (hepatitis token), and one to liver-1. No particular saying event is recognized, so the default Describe (descr-2) is posted, with the event become-2 as object. become-2 as organ state change is a consequence of diso-1, confirming coextension from liver-2 to liver-1. Finally, the event become-2 is attributed to hepa-1 in the saying event attr-2, confirming coextension from hepa-2 to hepa-1. The expansion mapping from complex 2 to complex 1 thus consists of two Coext and one Cond expansion.

The third verb complex, like the second one, gives rise to describing an event (lose_ap-3 in this case), and attributing it to hepa-1.

We note that with the Attribute saying events, we can associate the two symptoms to the disease, without having to decide what subgoal relations obtain between saying events. Without composing saying events and forming a tree, the saying events correspond exactly to what a more semantic analysis — i.e. geared specifically to disease description — would give.

4 Activity in the model

We describe in this section the operations and control used to derive the discourse elements from text data in the form of surface semantic triples.

The overall control is pattern-directed. For each verb complex added to the working base, all patterns that can apply are executed, changing the working base by adding or modifying discourse elements. Analysis is completed when no more patterns apply.
Figure 7: Roles, events and saying events of the Hepatitis text
Figure 8: The elements in graphic format
Figure 9: The Lexical, Situation and Exposition experts

Lexical expert
From dependency, specialize role and role relation
From lexical pattern, add saying event

Situation expert
From two roles or events, suggest Categorial expansion
From explicit signal, suggest expansion
From two verb complexes, search for all expansions
(possibly confirming suggestions)

Exposition expert
From current verb complex, add saying event
if not already added
From saying event pattern, add saying event (possibly confirming suggestions)

Figure 10: Activity of each expert
Conceptually, we can separate the patterns in three experts, namely a Lexical, Situation, and Exposition experts, corresponding to the three sources of plausibility used in reading. Figure 9 shows this, assuming that surface semantic triples are output by a dependency parser.

The operations effected by each expert are given in Figure 10. Two points are important. First, elements (both suggested and certain ones) are not posted in any prespecified order. For instance, it is not the case that exposition elements necessarily appear after situation elements for a given complex. They may appear early on, by lexical pattern application.

Second, the search for expansions follows a suggest/confirm cycle. Possible Categorial expansions are suggested first, then all other expansions are attempted using world concept coherence. In the process, any suggestion needed for a proof is confirmed as certain (this is a simplistic confidence mechanism using just two values).

5 The notion of rhetorical relation

In this section, we contrast the typical notion of rhetorical relation, using RST as reference, with the notion of rhetorical relation as expansion mapping. The latter has several advantages, but is less informative, than the former.

5.1 The typical notion

There are three main questions about rhetorical relations. We examine them in turn, using RST.

The definition question What content do the relations have? In particular, do they hold between linguistic units, or between conceptual units? Given two such units, can there be more than one relation between them? Do they induce a structuring of the text, as dependencies do for syntactic constituents? Do they depend only on the elements they link, or also on the embedding text?

The motivation question How necessary is it to take the relations into account, and what happens if we don’t? The question can be divided between generation and analysis. Norvig [Nor87] and Grosz and Sidner [GS86] for instance doubt that the relations are necessary for the reader, but they think the writer must know them. For analysis, the motivation question is raised in Allen’s survey [Al137a]: are the
Text

1. I know you have great credentials.
2. You don't fit the job description
3. because this job requires someone with extensive experience.

Rhetorical structure

```
4 Concession

1 ____________________________ 5 Reason

2 3
```

Figure 11: An example RS

relations useful in ongoing interpretation, or are they classified after interpretation? In the first case, interpretation is negatively affected if we subtract their recognition, in the other it is not.

The recognition question Assuming answers to the definition and the motivation questions, what is a reasonable way to recognize the relations in actual data?

