Automatic Discourse Structure Generation Using Rhetorical Structure Theory

LeThanh, H.

PhD 2004

Automatic Discourse Structure Generation Using Rhetorical Structure Theory

A thesis submitted to Middlesex University
In partial fulfilment of the requirements for the degree of
Doctor of Philosophy

Huong LeThanh

School of Computing Science

Middlesex University

September 2004

Abstract

This thesis addresses a difficult problem in text processing: creating a system to automatically derive rhetorical structures of text. Although the rhetorical structure has proven to be useful in many fields of text processing such as text summarisation and information extraction, systems that automatically generate rhetorical structures with high accuracy are difficult to find. This is because discourse is one of the biggest and yet least well defined areas in linguistics. An agreement amongst researchers on the best method for analysing the rhetorical structure of text has not been found.

This thesis focuses on investigating a method to generate the rhetorical structures of text. By exploiting different cohesive devices, it proposes a method to recognise rhetorical relations between spans by checking for the appearance of these devices. These factors include cue phrases, noun-phrase cues, verb-phrase cues, reference words, time references, substitution words, ellipses, and syntactic information. The discourse analyser is divided into two levels: sentence-level and text-level. The former uses syntactic information and cue phrases to segment sentences into elementary discourse units and to generate a rhetorical structure for each sentence. The latter derives rhetorical relations between large spans and then replaces each sentence by its corresponding rhetorical structure to produce the rhetorical structure of text. The rhetorical structure at the text-level is derived by selecting rhetorical relations to connect adjacent and non-overlapping spans to form a discourse structure that covers the entire text. Constraints of textual organisation and textual adjacency are effectively used in a beam search to reduce the search space in generating such rhetorical structures. Experiments carried out in this research received 89.4% F-score for the discourse segmentation, 52.4% F-score for the sentence-level discourse analyser and 38.1% F-score for the final output of the system. It shows that this approach provides good performance comparison with current research in discourse.

Acknowledgements

I wish to express my deep gratitude to my supervisors, Dr. Geetha Abeysinghe and Dr. Christian Huyck, whose guidance and encouragement made this work possible. Geetha Abeysinghe has been a wonderful supervisor. This work could not be finished without her support. She taught me how to write and patiently corrected my papers even when she had millions of other things waiting. I am most grateful to her. I wish to express my sincere appreciation to Christian Huyck for his helpful insights, suggestions, and feedback on many different aspects of this work, and even for all of his questions that led me to think about parts of this work in new ways.

I would like to thank my examiners, Prof. Donia Scott and Dr. Siri Bavan, for theirs detailed comments and suggestions (which have greatly improved this dissertation) on the manuscript and for a very interesting discussion during the viva.

A big thank you to all of the following, whose feedback was much appreciated: Dr. Wendy Wu, Dr. Dan Diaper, Prof. William Wong, Dr. Ellie Maclaren, Dr. Soodamani Ramalingam, and Prof. Manfred Stede. I am very grateful to them for their feedback on the draft of this thesis.

I am also indebted to the Natural Language Processing Research Group of the University of Sheffield who provides a software package named a General Architecture for Text Engineering (GATE), which was used as the infrastructure of the system implemented in this thesis. I thank for their technical support in using GATE.

I am grateful to the School of Computing Science, Middlesex University, for financially supporting me during my study in the United Kingdom.

Last but not the least, I would like to thank my friends: Nan, Viviane, KehKok, Leonard, Amala, Hanh, Duc, Duong, Yen, Hai, Hieu, Anh, Anita, Jenny, and Kamal. The friendship I have with them is what kept me going on during the tough times.

This thesis is dedicated to my parents and brother, whose encouragement and unwavering support over the years spent on this work has been a source of inspiration.

Contents

List of Figures

List of Tables

1 Introduction

The current boom in information technology has produced an enormous amount of information. From the abundance of information available, getting the information that we need is not an easy task. Using the vast amount of on-line text has become unmanageable without tools for retrieving and filtering. There are many World Wide Web search engines, which can locate possibly relevant texts, but they can provide hundreds of results, from which only a few may be really useful or relevant. Obviously, we do not have time to read every document presented by such search engines to find the most relevant documents. Sometimes the title of these documents may not represent their contents very well. If we skip some of the documents by looking at the title or the search engine index by which they are presented, we might miss valuable information. This problem can be overcome by representing documents by their summarisations. Therefore, effective methods of automatic text summarisation are necessary today.

Generating multi-document summaries also has a lot of demands. For example, a doctor needs information about a specific disease. He then extracts information involving that disease from medical digital libraries. What he needs is a document that summarises all the information from this search. This document needs to be well-organised and coherent. This is the area of information extraction and multi-document summarisation.

Most existing text summarisation systems are based on text extraction (Rau et al., 1994; Mitra et al., 1997). These systems identify and extract key sentences or paragraphs from an article using statistical techniques. The most important parts of the text are then copied and pasted into the summary. This approach often gives us incoherent texts since the summary may consist of sentences that do not naturally follow one another. A new trend in text summarisation solves the incoherence problem by using discourse strategies (Jones, 1993; Rino and Scott, 1994; Marcu, 2000; Polanyi et al., 2004), which analyse the coherence of a text by a rhetorical structure[1] that describes rhetorical relations between different parts of

[1] Terminologies "*discourse*" and "*rhetorical*" are used interchangeably in this thesis.

a text.[2] The salient units from these rhetorical relations, which are called nuclei by Mann and Thompson (1988), are selected and organised by using heuristics to generate a summary. Experiments from different summarisation systems have shown that the approach based on discourse strategies achieves better results than systems based on other strategies.

Discourse strategies not only improve the performance of text summarisation systems, but also support other fields of text processing such as information retrieval (Miike et al, 1994; Morato, 2003), text translation (Marcu et al., 2000), and text understanding (Rutledge et al., 2000; Torrance and Bouayad-Agha, 2001). Let us have a brief look at the impact of discourse strategies on one of the above text processing applications - information retrieval. In a normal information retrieval system, documents (or parts of a document) are selected by statistical techniques based on the relevant words between documents and a search query. Discourse analysis can improve the performance of these systems by concentrating on salient parts (the nuclei) of the search query and documents, since the nuclei are the most important parts in realising the writer's communicative goals (see Section 2.2.1).

Although research in text processing has proved that discourse analysis is an efficient approach in constructing automatic text processing systems, systems using discourse strategies are still rare. This is because discourse is complex and vague. Discourse analysis is difficult for linguistic analysts and it is much more difficult to do automatically by a computational system. Literature shows that a considerable amount of work has been carried out in this area (Grosz and Sydner, 1986; Mann and Thompson, 1988; Hovy, 1993; Marcu, 2000; Forbes et al., 2003). However, most research has concentrated on specific discourse phenomena (Schiffrin, 1987; Hirschberg and Litman, 1993; Kehler, 1994; Forbes and Webber, 2002). Only a few algorithms for implementing rhetorical structures have been proposed so far (Marcu, 2000; Corston, 1998; Forbes et al., 2003; Polanyi et al., 2004). Realising the lack of discourse systems and the great demand for text processing applications, we have carried out research in discourse analysis,

[2] See Section 2.2.1 for a detailed description of rhetorical structures and rhetorical relations.

aiming to construct a system that automatically derives rhetorical structures for text. The next section points out the main tasks of a discourse analysing system and identifies the research targets of this thesis by briefly reviewing the remaining problems of existing research in discourse.

1.1 Problem Statement

This thesis aims to construct a Discourse Analysing System (DAS) that automatically generates rhetorical structures of text. The main tasks of a normal discourse system are:

1. **Segment text into discourse units.** The discourse units should have independent functional integrity, which are essentially clauses.
2. **Posit rhetorical relations between text spans**[3]. After the text is segmented into elementary discourse units, the next task of a discourse system is to recognise all possible rhetorical relations between these units and between larger text spans.
3. **Generate rhetorical structures that best describe the text.** The hypothetical rhetorical relations created in the previous task (task 2) are selected and applied to construct a rhetorical structure that represents the text. A text may have more than one rhetorical structure that can describe it.

Although many attempts have been carried out to build discourse systems, the performances of existing discourse systems are still low. For this reason, this thesis concentrates on improving both speed and quality of a discourse analyser. Inspired by Marcu (2000) and Corston (1998), the thesis focuses on the following issues:

Improving the correctness of discourse segment boundaries

Discourse segmentation is the first step in discourse analysis. The output of the discourse segmentation process is used to generate rhetorical structures of text. Therefore, a high performance discourse segmenter is critical for a discourse

[3] Terminologies "*text span*", "*span*", and "*discourse unit*" are used interchangeably in this thesis.

system, which derives discourse trees for the entire text. Many efforts have been put in this task (Passonneau, 1997; Marcu, 1999; Forbes and Miltsakaki, 2002; Heusinger, 2001). However, the performances of existing discourse segmenters are still not good enough to assist the task of generating discourse trees. For this reason, exploring aspects that improve the correctness of discourse segments is one of the main targets of this thesis. Syntactic information and cue phrases are used to tackle this problem. This method is discussed in Chapter 3.

Exploring new factors to recognise rhetorical relations

Most research in discourse analysis is based on cue phrases to recognise rhetorical relations between text spans (Schiffrin, 1987; Marcu, 2000; Forbes and Webber, 2002). However, from the earliest stages of discourse theoretical development, it has been clear that in most texts, a large fraction of relations were not signalled by any word, phrase, or syntactic configuration. Different studies use different recognition factors to deal with such cases. Nevertheless, most of these studies are empirical and only concentrate on specific discourse situations (Kehler and Shieber, 1997; Poesio and Di Eugenio, 2001). In this thesis, different recognition factors are explored and integrated into the system. In addition to exploiting new properties of the factors that have been investigated in other research (syntactic information, cue phrases, time references, reiterative devices, reference words, substitution words, and ellipses), we propose new recognition factors (noun-phrase cues and verb-phrase cues). These factors are discussed in Chapter 4.

Improving the efficiency and reducing the computational complexity of the discourse analyser

Unlike syntactic parsers which have a long history, discourse analysing has only received attention since 1980s. As such only a few algorithms for generating rhetorical structures have been proposed (Marcu, 2000; Corston, 1998; Forbes et al., 2003); and fewer algorithms have been implemented. As discussed in Section 2.1.1, although the discourse systems created by Marcu (2000) and Corston

(1998) are advanced when compared with other discourse systems, the computational complexity of these systems is still high; the system's performances still need to be improved. For this reason, this thesis concentrates on reducing the computational complexity of the discourse analyser and improving the system's performance. The solution to these problems is presented in Chapter 5.

1.2 Outline of the Thesis

We began Chapter 1 by introducing the motivation for carrying out this research. We then clarified the problems that this thesis attempts to solve. A summary of the rest of this thesis is given below.

Chapter 2: Literature Review. We introduce existing approaches to discourse analysis, aiming to understand the state of the art of the field and to determine an approach for this thesis. Since the Rhetorical Structure Theory (RST) was chosen as the framework for this research, we present an overview of the RST to inform the reader of the basic concepts of this theory. After that, the unresolved issues of the RST are highlighted. Finally, we introduce the corpus that is used in this research in constructing a discourse system and in doing experiments.

Chapter 3: Discourse Segmentation. A method that uses sentential syntactic structures and cue phrases to segment text into elementary discourse units is proposed. A sentence is first segmented into clauses by using its syntactic structure. After that, DAS searches for strong cue phrases from these clauses and continually splits the clauses that contain a strong cue phrase. Finally, a post segmenting process is used to refine segment boundaries.

Chapter 4: Positing Rhetorical Relations between Elementary Discourse Units. We introduce the relation set that is used in this thesis to posit rhetorical relations. Several factors that contribute to the process of recognising relations are analysed. They are syntactic information, cue phrases, noun-phrase cues, verb-phrase cues, time references, reiterative devices, reference words, substitution words, and ellipses, among which noun-phrase cues and verb-phrase cues are new factors proposed in this thesis. We present a method of positing rhetorical relations based on these factors. Different scores are assigned to these factors, so that DAS can determine which relation is stronger than the others. Conditions for recognising a *List*, *Elaboration*, and *Circumstance* relation are then introduced as representative samples of this method. The complete set of conditions for recognising relations is in Appendix 6.

Chapter 5: Constructing Rhetorical Structures. This chapter introduces a method for deriving rhetorical structures of text at two levels: sentence-level (intra-sentence) and text-level (inter-sentences), concentrating on improving the system's performance and reducing the computational complexity. At the sentence-level, information about sentential syntactic structure permits DAS to generate one and only one rhetorical structure for each sentence. At the text-level, constraints about textual organisation and textual adjacency are integrated into a beam search to reduce the size of the space in searching for the best discourse trees.

Chapter 6: Evaluation. Since a standard benchmark for evaluating a discourse system does not exist, this chapter proposes a method to evaluate the discourse system based on different levels of processing. Experiments and their results are reported, discussed and compared with the most recent and best performance discourse systems. The experiments show that syntactic information and cue phrases are efficient in constructing discourse structures at the sentence-level, especially in discourse segmentation. The current version of DAS provides promising results compared to discourse trees generated by humans.

Chapter 7: Conclusions. This chapter summarises the thesis and outlines its contributions. We list some of the open issues that have not been addressed in this work, and we suggest directions for future research.

The contributions of this thesis are on several points. A new segmentation method, a new method for deriving sentential discourse trees, and new factors to signal discourse relations are proposed. We optimise the procedure to posit relations that is first proposed by Corston (1998). We extend Marcu's (2000) proposition that is used to posit relations between large spans to make the most of cue phrases. We improve the algorithm to construct discourse trees from hypothetical relations between text spans, aiming to reduce the computational complexity of the algorithm and to improve the system's performance.

The architecture of DAS is described in Appendix 1. The extended version of algorithms implemented in this thesis is presented in Appendix 2. Appendices 3 and 4 list the cue phrases, NP-cues and VP-cues that are used in this research. The syntactic chains that are used in DAS to segment a sentence into elementary discourse units are shown in Appendix 5. Finally, a definition of rhetorical relations and conditions to recognise rhetorical relations are introduced in Appendix 6.

2 Literature Review

In this chapter, we first discuss existing work on discourse analysis. Then, the discourse theory that this research was based on – the Rhetorical Structure Theory – is introduced. A brief description of the data used in experiments of this research is given at the end of this chapter.

2.1 Existing Work on Discourse Analysis

We review research on discourse analysis focussing on two aspects: one relates to generating an entire discourse structure of text, the other relates to solving specific tasks of discourse analysis (e.g., discourse segmentation). Section 2.1.1 introduces our survey which is based on the first aspect. The purpose of this survey is to identify different theories in discourse analysis and to select a discourse theory to be used as the framework of our research. Section 2.1.2 carries out another survey based on the second aspect mentioned above, aiming to investigate all appoaches that have been used to solve each task.

2.1.1 Existing Work on Generating Discourse Structures

In this section, the main theories that inspire most research in discourse (Grosz and Sidner, 1986; Mann and Thompson, 1988) are described. Thereafter we introduce some of the most recent studies on generating a discourse system that follow these theories.

2.1.1.1 Grosz and Sidner (1986)

One of the main discourse theories is proposed by Grosz and Sidner (1986). In this approach, the intention of the author in creating a text is crucial in leading the rhetorical structure of that text. According to Grosz and Sidner, a rhetorical structure is composed of three components: a linguistic structure, an intentional structure, and an attentional state. The linguistic structure consists of Discourse Segments (DSs) and an embedding relationship that can hold between them. The intentional structure is achieved by recognising the particular purpose of the author in producing the text (called Discourse Purpose or DP), and the way each DS contributes to the overall discourse purpose (called Discourse Segment

Purpose or DSP). Relations between intentions indicate whether one intention contributes to the satisfaction of another (dominance) or whether one intention must be satisfied before another (satisfaction-precedence).

The attentional state of a rhetorical structure is modelled by a set of focus spaces[4] and a set of transition rules[5]. The transition rules push a new focus space onto the focus stack when a text segment is open and pop it out when the segment is closed. Grosz and Sidner have proposed a method to recognise focus spaces based on cue phrases and anaphora resolution. They argue that the primary role of the stack of the focus space is to determine the DSPs that have a relationship with the DSP of the current segment. In other words, the focus space reflects the intentional structure.

The discourse theory proposed by Grosz and Sidner leaves many issues unresolved. It would require much more intensive work in order to transform it from theory into a real system, capable of automatically generating rhetorical structures.

2.1.1.2 Mann and Thompson (1988)

Another discourse theory, which exists in parallel with the one proposed by Grosz and Sidner (1986), is the Rhetorical Structure Theory proposed by Mann and Thompson (1988). Mann and Thompson have proposed and defined a set of 23 rhetorical relations for deriving rhetorical structures and a definition for each of these relations. They suggest that this relation set is not a closed list, but could be extended and modified for the purposes of particular genres and cultural styles. In order to derive the rhetorical structure of texts, one must first divide text into clauses and clause-like units, and then recognise relations between these units using the set of 23 rhetorical relations mentioned above. The reader is referred to Section 2.2 for a detailed description of the RST. Mann and Thompson admit the existence of multiple analyses in the RST, which causes difficulties in deriving and evaluating discourse systems. The main reasons for multiple analyses are:

---

[4] A focus space consists of representations of entities (i.e., objects, properties, and relations).

[5] A transition rule is the rule that specifies conditions to change an attentional state.

1. The *difference of boundary judgements* between analysts
2. The *text structure ambiguity*
3. The phenomenon of *simultaneous analyses*: Several analyses are acceptable for a specific text.
4. The *differences between analysts*: One text is analysed in different ways by different analysts.
5. The *analytical errors*: This situation happens with inexperienced analysts.

The theory introduced by Grosz and Sidner (1986) and the RST agree that discourse is structured in a hierarchy of non-overlapping constituents. However, the internal structure of a segment in the theory of Grosz and Sidner (1986) is different to that of a text span in the RST. The former consists of **an utterance of the discourse segment purpose** and **any number of embedded segments**. The latter consists of **a nucleus**, which expresses the content that the writer wants to convey, and **a satellite**, which provides additional information to the nucleus.

2.1.1.3 Poesio and Di Eugenio (2001)

Poesio and Di Eugenio (2001) have evaluated the work of Grosz and Sidner (1986) by carrying out an empirical study of the relation between rhetorical structure and anaphoric accessibility. The Sherlock corpus used in their experiment is a collection of seventeen tutorial dialogues annotated according to Relational Discourse Analysis (RDA) (Moore and Pollack, 1992). RDA is a theory of rhetorical structure that attempts to merge the RST with Grosz and Sidner's (1986) theory. In the RDA, the utterance of the discourse segment purpose is a core, whereas its embedded segments are contributors. A core can have any number of contributors, each of which plays a role in serving the purpose expressed by the core.

One goal in Poesio and Di Eugenio's research is to find out when focus spaces should be opened and closed. They assume that contributors stay on the stack until the RDA-segment is completed. They claim that only the simplest treatments of contributors and cores probably are consistent with this assumption, the most complex ones probably are not. In order to deal with this, the attentional state has to be seen as a list instead of a stack as in Grosz and Sidner's (1986) theory.

Poesio and Di Eugenio (2001) leave two open issues. The first issue is how to supervise discourse entities on the stack: the more entities are in the stack, the more likely that an antecedent will be found. However, this may result in losing the crucial property of the attentional state and increasing search ambiguity. As such an anaphoric resolution mechanism is proposed to deal with this issue, which is something that can be explored in future work. The second issue is the multiple analyses of discourse (see Section 2.1.1.2) and because of this, they do not expect everybody to agree on the particular analyses proposed in their paper.

2.1.1.4 Kurohashi and Nagao (1994)

Kurohashi and Nagao (1994) propose a method of detecting rhetorical structures automatically using surface information in sentences: cue phrases, chains of identical and similar words, and similarity between two sentences. The rhetorical structures derived by their system are similar to rhetorical structures in the RST. However, the elementary discourse units in Kurohashi and Nagao's system are sentences instead of clauses as in the RST. Their system starts with an empty discourse tree. Each step a new sentence is connected to the node on the right most edge in the discourse tree (Figure 2.1).

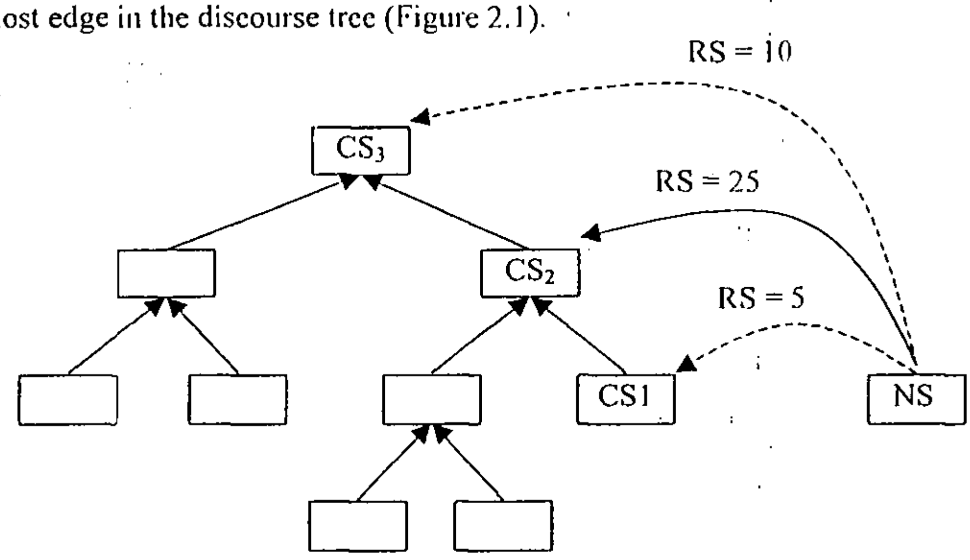


Figure 2.1: Connecting a New Sentence into a Discourse Tree (Kurohashi and Nagao, 1994)

CS - A Possible Connected Sentence on the Right Edge in the DS Tree

DS – Discourse Structure RS – Ranking Score NS – New Sentence

The node which is chosen to be connected with is the one with the highest ranking score. This score is computed by the three types of surface information mentioned above (i.e., cue phrases, chains of identical and similar words, and similarity between two spans).

2.1.1.5 Marcu (2000), Marcu (1999)

Marcu (2000)[6] presents a rhetorical parsing model that uses manually derived rules to construct rhetorical structures. This approach uses cue phrases to segment a text into elementary discourse units. To posit hypothetical rhetorical relations, Marcu uses a discourse-marker-based algorithm and a word co-occurrence-based algorithm. The co-occurrence-based algorithm is used to detect whether two sentences or two paragraphs "*talk about*" the same thing or not. Since this algorithm is based on the co-occurrence of words, it cannot deal with the case when the two sentences or paragraphs use synonyms or similar expressions to refer to one meaning. Marcu (2000) proposed a theory, which states that, "*If a rhetorical relation R holds between two text spans of the tree structure of a text that relation also holds between the most important units of the constituent spans*". From this point of view, Marcu (2000) analyses relations between text spans by considering only recognition factors from their nuclei. Although Marcu's algorithm for constructing RST representations is considerably advanced compared to other methods, several problems have not been considered. Since Marcu's system is heavily dependent on cue phrases, it has problems when cue phrases are not present in the text. In addition, as Marcu's system produces all RST trees compatible with the relations that might hold between pairs of RST terminal nodes, his system suffers from combinatorial explosion when the number of relations increases exponentially (see Section 2.1.2.3).