Let us briefly describe RST, in order to address the definition question. In RST, rhetorical relations hold between clauses, or between larger segments connected by rhetorical relations themselves. A relation always takes one main argument (the nucleus) and at least one subordinate argument (satellite), to form a larger segment, with the properties:

1. The nucleus is the one argument whose function most nearly represents the function of the larger segment [Man84],

2. Among the propositions embodied in the larger segment figures a relational proposition, contributed by the rhetorical relation.
An example of resulting rhetorical structure (RS) is given in Figure 11. In graphic notation, a rhetorical relation is an arc from satellite to nucleus, and a segment is linked to its nucleus by a vertical line. Numbered horizontal lines stand for lists of propositions embodied in a text segment. Line 5 for instance stands for propositions embodied by clauses 2 and 3, plus the relational proposition that the 'job requirement' is a reason for the 'non fitting'.

The answer RST gives to the definition question is the following. Rhetorical relations hold between linguistic units, either clauses or larger segments. Their content is implicit propositions, called relational propositions. The relations are recursive, and distinguish their main argument (the nucleus) from the others. They induce a structure on the text which is formally similar to a syntactic constituent structure. Several relations may hold between the same two segments (the authors in [MT86] mention Sequence and Cause as an example). In such a case, there is always a most salient one; deciding which one it is depends on the embedding text.

For the motivation question, there are (at least) two answers. One is expanded in [MT86], to the effect that relational propositions are crucial among all propositions, and they share communicative properties with explicit propositions:

- Adding a direct assertion of the proposition makes the text redundant.
- Adding a direct assertion of the negated proposition makes the text incoherent.
- The author is held responsible for what the proposition says; if it is contrary to fact, he can be accused of being insincere.

Another answer is that ambiguity regarding an argument of a proposition may arise without the relation. A simple example is given by Nirenburg [NC87]:

a) Horses are not reliable animals, and neither are mules. These beasts of burden have caused more trouble than they are worth.

b) Horses are reliable animals, but mules are not. These beasts of burden have caused more trouble than they are worth.

In this example, a parallel (respectively contrast) relation is responsible for attributing the 'causing trouble' to 'horses and mules' (respectively 'mules'). The second clauses in a) and b), even though they are equal in English — they would not be so in Spanish — do not embody the same propositions.
The recognition question is the most uncertain of the three questions. According to Mann and Thompson\textsuperscript{4}, a relational proposition is posited on the basis of very abstract features of clause content, such as tense, aspect, and verb classification of the cTause verbs. A plausibility judgment is then made on the whole text interpretation, assuming the proposition holds. As a result, the conjectured proposition is confirmed as part of the text, or discarded.

This description seems correct, but is not amenable to computer implementation. The confirming part of the posit/confirm cycle involves reasoning not only about the concepts involved, but also about the communicative goals of the writer.

5.2 Rhetorical relations as expansion mappings

Regarding the definition question, we saw that rhetorical relations in our model consist of expansions, each holding between two concept tokens (roles or events). Expansions only depend on world concept coherence, not on the embedding text or any factor of the exposition. They do not induce a structuring of the text.

For the recognition question, our notion has advantages over the typical one. The categories of expansions are more abstract, thus easier to decide, than e.g. RST relations. Instead of one specific relation between two segments, the continuation is described by a mapping of expansions, each easier to decide. In general, the mapping cannot reduce to one conceptual path. For instance with the text "While the input valve is open, the output valve is closed.", there is a contrast expansion between ‘open’ and ‘closed’, as well as a cotemporal expansion between the ‘open state’ and the ‘closed state’. In these cases, consistency of deciding multiple expansions is greater than that of deciding one most salient RST relation.

In sum, the notion of rhetorical relation as expansion mapping fares better on the recognition question, but is less informative than the typical notion. An additional drawback is that mappings do not induce a structuring of the text, contrary to RST relations. This prevents a general summarization procedure by truncation, based on rhetorical relations.

6 Summary

We showed in this paper a computational representation of expository discourse, which is clear, can be computed from surface semantic format, and supports a better notion of

\textsuperscript{4}Personal communication.
rhetorical relation than the typical one.

The representation separates on one hand roles and events for the described situation, and on the other saying events for the exposition itself. Saying events capture, for a given text genre, the kinds of communicative intents directly reflected in the text. They are necessary to determine the relative import of roles and events.

Rhetorical relations only concern relations between situation concepts (roles and events), so that they do not depend on the writer’s intentions. They are at a level of abstraction such that complex judgmental decisions are avoided. Both of these features make their recognition easier than that of RST relations.

Finally, they do not necessarily reduce to a single relation between concept nodes; rather they are mappings of such relations. Instead of a single specific relation which is hard to decide, there are typically several expansions, each easily decidable.

If the main advantage of mappings as opposed to single relations is mechanical recognition, the main drawback is that they do not induce a structuring on the text. Deriving a text structure in our representation would depend on both expansion mappings and saying events.

7 Acknowledgements

I would like to thank Dr. Robert Simmons and Wing-Kwong Wong for their helpful comments on earlier drafts of this paper.

References


A  Text data and their discourse elements

N.B. The texts are extracted with minor change from [BFMK86]. For each clause, we show roles, events, and saying events. Relations [A,R,B] under token A are written [R,B], omitting A.

**Text**: Most of our energy is produced by the sun. The sun gives us heat and light. We call this energy \textit{\textbf{\textit{\bf{radiant energy}}}}. It warms the earth, sustains life, and produces our changing weather conditions.

*Roles
ener-1 (tok energy) (poss-by we-1)
we-1 (tok we)
sun-1 (tok sun)

*Events
prod-1 (tok produce) (actor sun-1) (patient ener-1)

*Sayevents
descr-1 (tok describe) (obj prod-1) (focus ener-1)

*Roles
sun-2 (tok sun) (coext sun-1)
we-2 (tok we) (coext we-1)
heat-2 (tok heat)
light-2 (tok light)
list-2 (tok *list*) (of heat-2) (of light-2)

*Events
give-2 (tok give) (actor sun-2) (recip we-2) (patient list-2)
(-cond prod-1)

*Sayevents
descr-2 (tok describe) (obj give-2) (focus sun-2)

*Roles
we-3 (tok we) (coext we-1)
ener-3 (tok energy) (coext list-2)

*Events
*Sayevents
rename-3 (tok rename) (obj ener-3) (name radiant-energy)

*Roles
it-4 (tok it) (coext list-2)
earth-4 (tok earth)
life-4 (tok life)
wcond-4 (tok weather-condition)

*Events
Rail transportation is transportation in vehicles that run on rails. There are several different kinds of rail transportation. For instance, \{\bf subways\} are electrically powered trains. They run on special tracks in tunnels underground. \{\bf Elevated trains\}, or Els, run on tracks above the ground. However, the most common kind of rail transportation takes place at ground level, or \{\bf on-grade\}. Diesel- or coal-powered engines are used to move goods over steel tracks.

\*Roles
rtrans-1 (tok rail-transportation)
trans-1 (tok transportation)
vehic-1 (tok vehicle) (-part trans-1)

\*Events
\*Sayevents
title-1 (tok title) (obj rail-transportation)
class-1 (tok classify) (obj rtrans-1) (class trans-1)

\*Roles
rail-2 (tok rail)

\*Events
run-2 (tok run) (instr vehic-1) (on rail-2)
\*Sayevents
descr-2 (tok describe) (obj run-2) (focus vehic-1)

\*Roles
rtrans-3 (tok rail-transportation) (coext rtrans-1)

\*Events
\*Sayevents
exemp-3 (tok exemplify) (obj rtrans-3) (nb several)

\*Roles
subway-4 (tok subway)
train-4 (tok train) (coext vehic-1) (power electric)