Marcu (1999) introduces a decision-based approach to rhetorical parsing. This approach relies on a corpus of manually built discourse trees and the adoption of a shift-reduce parsing model to automatically derive rules. By evaluating the experimental results, Marcu (1999) claims that his system is sufficient for determining the hierarchical structure of a text and the nuclearity status of

---

[6] The research in Marcu (2000) is first established in Marcu's thesis (Marcu, 1997).

discourse segments. However, it is not very good at determining correctly the elementary discourse units and the rhetorical relations that hold between discourse segments.

2.1.1.6 Corston (1998)

Corston (1998) follows Marcu's (2000) research and creates a discourse processing component named RASTA. He proposes a set of thirteen rhetorical relations and builds RST trees for articles from the Encarta corpus. Corston reports a list of conditions that must be met for a text segment to be considered as an elementary discourse unit. These conditions are based on the syntactic structure of sentences. This syntactic-based approach is more reliable than the cue-phrase-based approach proposed by Marcu (2000) since most elementary discourse units are clauses. Furthermore, the sentential syntactic structure is always present, whereas cue phrases can be absent in a text. Unfortunately, it is not clear how the syntactic conditions reported by Corston (1998) are implemented in RASTA.

Corston combines cue phrases with anaphora and referential continuity to recognise rhetorical relations. A relation is posited between two discourse units if they satisfy several criteria that characterise this relation. If two or more relations are suggested by the system, heuristic scores are used to choose the best one. These scores also help in filtering out all ill-formed trees and in reducing the algorithm's complexity.

Corston uses the formal model of discourse that was presented by Marcu (2000) and improves Marcu's (2000) discourse parsing algorithm in order to reduce its search space. Although considerable improvement has been made when compared with Marcu's (2000) system, the search space of RASTA still contains redundancy, which increases the computational complexity of the discourse analyser. In addition, Corston does not consider the case of multiple discourse connectives.

2.1.1.7 Forbes et al. (2003)

Forbes et al. (2003) have developed an approach to discourse analysis by applying Lexicalised Tree Adjoining Grammar to Discourse (D-LTAG). In this approach, discourse connectives are used as anchors to connect discourse sub-trees into bigger discourse trees. The typical grammar rule for this system is:

Tree → SubTree1 + [anchor] + SubTree2

The anchor that connects two discourse sub-trees can be an overt connective or a lexically unrealised anchor such as a comma or a full stop. For example, Example (2.1) presents two different situations (a) and (b). In Example (2.1.a), two clauses "*you shouldn't trust John*" and "*he never returns what he borrows*" are connected by the connective "*because*". The discourse tree of Example (2.1.a) is illustrated in Figure 2.2.a. It consists of two sub-trees T1 and T2, which correspond to the two clauses mentioned above, and the anchor "*because*". In Example (2.1.b), two sentences "*you shouldn't trust John*" and "*he never returns what he borrows*" are separated by a full stop. Figure 2.2.b shows how this situation is presented by a D-LTAG tree. The two sub-trees T3 and T4 in Figure 2.2.b correspond to the two sentences in Example (2.1.b). The anchor that connects these sub-trees is the full stop.

(2.1) a. You shouldn't trust John *because* he never returns what he borrows.

b. You shouldn't trust John. He never returns what he borrows.

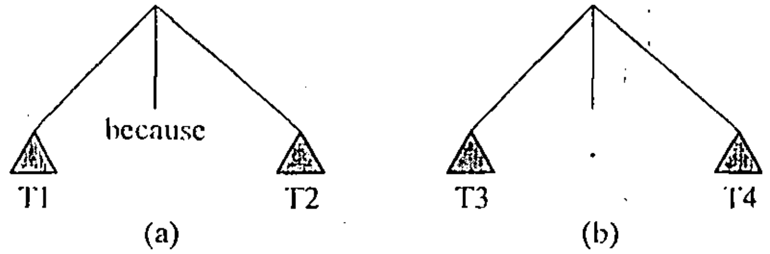


Figure 2.2. Derivation of Example (2.1)

Larger D-LTAG trees are achieved with two operations, *adjunction* and *substitution*. Adjunction adds an auxiliary tree with at least one tree node, whereas substitution replaces each tree node with a corresponding D-LTAG sub-tree.

Forbes et al. (2003) show that the D-LTAG grammar simplifies rhetorical structures, while allowing the realisation of a full range of rhetorical relations. Despite its potential ability in discourse analysis, many problems in the D-LTAG still need to be resolved. Although anaphoric and presuppositional properties of lexical items are proposed by Forbes et al. (2003) to extract certain aspects of discourse meaning, these features have not been investigated. Also, Forbes et al.'s (2003) system cannot be successful without discourse connectives. Tasks such as discourse segmenting, determining connections between discourse units, and recognising relation names in the absence of overt connectives are not mentioned in their research. Since an implementation of the D-LTAG and its experimental results have not been reported, it is impossible to compare the performance of this approach with other research in this field.

2.1.2 Existing Work on Specific Tasks of Discourse Analysis

In this section, we perform a critical survey of the research that deals with specific tasks of discourse analysis, including discourse segmentation (Section 2.1.2.1), relation recognition (Section 2.1.2.2), discourse structure generation (Section 2.1.2.3), and system evaluation (Section 2.1.2.4). From that point of view, we propose our solution for each task, which will be introduced later in Chapters 3, 4, 5, and 6.

2.1.2.1 Discourse Segmentation

Discourse has been automatically segmented using disparate phenomena: cue phrases (Grosz and Sydner, 1986; Marcu, 2000; Passonneau and Litman, 1997), syntactic information (Batliner et al., 1996; Corston, 1998), and semantic information (Polanyi et al., 2004). However, the criteria to indicate the exact discourse segment boundaries are still not certain.

The shallow parser introduced by Marcu (2000) splits text into elementary discourse units by mapping cue phrases and punctuation marks. Marcu does not provide any solution to deal with the case when cue phrases are not present in text, and his system fails in this situation. For example, Marcu's system cannot detect the two discourse units presented in Example (2.2) below:

(2.2) [As part of the upscale push, Kidder is putting brokers through a 20-week training course,] [*turning them into "investment counselors" with knowledge of corporate finance.*][7]

Passonneau and Litman (1997) propose two sets of algorithms for linear segmentation based on linguistic features of discourse. The first set is based on referential pronoun phrases, cue phrases, and pauses. The second set uses error analysis and a machine learning method. The performance of their segmentation modules are quite advanced in comparison with other research at that time. However, the machine learning method requires training, which heavily depends on manually annotated corpora. A small training corpus may lead to lack of generality and a large discourse corpus for such a training purpose is difficult to find.[8]

One well-organised system using the syntactic approach is proposed by Corston (1998). He defines a list of grammatical conditions that a text segment must satisfy in order to be considered as an elementary discourse unit. Unfortunately, the segmentation algorithm used by him is not fully explained in his thesis. Corston's system does not consider the cases when strong cue phrases make noun phrases become elementary discourse units. The two elementary discourse units in Example (2.3) are recognised as only one by Corston's system.

(2.3) [*According to a Kidder World story about Mr. Megargel,*] [all the firm has to do is "position ourselves more in the deal flow."]

Polanyi et al. (2004) propose a new approach based on discourse semantics. Rather than posit which syntactic objects function as discourse segments, they start by establishing the semantic basis for functioning as a segment and then identify which syntactic constructions carry the semantic information needed for discourse segment status. The segments that have the potential to independently establish an anchor point for future continuation are identified as Basic Discourse Units (BDUs). After that, they draw a further distinction between BDUs as a class

[7] The square brackets indicate the boundaries of discourse units.

[8] The biggest discourse corpus which currently exists to our knowledge is the RST Discourse Treebank (RST-DT, 2002), with 385 Wall Street Journal articles.

of syntactic structures with the potential to establish anchor points and the BDUs in a given sentence, which function as indexical anchor points in a specific discourse. Despite being cumbersome, this approach has a potential to provide a discourse segmenter with high accuracy. Unfortunately, no experimental result of discourse segmentation was reported in this research. It is thus difficult to compare this approach with the others.

2.1.2.2 Recognising Discourse Relations

Recognising rhetorical relations is the most crucial and difficult task in deriving the rhetorical structure of text. Although much research has been carried out in this area, there are many situations in which no simple rule can be established to recognise relations.

Cue phrases have been the centre of research in this area since using cue phrases is the most efficient method of recognising relations (Schiffrin, 1987; Marcu, 2000; Forbes and Webber, 2002). Several corpus-based works have attempted to build a set of potential cue phrases (Grosz and Sidner, 1986; Hirschberg and Litman, 1993; Knott and Dale, 1994; Marcu, 2000). Some studies pay attention to the diversity of meanings associated with some specific cue phrases, based on the context or the conversational moves (Halliday and Hassan, 1976; Schiffrin, 1987; Korbayov and Webber, 2000).

Studies on the disambiguation between the discourse sense and the sentential sense of a cue phrase include Hirschberg and Litman (1993), Litman (1996), Siegel and McKeown (1994), and Marcu (2000) (see Section 4.2.1 for the concept of "*disambiguation of a cue phrase*"). Siegel and McKeown (1994) develop an approach that is based on decision trees to test adjacent punctuation marks, cue phrases, and near-by words, and to discriminate between cue phrases. They use a genetic algorithm to automatically determine which words or punctuations near a cue phrase are important for disambiguation. Litman (1996) uses machine learning techniques to classify the discourse sense and sentential sense of cue phrases, using features of cue phrases in text and speech. These approaches have shown that the sense of a cue phrase can be determined by the orthographic environment of the cue phrase.

As Redeker (1990) stated, only 50% of clauses contain cue phrases. Therefore, although cue phrases are the easiest means of signalling rhetorical relations, other recognition factors still need to be investigated. Research has shown that lexical cohesion can be used to identify the movement of topics (Grosz and Sidner, 1986); sometimes it can even determine rhetorical relations between small text spans (Harabagiu and Maiorano, 1999). Several forms of lexical cohesion have been exploited, including anaphoric references (Poesio and Di Eugenio, 2001; Webber et al., 2003), and VP-ellipsis (Kehler, 1994; Kehler and Shieber, 1997). (See Sections 4.2.4 and 4.2.7 for the concept of "*anaphoric references*" and "*VP-ellipsis*" respectively.) It is more difficult to recognise relations using lexical cohesion than using cue phrases because of two reasons. First, cohesive devices cannot be simply detected by pattern matching like cue phrases. It requires a more complicated mechanism (see Chapter 4). Second, the lexical cohesion cannot directly signal a rhetorical relation most of the time. Instead, it often indicates a semantic link among text spans. For this reason, in order to recognise rhetorical relations, research often uses the solution of combining different linguistic factors (e.g., Corston, 1998).

2.1.2.3 Generating Discourse Structures

The aim of this task is to generate discourse structures of text, given all possible relations that hold between text spans. Since we concentrate on minimising the search space when producing well-formed discourse structures, we will look at different approaches related to this problem both in a direct way (i.e., generating discourse structures of text such as Marcu (2000)) and in an indirect way (i.e., generating a text using discourse structures such as Hovy (1993)).

Hovy (1993) describes methods of automated planning and generating multi-sentential texts using rhetorical structure. He proposes a method based on predefined structures or schemas. Hovy's method is based on the idea that text structure reflects the intention of the writer. With the predefined knowledge about the text (the structure of a scientific paper, a dialogue with a communicative goal, etc.), a schema is developed for that text, and then the content of the text is mapped to this schema to construct a rhetorical structure. This approach produces acceptable results in restricted domains where the library of schemas is provided. The schema is especially efficient in reducing the combination of spans in a text that consists of two or more paragraphs. However, it is difficult to extend it to freer text types, as this approach relies on a library of schemas.

In contrast, Marcu's (2000) system can apply to unrestricted text, but is faced with combinatorial explosion. Marcu's system generates all possible combinations of nodes according to the hypothesized relations between spans, and then filters out ill-formed trees based on the heuristics about tree quality. These heuristics are constraints about the order of the two facts involved in a rhetorical relation and the adjacency of discourse segments. The disadvantage of Marcu's approach is that it produces great numbers of ill-formed trees during its process, which is its essential redundancy in computation. As the number of relations increases, the number of possible discourse trees increases exponentially. The construction of these ill-formed trees can be eliminated before hand if the above heuristics are applied during the process of generating discourse trees instead of applying them after generation. Another problem of Marcu's (2000) system involves the evaluation of discourse trees to select the most preferred one. According to Marcu (2000), the right-branching structures are preferred because they reflect basic organisational properties of a text. This observation, as Corston (1998) pointed out, is not valid for all genres of text. The right-branching structure is also not the chosen one in Example (5.6) (shown later in Chapter 5), which is taken from the RST-DT corpus (RST-DT, 2002).

The tree-constructing algorithm in RASTA, proposed by Corston (1998), solves the combinatorial problem in Marcu (2000) by using a recursive, backtracking algorithm that produces only well-formed trees. If RASTA finds a combination of two spans leading to an ill-formed tree, it will backtrack and go to another direction, thus reducing the search space. By applying the higher score hypotheses before the lower ones, RASTA tends to produce the most reliable RST trees first. Although a lot of improvement has been made over Marcu's (2000) method, RASTA search space is still not optimal because of two reasons. First, RASTA does not consider the adjacency constraint when it constructs discourse trees. Second, RASTA does not trace the already visited routes that generate ill-formed trees. As a result, RASTA continues to check the same combinations of discourse units again and again. These problems are further discussed in Section 5.3.1.

2.1.2.4 Evaluation Methods

Evaluating a discourse system is difficult. Most research in discourse analysis concentrates on proposing discourse analysing methods (Poesio and Di Eugenio, 2001; Forbes et al., 2003). There are only a few efforts to install a real discourse system (Kurohashi and Nagao, 1994; Marcu, 2000; Corston, 1998), and fewer efforts to evaluate the system's performance (Marcu, 2000; Soricut and Marcu, 2003). One reason is that discourse is too complex and ill defined to generate rules that can automatically derive rhetorical relations, and even if a system that can generate rhetorical structures is available, it is difficult to choose which discourse trees should be used to compare because of the multiple analysis property of discourse. To our knowledge, there is no standard benchmark to evaluate a discourse system. Each researcher has evaluated their discourse system using different data. Also, they do not compare the system's performance with others.

Corston (1998) assesses discourse trees by scores of the trees, which are calculated by heuristic scores of recognition factors that contribute to the relation. No experimental result has been reported by Corston (1998). Marcu (2000) manually evaluates discourse trees of five texts by comparing rhetorical relations of the discourse trees built by his system and by human annotators. However, the method of manually evaluating rhetorical structures would be extremely costly and inefficient for a larger number of texts. Soricut and Marcu (2003) evaluate their sentence-level discourse parser at different tasks: discourse segmentation and discourse parsing. These tasks are evaluated in both cases, when correct data or automatically generated data are used as the input. The evaluating approach of Soricut and Marcu is better than those reported in Marcu (2000) and Corston (1998) since it can assess the real performance of each module, as well as the effect of the previous module on the next one.

2.1.3 Summary

Two main discourse theories have been reported in this section. The first discourse theory proposed by Grosz and Sidner creates a rhetorical structure using three components: the linguistic structure, the intentional structure, and the attentional state. An example of research following this theory is Poesio and Di Eugenio (2001). The second discourse theory, which is proposed by Mann and Thompson (1988), is called the Rhetorical Structure Theory. A rhetorical structure that follows the RST framework is derived by rhetorical relations between adjacent, non-overlapping text spans. Corston (1998), Mellish et al. (1998), and Marcu (2000) follow this theory. Several studies are close to the representation of the RST, but do not follow exactly the RST framework (Kurohashi and Nagao, 1994; Cristea, 2000; Forbes et al., 2003).

The discourse segmentation has been done using various means: cue phrases (Grosz and Sydner, 1986; Passonneau and Litman, 1997), syntactic information (Corston, 1998), and semantic information (Polanyi et al., 2004).

Studies on recognising relations between discourse constituents can be divided into three main trends, depending on the features being used. These trends are cue-phrase-based (Hirschberg and Litman, 1993; Knott and Dale, 1994; Kurohashi and Nagao, 1994; Marcu, 2000; Forbes et al., 2003), lexical-cohesion-based (Poesio and Di Eugenio, 2001; Webber et al., 2003; Kehler and Shieber, 1997), and a combination of cue phrases and lexical information (Corston, 1998). Discourse relations are generated using two basic methods: rule-based (Kurohashi and Nagao, 1994; Corston, 1998; Forbes et al., 2003) and machine-learning-based (Marcu, 1999; Marcu and Echihabi, 2002).

In respect of constructing discourse structures, we have introduced two approaches: schema-based (Hovy, 1993) and non-schema-based (Marcu, 2000; Corston, 1998). The former relies on predefined knowledge about the text. The latter can be applied to unrestricted text, but is faced with combinatorial explosion.

We follow the approach proposed by Marcu (2000) and extended by Corston (1998), using the RST as a framework for the discourse system. Before introducing our approach to the problem of discourse structure generation, let us have an overview of the Rhetorical Structure Theory, informing the reader of the basic definition of a rhetorical structure. This information is presented in Section 2.2. Section 2.2 also analyses the open issues in the Rhetorical Structure Theory that research on this theory have been facing.

2.2 Rhetorical Structure Theory

2.2.1 Overview

Rhetorical Structure Theory is a method of representing the coherence of texts, in order to understand discourse structure. It models the rhetorical structure of a text by a hierarchical tree that labels relations between text spans (typically clauses or larger linguistic units). This hierarchical tree diagram is called "rhetorical tree", "discourse tree", or "RST tree". The leaves of a discourse tree correspond to clauses or clause-like units with independent functional integrity. Meanwhile, the internal nodes of a discourse tree correspond to spans that are larger than clauses.

The children of an RST tree correspond to adjacent, non-overlapping spans, which are joined by a *rhetorical relation*. This relation can be asymmetric or symmetric. An asymmetric relation, also called a *nuclear-satellite* relation, involves two spans, one of which is more essential to the writer's goals than another. The more important span in a rhetorical relation is labelled a *nucleus* (N); whereas the less important one is labelled a *satellite* (S). The nucleus of a rhetorical relation is comprehensive and independent of the satellite, but not vice-versa. An asymmetric relation is shown in Example (2.4) below:

(2.4) [Its 1,400-member brokerage operation reported an estimated $5 million loss last year,][*although Kidder expects it to turn a profit this year.*]

The deletion of the second clause in Example (2.4) does not significantly affect the meaning of the whole text. The first clause is still understandable without the second clause. Meanwhile, the second clause is not understandable without the first clause. For this reason, the first clause is more important than the second clause in respect to the writer's purpose. Therefore, the first clause is the nucleus, and the second clause is the satellite of an asymmetric relation between them.

A symmetric relation, also called a *multi-nuclear* relation, involves two or more spans,[9] each of which is equally important in respect to the writer's intention in producing texts, such as the two clauses "*Three seats currently are vacant*" and "*and three others are likely to be filled within a few years*" in Example (2.5) below. Each node in a symmetric relation is a *nucleus*.

(2.5) [Three seats currently are vacant][and three others are likely to be filled within a few years.].

A rhetorical relation is recognised by constraints on the nucleus, on the satellite, and on the combination of the nucleus and the satellite (Mann and Thompson, 1988). Figure 2.3 illustrates this recognition process by a representative sample relation – the *Purpose* relation.

Constraints on N:	presents an activity.
Constraints on S:	presents the situation that is unrealised.
Constraints on the N+S combination:	S presents a situation to be realised through the activity in N.
The effect:	Reader recognises that the activity in N is initiated in order to realise S.

Figure 2.3. Definition of the *Purpose* Relation

Example of the *Purpose* relation:

(2.6) [*To answer the brokerage question*,$_{2.6.1}$][Kidder, in typical fashion, completed a task-force study.$_{2.6.2}$][10]

The first clause "*To answer the brokerage question*" in Example (2.6) presents an incompleted statement without the second clause. It is only understood when

[9] Only binary relations are considered in this thesis. N-ary relations can be easily constructed from binary relations by a binary-to-n-ary conversion procedure. (An n-ary relation is a multi-nuclear relation that consists of three or more nuclei.)

[10] The superscripts such as 2.6.1 and 2.6.2 are used to distinguish different discourse units focused on in each example.

the second clause is pronounced. The effect of the relation is that the activity "*Kidder, in typical fashion, completed a task-force study*" needs to be carried out in order to perform the situation presented in the first clause. In other words, the first clause is a purpose of the activity mentioned in the second clause. Span (2.6.2) is understandable without span (2.6.1), but not vice-versa. Therefore, a *Purpose* relation holds between the nucleus (2.6.2) and the satellite (2.6.1). The *Purpose* relation is an asymmetric relation.

Definitions of the rhetorical relations in the RST (Mann and Thompson, 1988) are only guidelines for the reader to understand and to be able to recognise relations in a text. These definitions do not provide any signal that can be used to computationally posit relations. Finding the factors that can signal relations is the centre of many studies in discourse analysis, including the research in this thesis.

Rhetorical relations are represented in discourse trees on the basis of five schemas (see Mann and Thompson (1988) for details). These schemas are also applied in this thesis. For the clarification of presentation, we present the discourse trees corresponding to the two basic types of rhetorical relations (i.e., asymmetric relation and symmetric relation) in Figure 2.4 below.

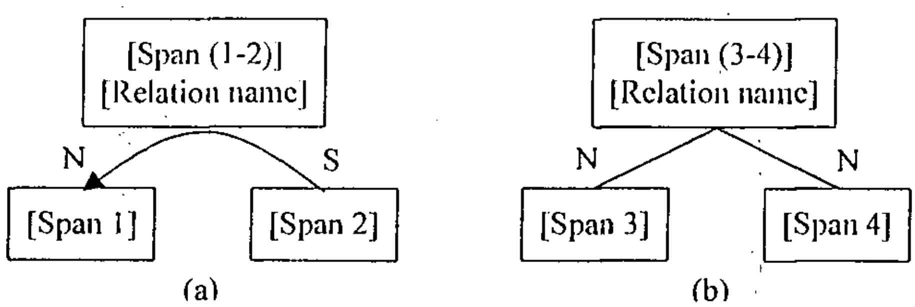


Figure 2.4. Basic Discourse Trees Used in this Thesis

Figure 2.4.a represents the discourse tree of an asymmetric relation. An arc goes from the satellite (span 2) to the nucleus (span 1) of the relation, whose arrow-head points to the nucleus. A tree node is created as the parent node of the nucleus and the satellite, which contains a relation name and the text span corresponding to this tree node. This new span (span 1-2) is the combination of the spans of the children nodes. Each child node in the discourse tree is marked with a nuclearity role (i.e., nucleus or satellite).