\*Events
Sayevents
class-4  (tok classify) (obj subway-4) (class train-4)

Roles
they-5   (tok they) (coref subway-4)
track-5  (tok track) (in tunnel-5) (simil rail-2)
tunnel-5 (tok tunnel) (loc underground)

Events
run-5    (tok run) (instr they-5) (on track-5)
          (simil run-2)

Sayevents
descr-5  (tok describe) (obj run-5) (focus they-5)
attrib-5 (tok attribute) (obj subway-4) (obj run-5)
exemp-5  (tok exemplify) (obj run-2) (with run-5)

Roles
eletrain-6 (tok elevated-train)
track-6  (tok track) (loc above-ground)
          (simil track-5) (simil rail-2)

Events
run-6    (tok run) (instr eletrain-6) (loc track-6)
          (simil run-5) (simil run-2)

Sayevents
descr-6  (tok describe) (obj run-6) (focus eletrain-6)
rename-6 (tok rename) (obj eletrain-6) (name el)
exemp-6  (tok exemplify) (obj run-2) (with run-6)

Roles
kind-7   (tok kind) (subc rtrans-7)
rtrans-7 (tok rail-transportation) (coext rtrans-1)
glevel-7 (tok ground-level)

Events
tplace-7 (tok take-place) (instr kind-7) (at glevel-7)
          (disimil run-6) (disimil run-5) (simil run-2)

Sayevents
descr-7  (tok tplace-7) (focus kind-7)
rename-7 (tok rename) (obj glevel-7) (name on-grade)
exemp-7  (tok exemplify) (obj run-2) (with tplace-7)

Roles
gen-8    (tok engine) (power list-8)
          (-part kind-7)
list-8   (tok *orlist*) (of diesel) (of coal)

Events
use-8  (tok use) (instr engine-8)
*Sayevents
descr-8  (tok describe) (obj use-8) (focus engine-8)
attrib-8  (tok attribute) (obj kind-7) (prop use-8)

*Roles
good-9  (tok good)
track-9  (tok track)
          (simil track-6) (simil track-5) (simil rail-2)
*Events
move-9  (tok move) (instr engine-8) (patient good-9) (on track-9)
          (-cond use-8)
*Sayevents
descr-9  (tok describe) (obj move-9) (focus engine-8)

B  The small Prolog analyzer

% File: miniop
% Author: ow
% Date: 7/88
% Purpose: Simple pattern-directed program and interpreter

:- ensure_loaded([
  library(lists) % reverse/2
  ]).

:- dynamic obj/2. % for objects
:- dynamic sug/2. % same as obj/2, but for suggested objects
:- dynamic current/1. % to keep index of current verb complex
:- dynamic closed/1. % to keep indices of closed anterior complexes
:- dynamic done/1.

:- op(300,yfx,-->). % to make priority lower than /
:- op(800,xfx,--->). % rule operator
:- op(900,fy,not). % to have a readable not

not X :- \+X.

% obj(\text{Index,\text{Triple}}) as atomic element of interpretation
% Possible syntax: \textbf{Index} ::= \text{Level/Relation_type/Ref}
% \textbf{Triple} ::= \text{[Ref,Relation,Ref1]}
% Example of initial base
% Just so rules
% Rules just add or replace obj/2 predicates, and record their execution
% so that they are not infinitely reexecuted (with done/1 predicate).

% If not done(setcurrent), update complex and assert done
[ not done(setcurrent) ]
---> [update_current,assertz(done(setcurrent)) ].

% Lexical patterns
% If 'A being B', add a classify saying event
[ obj(s/rel/A,[A,being,B]) ]
---> [replace(obj(s/rel/A,[A,being,B]),obj(s/rel/A,[A,subc,B])),
  A = _ -N,addlist([,[e/say/class-N,[class-N,tok,classify]],
  [e/say/class-N,[class-N,obj,A]],
  [e/say/class-N,[class-N,class,B]]]),
  assertz(done([A,B])) ].

% If 'In B, A', and B has a coextention, suggest a part expansion
[ obj(s/rel/A,[A,in,B]),not done([A,B]),
  obj(s/mark/B,[B,force,fronted]),
  canhold([B,coext,B1]) ]
---> [addsug(s/exp/A,[A,-part,B1]),
  assertz(done([A,B])) ].

% If 'A Rel B', add relations resulting from grouping A with B along Rel
[ role_or_event(A),obj(s/rel/A,[A,Rel,B]),role_or_event(B),
  not done([A,B]),not done([B,A]) ]
---> [infer0(A,B,Rel),assertz(done([A,B])) ].

% Situation rules
% If T token in current complex and T1 in anterior nonclosed complex,
% propose Coext, Part or Role if possible from T to T1
[ obj(s/tok/T,_,),incurrent(T),
  obj(s/tok/T1,_,),not insame(T,T1),
  not done([T,T1]),not done([T1,T]) ]
--- [inferi(T,T1), assertz(done([T,T1])))].

% Exposition

% if no saying event for current complex, take asserted point and focus
% and add a describe saying event
[ current(N), not obj(e/say/_-N,_) ]
--- [complex(N,Point,Focus), addlist([ e/say/descr-N,[ descr-N,tok,describe] ],
 [e/say/descr-N,[ descr-N,obj,Point]],
 [e/say/descr-N,[ descr-N,focus,Focus]]]) ]

% if current complex B and anterior nonclosed complex A not done yet,
% best expand(A,B) and assert done
[ current(N), complex(N,B,_) ,
 complex(M,A,_),M<N, not closed(M), not done([M,N]) ]
--- [best expand(A,B), assertz(done([M,N]))) ]

% if current complex has a describe say, and an anterior complex has
% a classify, and the description focus is slot of the classified,
% attribute the description to the classified
[ obj(e/say/B,[B,tok,describe]), incurrent(B), current(N),
 obj(e/say/A,[A,tok,classify]), not done([A,B]), not insame(A,B),
 holds([B,obj,Y]), holds([B,focus,F]), holds([A,obj,O]),
 self_or_slot(O,F) ]
--- [addlist([e/say/attr-N,[ attr-N,tok,attribute] ],
 [e/say/attr-N,[ attr-N,obj,O]],
 [e/say/attr-N,[ attr-N,prop,Y]]),
 assertz(done([A,B]))) ]

update_current :-
(current(L) -> % some anterior complex already in base
 obj(e/_-N,_,N>L, replace(current(L),current(N))
 ; obj(e/_-N,_,assertz(current(N))))). % first complex

incurrent(A) :- current(N), A=_-N.
inside(A,B) :- A=_-N, B=_-N.
iclosed(A) :- A=_-N, closed(N).

role_or_event(A) :- obj(s/Type/A,[A,tok,_]), (Type=tok;Type=evt).

% inferO(A,B,Rel) :- A and B are two conceptual tokens functionally related
% with Rel; A and B are collated, which adds a role relation, and possibly
% concretion of either A and B.

infer0(A,B,of) :- canbe(B,organ), canbe(A,organ_disorder), replace(obj(s/rel/A,[A,of,B])), obj(s/rel/A,[A,affected орган,B])).
infer0(A,B,cARRIER) :- canbe(B,organ), canbe(A,organ_state_change), replace(obj(s/rel/A,[A,cARRIER,B])), obj(s/rel/A,[A,changed орган,B])).
infer0(A,B,with) :- canbe(B,infect), canbe(A,diseased_person), retract(obj(s/rel/A,[A,with,B])).
infer0(_,_,_).

canbe(A,C) :- obj(s/Type/A,[A,tok,C1]), token(A,C),
           (not C=C1
           -> add(s/Type/A,[A,subc,C])
           ; true).
canbe(A,C) :- token(A,C1), isa(C,C1), % concretion case
           replace(obj(s/Type/A,[A,tok,_,_]), obj(s/Type/A,[A,tok,C])).