Figure 2.4.b illustrates the discourse tree of a symmetric relation. Both spans, which correspond to the nuclei of this relation, are connected with their parent node by straight lines. The parent node contains information about its text span and the name of the relation between its child nodes.

The discourse tree of the asymmetric relation in Example (2.6) is displayed in Figure 2.5. An arc with an arrow-head goes from the satellite "*To answer the brokerage question*" to the nucleus "*Kidder, in typical fashion, completed a task-force study*". Instead of displaying the span "*To answer the brokerage question, Kidder, in typical fashion, completed a task-force study*" in the parent node of the two nodes (2.6.1) and (2.6.2), we only represent the index of the first and last spans contributing to the parent node.

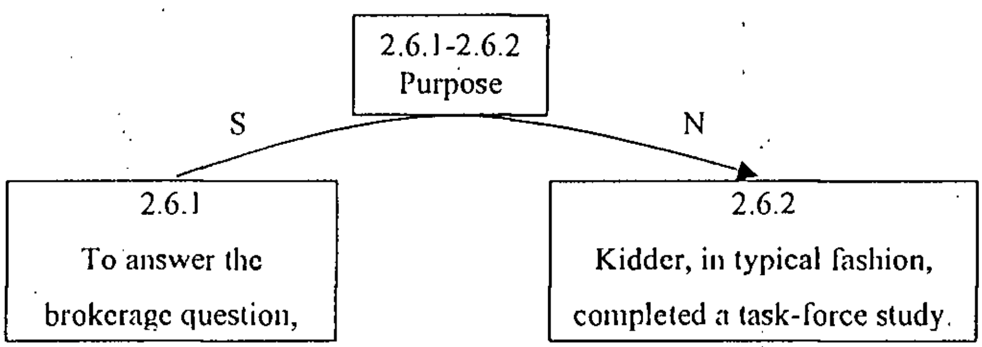


Figure 2.5. Discourse Tree of Example (2.6)

Figure 2.6 represents the discourse tree of the symmetric relation in Example (2.5). A *List* relation[11] holds between the two spans in this example.

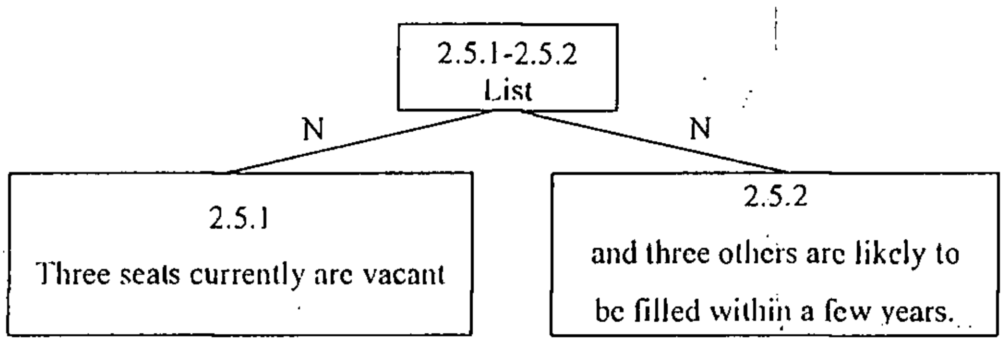


Figure 2.6. Discourse Tree of Example (2.5)

[11] See Section 4.3.2 and Appendix 6 for the definition of rhetorical relations.

To represent the discourse tree of a relation between large spans, each of which contains more than one clause, the tree nodes of these spans are replaced by their correspondent discourse trees. This is illustrated by Example (2.7) shown below.

(2.7) [Only a few months ago, the 124-year-old securities firm seemed to be on the verge of a meltdown, $_{2.7.1}$][racked by internal squabbles and defections. $_{2.7.2}$] [Its relationship with parent General Electric Co. had been frayed since a big Kidder insider-trading scandal two years ago. $_{2.7.3}$][Chief executives and presidents had come and gone. $_{2.7.4}$]

The text in Example (2.7) consists of four elementary discourse units. The clause "*racked by internal squabbles and defections*" elaborates the information in the clause "*Only a few months ago, the 124-year-old securities firm seemed to be on the verge of a meltdown*". The second sentence relates to the first sentence by a *List* relation. The last sentence elaborates two sentences before it. Figure 2.7 displays the discourse tree for the text in Example (2.7). Instead of displaying the content of each leaf, we only show the indexes of the corresponding spans.

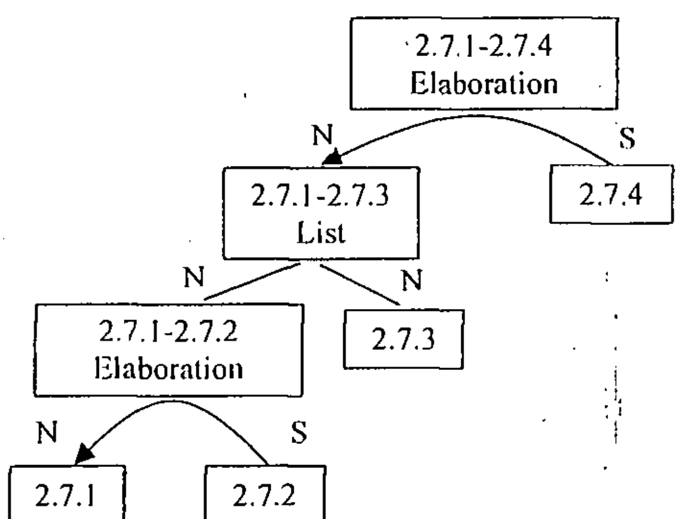


Figure 2.7. Discourse Tree of Example (2.7)

In Figure 2.7, an *Elaboration* relation holds between two leaves (2.7.1) and (2.7.2). The internal tree node corresponding to the text span that covers the two spans (2.7.1) and (2.7.2) is represented by the index of its first and last spans (2.7.1-2.7.2), and the name of the relation that holds between these spans. The arc with an arrow-head that goes from (2.7.2) to (2.7.1) indicates that span (2.7.1) is the nucleus in a rhetorical relation between (2.7.1) and (2.7.2).

A discourse tree is created for the *List* relation between spans (2.7.1-2.7.2) and (2.7.3). Span (2.7.1-2.7.2) in this tree is represented by the tree that contains two children (2.7.1) and (2.7.2). Similarly, a tree with the parent node that contains spans (2.7.1-2.7.4) and an *Elaboration* is generated, as shown in Figure 2.7.

According to Mann and Thompson (1988), a valid RST tree that describes the structural analysis of a text must satisfy the following constraints:

- **Completeness**: One RST tree covers the entire text.
- **Connectedness**: Except for the entire text as span, each span in the analysis is either a minimal unit or a constituent of another tree of the analysis.
- **Uniqueness**: Each RST tree consists of a different set of spans.
- **Adjacency**: Only adjacent spans can be connected to form larger spans.

These constraints are employed in this thesis as principles to generate discourse trees. They are not only used to check the well-formedness of the final RST trees, but also considered as conditions to limit the search space of discourse trees during the generation process (see Section 5.3.1 for more detail).

2.2.2 Discussion

Although the Rhetorical Structure Theory has been widely used in most research in discourse analysis, many issues still need to be addressed. The theory in this research only provides some basic ideas that may need further studies to be validated, both from a theoretical and computational point of view. These problems are:

1. No standard set of rhetorical relations has been defined. Mann and Thompson (1988) have proposed a set of 23 relations. However, as stated in their report, this relation set can vary, depending on the purpose of particular genres and cultural styles.
2. Although Mann and Thompson have provided a definition for each rhetorical relation, little guidance is given on how to recognise rhetorical

relations. At present there are many debates involving positing the most suitable relation for a specific example.

3. Mann and Thompson have not given us any method to recognise rhetorical relations in a computational way. According to Mann and Thompson, the process of relation recognition depends on functional and semantic judgements alone, not on morphological or syntactic signals.
4. It is not clear from Mann and Thompson as to what order of spans to form a discourse tree. This paper only presents an example of an eight-sentence-text being analysed using five kinds of schemas. There is no rule to say which spans should be connected in a rhetorical relation.
5. The above problems cause the problem of multiple analyses. Different people may create different discourse trees for the same text and we cannot say which trees are incorrect. Even one person may generate two different trees for the same text. The RST does not give any instruction on how to evaluate the correctness and the quality of discourse trees, nor the similarity among different discourse trees.
6. Although the RST has been popular among studies in discourse, there are other discourse theories, which have been used by other researchers (e.g., Grosz and Sidner, 1986; Polani, 1988). Therefore, it is necessary to understand the compatibility between the RST and other discourse theories.

The RST has been understood as a method to understand the coherence of text. It has the potential ability to be used in or to inspire many text processing applications such as text generation, automatic text summarisation, and evaluation of students' compositions. Therefore, it is necessary to turn the theory of rhetorical structure into a real computational discourse system, which can automatically generate rhetorical structures. To achieve this purpose, it is ideal to solve all the problems mentioned above. However, to do so would be too ambitious for the scope of this thesis. For the present study, we concentrate on three main unsolved issues in discourse analysis discussed in Section 1.1. The data used in the experiments of this research are documents taken from the RST Discourse Treebank (RST-DT, 2002), which is described next.

2.3 Overview of the Corpus

Discourse analysis has begun to receive the attention of the computational linguistics community in recent years. Each researcher in discourse uses different data to evaluate the system. Because of that, it is difficult to compare the performance of one system with the others. Several efforts have been made to annotate discourse structures. The main studies among these efforts are done by Carlson et al. (2002) and Forbes et al. (2003). The corpus created by Carlson et al. (2002) is based on the RST framework, whereas the one created by Forbes et al. (2003) reflects the theory of the D-LTAG. Since the RST corpus (RST-DT, 2002) created by Carlson et al. is the only available corpus that follows the Rhetorical Structure Theory, it is used in the experiments carried out in this research. An overview of this corpus is presented in the remainder of this section.

The RST-DT corpus contains 385 Wall Street Journal (WSJ) articles from the Penn Treebank (1999). These articles have been manually annotated with rhetorical structures in the RST framework. Each article is accompanied with an .edus file and a .dis file. The .edus file contains the elementary discourse units of the article with one discourse unit per line. These discourse units have been generated by a human. The .dis file contains the manually annotated rhetorical structure of that article. This file has a structure similar to the LISP language. Fifty three articles in the corpus have been independently annotated by a second analyst. These 53 documents have been used to compute human agreement on the rhetorical structures derived from the same texts. One hundred and ten different rhetorical relations are used in the RST-DT corpus. This corpus also contains extract and abstract documents of the WSJ articles, which can be used for summarisation tasks.

The next chapter analyses the problem of discourse segmentation, whose purpose is to split text into discourse units with independent functional integrity. In order to solve this task, we combine two processes: discourse segmentation by syntax and discourse segmentation by cue phrase.

3 Discourse Segmentation

According to Mann and Thompson (1988), a rhetorical structure is constructed from smaller discourse segments. All discourse units should have independent functional integrity, such as independent clauses. The smallest discourse unit is called *elementary discourse unit* (edu) (Marcu, 2000). Therefore, the first problem in constructing the rhetorical structure is to segment text into elementary discourse units.

Considering the advantages and disadvantages of different approaches in discourse segmentation discussed in Section 2.1.2.1, a new discourse segmenting method that combines the syntactic approach with the cue phrase approach is proposed in this thesis (LeThanh et al., 2004a). The principles used in our approach to segment text into elementary discourse units are mainly based on previous research in discourse segmentation (Carlson et al., 2002). Since a typical discourse unit is an independent clause or a simple sentence, the text is first split into elementary discourse units using syntactic information. One may argue that using syntactic information is complicated since a syntactic parser is needed to generate this kind of information, but there are a number of good syntactic parsers available nowadays. To deal with the case where strong cue phrases make a noun phrase become a separate elementary discourse unit, a further segmentation process is undertaken after segmenting by syntax. The purpose of the latter process is to detect strong cue phrases. Both these processes will be discussed in more detail in the following sections.

The rest of this chapter is organised as follows. The first step of discourse segmentation (Step 1) - Discourse Segmentation by Syntax - is described in Section 3.1. Discourse Segmentation by Cue Phrases (Step 2) is introduced in Section 3.2. Section 3.3 summarises the segmentation method and discusses the possible future work.

3.1 Discourse Segmentation by Syntax – Step 1

In this section, we introduce a method to segment text by using sentential syntactic structures. There are two methods to get the syntactic structure of sentences. One method is using a syntactic parser to generate the syntactic information from the plain text. Another method is using a gold standard corpus that contains syntactic parsed documents annotated by human analysts. Since we concentrated on the discourse analysing task, we chose the second method – using a syntactic annotated corpus (the Penn Treebank (1999)) – to get sentential syntactic structures. The Penn Treebank (1999) is chosen because of two reasons. First, this syntactic corpus is widely accepted and used in much syntactic research. Second, documents from the RST Discourse Treebank (RST-DT, 2002) are also taken from the Penn Treebank.

A requirement for the input syntactic information is that the clausal boundaries should be correctly assigned. If this syntactic information, which comes either from the annotated corpus or from the output of a syntactic parser, contains incorrect clausal boundaries, it will affect the system's performance (see Section 6.2.1).

The input to this module is a sentence[12] and its syntactic information. It checks segmentation rules, which are based on sentential syntactic structures, to split sentences into discourse segments. This process also provides initial information about rhetorical relations between spans within a sentence, such as which spans should be connected, and the nuclearity status (i.e., nucleus, satellite) of spans in a rhetorical relation. A brief description of the segmentation rules is presented in Section 3.1.1. An implementation of the segmentation algorithm that is based on these rules is introduced in Section 3.1.2. The post segmenting process is discussed in Section 3.1.3.

3.1.1 Discourse Segmentation Rules

The rules for dividing sentences into discourse segments[13] in this step are based on the syntactic structure of the sentence. These rules are based on the segmentation principles proposed by Carlson et al. (2002). The contribution of this thesis here is

[12] A text is split into sentences by another procedure before being used as the input of this module.

[13] In this chapter, "discourse segment" refers to a segment of a sentence that is generated during the segmentation process. "Elementary discourse unit" refers to the final output of the segmentation process. A discourse segment may be larger than an elementary discourse unit.

to propose a method that automatically detects discourse segments, instead of a segmention process that depends on humans as in Carlson et al. (2002).

In this section, we first analyse three representative samples of segmentation principles (principles i to iii) and describe our method of implementing these principles. These samples represent the main segmentation categories in respect of syntactic roles: segmenting a clause from a noun phrase; segmenting a clause from a verb phrase; and segmenting a clause from a sentence or a complex clause. The complete set of segmentation principles can be found in Carlson et al. (2002). After the representative principles have been given, we introduce the basic segmentation rule and the syntactic chains that correspond to the sample principles. An implementation of the segmentation process is described in Section 3.1.2.

Principle (i) - *The clause that is attached to a noun phrase (NP) can be recognised as an embedded unit.*

For example:

(3.1) [Mr. Silas Cathcart built a shopping mall on some land][he owns.]

Principle (ii) - *Coordinate clauses and coordinate elliptical clauses of verb phrases (VPs) are elementary discourse units. Coordinate VPs that share a direct object with the main VP are not considered as a separate discourse segment.*

For example:

(3.2) [The firm seemed to be on the verge of a meltdown,][racked by internal squabbles and defections.]

Principle (iii) - *Coordinate clauses and coordinate sentences of a complex sentence are elementary discourse units.*

For example:

(3.3) [The firm's brokerage force has been trimmed][and its mergers-and-acquisitions staff increased to a record 55 people.]

The basic segmentation rule that corresponds to the segmentation principle is:

If: a sentence satisfies the segmentation conditions of a segmentation principle

Then: split the sentence into discourse segments

The '*If*' part of the rule checks whether the syntactic structure of the sentence contains the syntactic chain suggested by the segmentation principles or not. Using the syntactic assignments of the Penn Treebank (Bies et al., 1995), syntactic chains that correspond to principle (i) are:

(i-a) (NP|NP-SBJ <text1> (X <textx>)* (SBAR|RRC <text2>))

(i-b) (NP|NP-SBJ <text1> (X <textx>)* (PRN <text2> (Y <texty>)* (S <text3>)))

(i-c) (NP|NP-SBJ <text1> (X <textx>)* (PP <text2> (Y <texty>)* (S|VP <text3>)))

SBJ, SBAR, RRC, PRN, S, and PP stand for subject, subordinate clause, reduced relative clause, parenthetical, sentence, and prepositional phrase respectively. '|' stands for 'or'. <text1>, <text2>, and <text3> are parts of the text of a sentence. (X <textx>)* and (Y <texty>)* stand for any syntactic string (or none of them). For example, consider the sentence:

(3.4) The land he owns is very valuable.

The syntactic chain which represents the noun phrase "*The land he owns*" in the above sentence can be written as (NP *The land* (SBAR *he owns*)).

According to principle (i), <text2> in syntactic chain (i-a), and the combination of <text2> and <text3> in syntactic chains (i-b) and (i-c) are recognised as embedded units. To simplify syntactic chains (i-b) and (i-c), DAS creates two labels named PRS (parenthetical-sentence) and PS (prepositional-sentence). These two labels are described respectively in (i-d) and (i-e) below:

(i-d) (PRN <text2> (Y <texty>)* (S <text3>)) → (PRS <text2-3>)

(i-e) (PP <text2> (Y <texty>)* (S|VP <text3>)) → (PS <text2-3>)

"→" can be interpreted as "*convert to*". <text2-3> is the concatenated string of <text2> and <text3>. By using syntactic chains (i-d) and (i-e), syntactic chains (i-

a) to (i-c) can be combined into one syntactic chain as given in (i-a'). It should be noted that <text2'> in (i-a') is <text2-3> in (i-d) and (i-e).

(i-a') (NP|NP-SBJ <text1> (X <textx>)* (SBAR|RRC|PS|PRS <text2'>))

The syntactic chains that map to principles (ii) and (iii) are given in (ii-a) and (iii-a) respectively. In the syntactic chains corresponding to principles (ii) and (iii), Sx stands for basic clause types such as subordinate clause (SBAR) and participial clause (S-ADV). <conjunction> stands for a conjunction such as "*and*", "*or*", comma, and semicolon.

(ii-a) (VP (VP <text1>) <conjunction> (X <textx>)* (VP|Sx|RRC|PPS <text2>))

(iii-a) (Sx <text1> (X <textx>)* (Sx <text2>) <conjunction> (Y <texty>)* (Sx <text3>))

All segmentable syntactic chains are presented in Appendix 5. The algorithm that automatically splits text into discourse segments using the segmentation principles is described next.

3.1.2 Segmentation Algorithm

The segmentation algorithm that we propose in this section is outlined in Figure 3.1. The input to this algorithm is the syntactic string of a sentence, in which <text> is replaced by a token #x,y# (where x,y is the begin and end position of <text> in the sentence being analysed). Each token of the sentential syntactic string is separated by a space. The syntactic string from the Penn Treebank of Example (3.5) is given in (3.5.a).

(3.5) "The book I read yesterday is interesting."

(3.5.a) ((S (NP-SBJ (NP The book) (SBAR I read yesterday)) (VP is (ADJP interesting))).)

The input to the discourse segmenter by syntax in this case is given in (3.5.b).

(3.5.b) ((S (NP-SBJ (NP #0,7#) (SBAR #9,24#)) (VP #26,27# (ADJP #29,39#))) .)

The segmentation algorithm uses a stack to store tokens of the syntactic string during the reading process. The algorithm ends when the syntactic string is reduced to the string "((S #x,y#) .)".

Input: The syntactic information of a sentence.

Output: Discourse segments (DSs).

1. Read each character from the input string from left to right and put them onto a stack, until a space is found.
2. Repeat Step 1 until two consecutive close brackets (')') are found on top of the stack.
3. Pop off strings from the top of the stack into a separate string called "*compared string*" until the number of open brackets and the number of close brackets in the *compared string* are equal.
4. Check whether the *compared string* maps to the *syntactic chain* (syntactic strings (i-a'), (ii-a), (iii-a), etc.) or not. If they map, segment the text corresponding to the *compared string*. A rhetorical relation is created by using these two segments as its left and right spans. Assign nuclearity roles if there is enough information.[14]
5. Encode the *compared string* as a text that consists of the syntactic category of the *compared string* in the sentence and its position tag #x,y#. Push the encoded string onto the stack.
6. Repeat Step 1 to Step 5 until the input string is empty and the stack contains the following tokens, considering from the bottom of the stack: "(", "(", "S", "#x,y#", ")", ".", ")".

Figure 3.1. Outline of the Discourse Segmentation by Syntax Algorithm

Figure 3.2 represents the segmentation progress of Example (3.5). Due to space constraints, some steps of the segmentation process are not described in detail. In

[14] In some cases, DAS can assign the nuclearity roles at this process. For example, if the segmentation rule involves a noun phrase and its subordinate clause, the noun phrase is assigned as the nucleus, the subordinate clause is considered as the satellite.

Figure 3.2.d, DAS continues pushing characters from the left side of the input string onto the stack. When there are two intermediate close brackets on top of the stack (Figure 3.2.e), the segmenter pops from the top of the stack into a *compared string* until the number of open brackets and the number of close brackets in the *compared string* are equal (Figure 3.2.g). The *compared string* is then compared with the *syntactic chain*. Since the *compared string* maps to (i-a'), the segmenter produces discourse boundaries at the beginning and the end position of the SBAR clause (at characters 9 and 24 of the input sentence). A rhetorical relation is then created between the noun phrase "*the book*" and its embedded unit "*I read yesterday*" (Step 4). The noun phrase is the nucleus; whereas the embedded unit is the satellite of this relation. This relation is stored in a list of rhetorical relations of the sentence. After that, the segmenter encodes the *compared string* (NP-SBJ (NP #0,7#) (SBAR #9,24#)) as (NP-SBJ #0,24#) and pushes the new string onto the stack (Figure 3.2.h). The segmenter continues its loop by carrying out the operators of pushing onto and poping off the stack, mapping rules, segmenting text, and encoding syntactic strings. If the *compared string* does not map to any *syntactic chain*, the *compared string* is simply encoded and pushed back onto the stack (Figure 3.2.k and Figure 3.2.l). The segmenter finishes its process when the stack consists of characters "(", "(", "S", "#0,39#", ")", ".", ")" (Figure 3.2.r).