% C is taxo. father or grandfather of token or event A
token(A,C) :- obj(s/Type/A,[A,tok,C1]), (Type=tok; Type=evt).
           (C1=C; isa(C1,C)).

isa(liver,organ).
isa(organ_disorder,disorder).
isa(become, state_change).
isa(organ_state_change, state_change).
isa(diseased_person, person).
isa(lose_appetite, subject_state_change).

% infer1(A,B) :- A is a role or event in the current complex, and B is in % some anterior nonclosed complex. Then Coext or Part expansions are % suggested from B to A.
infer1(T,T1) :- coextensive(T,T1), addsug(s/exp/T,[T,coext,T1]).
infer1(T,T1) :- token(T,C), token(T1,C1), world(part/[C1,C],[T1,R,T]),
           addsug(s/exp/T1,[T1,R,T]).
infer1(_,_).

coextensive(T,U) :- obj(s/tok/T,[T,tok,A]),
          obj(s/tok/U,[U,tok,B]), ref2(A,B).

ref2(A,A).
ref2(A,B) :- isa(A,B); isa(B,A).

% canhold(Triple) :- Triple holds as object, or it is just a suggestion,
% but since it is needed, it is enhanced as object.
holds(Trip) :- obj(_, Trip).

canh( Trip ) :- holds( Trip ).
canh( Trip ) :- sug(Index, Trip), replace(sug(Index, Trip), obj(Index, Trip)).

% Situation

% world( Type/Concept_pair, Expansion_fact ) :- Constraints.
% used by best_expand
world( temp/ [organ_disorder, organ_state_change], [Diso, cond, Chan] ) :-
canh( [[Diso, affected_organ, O01]], cananh( [[Chan, changed_organ, O02]],
( O1=O02; cananh( [[O2, coext, O1]]))).
world( temp/ [organ_disorder, subject_state_change], [Diso, cond, Chan] ) :-
canh( [[Diso, disorder_subject, S1]], cananh( [[Chan, subject, S2]],
( S1=S2; cananh( [[S2, coext, S1]]))).
world( part/ [organ_disorder, diseased_person], [_, disorder_subject, _]).

% Exposition

complex(W, Point, Focus) :- obj(s/mark/Point, [Point, force, assert]),
Points=\W,
obj(s/rel/Point, [Point, _, Focus]). % the first token in arg order

best_expand(A, B) :- token(A, C1), token(B, C2),
world(temp/[C1, C2], [A, Exp, B]),
add(s/exp/B, [B, Exp, A]).
best_expand(A, B) :- token(A, C2), token(B, C1),
world(temp/[C1, C2], [B, Exp, A]),
add(s/exp/B, [B, Exp, A]).
best_expand(_, _).

self_or_slot(A, A).
self_or_slot(A, B) :- cananh( [[B, coext, A]]).
self_or_slot(A, B) :- obj(s/exp/B, [B, _, A]).
self_or_slot(A, B) :- cananh( [[A, subc, C]]),
( obj(s/exp/B, [B, _, C]); obj(s/exp/C, [C, _, B])).

%-----------------------------------------------
% Interpreter
% From: Bratko, Chap. 16, Pattern-directed programming
% Purpose: Interpreter for pattern-directed programs
% Rules are examined in Prolog base order, all of them at each cycle. The
% first Condition to match is executed.
run :- Condition --> Action,
test(Condition), execute(Action).
run :- !, write('No more patterns apply.'), nl, nl.

test([]).
test([First|Rest]) :- call(First), test(Rest).

execute([stop]) :- !.
execute([]) :- run.
execute([First|Rest]) :- call(First), execute(Rest).
%---------------------------------------------------------

replace(sug(I,F), obj(I,F)) :- retract(sug(I,F)), !, assertz(obj(I,F)),
write(F), write(' confirmed'), nl. % special case
replace(A,B) :- retract(A), !, assertz(B), % general case
(B=obj(_,Fact), write(Fact), nl % for user interface
; true).

% Top functions and tests

% addrun(List) :- List is a list of objects [Index,Fact], which are recorded,
% then processed by the patterns.
addrun(List) :- addlist(List), nl,
(done(setcurrent) -> retract(done(setcurrent))
 ; true), run.
addlist([]).
addlist([[Index,Fact]|Rest]) :- assertz(obj(Index,Fact)),
write(Fact), nl, % for user interface
addlist(Rest).

add(Index,Fact) :- assertz(obj(Index,Fact)),
write(Fact), nl.
add sug(Index,Fact) :- assertz(sug(Index,Fact)),
write(Fact), write(' suggested'), nl.

cleanup :- retractall(obj(_,_)), retractall(sug(_,_)),
retract(current(_)), retractall(closed(_)), retractall(done(_)).

% showbase :- all current objects in the base are displayed, ordered by
% referent, with s objects (Roles and Events) before e objects (Sayevents)

showbase :- write('** Current working base'),nl,
            write('** Roles'),nl,showroles,
            write('** Events'),nl,showevents,
            write('** Sayevents'),nl,showsays.

showroles :- obj(s/tok/A,[A,tok,_]),showabout(s,A),fail.
            showroles.

showevents :- obj(s/evt/A,[A,tok,_]),showabout(s,A),fail.
            showevents.

showsays :- obj(s/say/A,[A,tok,_]),showabout(s,A),fail.
            showsays.

showabout(T,A) :- showabout1(T,A),!.
showabout1(Type,A) :- write(A),nl,
             setof(T,Rel~(obj(Type/Rel/A,T);sug(Type/Rel/A,T)),Trips),
             reverse(Trips,Trips1), % from library(lists)
             showtrips(Trips1).
             showtrips([]).
             showtrips([T|Trips]) :- tab(4),write(T),nl,showtrips(Trips).

%-------------------------------------------------------------------
% Text data in surface semantic triples

% test1,test2,test3: 'HEPATITIS
% Hepatitis is a disorder of the liver. In hepatitis, the liver becomes
% inflamed. The infected person loses appetite.'

test1 :- addrun([]
              [s/tok/hepa-1,[hepa-1,tok,hepatitis]],
              [s/tok/hepa-1,[hepa-1,tok,hepatitis]],
              [s/tok/hepa-1,[hepa-1,tok,hepatitis]],
              [s/tok/hepa-1,[hepa-1,tok,hepatitis]],
              [s/tok/dis-1,[diso-1,force,assert]],
              [s/tok/dis-1,[diso-1,tok,disorder]],
              [s/tok/dis-1,[diso-1,of,liver-1]],
              [s/tok/liver-1,[liver-1,tok,liver]])).

test2 :- addrun([]
              [s/tok/hepa-2,[hepa-2,tok,hepatitis]],
test3 :- addrun([  
[s/tok/pers-3,[pers-3,tok,person]],  
[s/rel/pers-3,[pers-3,with,infect-3]],  
[s/evt/infect-3,[infect-3,tok,infect]],  
[s/mark/lose_ap-3,[lose_ap-3,force,assert]],  
[s/evt/lose_ap-3,[lose_ap-3,tok,lose_appetite]],  
[s/rel/lose_ap-3,[lose_ap-3,subject,pers-3]]  
]).

text1 :- test1,test2,test3.