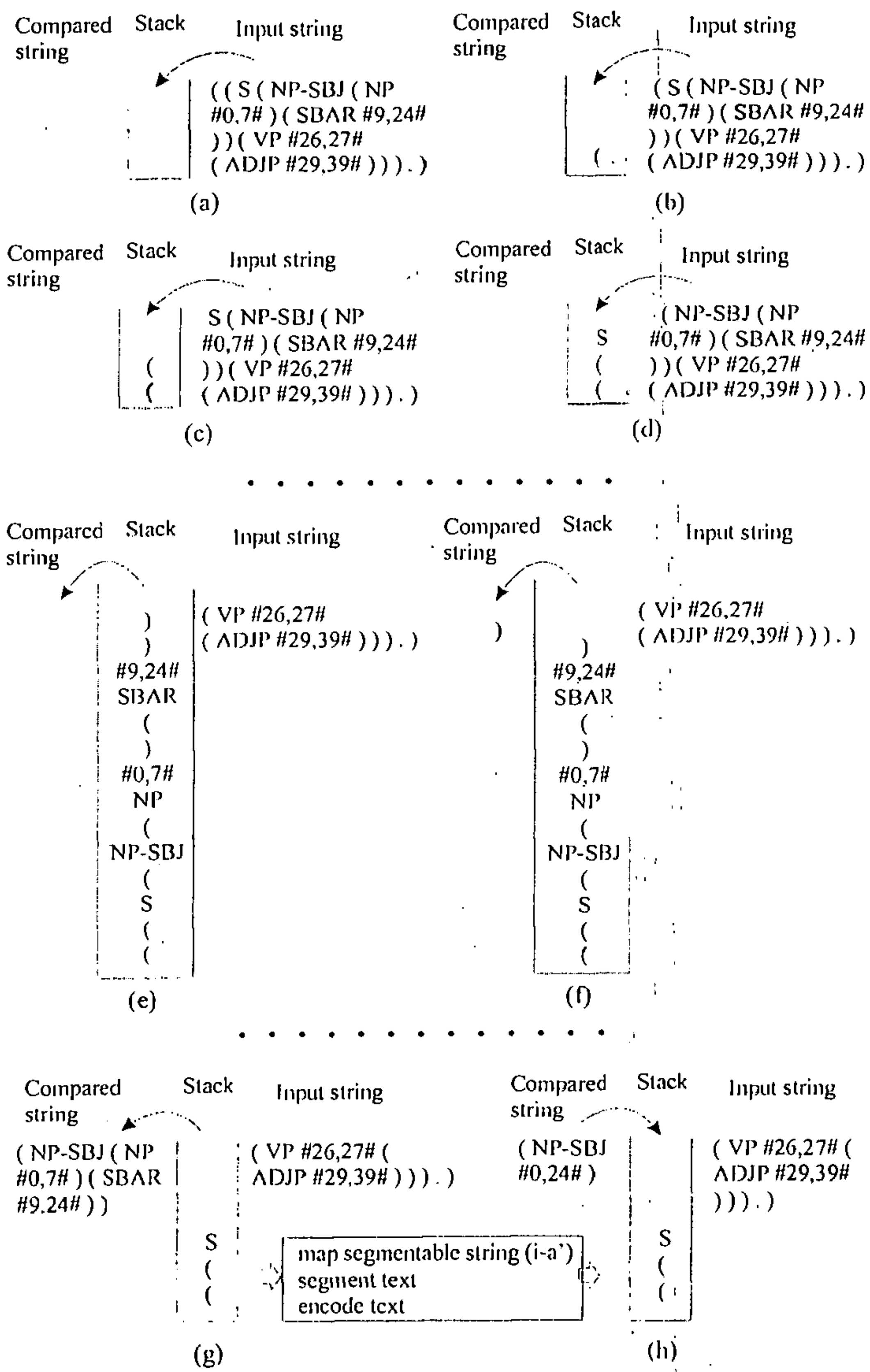


Figure 3.2. Segmenting Example (3.5) Using Syntactic Information

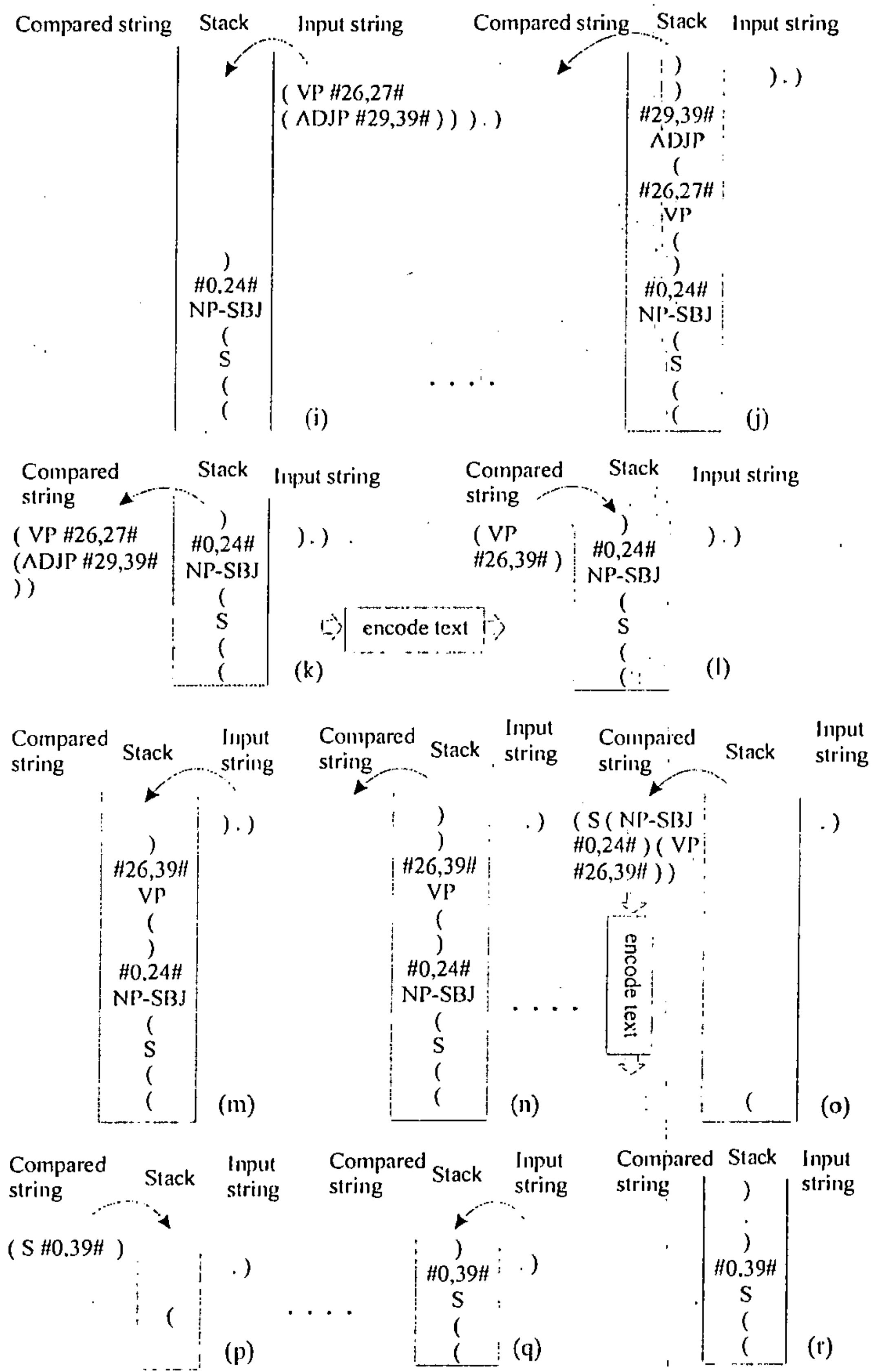


Figure 3.2. Segmenting Example (3.5) Using Syntactic Information (con't)

3.1.3 Post Segmenting Process

The purpose of the processing described in this section is to refine the output of the segmentation procedure described in Section 3.1.2. This post segmenting processing is required due to two problems, which arise from the output of the segmentation process presented in Figure 3.1. The first problem is that the segmentation of embedded units fragments the sentence.[15] We call the text being left out of the relation as "*Unknown*", as it does not belong to any relation. There are two cases under the first problem that need to be treated differently. In the first case, the *Unknown* part is adjacent to the satellite of a nuclear-satellite relation, such as in Example (3.5) given in Section 3.1.2. For the convenience of the reader, this example is repeated here as Example (3.6).

(3.6) The book I read yesterday is interesting.
N S Unknown

After Example (3.5) undergoes the segmentation process described in Figure 3.1, two segmentation boundaries are produced at the positions of characters 9 and 24 of this sentence. The sentence is divided into three parts: "*the book*", "*I read yesterday*", and "*is interesting*". "*I read yesterday*" is an embedded unit of the noun phrase "*the book*". "*The book*" and "*is interesting*" are not two separate discourse segments since they are not clauses. We deal with this case by considering "*I read yesterday*" as an embedded unit of "*the book*" and "*The book I read yesterday is interesting*" as a discourse segment. DAS generates two relations in this case. One relation relates the nucleus and the satellite. Another relation called *Same-Unit*[16] connects the span that covers the nucleus and the satellite, and the *Unknown* span. Both spans in the *Same-Unit* relation are nuclei. As the post segmenting process connects all the *Unknown* spans with the rest of the sentence, it creates an initial rhetorical structure for the sentence. Figure 3.3 displays the rhetorical structure of Example (3.6). The name of the relation

[15] This fragmentation is because of the artificial segmentation principles of the RST Treebank (RST-DT, 2002).

[16] *Same-Unit* is not a rhetorical relation. It is an artificial relation to connect two strings, which belong to the same discourse unit, being fragment by the annotation of the embedded unit.

between "*The book*" and "*I read yesterday*" is decided later in the discourse recognition process.

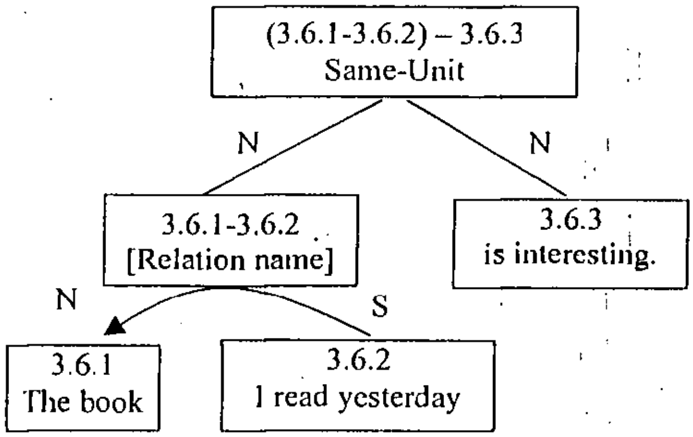


Figure 3.3. Discourse Tree of Example (3.6)

In the second case, the *Unknown* part is adjacent to the nucleus of a nuclear-satellite relation, such as in Example (3.7) shown below.

(3.7) Mr. Silas Cathcart built a shopping mall on some land *he owns.*

Unknown N S

In this case, DAS produces only one relation. It merges the *Unknown* span with the nucleus. The previous relation between the old nucleus and satellite now becomes the relation between the new nucleus, whose span covers the *Unknown* span and the old nucleus, and the old satellite. In Example (3.7), the discourse segmenter by syntax described in Section 3.1.2 produces a nuclear-satellite relation between the noun phrase "*some land*" and its embedded unit "*he owns*". The string "*Mr. Silas Cathcart built a shopping mall on*" becomes an *Unknown* span. The post segmenting process reconstructs the discourse tree of Example (3.7), as shown in Figure 3.4.

The dotted arrow shows the relation between "*some land*" and "*he owns*" created by the algorithm described in Figure 3.1. The solid arrow shows a rhetorical relation between the two discourse units after the post segmenting process. This case is treated similarly in the RST-DT corpus, as shown in Example 3.8.

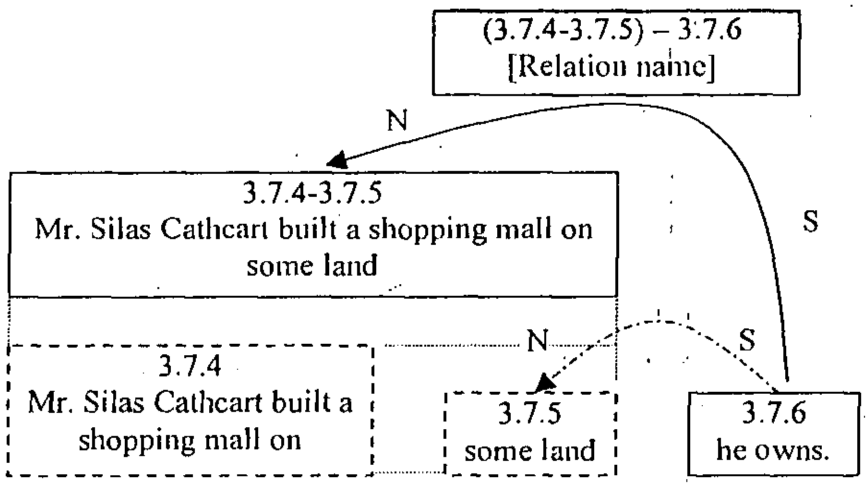


Figure 3.4. Discourse Tree of Example (3.7)

(3.8) [A new specialty court was sought by patent experts,][who believed][that the generalists had botched too many important, multimillion-dollar cases.]

The clause "*who believed that the generalists had botched too many important, multimillion-dollar cases*" is the subordinate clause of the noun phrase "*patent experts*". "*Patent experts*" is not treated as a separate discourse unit, but a part of the discourse segment "*A new specialty court was sought by patent experts*".

The second problem that needs the post segmenting process involves the placement of adverbs in discourse segments. Some adverbs, which should stand at the beginning of the right clause, are put at the end of the left clause by the segmenting process in Section 3.1.2. An example of such a case is shown in (3.9).

(3.9) [They had to give up that campaign, *mainly*][because they did not have enough people.]

DAS recognises the second clause in Example (3.9) as a subordinate clause of the first clause. It produces a segment boundary at the first position of the subordinate clause, which is the position before the word "*because*". However, the correct segmentation in this case should be before the word "*mainly*", as shown in (3.10).

(3.10) [They had to give up that campaign,] [*mainly* because they did not have enough people.]

To deal with this situation, DAS checks all adverbs that are adjacent to the left boundary of the right clause. If these adverbs do not belong to the syntactic structure of the left clause, they will be moved to the right clause. After undergoing the post segmenting process, the segment boundary between "*mainly*" and "*because*" in Example (3.9) created by the previous discourse segmentation process is moved to the left, which means to the position between the comma and the adverb "*mainly*" as given in Example (3.10). The RST-DT corpus analyses Example (3.10) in the same way as DAS does.

The input to the post processing procedure is the output of the segmentation algorithm given in Section 3.1.2. The outputs of the post processing procedure are the discourse segments after refining segment boundaries. In doing this, discourse segments are connected into pairs of adjacent and non-overlapping spans; the longest pair covers the entire sentence. In other words, besides refining discourse boundaries, the post segmenting process also constructs rhetorical relations between spans within a sentence. The nuclearity roles and relation names, which have not been assigned in this process, will be posited later in the sentence-level discourse analysing process (see Section 5.2). The post segmenting process includes two components: the first component corrects the position of adverbs in a sentence (see Examples 3.9 and 3.10), the second one deals with the text fragments caused by the segmentation of embedded units (see Examples 3.6 and 3.7). The latter is the centre of the post segmenting process. The pseudo-code for the second component mentioned above, which is called *Defragment*, is given in Figure 3.5. An extended version of this algorithm is given in Appendix 1.

Let us apply the *Defragment* algorithm to the discourse segments of Example (3.11), which are created by the segmentation algorithm presented in Figure 3.1.

(3.11) [Three seats currently are vacant][and three others are likely to be filled within a few years,][so patent lawyers and research-based industries are making][a new push][for specialists to be added to the court.]

Input:

- *start* and *end* position of the input phrase needed to be processed.
- A list of relations *sentNodes* created by the segmentation procedure presented in Figure 3.1.

Output:

- Rhetorical relations after refining boundaries.

Defragment(start, end)

1. Find a node *changenode* that starts at the left most boundary of relations within the input phrase (*minsta*), ends at the right most boundary of relations that are within the input phrase and start at *minsta*, (*maxend*). The maximum position of boundaries between the left and right node of the tree nodes that starts at *minsta* and ends at *maxend* is *middle*.
2. if (*minsta* > *start*):
   2.1 if (*changenode.leftrole* = 'N'): Expand the left node of the *changenode* to the start position.
   2.2 else: Create a new node, whose left node corresponds to the remaining span, and the right node is the *changenode*. Assign nuclearity roles for these nodes.
3. if (*maxend* < *end*):
   3.1 if(*changenode.leftrole* = 'S'): Expand the right node of the *changenode* to the end position.
   3.2 else: Create a new node, whose left node is the *changenode*, and the right node corresponds the remaining span.
4. if(*middle* < *end*) *Defragment(start, middle);*
5. if(*middle* > *start*) *Defragment(middle, end);*
6. Return.

Figure 3.5. Pseudo-code for the *Defragment* Process of Discourse Segmentation by Syntax

After being segmented by the algorithm in Figure 3.1, three rhetorical relations are created: one between "*Three seats currently are vacant*" and " *and three others are likely to be filled within a few years,*", one between " *a new push*" and " *for specialists to be added to the court.*", and one between "*Three seats currently* *are vacant and three others are likely to be filled within a few years,*" and " *so patent lawyers and research-based industries are making a new push for specialists to be added to the court.*". The positions of the left and right tree nodes of the first, the second, and third relations are (0,33) and (33,95); (155,166) and (166,208); and (0,95) and (95,208), respectively. Figure 3.6 shows the positions of these spans and their relations in the sentence.

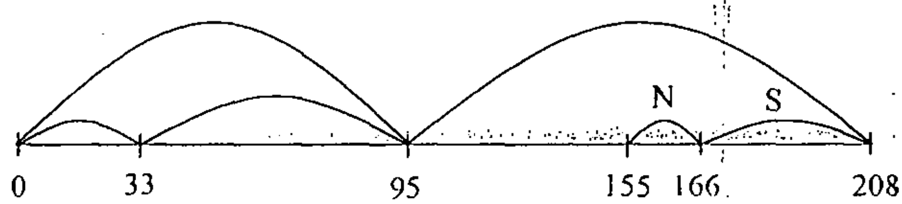


Figure 3.6. Relations within Example (3.11) before Applying *Defragment* Process

It is clear that the rhetorical relations presented in Figure 3.6 cannot form an RST tree because the connectedness constraint of the RST (Mann and Thompson, 1988) is not satisfied here (see Section 2.2.1 for the statement of the connectedness constraint). This problem is fixed by the *Defragment* procedure outlined in Figure 3.5.

start	end	minsta	maxend	middle	changenode before	newnode	changenode after	sentNodes
0	208	0	208	95	(0,95,208)			(0,33,95) (155,166,208) (0,95,208)
0	95	0	95	33	(0,33,95)			-
0	33	0	NA	NA				-
33	95	NA	NA	NA				-
95	208	155	208	166	(155,166,208)		(95,166,208)	(0,33,95) (95,166,208) (0,95,208)
95	166	NA	NA	NA				-
166	208	NA	NA	NA				-

Table 3.1. *Defragment* Process for Example (3.11)

Table 3.1 represents the progress of the *Defragment* process to refine segment boundaries and generate rhetorical relations. In Table 3.1, "*NA*" is the abbreviation for "*Not Available*"; "-" means "*same as the above row*"; each node is represented by a triple of the node properties (from, leftnode.to, to).

After undergoing the *Defragment* process, the three relations of Example (3.11) are now updated as (0,33) and (33,95); (95,166) and (166,208); (0,95) and (95,208). These relations are shown in Figure 3.7.

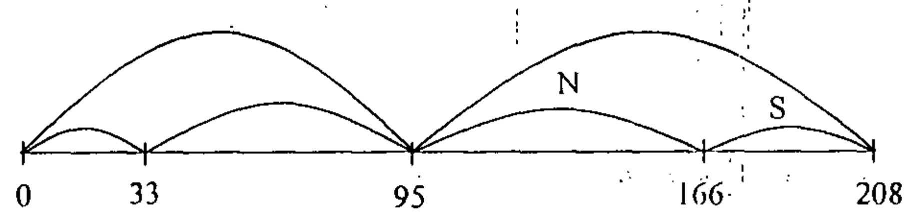


Figure 3.7. Relations within Example (3.11) after Applying *Defragment* Process

The three relations showed in Figure 3.7 now form an RST tree that satisfies the four constraints of the RST (Mann and Thompson, 1988).

When elementary discourse units are clauses, syntactic information is reliable enough to be used in segmenting text. However, syntactic information could not detect the case when an elementary discourse unit is a noun phrase. This case will be analysed and solved in Section 3.2.

3.2 Discourse Segmentation by Cue Phrases – Step 2

Several noun phrases are considered as elementary discourse units when they are accompanied by a strong cue phrase[17] (Examples 3.12). These cases cannot be recognised by syntactic information. Therefore, another segmentation process is integrated into DAS to deal with such cases. This process searches for a strong cue phrase in each discourse segment generated by Step 1. When a strong cue phrase is found, the algorithm splits the discourse segment into two elementary discourse units: one unit is the noun phrase that contains the strong cue phrase,

[17] In this thesis, a cue phrase that is strong enough to make a noun phrase become an elementary discourse unit is called a "*strong cue phrase*". Otherwise, it is a normal cue phrase or a weak cue phrase.

and another unit is the rest of the discourse segment. The set of strong cue phrases used in the experiments described in this thesis are: *according to, as a result of, although, because of, but also, despite, despite of, in spite of, irrespective, not only, regardless, without, --*. It is created basing on previous research about elementary discourse units such as Marcu (1997) and on the observation of the annotated documents from the RST-DT corpus (2002).

There are two cases of strong cue phrases that are treated differently by DAS, as shown in Examples (3.12) and (3.13):

(3.12) [*According to* a Kidder World story about Mr. Megargel,][all the firm has to do is "position ourselves more in the deal flow."]

(3.13) [In 1988, Kidder eked out a $46 million profit,][mainly *because of* severe cost cutting.]

In the first case, there is no adverb that is adjacent to the strong cue phrase and on the left of the cue phrase (Example 3.12). A new elementary discourse unit is created from the beginning position of the cue phrase to the end boundary of the noun phrase. The end boundary of a noun phrase is identified by punctuation such as a comma, a semicolon, or a full stop.

In the second case, some adverbs are left-adjacent to the strong cue phrase (Example 3.13). If these adverbs do not belong to the syntactic structure of the left part of the old discourse segment, a new elementary discourse unit is created from the left most position of these adverbs to the end boundary of the noun phrase. Otherwise, the new elementary discourse unit is created in the same way as in the first case.