C Execution trace

yes  
| ?- text1.  
[tit-1,tok,title]  
[tit-1,obj,hepatitis]  
[hepa-1,tok,hepatitis]  
[hepa-1,being,diso-1]  
[diso-1,force,assert]  
[diso-1,tok,disorder]  
[diso-1,of,liver-1]  
[liver-1,tok,liver]  
[hepa-1,subc,diso-1]  
[class-1,tok,classify]  
[class-1,obj,hepa-1]  
[class-1,class,diso-1]  
[liver-1,subc,organ]  
[diso-1,tok,organ_disorder]  
[diso-1,affected_organ,liver-1]  
No more patterns apply.
[hepa-2, tok, hepatitis]
[hepa-2, force, fronted]
[become-2, force, assert]
[become-2, tok, become]
[become-2, in, hepa-2]
[become-2, carrier, liver-2]
[become-2, attrib, inflamed]
[liver-2, tok, liver]

[liver-2, subc, organ]
[become-2, tok, organ_state_change]
[become-2, changed_organ, liver-2]
[hepa-2, coext, hepa-1] suggested
[liver-2, coext, liver-1] suggested
[descr-2, tok, describe]
[descr-2, obj, become-2]
[descr-2, focus, hepa-2]
[liver-2, coext, liver-1] confirmed
[become-2, -cond, diso-1]
[hepa-2, coext, hepa-1] confirmed
[attr-2, tok, attribute]
[attr-2, obj, hepa-1]
[attr-2, prop, become-2]
No more patterns apply.

[pers-3, tok, person]
[pers-3, with, infect-3]
[infect-3, tok, infect]
[lose_ap-3, force, assert]
[lose_ap-3, tok, lose_appetite]
[lose_ap-3, subject, pers-3]

[pers-3, tok, diseased_person]
[diso-1, disorder_subject, pers-3] suggested
[descr-3, tok, describe]
[descr-3, obj, lose_ap-3]
[descr-3, focus, pers-3]
[diso-1, disorder_subject, pers-3] confirmed
[lose_ap-3, -cond, diso-1]
[attr-3, tok, attribute]
[attr-3, obj, hepa-1]
[attr-3, prop, lose_ap-3]
No more patterns apply.
** Current working base

* Roles

hepa-1
[hepa-1, tok, hepatitis]
[hepa-1, subc, diso-1]
liver-1
[liver-1, tok, liver]
[liver-1, subc, organ]
diso-1
[diso-1, tok, organ_disorder]
[diso-1, force, assert]
[diso-1, disorder_subject, pers-3]
[diso-1, affected_organ, liver-1]
hepa-2
[hepa-2, tok, hepatitis]
[hepa-2, force, fronted]
[hepa-2, coext, hepa-1]
liver-2
[liver-2, tok, liver]
[liver-2, subc, organ]
[liver-2, coext, liver-1]
pers-3
[pers-3, tok, diseased_person]

* Events

become-2
[become-2, -cond, diso-1]
[become-2, tok, organ_state_change]
[become-2, in, hepa-2]
[become-2, force, assert]
[become-2, changed_organ, liver-2]
[become-2, attrib, inflamed]

infect-3
[infect-3, tok, infect]

lose_ap-3
[lose_ap-3, -cond, diso-1]
[lose_ap-3, tok, lose_appetite]
[lose_ap-3, subject, pers-3]
[lose_ap-3, force, assert]

* Sayevents

tit-1
[tit-1, tok, title]
[tit-1, obj, hepatitis]
class-1
  [class-1, tok, classify]
  [class-1, obj, hepa-1]
  [class-1, class, diso-1]
descr-2
  [descr-2, tok, describe]
  [descr-2, obj, become-2]
  [descr-2, focus, hepa-2]
attr-2
  [attr-2, tok, attribute]
  [attr-2, prop, become-2]
  [attr-2, obj, hepa-1]
descr-3
  [descr-3, tok, describe]
  [descr-3, obj, lose.ap-3]
  [descr-3, focus, pers-3]
attr-3
  [attr-3, tok, attribute]
  [attr-3, prop, lose.ap-3]
  [attr-3, obj, hepa-1]
yes
| ?-