3.3 Summary and Discussion

In this chapter, we have presented a discourse segmenting method based on syntactic information and cue phrases. The discourse segmentation by syntax consists of two processes. The first process divides text into discourse segments based on syntactic information. The second process refines the output of the first process to guarantee the independent functional integrity of each discourse segment. After the input text has been segmented by using syntactic information, noun phrases that have the role of elementary discourse units are recognised by detecting strong cue phrases from text.

At the moment, the rules for the discourse segmentation by syntax are manually created based on previous research in discourse analysis (Carlson et al., 2002). The experiments carried out in this research show that this discourse segmenter has good performance when compared with other discourse segmenters known to us (Section 6.2.1). It shows that the combination of sentential syntactic structures and cue phrases are reliable enough for intra-sentential discourse segmentation.

A problem faced by all discourse segmenting systems is that different people may create different elementary discourse units for the same text. For that reason, a flexible rule set that can adapt to new segmentation approaches is preferred by DAS. For future work, a method for automatically learning syntactic-based rules from a discourse corpus can be considered and then these rules can be used to segment text into elementary discourse units. Cue phrases can also be considered in the future system since they are the strongest signals and they provide the simplest way to annotate rhetorical structures.

After a text is segmented into elementary discourse units, the next task in discourse analysis is to find all possible rhetorical relations between them. This problem is addressed in Chapter 4.

4 Positing Rhetorical Relations between Elementary Discourse Units

Rhetorical relations have been recognised based on different factors. The factors that have been mostly used by researchers are cue phrases, anaphora resolution, and VP-ellispis (see Section 2.1.2.2). We apply some cohesive devices that have already been exploited by other researchers and propose new factors including noun-phrase cues and verb-phrase cues (LeThanh and Abeysinghe, 2003b).

This chapter is organised as follow. The set of relations that is used to represent the rhetorical structure of text is introduced in Section 4.1. Section 4.2 introduces the factors that are used in DAS to signal rhetorical relations. Section 4.3 describes the method that uses these factors to recognise relations. An implementation of the recognition process is presented in Section 4.4. Finally, a summary of this chapter is provided in Section 4.5.

4.1 The Set of Relations

Before considering constructing a rhetorical structure, it is necessary to define a set of relations to describe this structure. Which relations and how many relations are going to be enough to describe the text coherence by a rhetorical structure? According to Mann and Thompson (1988), the set of rhetorical relations is open. It can be modified for the purposes of particular genres and cultural styles. If the relation set consists of just a few relations, the discourse trees will be easier to construct, but they will not be informative. On the other hand, if it is a large relation set, the trees will be informative, but they will be difficult to build.

The number of relations proposed by researchers varies from two (Grosz and Sidner, 1986) to over a hundred (Carlson et al., 2002). According to Hovy (1990), the two relations proposed by Grosz and Sidner (1986), *Dominance* and *Satisfaction-Precedence*, may satisfy from the point of view of text summarisation, but it is not so from the point of view of text generation. Hovy (1990) has carried out a survey of the works of approximately 30 researchers and identifies more than 350 relations from their research. He then proposes a set of

70 relations, which is achieved by fusing and classifying these relations. His motivation in defining this set is to produce a standardised and covering set of relations. However, this taxonomy is then replaced by Maier and Hovy (1991), as they state the set proposed by Hovy (1990) fails to recognise the communicative differences between the various relations.

The problem arising from this work is how to justify whether one set of relations is adequate or not, and how to justify whether one set is more appropriate than another. Mann and Thompson (1988) use five different relations to describe causal relations (*Volitional Cause, Non-Volitional Cause, Volitional Result, Non-Volitional Result,* and *Purpose*). All these five relations are grouped together by Scott and de Souza (1990) for the task of textual realisation.

According to Knott (1996), in order to justify the set of relations, we have to have a way of deciding on an appropriate level of detail. Knott states:

> "The standards of adequacy are set by the demands of the theory in which the relations figure. The theory will determine what information about a text relations are supposed to capture; we can then ask whether the description they provide is in fact sufficient to capture that information." (Knott, 1996, pp.40)

Corston (1998) uses a set of thirteen relations in RASTA, as he claims at least these relations are required for the analyses of *Encarta 96* articles. He eliminates six relations from the original set of relations in Mann and Thompson (1988) (*Antithesis, Enablement, Evaluation, Interpretation, Motivation,* and *Solutionhood*), as these relations do not participate in building the rhetorical structures for articles in *Encarta 96*.

The articles from the RST discourse corpus (RST-DT, 2002) used in this thesis were manually analysed using 110 different relations (see Section 2.3). It is very difficult to automatically construct RST trees based on such a large set. Therefore, we propose a smaller set by merging relations with similar characteristics in these 110 relations, resulting in a set of 22 relations: *List, Sequence, Condition, Otherwise, Hypothetical, Antithesis, Contrast, Concession, Cause, Result, Cause-Result, Purpose, Solutionhood, Circumstance, Manner, Means, Interpretation,* *Evaluation, Summary, Elaboration, Explanation,* and *Joint.* We use three different relations *Couse, Result,* and *Cause-Result* to emphasise the essential text span in each rhetorical relation (see Appendix 6 for definitions of these relations). This relation set is created by taking the most widely used relations by researchers on discourse analysis (e.g., Mann and Thompson, 1988; Hovy, 1990; Marcu, 2000; Corston, 1998). Also, these relations are separate enough so that DAS can recognise one relation from another.

As mentioned at the beginning of this section, since the set of rhetorical relations is an open set, we make no claim that this set covers all other relations or is correct in all details. It can be reduced, extended, or modified depending on different purposes and data. The modification of the set of relations does not affect the approach used in this thesis. In order to fit with the new set of relations, DAS is easily modified by changing the conditions for recognising relations based on recognition factors proposed in Section 4.2. Other analysing modules used in DAS, i.e., discourse segmentation (Chapter 3) and discourse analysing (Chapter 5), still remain the same since they are independent of the set of relations.

4.2 Factors Used for Signalling Rhetorical Relations

This section presents different recognition factors used in DAS for signalling rhetorical relations. In addition, to exploit new properties of the factors that have been investigated in other research (cue phrases, syntactic information, time references, reiterative devices, reference words, substitution words, and ellipses), we propose new recognition factors (noun-phrase cues and verb-phrase cues). Similar to cue phrases, these new factors are very useful in recognising relations. The recognition factors are briefly presented below.

4.2.1 Cue Phrases

Cue phrases (e.g., *however, as a result*), also called discourse connectives, conjunctions, or discourse markers, are words or phrases that connect clauses, sentences, or larger textual units. Cue phrases are the most simple and obvious means of signalling relations in text because of two reasons. First, they explicitly express the cohesiveness among textual units most of the time. Second, identifying cue phrases is essentially based on pattern matching. For example, the cue phrase "*when*" in Example (4.1) determines a *Circumstance* relation between two clauses "*He was staying at home*" and "*the police arrived*":

(4.1) [He was staying at home][*when* the police arrived.]

Cue phrases have been widely and systematically investigated in both linguistic and computational literature. Therefore, we created a set of cue phrases for recognising rhetorical relations based on previous studies of cue phrases. This set is inherited from those in Grosz and Sidner (1986), Hirschberg and Litman (1993), Knott and Dale (1994), and Marcu (2000). The list of cue phrases used is shown in Appendix 3.

Some problems in using cue phrases

Although cue phrase has shown to be the simplest factor to signal rhetorical relations, they are not without problems. Previous studies on cue phrases (Litman, 1996; Marcu, 2000; Webber et al., 1999a) have drawn out several difficulties in recognising cue phrases and using them in signalling rhetorical relations. These problems are:

A. Ambiguity between the discourse sense and the sentential sense of a cue phrase;
B. Ambiguity about rhetorical relations;
C. Effective scope of cue phrases;
D. Multiple discourse connectives.

We will address these problems and propose our solutions to each case.

A - *Ambiguity between the discourse sense and the sentential sense of a cue phrase.*

A word or phrase can have a discourse sense in some cases, but it may not do so in the others. For example, the word "*and*" is a cue phrase in Example (4.2), but not in Example (4.3) as shown below.

(4.2) [Mary borrowed that book from our library last Monday,] [*and* she returned it this morning.] Sequence

(4.3) Mary has a eat *and* a dog.

The word "*and*" in Example (4.2) starts a new action "*she returned it this morning*" that happens after the first action represented by the first clause of the sentence. "*And*" has a discourse sense here by signalling a *Sequence* relation between these two clauses. On the other hand, the word "*and*" in Example (4.3) does not give any information about the rhetorical relation. Instead, it is only a conjunction that connects two noun phrases in that sentence. When a word only expresses a sentential meaning in the current context such as "*and*" in (4.3), it only has a sentential sense. In our experiments we noticed that a cue phrase in the discourse sense has a different effect to a sentence than a cue phrase in the sentential sense. The sentence is still grammatically correct when the "*discourse sense*" cue phrase is removed from the sentence, but it is not so with the "*sentential sense*" cue phrase. Therefore, we used syntactic information to deteet the sense of a cue phrase.

Examples (4.2) and (4.3) show that the position of a word in a sentence is important in deciding the discourse role of that word. The word "*and*" has a discourse sense only when it stands at the beginning of the right span of a rhetorical relation. Because of this, we added information about the possible positions of a cue phrase in a span to each cue phrase. If a cue phrase has a discourse sense only in some special positions of a sentence, the information about its position will be attached to the cue phrase. From now on, when mentioning a cue phrase, we only refer to the cue phrase in its discourse sense.

<u>*B – Ambiguity about rhetorical relations*</u>

Finding a cue phrase in its discourse sense does not mean a rhetorical relation can be immediately posited since a cue phrase may signal two or more relations. In Example (4.4), "*since*" ean be interpreted as a notation about the time "*I have not seen John*". Meanwhile, the clause after "*since*" in Example (4.5) explains the reason why "*He came back to Berlin*". As a result, the cue phrase "*since*" signals a *Circumstance* relation in Example (4.4) and an *Explanation* relation in Example (4.5).

(4.4) I have not seen John *since* he came back from Austria.

(4.5) He came back to Berlin *since* he prefers to live there.

In order to posit a relation between elementary discourse units in these cases, other information has to be taken into account. In a *Circumstance* relation, the event in the circumstance clause always happens before the event in the main clause; or the event in the main clause happens during the time of the event in the circumstance clause. Therefore, the tense of spans is checked by DAS in order to extract this information. In addition, other coherence information is also computed to posit the most suitable relation in each situation. A detailed description about the process to recognise rhetorical relations is discussed in Section 4.3.

In some cases, more than one relation can be posited between spans as in Example (4.6) shown below.

(4.6) I have not seen John for a while *since* he moved to a new town far away from here.

The clause after "*since*" in Example (4.6) can be understood as an explanation for the clause before it. The reason "*I have not seen John for a while*" is that he lives too far from the writer or the speaker. This clause can also be considered as the answer for the question "*Since when have you not seen John?*". Therefore, it also has a *Circumstance* relation with the first clause. Both relations are acceptable in this case. They are kept as candidates in DAS to derive discourse trees.

C- Effective scope of cue phrases

Deciding the spans that are affected by a cue phrase is sometimes not easy, such as in Example (4.7) below.

(4.7) a. *As* the crystal grows larger,

b. the corners of the hexagon grow a bit faster.

c. The slightly faster growth at the corners soon *causes* the hexagon to sprout arms.

d. ***And since*** the ambient atmospheric conditions are nearly identical across the crystal,

e. all six budding arms grow at roughly the same rate.

The cue phrase "*as*" signals a *Circumstance* relation between the two clauses (a) and (b). The VP cue "*cause*" indicates that "*The slightly faster growth at the corners*" is a cause of "*the hexagon to sprout arms*". Since "*The slightly faster growth at the corners*" is another way of expressing the main clause (b) of the first sentence (a-b), a *Cause-Result* relation holds between the first sentence (a-b) and the second sentence (c). Both cue phrases "*and*" and "*since*" stand at the beginning of the last sentence in Example (4.7). Only the cue phrase "*since*" signals a rhetorical relation (a *Cause* relation) between the two clauses (d) and (e). The cue phrase "*and*" connects the second sentence (c) with the third sentence (d-e). It posits an *Elaboration* relation between these sentences. Two questions arise from this example:

1. How to make DAS take "*since*", not "*and*", as the cue phrase for the two clauses (d) and (e)?

2. How to make DAS decide whether the cue phrase "*and*" connect the last two sentences (Figure 4.1a) or the first two sentences and the third one (Figure 4.1b)?

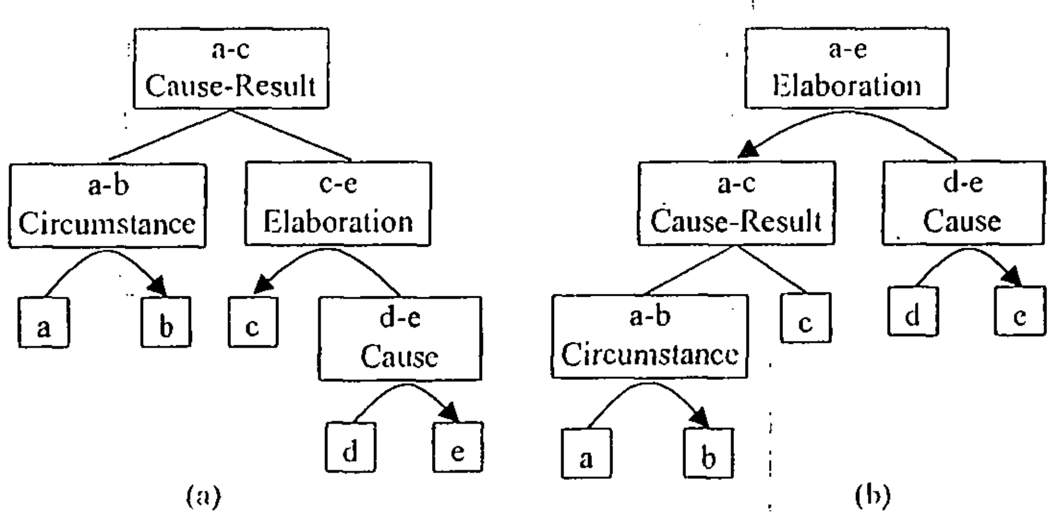


Figure 4.1. Two Possible Discourse Trees for Example (4.7)

With the first problem, we used information about positions of cue phrases. The cue phrase "*and*" has to stand at the beginning of the right span (see Appendix 3). Therefore, "*and*" cannot be used as a cue phrase for the relation between the two spans "*since the ambient atmospheric conditions are nearly identical across the crystal*" and "*all six budding arms grow at roughly the same rate*" in Example (4.7).

The second problem is solved by using the scope of cue phrases. Since some cue phrases can only connect clauses or sentences (e.g., *since*, *although*), and the others can connect paragraphs (e.g, *firstly*, *secondly*), we used this information to control the effective scope of cue phrases. As the cue phrase "*and*" is to connect clauses or sentences (Appendix 3), the discourse tree in Figure 4.1.a is chosen as the representation for Example (4.7).

D - Multiple discourse connectives

Let us consider the following situations of recognising rhetorical relations between two text spans, which are referred to as the problem of multiple discourse connectives:

1. Several adjacent cue phrases in a span.
2. Several non-adjacent cue phrases in a span.
3. Several non-adjacent cue phrases in both spans.

Example (4.7) represents the first situation. In this case, we proposed to use the following rules:

1. *If several cue phrases stand at the beginning of the left span, the right most cue phrase will decide the relation. The left most cue phrase will decide the relation between the span on its left and the span that covers the left and right span.*

2. *If two cue phrases stand at the beginning of the right span, the left most cue phrase will decide the relation.*

It is necessary to note that the chosen cue phrase needs to satisfy all of its properties (e.g., its position in a span) before it can be used to posit a relation.

The second rule is applied for a situation that has been discussed in Webber et al. (1999b), which is represented in Example (4.8):

(4.8) You shouldn't trust John because, for example, he never returns what he borrows.

The cue phrase "*because*" is chosen by the second rule to connect "*You shouldn't trust John*" and "*for example, he never returns what he borrows.*"

According to Webber et al. (1999b), the presupposition of "*for example*" is grounded through the adjacent discourse connectives "*because*", which provides evidence for a set of reasons. Thus, the right clause "*for example, he never returns what he borrows*" is a cause for the left clause "*You shouldn't trust John*". Webber et al. use the D-LTAG (Discourse Lexicalized Tree-Adjoining Grammar) to parse the sentence in Example (4.8), which produces the tree in Figure 4.2.

Both approaches, which are used in DAS and in Webber et al. (1999b), posit the same relation for Example (4.8). The difference between them is that DAS generates an RST tree, whereas the method used in Webber et al. (1999b) constructs a D-LTAG tree for the text.

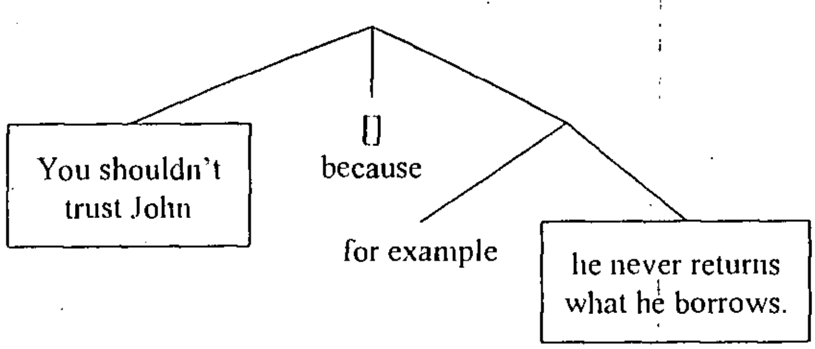


Figure 4.2. The D-LTAG Derivation for Example (4.8)

For the second situation (i.e., several non-adjacent cue phrases in a span), the cue phrase that is contiguous with the segment boundary will decide the relation. For the third situation (i.e., several non-adjacent cue phrases in both spans), all relations corresponding to these cue phrases will be checked. The procedure to check a relation is discussed in Section 4.4.

In summary, the following properties are added to each cue phrase in order to assist the process of recognising relations:

- **The possible position of a cue phrase in a span.** A cue phrase can be at the beginning, in the middle, or at the end of a span. Its respective positions are 'B', 'M', and 'E'. If the cue phrase can be at any position inside a span, then its position is 'A', which means "*any position*".
- **The span that the cue phrase belongs to.** This is indicated by the letter 'L' (for left span) or 'R' (for right span). If the cue phrase can be in either side, this property is indicated by the letter 'A', which means "*any side*".
- **The effective scope of a cue phrase.** If a cue phrase can be used only to connect clauses, its effective scope is 'C'. If the maximum size of a span that a cue phrase can connect with is the size of a sentence, the effective scope is 'S'. Otherwise, this value is 'P' (paragraph).
- **The relation name suggested by the cue phrase** (e.g., *Elaboration*, *Circumstance*).
- **The score of the cue phrase for the relation**, whose values ranges between 0 and 1. For example, "*in spite of*" is the cue phrase for the *Concession* relation; it has a score of 1. "*And*" can be a cue phrase for a *List*, *Sequence*, or *Elaboration* relation, its score for each of these relations should be lower than 1. This score is initially assigned according to human linguistic intuitions. It can be adjusted during a training process of cue phrases.

For example, the cue phrase "*and*" is stored in the set of cue phrases for the *List* relation as "*and*:B:R:S:*List*:0.8". The information for the cue phrase "*in spite of*" is "*in spite of*:B:A:C:*Contrast*:1".

4.2.2 Noun-Phrase Cues and Verb-Phrase Cues

In this section, we introduce two new types of cue phrases. They are noun-phrase cues (NP cues) and verb-phrase cues (VP cues). Examples of NP cues and VP cues are shown below:

(4.9) [New York style pizza meets Californian ingredients,][and the *result* is the pizza from this Church Street pizzeria.]

(4.10) [By the end of this year, 63-year-old Chairman Silas Cathcart retires to his Lake Forest, Ill., home, possibly to build a shopping mall on some land he owns. "I've done what I came to do at Kidder", he says.] [And that *means* 42-year-old Michael Carpenter, president and chief executive since January, will for the first time take complete control of Kidder and try to make good on some grandiose plans.]

In Example (4.9) the noun "*result*" indicates a *Result* relation; whereas in Example (4.10) the verb "*means*" signals an *Interpretation* relation between two sentences. The phrases in the main noun phrases (i.e., subjects and objects) of a sentence that signal rhetorical relations are called NP cues. These phrases can be nouns, adjectives, or adverbs. For example, the adjective "*following*" in the noun phrase "*the following week*" signals a *Sequence* relation. This word is considered as a NP cue. Similarly, the phrases in the main verb phrase of a sentence that signal relations are called VP cues. These phrases can contain verbs, adjectives, or adverbs.

NP cues, VP cues, and cue phrases are considered as separate recognition factors because of their different behaviours in recognising relations. The same word in a NP, a VP, and a clause may signal different relations or may not signal any relation at all. Let us illustrate this statement using the word "*means*". When "*means*" acts as a verb, it often signals an *Interpretation* relation (Example 4.10). When the noun "*means*" is in the main noun phrase of a sentence, it does not signal any relation (Example 4.11). Meanwhile, when the noun "*means*" is not in a main noun phrase of a sentence, but it is in the cue phrase "*by means of*", it indicates a *Means* relation (Example 4.12).

(4.11) [These *means* of transport are sometimes called accidental,][but this is not strictly correct.]

(4.12) [It is the magician's wand,][*by means of* which he may summon into life whatever form and mould he pleases.]

In addition, the cue phrases are identified based on pattern matching, whereas the NPs or VPs of text spans have to be stemmed before being compared with the NP or VP cues. The sets of NP cues and VP cues were created by us and are listed in Appendix 4. The information stored for each NP or VP cue includes the span that the NP or VP cue belongs to, relations that the NP or VP cue signal, and the score of NP or VP cue whose value is between 0 and 1. Similar to cue phrases, if a NP cue or a VP cue signals more than one relation, its score for each relation is lower than 1. This score can be adjusted during a training process.

A detailed description of the use of NP cues and VP cues will be discussed in Sections 4.3 and 4.4.

4.2.3 Reiterative Devices

The reiterative devices include word repetition, synonyms, hypernyms, co-hyponyms, and antonyms.

4.2.3.1 Word Repetition and Synonyms

Word repetition has been used in previous studies on text summarisation and information retrieval to segment text into topics (Utiyama and H. Isahara, 2001; Salton et al., 1999; Choi, 2000). The idea is that if two spans refer to the same topic (i.e., some specific words are repeated many times), there must be a discourse connection between these spans.

Synonyms can be considered as a variant of word repetition phenomenon, which is often used when people do not want to repeat a specific word so many times. In Example (4.13), the words "*employer*" and "*boss*" refer to the same person:

(4.13) Amada's *employer*, however, was less sympathetic. 'My *boss* gave me an envelope and told me it was redundancy money – two weeks' pay - $280. I was shocked.' (Salkie, 1995)

In relation recognition, the information about word repetition and synonyms is used to detect the discourse connection and relation name between spans. For example, a *Contrast* relation often occurs when most words in two spans are similar, and one span has the word "*not*". When both sentences have the same subject or object, and the same syntactic structure, it is likely that a *List* relation holds between these sentences.

4.2.3.2 Hypernyms

Hypernyms are used when one refers back to a word that has been used in the previous text, or mentioned a more general or more specific situation, such as the words "*Brazil*" and "*country*" in Example (4.14) shown below.

(4.14) *Brazil*, with her two-crop economy, was even more severely hit by the Depression than the other Latin American states. The *country* was on the verge of complete collapse.

"*Brazil*" is a specific instance of the more general word "*country*". The general word is called a hypernym; the more specific one is called a hyponym. If the specific word is used before the general word, and no cue phrase is present in the text, it is possible that an *Elaboration*, *Evaluation*, *Interpretation*, *Explanation*, or *Circumstance* relation holds between two spans. On the other hand, if the general word is used before the specific word, it often signals an *Elaboration* relation. As such, DAS uses hypernyms to limit the range of rhetorical relations needed to be examined.

4.2.3.3 Co-hyponyms and Antonyms

Co-hyponyms are the words that have the same hypernym. For example, "*Brazil*", "*Vietnam*", and "*Poland*" are co-hyponyms since all of them are all hyponyms of country. Antonyms are opposite words such as "*hot*" and "*cold*". Since the meaning of antonyms is opposite, they often express a *Contrast* relation. The co-hyponyms often refer to a multi-nuclear relation such as *Contrast*, *List*, and *Sequence*.

4.2.4 Combining Reiteration Devices with Reference Words

The reference words include personal pronouns (*I, you, he*), their object forms (*me, him*) and their possessive forms (*my, mine, your, yours*), demonstratives (*this, that, these, those*) and comparative constructions (*the same thing, a different person*, etc.). The meanings of reference words do not exist in isolation. Each word must be interpreted in its context. One needs to look back at the previous text to understand which entity the reference word refers to. For example, the pronoun "*he*" in the second sentence of Example (4.15) refers to the private name "*Graham*", whereas the pronoun "*it*" refers to the noun phrase "*his bicycle*".

(4.15) *Graham* sold **his bicycle**. *He* said *he* did not need **it** anymore.

The reference words are often combined with the reiterative devices to refer to the previous entity (see Example 4.16). The phenomenon of reiteration of an entity (called "*antecedent*") by a reference (called "*anaphor*") that points back to that entity is called "*anaphora*".

(4.16) I have read *a novel* written by Barbara Erskine. *The book* is fascinating, absorbing, and hypnotic.

Normally, after an entity is initiated, this entity is elaborated by succeeding sentences, using reference devices. When another entity is initiated, the text changes its focus to the new entity. Researchers such as Grosz and Sydner (1986) have proved that identifying the movement of focus is important in defining the rhetorical structure of text. The process of identifying the antecedent of an anaphor is called "*anaphora resolution*". Anaphora resolution has been extensively investigated in many studies (e.g., Cristea and Dima, 2001; Mitkov, 2002).

In this research, we use a simple model of anaphora resolution to recognise relations. The main noun phrases (i.e., subjects and objects of sentences), verb phrases, and adjective phrases are extracted from the syntactic information of these spans. These phrases are then stemmed into their original forms (e.g., "*books*" is converted into "*book*"). After that, DAS computes the semantic relation between these phrases using a thesaurus called WordNet (2004). The relations needed to be computed are word repetition, synonyms of nouns, hypernyms of nouns, co-hyponyms and antonyms of nouns, verbs, adjectives, and adverbs, and references.

Let us consider the following example:

(4.17) Fire is hot. Ice is cold.

The two subjects, "*ice*" and "*fire*", are co-hyponyms, since both of them have a hypernym "*substance*". The two adjectives, "*cold*" and "*hot*", are antonyms. The head verbs of two sentences have the same base form "*be*". This information signals a *Contrast* or a *List* relation between these two sentences.

A detailed description about using cohesive devices in recognising relations is described in Sections 4.3 and 4.4.

4.2.5 Time References

Discourse connection can be established by time relations between spans. If the time of a narrative changes from the present to the past, it is likely that the writer refers to a previous event that is the cause (Example 4.18), the hypothetical (Example 4.19), or the elaboration (Example 4.20) of the current event.

(4.18) Mark *has* a terrible headache today. He *drank* too much last night.

(4.19) Mark *has* a terrible headache today. He *must have drunk* too much last night.

(4.20) Mark *has* a restaurant now. He *had to work* in several restaurants before he opened his own.

If the time of the second span covers the time of the first span, a *Circumstance* relation usually holds in this case (Example 4.21).

(4.21) Mark *knows* every person in this village. He *has been living* here for more than ten years.

If no cue phrase is present in a sentence and the subject of this sentence contains a temporal NP cue (e.g., previous, next), a *List*, *Sequence*, *Explanation*, or *Elaboration* relation may hold. For example, an *Explanation* relation exists in Example (4.22).

(4.22) Mark bought a new car today. His *previous* car was stolen.

The time reference can also be used to check the validity of a relation as mentioned in Section 4.3.2.1. Since the time reference can signal discourse connection and limit possible relations, it is combined with other factors to posit rhetorical relations, as described in Section 4.3.

4.2.6 Substitution Words

Substitution serves as a place-holding device, where the missing expression is replaced by a special word (*one, do, so,* etc.), in order to avoid the repetition. "*One*" and "*some*" replace a noun phrase (Example 4.23). "*Do*" and its other forms ("*did*", "*have done*", etc.) replace a verb phrase (Example 4.24). "*So*" replaces a clause (Example 4.25).

(4.23) This *television* is too small for your room. You should buy a bigger *one*.

(4.24) I *have* never *read that book* before. I wish I *did*.

(4.25) - *Steve will get the first prize*.

- I think *so*.

By replacing words that have already been used in the preceding text, a strong link is created between one part of the elided text and another. While reiterative devices or reference words can be distant from their antecedents, the substitution words only refer to the entities or the actions that have just been mentioned. Therefore, substitution words are used for local focus.

The substitution words "*one*", "*do*", and "*so*" also have other uses where they do not substitute for anything. For example, "*one*" can be a number; "*do*" can be an auxiliary; and "*so*" is not a substitute in "*so many*". Syntactic information is used in DAS to distinguish these cases.

4.2.7 Ellipses

Ellipsis is a special form of substitution words where a part of a sentence is omitted. The use of VP-ellipsis has been discussed in previous research in discourse analysis such as Kehler and Shieber (1997). In this research, other types of ellipses are also considered, including NP-ellipsis and clause-ellipsis. Examples of NP-ellipsis, VP-ellipsis, and clause-ellipsis are shown in Examples (4.26), (4.27), and (4.28) respectively. In these examples, the word or clause in *italics* is left out at the position that is marked with "<>".

(4.26) Steve has always been a good *student* in my class. Actually, he is the best <>.

(4.27) I *went* to the dentist, and he <> to the airport.

(4.28) *I have a feeling that this cottage is very familiar*. But I cannot explain why <>.

Many studies on VP-ellipsis have been carried out by Kehler (e.g., Kehler, 1996; Kehler and Shieber, 1997). He claims that VP-ellipsis exists in two levels: the syntactic level and the semantic level. The data support a syntactic account when a *Resemblance* relation is operative between the clauses, whereas the data support a semantic account when a *Cause-Effect* relation is operative. This observation about VP-ellipsis, as well as other aspects of NP-ellipsis and clause-ellipsis, is considered in DAS to posit rhetorical relations. The elliptical phenomenon in a text is recognised by analysing the syntactic information of sentences.

4.2.8 Syntactic Information

According to Matthiessen and Thompson (1988), clausal relations reflect rhetorical relations within a sentence. The rhetorical relation between a main clause and its subordinate clause is an asymmetric relation, in which the main clause is the nucleus, and the subordinate clause is the satellite. This proposal is applied in DAS to suggest the nuclearity role of spans and to eliminate unsuitable relations (e.g., a *List* relation cannot hold between a main clause and a subordinate clause). If two clauses are in coordination, their relation can be symmetric or asymmetric.

Syntactic information can also be used to suggest relation names. For example, the reporting and reported clauses of a sentence are considered as the satellite and the nucleus in an *Elaboration* relation:

(4.29) [*Mr. Carpenter says*][that Kidder will finally tap the resources of GE.]

In Example (4.29), the reporting clause "*Mr. Carpenter says*" is considered as the satellite, whereas the reported clause "*that Kidder will finally tap the resources of GE*" is considered as the nucleus. This is described in Chapter 3.

4.3 Conditions to Posit a Rhetorical Relation

Mann and Thompson (1988) have stated that a rhetorical relation is identified by constraints on the nucleus, on the satellite, and on the combination of the nucleus and the satellite (see Figure 2.3 in Section 2.2.1 for an example). This process depends on the reader's understanding of the text. To recognise relations in a computational way, DAS uses two kinds of recognition rules. The rules that are used to signal relations are called heuristic rules. The rules that are used to check the validity of a relation are called necessary conditions.

The heuristic rules are the applications of recognition factors to a specific relation. For example, the heuristic rule to recognise a *List* relation "The right span contains *List* cue phrases" (Section 4.3.2.1) is the application of the recognition factor *cue phrases*.

The purpose of separating two kinds of recognition rules is to reduce the work-load of the recognition process. To posit relations, DAS starts by finding recognition factors from text spans. If these factors are strong enough to signal a relation, which means that the total scores of the heuristic rules that contribute to that relation are more than or equal to a threshold θ (see Section 4.3.1), then the necessary conditions of that relation will be checked. That relation will be posited if all necessary conditions are satisfied (see Section 4.4). Since a factor often signals a limited number of relations, DAS does not need to check all relations from the relation set.

4.3.1 Scoring Heuristic Rules

Cue phrases, NP cues, VP cues, and cohesive devices have different importances in recognising rhetorical relations. The cue phrases are the ones that explicitly express discourse relations most of the time. Meanwhile, ellipsis, which is one type of cohesive devices, can only create a link between text spans and cannot determine a relation name. Therefore, the heuristic rules using cue phrases are stronger than the heuristic rules using ellipsis. To control the influence of these factors to the relation recognition, each heuristic rule is assigned a heuristic score. The cue phrase rules have the highest score of 100 because cue phrases are the strongest factor to signal relations. NP cues and VP cues are also strong factors but they are weaker than cue phrases since they do not express relations in a straightforward way like cue phrases. As a result, the heuristic rules involving NP cues and VP cues are assigned a score of 90. The heuristic rules corresponding to the remaining recognition factors receive scores ranging from 20 to 80 because these factors are weaker than NP cues and VP cues.

In this research, we separate two types of scores: the score of a heuristic rule and the score of a specific cue phrase, NP cue, and VP cue (see Sections 4.2.1 and 4.2.2). The heuristic rule involving cue phrases has the score of 100, which means DAS is one hundred per cent certain that the relation signalled by the cue phrase holds. However, it is only correct when that cue phrase explicitly expresses a relation. As mentioned in Sections 4.2.1, each cue phrase has a different level of certainty in signalling relations. The cue phrase "*instead of*" always signals an *Antithesis* relation; whereas the cue phrase "*and*" may signal a *List*, *Sequence*, or *Elaboration* relation. That means the cue phrase rule that applies to the cue phrase "*and*" is not one hundred per cent certain that a *List* relation holds. In other words, the score of a cue phrase rule should be reduced when this rule is applied to a weak cue phrase. Since the score of a cue phrase is between 0 and 1, DAS calculates the actual score of a heuristic rule involving cue phrases as follow:

Actual-score(heuristic rule) = Score(heuristic rule) * Score(cue phrase).

This treatment is also applied to NP cues and VP cues. Since a NP or VP cue can signal two or more relations, each NP or VP cue may have a different score. It follows that the actual score for the heuristic rule of a NP or VP cue is:

Actual-score(heuristic rule) = Score(heuristic rule) * Score(NP cue or VP cue).

The actual score of other heuristic rules that do not involve cue phrase, NP or VP cue is:

Actual-score(heuristic rule) = Score(heuristic rule)

If several heuristic rules of a relation are satisfied, the score of that relation will be the total scores of all factors that contribute to this relation.

Total-heuristic-score = $\sum$ Actual-score (heuristic rule)

At present, heuristic scores are assigned by human linguistic intuitions. They can be optimised by a training method. Unfortunately, we know of no discourse corpus that is large enough for this training purpose. Therefore, this training process has not been addressed in this thesis.

DAS seeks the recognition factors in the following order: cue phrases, NP cues, VP cues, and the remaining recognition factors. A rhetorical relation will be posited if the *total-heuristic-score* of this relation is greater than or equal to a threshold θ. Choosing a reasonable value for this threshold is very important since a modification of this value may affect many decisions in positing relations, therefore changing rhetorical structures of text. The threshold is initially assigned the score of 30 (compare to 100 as the maximum score of a heuristic rule), as by observation we found that recognition factors can be very weak in many cases. For a better use of the threshold, a training method to optimise this value will be considered in future work.

4.3.2 Criteria to Recognise Relations

The criteria to recognise relations in this research are inherited from Corston (1998) and then modified and extended by us. However, DAS and RASTA (Corston, 1998) use the necessary conditions and the heuristic rules for different purposes and in different orders. The necessary conditions are used in DAS to eliminate the unsuitable relations that have been signaled by heuristic rules. Meanwhile, these conditions are used in RASTA to filter out unsuitable relations from the relation set before considering any heuristic rule. DAS has to test fewer relations than RASTA since the number of relations satisfied by the heuristic rules is always less than the number of relations satisfied by the necessary conditions. Therefore, the computational cost in DAS is less than that in RASTA.

The criteria to recognise relations are described by three representative samples of a *List* relation (Section 4.3.2.1), a *Circumstance* relation (Section 4.3.2.2), and an *Elaboration* relation (Sections 4.3.2.3). The heuristic rules that are used to recognise the remaining relations are given in Appendix 6.

4.3.2.1 List

A *List* relation is a multi-nuclear relation whose elements can be listed. A *List* relation is often considered as a *Sequence* relation if there is an explicit indication of temporal sequence. The necessary conditions for a *List* relation between two text units, $Unit_1$ and $Unit_2$ ($Unit_1$ precedes $Unit_2$) are shown in Table 4.1. The first condition is based on syntactic information to guarantee that the two units are syntactically independent. The second condition in Table 4.1 checks the linkage between the two units by using reiterative and co-reference devices. Syntactic and semantic information determine the subject of these units and their relations. The third condition distinguishes a *List* relation from a *Sequence* relation. The last condition ensures that a *Contrast* relation is not present.

Index	Necessary Condition
1	Two units are two co-ordinate clauses or two sentences.
2	If both units have subjects and do not contain attribution verbs, then these subjects need to meet the following requirement: they must either be the same, identical, synonyms, co-hyponyms, hypernym/hyponym, or the subject of $Unit_2$ is a pronoun or a noun phrase that can replace the subject of $Unit_1$.
3	There is no explicit indication that the event expressed by $Unit_1$ temporally precedes the event expressed by $Unit_2$.
4	The *Contrast* relation is not satisfied.

Table 4.1. Necessary Conditions for the *List* Relation

The heuristic rules for the *List* relation is shown in Table 4.2. Let us apply the criteria to recognise the *List* relation to Example (4.30).

(4.30) [Mr. Cathcart is credited with bringing some basic budgeting to traditionally free-wheeling Kidder.$_{4.30.1}$] [He *also* improved the firm's compliance procedures for trading.$_{4.30.2}$]

In Example (4.30), the cue phrase "*also*" signals a *List* relation between the two sentences (4.30.1) and (4.30.2). Since only the heuristic rule 1 (Table 4.2) is satisfied here, the total-heuristic-score is:

Total-heuristic-score = Actual-score(heuristic rule 1)

= score(heuristic rule 1) * score("*also*").

Index	Heuristic Rule	Score
1	$Unit_2$ contains *List* cue phrases.	100
2	Both units contain enumeration conjunctions (*first, second, etc*).	100
3	Both subjects of $Unit_1$ and $Unit_2$ contain NP cues.	90
4	If both units contain attribution verbs, the subjects of their reported clauses are similar, synonyms, co-hyponyms, or hypernyms/hyponyms.	80
5	If the subjects of two units are co-hyponyms, then the verb phrase of $Unit_2$ must be the same as the verb phrase of $Unit_1$, or $Unit_2$ has the structure "*so + auxiliary + sbj*".	80

Table 4.2. Heuristic Rules for the *List* Relation

The cue phrase "*also*" has the score of 1 for the *List* relation, so the total-heuristic-score is 100*1=100>0. Therefore, the necessary conditions of the *List* relation are checked. Spans (4.30.1) and (4.30.2) are sentences, the first condition is thus satisfied. The subject of text span (4.30.2), "*he*", is a pronoun, which replaces the subject of text span (4.30.1), "*Mr. Cathcart*" (condition 2). There is no evidence of an increasingly temporal sequence (condition 3), and also no signal of a *Contrast* relation (condition 4). Therefore, a *List* relation is posited between text spans (4.30.1) and (4.30.2).

The cue phrase "*and*" is found in Example (4.31):

(4.31)[But the Reagan administration thought otherwise.$_{4.31.1}$] [*and* so may the Bush administration.$_{4.31.2}$]

"*And*" is considered as a cue phrase because it stands at the beginning of clause (4.31.2) (heuristic rule 1). The subjects of two spans, "*the Reagan administration*" and "*the Bush administration*", are co-hyponyms. In addition, clause (4.31.2) has the structure "*so* + *auxiliary* + *sbj*". With the score of 0.8 for the cue phrase "*and*" in the *List* relation, and with the satisfaction of the heuristic rule 5, the total-heuristic-score is:

Total-heuristic-score = Score(heuristic rule 1)*Score("*and*") + Score(heuristic rule 5) = 100*0.8 + 80 = 160 > 0.

As in Example (4.30), the necessary conditions of the *List* relation are checked and then a *List* relation is posited between two elementary discourse units (4.31.1) and (4.31.2).

4.3.2.2 Circumstance

A *Circumstance* relation is a nuclear-satellite relation. In a *Circumstance* relation, the situation presented in the satellite provides the context in which the situation presented in the nucleus should be interpreted. The satellite is not the cause/explanation of the situation presented in the nucleus.

For example:

(4.32) [Some evinced an optimism that had been rewarded][*when* they didn't flee the market in 1987.]

There is no necessary condition for the *Circumstance* relation. The heuristic rules for the *Circumstance* relation between two text units, $Unit_1$ and $Unit_2$ ($Unit_1$ precedes $Unit_2$) are shown in Table 4.3.

The heuristic rule 3 is to distinguish the *Circumstance* relation with the *Manner* relation (see conditions to posit a *Manner* relation in Appendix 6). It is illustrated in Examples (4.33) and (4.34) below:

(4.33) [*Walking slowly*,][we approached the main building.]

(4.34) [*Looking at Susan's face*,][he knew she was terrible angry.]

The adverb "*slowly*" describes the manner of "*walking*". The verb phrase "*walking slowly*" indicates the manner of "*we approached the main building*".

Meanwhile, "*looking at Susan's face*" denotes the circumstance when "*he knew she was terrible angry*". The total-heuristic-score of the *Circumstance* relation in Example (4.34) is:

Total-heuristic-score = Score(heuristic rule 3) = 80 > 0.

Index	Heuristic Rule	Score
1	One unit has *Circumstance* cue phrases.	100
2	The subject of $Unit_2$ contains a NP cue.	90
3	$Unit_1$ is a Verb + ing clause; that verb phrase does not contain any adverb.	80
4	The time of $Unit_2$ covers the time of $Unit_1$.	50

Table 4.3. Heuristic Rules for the *Circumstance* relation

Since the *Circumstance* relation does not require necessary conditions, a *Circumstance* relation is posited between two clauses in Example (4.34), with the total-heuristic-score of 80.

4.3.2.3 Elaboration

An *Elaboration* relation is a nuclear-satellite relation. In an *Elaboration* relation, the satellite gives additional information or detail about the situation presented in the nucleus. This is the most general relation since one span often provides additional information for its previous span. The necessary conditions for the *Elaboration* relation are given in Table 4.4.

Index	Necessary condition
1	Both units are not dominated by and do not contain cue phrases that are compatible with other relations. However, it is still acceptable if the cue phrase signals other relations as well as the *Elaboration* relation.
2	*List, Sequence, Circumstance* relations are not satisfied.

Table 4.4. Necessary Conditions for the *Elaboration* Relation

The heuristic rules for the *Elaboration* relation are given in Table 4.5. The heuristic rule 6 in Table 4.5 means that the *Elaboration* is the default relation when there is a signal of semantic relation between two spans and all other relations are not satisfied. It is checked only when no other heuristic rule is satisfied, or when it is signalled by syntactic information. Let us apply the criteria to recognise the *Elaboration* relation to Example (4.35):

(4.35) [But even on the federal bench, specialisation is creeping in,] [*and* it has become a subject of sharp controversy on the newest federal appeals court.]

Index	Heuristic rule	Score
1	One unit contains *Elaboration* cue phrases.	100
2	The VP of $Unit_2$ contains a VP cue or an attribution verb.	90
3	$Unit_2$ is a clause that is adjacent to the last NP of $Unit_1$ and has the syntactic role of PP, NP, VP, or SBAR.	80
4	The subject or object of $Unit_2$ is a hyponym of the subject or object of $Unit_1$, or the subject|object of $Unit_2$ is the pronoun *some* or contains the modifier *some*.	50
5	The subject or object of $Unit_2$ is a synonyms, co-hyponyms, or the subject of $Unit_2$ is a pronoun or a NP that relates to the subject or object of $Unit_1$.	30
6	There is an indication of a relation between two units (e.g., both units "talk about" the same subject). Also, other relations (except *Joint*) are not satisfied.	30

Table 4.5. Heuristic Rules for the *Elaboration* Relation

DAS considers the word "*but*" as a cue phrase if this word is at the beginning of the right span (see Appendix 3), which does not match with "*but*" in Example (4.35). Therefore, there is only one cue phrase "*and*" in this example. "*And*" signals the *List*, *Sequence*, and *Elaboration* relations. It has the score of 0.8, 0.8, and 0.7 for the *List*, *Sequence*, and *Elaboration* relation respectively. Since the score for the heuristic rule involving cue phrases is 100, the actual scores for the *List*, *Sequence*, and *Elaboration* relations are 80 (=100*0.8), 80 (=100*0.8) and 70 (=100*0.7) respectively. All of these scores are greater than the threshold θ (=30). The two clauses in (4.35) do not satisfy the necessary condition 2 of the *List* and *Sequence* relations (see Necessary Conditions of the *List* and *Sequence* relations in Section 4.3.2.1 and in Appendix 6). The *List* and *Sequence* relations therefore do not hold in Example (4.35). The necessary conditions of the *Elaboration* relation are satisfied. Heuristic rules 2, 3, and 4 of the *Elaboration* relation are not satisfied in Example (4.35). With the score of 0.5 for the cue phrase "*and*" of the *Elaboration* relation, and with the satisfaction of the heuristic rule 1, the total-heuristic-score of the *Elaboration* relation is:

Total-heuristic-score = Actual-score(heuristic rule 2)

= Score(heuristic rule 2) * Score("*and*") = 100 * 0.7 = 70 > θ

Therefore, an *Elaboration* relation with the score of 70 is posited in Example (4.35).

4.4 Procedure to Posit Rhetorical Relations

In order to posit one or several rhetorical relations between spans from the set of 22 relations, it is unnecessary to check all of 22 relations one by one. Instead, DAS starts by detecting recognition factors from texts. If two spans are clauses of a sentence, DAS first checks the syntactic rule that produces the segmentation boundary between these clauses. If no relation can be posited between two clauses by using syntactic information, or if two spans are not in the same sentence, DAS searches for cue phrases from the two spans. If cue phrases are found, DAS checks other conditions of the relations that correspond to these cue phrases.

If cue phrases are not found, DAS searches for other factors, in a decreasing order of heuristic scores. These factors are VP cues, NP cues, syntactic information, and semantic information. When searching for recognition factors, DAS calculates the accumulation score of all heuristic rules contributing to each relation signalled by the factors. If the accumulation score of one relation is higher or equal to the threshold θ, DAS examines necessary conditions of this relation. If the necessary conditions are satisfied, this relation will be posited. If no heuristic rule is satisfied, and there is evidence that two spans are related, an *Elaboration* is posited. Otherwise, if no heuristic rule is satisfied, and no evidence that these spans are related, a *Joint* relation is assigned. Figure 4.3 describes the algorithm to posit relations between spans. A detailed description of this algorithm is presented in Appendix 2.

Input:

- Two non-overlapping spans $Unit_1$ and $Unit_2$ and the syntactic rule that has been used to segment these spans.
- Lists of cue phrases, VP cues, and NP cues.

Output: All possible relation name of the relation between $Unit_1$ and $Unit_2$.

Algorithm:

1. If the input spans are clauses, find a name for the relation using the information of the syntactic rule. If it is found, posit this name.
2. If a relation name has not been assigned, find all cue phrases in $Unit_1$ and $Unit_2$. Compute a total score of all heuristic rules that have been found. If this score >= 0, and necessary conditions of the relation signalled by the cue phrases are satisfied, posit this relation name.
3. If a relation name has not been assigned, detect VP cues and NP cues from Unit1 and Unit2 and perform the same operations as in Step 2.
4. If a relation name has not been assigned, check other heuristic rules of each relation. With each relation, compute a total score of all heuristic rules that have been found. If this score >= 0, and necessary conditions of the relation signalled by the heuristic rules are satisfied, posit this relation name.
5. If a relation name has not been assigned, and there is a signal indicating that $Unit_1$ and $Unit_2$ has a semantic relation, posit an *Elaboration* relation. Otherwise, posit a *Joint* relation.

Figure 4.3. Outline of the Algorithm to Posit Relations Between Spans.

When positing a rhetorical relation between spans, the nuclearity role and the relation score are also assigned to this relation. The semantic relation mentioned in Step 5 is achieved by checking cohesive devices (e.g., word repetition and synonyms).

4.5 Summary and Discussion

This chapter has presented a method of positing rhetorical relations between spans based on several recognition factors. A set of 22 relations has been proposed to be used in analysing rhetorical relations. It is created by grouping relations in the RST-DT corpus according to a specific resemblance. As Mann and Thompson (1988) stated, the set of rhetorical relations can be varied depending on genres and cultural styles. The set of 22 relations used in DAS is enough for our research and for evaluating the result based on the RST-DT corpus. The number of relations in this set can be made smaller by grouping relations that share a number of characteristics into a relation; or it can be made larger by adding more relations into the set. In case of grouping similar relations into one, the easiest way to produce discourse trees is to get the output from DAS and then map the relations from this output to the relation in the new relation set. In case of adding more relations, the heuristic rules of the relations that have some common properties with the new relations need to be modified accordingly. The task of finding recognition factors (time references, substitution words, etc.) and the algorithm to posit relation are still the same.

Beside the traditional cue phrases that have been used in most research on discourse analysis, we exploit new recognition factors, including NP cues and VP cues. Time references, anaphora resolution, substitution words, ellipses, and syntactic information are also investigated in this research. Each heuristic rule, which is an application of these recognition factors to a specific relation, is considered as a piece of evidence that contributes to the recognition process. Each heuristic rule is assigned a score. The combination of these heuristic rules, represented by a total-heuristic-score, decides the relations.

DAS posits a rhetorical relation if all necessary conditions and at least one heuristic rule are satisfied. To recognise a relation, DAS does not check all 22 relations from the relation set. Instead, DAS uses recognition factors from the two spans to propose all possible relations. Only the relations that are signalled by the recognition factors are tested to posit relations between spans.

The problem of generating rhetorical structures of a text from rhetorical relations between text spans is discussed in Chapter 5.

5 Constructing Rhetorical Structures

This chapter concentrates on reducing the search space when constructing rhetorical structures of text, given all possible relations that hold between text spans. The discourse analyser developed in this thesis was inspired by Marcu (2000) and Corston (1998). It concentrates on further reducing the search space and finding the best rhetorical structures (LeThanh et al., 2004b). According to Matthiessen and Thompson (1988), syntactic structures of sentences have a close relation to the sentential rhetorical structures. Therefore, syntactic information is used in DAS to construct the sentential rhetorical structures (see Section 5.2). Based on the syntactic information, only one rhetorical structure is created for a sentence. No hypothetical span combination is created; no combinatorial problem happens; and no searching algorithm is needed to derive a rhetorical structure for a sentence.

In principle, the process of constructing text-level rhetorical structures is the same as that of sentence-level rhetorical structures (i.e., connecting text spans by rhetorical relations to create discourse trees). However, since there is no syntactic information to indicate the syntactic relations between sentences, DAS cannot use syntactic information to construct text-level rhetorical structures. Instead, DAS has to search for the best rhetorical structure covering the entire text from all hypothetical relations between text spans (see Section 5.3).

In order to take advantages of the clausal relations within a sentence, we divide the discourse analyser into two levels: sentence-level and text-level, each of which is processed in a different way. Nevertheless, both analysing levels have to posit rhetorical relations between large text spans. We modify the compositionality criterion of Marcu (2000) in order to take advantage of recognition factors that are situated in the satellite. This recognition process is presented next.

5.1 Positing Relations Between Large Spans

An important task in constructing discourse trees is to posit rhetorical relations between large spans. For example, DAS has to find rhetorical relations between two sentences in Example (5.1), each of which consists of two elementary discourse units.

(5.1) [Some of the associations have recommended Dr. Alan D. Lourie $_{5.1.1}$][who now is associate general counsel with SmithKline Beckman Corp. in Philadelphia. $_{5.1.2}$][Dr. Lourie says $_{5.1.3}$][the Justice Department interviewed him last July. $_{5.1.4}$]

Figure 5.1.a shows the discourse tree that connects two sentences in Example (5.1). The dotted arc connecting these sentences indicates that the nuclearity roles of these sentences have not been posited.

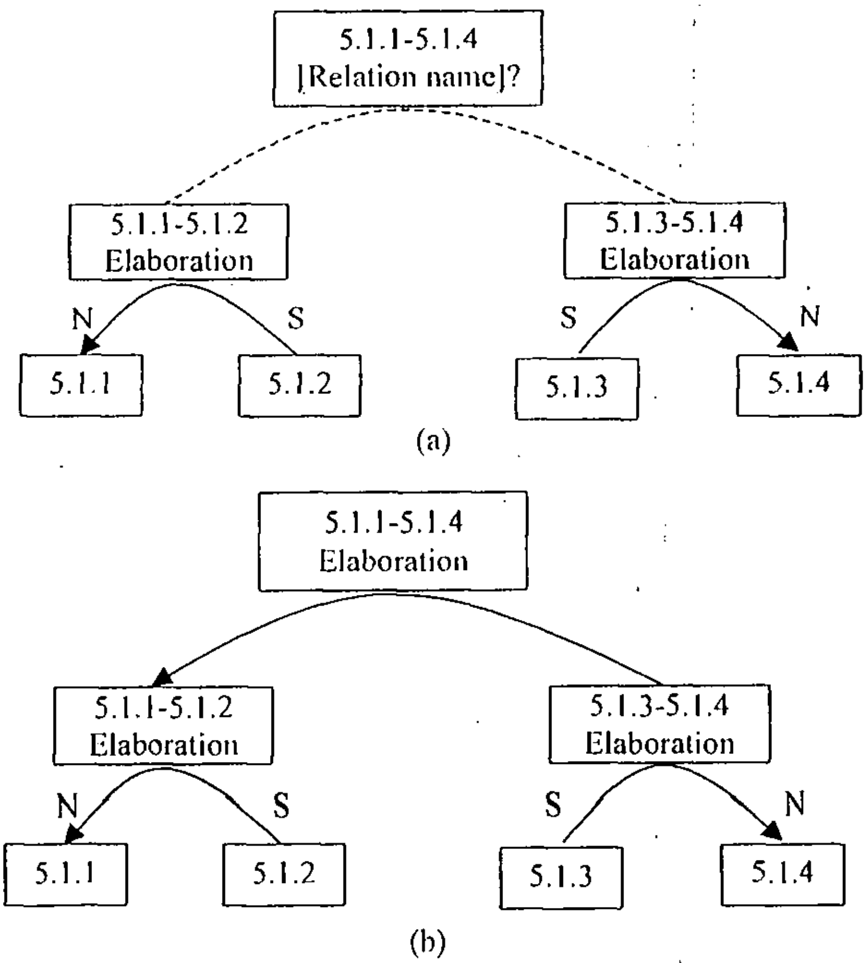


Figure 5.1. Discourse Tree of Example (5.1)

Marcu (2000) explains the rhetorical relations that are held between large spans in terms of the rhetorical relations that are held between elementary discourse units. According to the strong compositionality criterion of Marcu (2000), "*if a* *rhetorical relation R holds between two textual spans of the tree structure of a text, then it can be explained by a similar relation R that holds between at least two of the most important textual units of the constituent spans.*" From this point of view, Marcu analyses relations between large spans by considering only relations between their nuclei.

Let us apply the compositionality criterion proposed by Marcu to Example (5.1). In Example (5.1), the elementary discourse units (5.1.1) and (5.1.4) are the most important units of the first and the second sentences respectively. Therefore, the relation between the two sentences is the relation between (5.1.1) and (5.1.4). Since the span (5.1.4) elaborates the information in the span (5.1.1), an *Elaboration* holds between the spans (5.1.1) and (5.1.4), in which the span (5.1.1) is the nucleus and the span (5.1.4) is the satellite. Consequently, an *Elaboration* holds between spans (5.1.1-5.1.2) and (5.1.3-5.1.4), in which the span (5.1.1-5.1.2) is the nucleus and the span (5.1.3-5.1.4) is the satellite. The dotted arc now becomes a solid arc whose arrow-head points to the span (5.1.1-5.1.2) (see Figure 5.1.b)

Consider the case when one constituent span contains two nuclei (Example 5.2).

(5.2) [Some patent lawyers had hoped $_{5.2.1}$][that such a specialty court would be filled with experts in the field.$_{5.2.2}$][But the Reagan administration thought otherwise,$_{5.2.3}$][and so may the Bush administration.$_{5.2.4}$]

The elementary discourse unit (5.2.2) is the most important unit of the first sentence. Both elementary discourse units (5.2.3) and (5.2.4) have equal important roles in contributing to the discourse relation of the second sentence. Therefore, a relation holds between two sentences in Example (5.2) if it holds either between (5.2.2) and (5.2.3), or between (5.2.2) and (5.2.4). The cue phrase "*but*" signals a *Contrast* relation between (5.2.2) and (5.2.4), a *Contrast* relation thus holds between these spans with the heuristic score of 100 (see Appendix 6). Because of that, a *Contrast* relation holds between the two sentences in Example (5.2) (Figure 5.2.b).

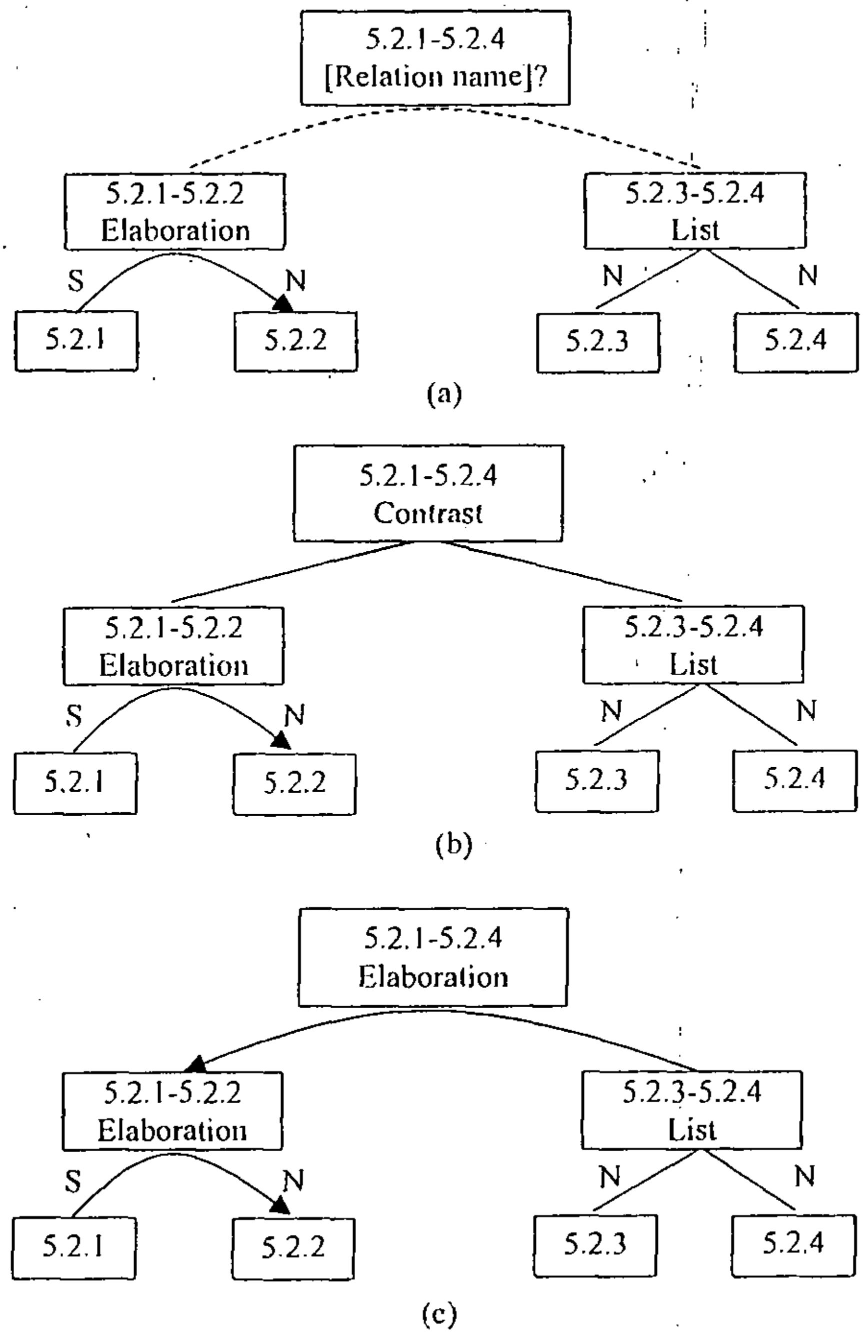


Figure 5.2. Discourse Tree of Example (5.2)

The span (5.2.4), "*and so may the Bush administration*", means the Bush administration did not think that "*such a specialty court would be filled with experts in the field*". Therefore, the context of the span (5.2.4) is also contrast with the context of the span (5.2.2). However, the current recognition factors used in DAS is unable to detect a *Contrast* relation between (5.2.2) and (5.2.4). The word "*and*" in the span (5.2.4) is not considered as a cue phrase in the relation between these spans since it has been used to signal a *List* relation between (5.2.3) and (5.2.4) (see "*Effective scope of cue phrases*" in Section 4.2.1). A default relation "*Elaboration*" is assigned between (5.2.1) and (5.2.4) with the heuristic score of 30 (Figure 5.2.c) (see Section 4.3.2.3). As a result, two relations are posited between two sentences in Example (5.2), a *Contrast* with the heuristic score of 100 and an *Elaboration* with the heuristic score of 30.

The compositionality criterion of Marcu (2000) skips recognition factors from the satellites of the constituent spans, which can also be used to signal relations between large spans. Example (5.3) illustrates this situation. Figure 5.3 shows the discourse tree that connects two sentences in Example (5.3). The name of the rhetorical relation between these sentences has not been recognised.

(5.3) [With investment banking as Kidder's "lead business," where do Kidder's 42-branch brokerage network and its 1,400 brokers fit in?$_{5.3.1}$][*To answer the brokerage question,*$_{5.3.2}$] [Kidder, in typical fashion, completed a task-force study.$_{5.3.3}$]

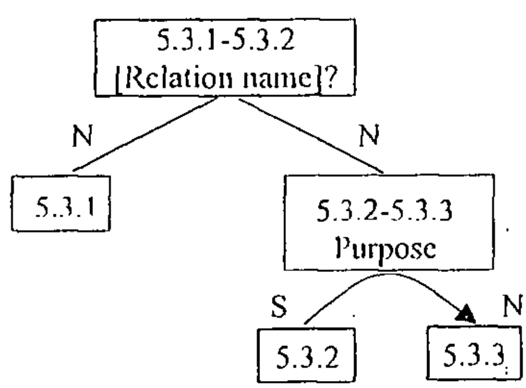


Figure 5.3. Discourse Tree of Example (5.3)

The VP cue "*To (+verb)*" in span (5.3.2) indicates a *Purpose* relation between two clauses (5.3.2) and (5.3.3), in which the span (5.3.2) is the satellite and the span (5.3.3) is the nucleus. The VP cue "*answer*" in the span (5.3.2) indicates a *Solutionhood* relation between two sentences; one is the span (5.3.1), another covers spans (5.3.2) and (5.3.3). If DAS ignores the satellite (5.3.2), it is difficult to recognise the relation that holds between these two sentences.

Example (5.3) shows that although the content of a satellite does not determine rhetorical relations of its parent span, recognition factors that belong to the satellite are still a valuable source. We noticed that the cue phrases, NP cues and VP cues of the left most elementary discourse unit of both large spans can contribute to the relation between the two large spans. Meanwhile, the other cue phrases inside these two spans contribute to the internal rhetorical relations within each large span. For this reason, the first elementary discourse units of the two large spans are always considered by DAS to contribute to the relation. In computing the relation between large spans, DAS does not use only the nuclei of the two large spans as Marcu (2000) did, but also their first elementary discourse units, whether they are nuclei or not.

We apply the compositionality criterion of Marcu and extended it for the case when a satellite stands at the beginning of the large span. To formalise the rules that are used to posit rhetorical relations between large spans, the following definitions are applied:

- <T> represents a span.
- $<T_i\ T_j>$ represents a span that covers two adjacent, non-overlapping spans <Ti> and <Tj>, which are related by a rhetorical relation. The possible roles of $<T_i>$ and $<T_j>$ in this rhetorical relation are Nucleus – Nucleus, Nucleus – Satellite, or Satellite – Nucleus. These states are encoded as $<T_i\ T_j \mid NN>$, $<T_i\ T_j \mid NS>$, and $<T_i\ T_j \mid SN>$, respectively.
- rhet_rels($<T_i>$,$<T_j>$) represents the rhetorical relations between $<T_i>$ and $<T_j>$.

The paradigm rules 1 to 4 in DAS given below are based on the proposition proposed by Marcu (2000).

Rule 1:

rhet_rels($<T_1\ T_2 \mid NS>$, <T>) ≡ rhet_rels($<T_1>$, <T>)

<u>If:</u> there is a relation between two spans $<T_1>$ and $<T_2>$, in which $<T_1>$ is the nucleus and $<T_2>$ is the satellite;

<u>Then:</u> rhetorical relations between span $<T_1\ T_2>$ and its right-adjacent span <T> are the relations that hold between $<T_1>$ and <T>.

Rule 2:

rhet_rels(<T>, $<T_1\ T_2 \mid NS$) ≡ rhet_rels(<T>, $<T_1>$)

<u>If:</u> there is a relation between two spans <T_1> and <T_2>; in which <T_1> is the nucleus and <T_2> is the satellite;

<u>Then:</u> rhetorical relations between span <T> and its right-adjacent span <T_1 T_2> are the relations that hold between <T> and <T_1>.

Rule 3:

rhet_rels(<T_1 T_2 | NN>, <T>) $\equiv$ rhet_rels(<T_1>, <T>) $\cup$ rhet_rels(<T_2>, <T>)

<u>If:</u> there is a relation between two spans <T_1> and <T_2>; both <T_1> and <T_2> are nuclei

<u>Then:</u> rhetorical relations between span <T_1 T_2> and its right-adjacent span <T> are the relations that hold either between <T_1> and <T>, or between <T_2> and <T>.

Rule 4:

rhet_rels(<T>, <T_1 T_2 | NN>) $\equiv$ rhet_rels(<T>, <T_1>) $\cup$ rhet_rels(<T>, <T_2>)

<u>If:</u> there is a relation between two spans <T_1> and <T_2>; and both <T_1> and <T_2> are nuclei

<u>Then:</u> rhetorical relations between span <T_1 T_2> and its left-adjacent span <T> are the relations that hold either between <T> and <T_1>, or between <T> and <T_2>.

In case a satellite stands at the beginning of a large span, we propose a different treatment than the rules reported in Marcu (2000). This situation is formalised as follows:

Rule 5:

rhet_rels(<T>, <T_1 T_2| SN>)

<u>If:</u> there is a relation between two spans <T_1> and <T_2>, in which <T_1> is the satellite and <T_2> is the nucleus;

<u>Then:</u> rhetorical relations between span <T_1 T_2> and its left-adjacent span <T> are either the relations that hold between <T> and <T_2>, or the relations that signal by cue phrases in <T_1>.

To recognise the relations rhet_rels(<T>, <T_1 T_2| SN>), DAS first finds all cue phrases *restCPs* in span <T_1> which have not been used to create the relation between <T_1> and <T_2>, then checks rhet_rels(<T>, <T_1>) by using *restCPs*. If a relation is found, it is assigned to rhet_rels(<T>, <T_1 T_2| SN>). Otherwise, rhet_rels(<T>, <T_1 T_2 | SN>) ≡ rhet_rels(<T> <T_2>).

Applying this rule to Example (5.3) with two spans (5.3.1) and (4.3.2-5.3.3), *restCPs* contains one VP cue "*answer*" since the VP cue "*To*" is used to signal the relation between (5.3.2) and (5.3.3). The relation between (5.3.1) and (5.3.2-5.3.3) is recognised as *Solutionhood* by using the cue "*answer*" in *restCPs*. If DAS uses Marcu's rules, rhet_rels((5.3.1), (5.3.2 5.3.3 | SN)) = rhet_rels((5.3.1), (5.3.3)). That means the VP cue "*answer*" is not considered in Marcu's system.

5.2 Constructing Discourse Trees at the Sentence-level

This module takes the output of the discourse segmenter as the input and generates a discourse tree for each sentence. The discourse segmenter has already generated elementary discourse units and initial rhetorical relations between elementary discourse units (see Chapter 3). The sentence-level discourse analyser only has to posit relation names and the nuclearity roles of discourse units that contribute to each relation. This information is achieved by applying the conversional rules described in Section 5.1 and the relation recognition module described in Section 4.4. Syntactic information and cue phrases are the main recognition factors for the recognition process. For example, the rhetorical relation between a reporting clause and a reported clause in a sentence is an *Elaboration* relation. The reporting clause is the satellite; the reported clause is the nucleus of that relation (see Example 5.4).

(5.4) [She said|| she went to the British library yesterday.]

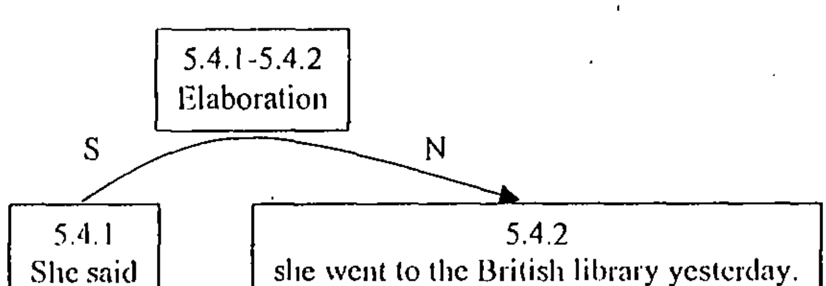


Figure 5.4. Discourse Tree of Example (5.4)

Cue phrases are also used to detect the connection between clauses in a sentence, as in Example (5.5) shown below:

(5.5) [He came late] [*because of* the traffic.]

The cue phrase "*because of*" in Example (5.5) recognises a relation between the clause containing this cue phrase and its left adjacent clause. The clause containing "*because of*" is the satellite of this relation.

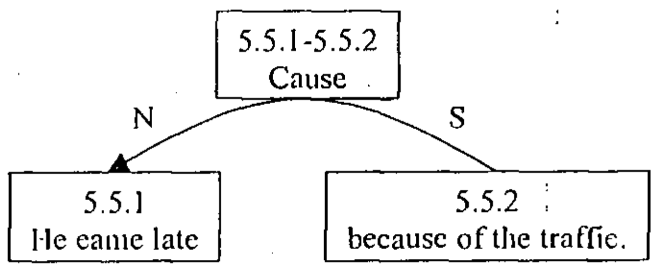


Figure 5.5. Discourse Tree of Example (5.5)

To construct the sentence-level discourse trees, after all relations within a sentence have been posited, all spans that correspond to a sub-tree are replaced by that sub-tree, such as in Example (5.6):

(5.6) [[She knows$_{5.6.1}$] [what time you will come$_{5.6.2}$]] [because I told her yesterday.$_{5.6.3}$]

The discourse segmenter outputs two discourse sub-trees, one with two spans "*She knows*" and "*what time you will come*"; another with two elementary discourse units "*She knows what time you will come*" and "*because I told her yesterday*". DAS combines these two sub-trees into one discourse tree, as shown in Figure 5.6.

With the presented method of constructing sentential discourse trees based on syntactic information and cue phrases, the combinatorial explosion can be prevented while DAS still gets accurate analyses.

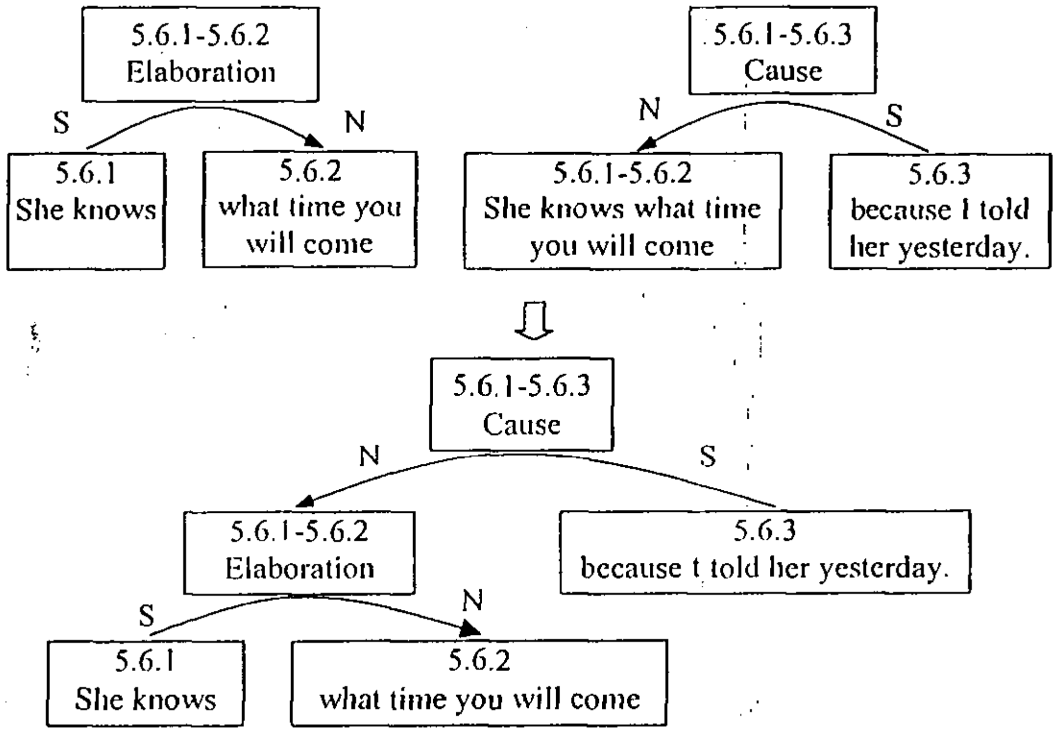


Figure 5.6. Discourse Tree of Example (5.6)

5.3 Constructing Discourse Trees at the Text-level

The discourse tree of a sentence can be constructed with high accuracy based on the syntactic structure and cue phrases of that sentence. It also prevents the combinatorial explosion by using only rhetorical relations that have been generated by the discourse segmenter. Since there is no syntactic structure between sentences, syntactic information cannot be used to determine rhetorical relations outside the scope of a sentence. In order to construct the discourse trees of a text at the text-level, other sources of information should be taken into account. First, constraints about textual organisation and textual adjacency are used to initiate all possible connections between spans (see Section 5.3.1 for the description of these constraints). Then, all possible adjacent rhetorical relations are posited based on different recognition factors mentioned in Section 4.2. The relation recognition procedure is discussed in Section 4.3. Based on these hypothetical relations, the discourse analyser chooses the best combination of relations between text spans to form a discourse tree representing the entire text.

Since a text can have more than one rhetorical structure, we set an upper bound N of rhetorical structures DAS has to generate, which means DAS should generate no more than N structures that best describe the text. The value N is decided by the user.

Section 5.3.1 discusses the approach used in DAS to reduce the search space in deriving best discourse trees representing the entire text, given all possible relations between text spans. The analysing algorithm is introduced in Section 5.3.2.

5.3.1 Search Space Reduction

This section describes the methods used in DAS to reduce the search space in searching for the best combinations of hypothetical relations between text spans, in order to generate discourse trees representing the entire text. The search space reduction is done by using two constrains: textual organisation and textual adjacency, which are introduced in the rest of this section.

The first factor that is used to reduce the search space is the predefined structure of the text, or "*textual organisation*". The application of this factor comes from the fact that each text normally has an organisational framework including sections, sub-sections, paragraphs, etc. to express a communicative goal. Each unit in a text completes an idea, an argument, or a topic that the writer intends to convey. Therefore, each span should have a semantic link to spans in the same textual unit before connecting with spans in a different one. We call it the textual organisational constraint. Based on this idea, in order to generate the rhetorical structure of a text, instead of testing every possible combination of discourse trees, only discourse trees whose spans are in the same text unit (a paragraph, a sub-section) are considered. This strategy reduces the search space significantly, especially with long texts. It is applied in Marcu (2000), but surprisingly not in Corston (1998).

Marcu's (2000) system generates rhetorical structures at each level of granularity (e.g., paragraph, section). The discourse trees at a particular level are used to build the discourse trees at the higher level, until the discourse tree for the entire text is generated. This approach does not optimise calculation time when one wants to derive only some rhetorical structures of the text instead of all of them. Since the discourse analyser does not know the number of discourse trees it should generate for each paragraph or section, it still has to generate all possible trees at each level of granularity.

DAS does not separate levels of granularity in this way. Instead of concentrating on only one level at a time, the entire text participates in the process of deriving rhetorical structures. The levels of granularity are controlled by using a *block-level-score*. One span is always connected with the spans that have the same *block-level-score* before connecting with the spans that have a different one. A detailed description of the *block-level-score* is presented in Section 5.3.2. The discourse analyser completes its task when the required number of rhetorical structures has been generated.

There are two situations that can affect this approach. First, some authors might put two or more topics in one paragraph, or one topic in two or more paragraphs. The use of a *block-level-score* is still effective here since the texts in different units still cannot have a closer semantic relation than the texts within the same one. Second, some texts such as articles on the web or texts written by inexperienced writers may not have structural mark-ups to separate paragraphs and sections. Some texts even may have weak semantic coherence by containing incorrect paragraph boundaries. The use of a *block-level-score* may have a problem here since this situation does not follow the assumption about the organisational framework of texts. However, since most writers create a new paragraph when they start a new argument, we still use the textual format to detect the boundaries of sections or paragraphs. The text segmentation solution is considered for future work.

The second factor used in reducing the search space is the adjacency criterion of rhetorical structures. Since the spans that contribute to a rhetorical relation must be adjacent (Mann and Thompson, 1988), only adjacent spans are considered to be connected in generating new rhetorical relations. This search space is smaller than the search space reported in Marcu (2000) since most discourse trees in his search space connect discourse trees that correspond to non-adjacent spans. Marcu's (2000) system generates all possible trees, and then uses the adjacency constraint to filter the inappropriate ones. We reduce the search space further by applying this constraint earlier, when the candidate solutions are generated, instead of filtering candidates after they are generated. Although Corston (1998) made considerable improvements to reduce the search space in Marcu's (2000) algorithm, his system still contains redundancy since Corston's algorithm does not check this property before generating trees.

To elaborate the efficiency of text adjacency, we make a comparison between DAS search space and the search space of RASTA, created by Corston (1998). Given a set of elementary discourse units, RASTA detects all possible rhetorical relations of every pair of elementary discourse units. These relations are called hypothetical relations or hypotheses. With N elementary discourse units $\{U_1, U_2, ..., U_N\}$, N(N-1) pairs of elementary discourse units $\{(U_1,U_2), (U_1,U_3), ..., (U_1,U_N), (U_2,U_3), ..., (U_{N-1},U_N)\}$ are examined. Then, all combinations of this set are tested in order to build the discourse trees. Meanwhile, DAS only detects rhetorical relations of N-1 pairs of adjacent elementary discourse units (U_1,U_2), (U_2,U_3), ..., (U_{N-1},U_N) and then tests their combination to build discourse trees. Each hypothetical relation has a score, as mentioned in Section 4.3. DAS picks relations from the hypotheses set starting from the highest score to the lowest score.

To illustrate this idea, let us consider a text with four elementary discourse units U_1, U_2, U_3, U_4, and the hypothesis set $H = \{(U_1,U_2), (U_1,U_3), (U_2,U_3), (U_3,U_4)\}$. The set H consists of all possible relations between every pair of elementary discourse units. (U_i,U_j) refers to the hypotheses that involve two elementary discourse units U_i and U_j. Since two elementary discourse units U_1 and U_3 are not adjacent, the hypothesis (U_1,U_3) is not selected by DAS. Figure 5.7 displays the search space for the set H. In this figure, each elementary discourse unit U_i has been replaced by the corresponding number i due to space restriction and for clarity.

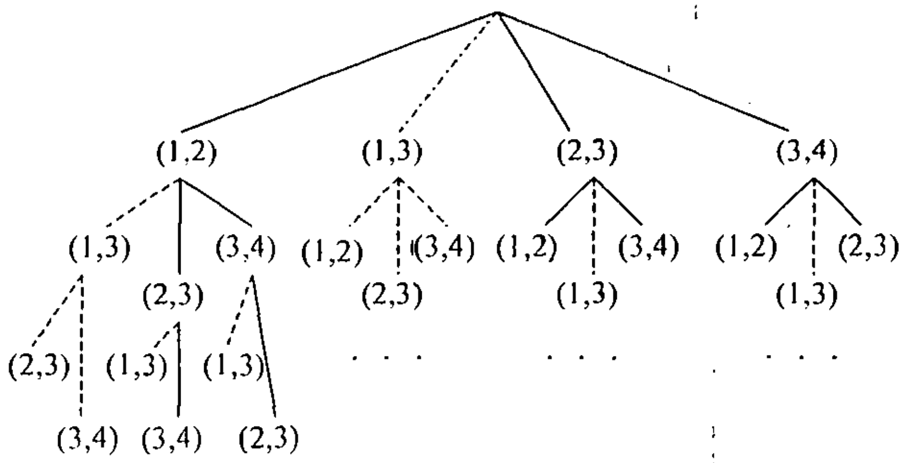


Figure 5.7. Search Spaces For the Hypothesis Set H. RASTA Visits all Branches in the Tree. The Branches Drawn by Dotted Lines Are Pruned by DAS.[18]

Although rhetorical relations between non-adjacent spans are not considered in DAS search space, these relations may be generated during the searching process when they are parts of two larger discourse trees that correspond to adjacent spans. The relations between non-adjacent spans are stored in a hypothesis set in order to be called when they are needed. Figure 5.8 illustrates a situation when the relation between two non-adjacent spans is called. T_1, T_2, T_3, T_4, T_5, T_6 are adjacent spans by this order. Rhet_rels(T_i,T_j) denotes the rhetorical relation between two spans T_i and T_j. T_{1-3}, T_{4-6} are two adjacent spans.

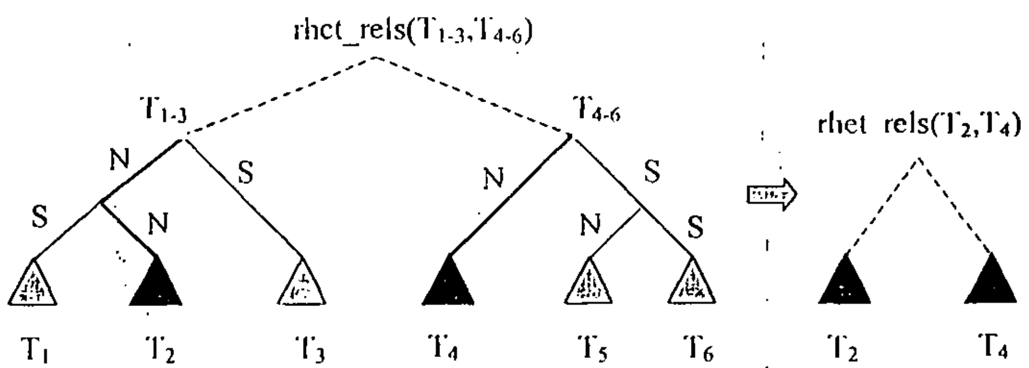


Figure 5.8. A Situation When the Rhetorical Relation Between Two Non-Adjacent Spans Is Called

[18] Due to lack of space, all nodes of this tree cannot be presented together in this figure. The non-displayed nodes are replaced by ". . .".

In Figure 5.8, the relation between the two non-adjacent spans T_2 and T_4 is called when DAS attempts to find the relation between two adjacent spans T_{1-3} and T_{4-6}. If the relation between T_2 and T_4 has not been generated before, it will be posited based on recognition factors given in Section 4.2.

Another problem with RASTA is that one RST tree can be created twice by grouping the same spans in different orders. If derived hypotheses of the set H contain $\{(U_1,U_2),(U_3,U_4)\}$, RASTA will generate two different combinations which create the same tree as shown below:

Connect U_1 and U_2 -> Connect U_3 and U_4 -> Connect (U_1,U_2) and (U_3,U_4).

Connect U_3 and U_4 -> Connect U_1 and U_2 -> Connect (U_3,U_4) and (U_1,U_2).

To deal with the redundancy problem faced by RASTA, DAS updates the hypothesis set every time a new branch on the search tree is visited. When the discourse analyser visits a new branch, the currently visited node is removed from the hypothesis sets, which only stores unvisited branches that are at the same level as the current branch. This action ensures that the algorithm does not create the same RST tree twice.

Let us assume that both RASTA and DAS start from the search space drawn by solid lines in Figure 5.7. DAS search space is explained in more detail using Figure 5.9.

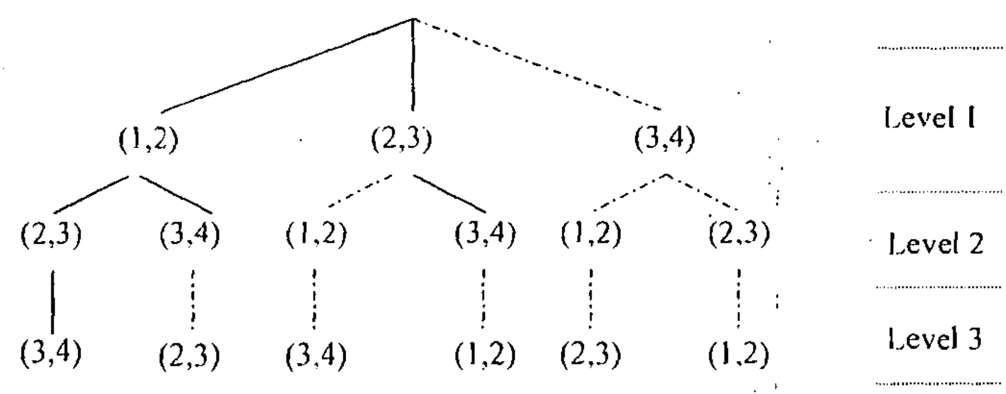


Figure 5.9. Routes Visit by the Two Analysers. RASTA Visits all Branches in the Tree. DAS only Visits the Branches Drawn by Solid Lines.

Firstly, DAS visits the branches that start with node (1,2) at Level 1. After all branches starts with branch (1,2) → (2,3) have been visited (here only one node (1,2) → (2,3) → (3,4)), DAS is going to visit nodes starting with node (3,4) at Level 1. Node (2,3) that belongs to branch (1,2) → (3,4) → (2,3) is removed from DAS search space since all branches that contain two nodes (1,2) and (2,3) have been visited before.

After all RST trees or sub-trees involving the node (1,2) are already visited, this node will not be revisited in the future. DAS removes all branches that contain nodes (1,2) from the hypothesis set of other nodes at the same level as the node (1,2) at Level 1. The branch that connects node (2,3) in Level 1 with node (1,2) in Level 2 is pruned from the search tree. As a result, DAS does not visit the route (2,3) → (1,2) → (3,4). The same reason is applied for other dotted lines in Figure 5.9. This figure shows that DAS search space is much smaller than RASTA search space.

5.3.2 Discourse Analysing Algorithm

The problem of deriving rhetorical structures from a set of hypothetical relations can be considered as the problem of searching for the best solutions for combining rhetorical relations. An algorithm that minimises the search space and maximises the tree quality needs to be found. We apply a beam search, which is an optimisation of the best-first search where only a predetermined number of paths are kept as candidates. The rest of this section will describe this algorithm in detail.

A set called *Subtrees* is used to store sub-trees that have been created during the constructing process. The sub-trees in this set correspond to adjacent and non-overlapping spans. At the beginning, *Subtrees* consists of sentential discourse trees. As sub-trees corresponding to contiguous spans are connected to construct bigger trees, *Subtrees* contains fewer and fewer members. When *Subtrees* contains only one tree, this tree will represent the rhetorical structure of the input text.

All potential relations between adjacent spans that can be used to construct bigger trees at a step (t) form a hypothesis set *PotentialH*. A rhetorical relation created by the system is called a hypothesis. Each relation has a *total-heuristic-score*, which is equal to the total score of heuristic rules that signal the relation as explained in Chapter 4. To control the textual block level (paragraph, section, etc.), each hypothesis is assigned a *block-level-score*, whose value depends on the block level of the spans that participate in the hypothesis. The *block-level-score* and the *heuristic-score* are set in different value-scale so that the combination of sub-trees in the same textual block always has a higher score than that in a different textual block.

- If two sub-trees are in the same paragraph, the relation that connects these sub-trees will have the *block-level-score* = 0. (The paragraph is considered as the lowest block level.)
- If two sub-trees are in different paragraphs, and a value Li is the lowest block level where two sub-trees are in the same unit, the *block-level-score* of the relation corresponding to their parent tree is equal to -1000 * Li. For example, if two sub-trees are in the same section but in different paragraphs; and there is no subsection in this section; then Li is equal to 1. The negative value (-1000) indicates the higher the distance between two spans, the lower the combinatorial priority they get. **The *block-level-score* of a relation is the lowest *block-level-score* among all relations between a sub-tree of the left node and a sub-tree of the right node.** This computation is illustrated in Example (5.7) at the end of this section.

When selecting a hypothesis, the hypothesis with the higher *block-level-score* is preferred. If two or more hypotheses have the same *block-level-score*, the one with higher *total-heuristic-score* is chosen. A variable *total-score* is used to store the sum of the *total-heuristic-score* and the *block-level-score* of a hypothesis.

To simplify the searching algorithm, an *accumulated-score* is used to store the value of the search path. The *accumulated-score* of a path at a step (t) is the highest *predicted-score* of that path at the previous step (t-1). A *predicted-score* of a hypothesis at the step (t) is equal to the sum of the *accumulated-score* of the previous step (t-1) and the *total-score* of the hypothesis. The searching process now becomes the process of searching for the hypothesis with the highest *predicted-score*. The method of calculating the *accumulated-score* and the *predicted-score* are illustrated in Figure 5.10. $h_i(t)$ stands for the hypothesis i at the step (t). $h^*(t-1)$ is the best hypothesis found at the step (t-1) that maximises the *accumulated-score* from the starting point to the step (t-1).

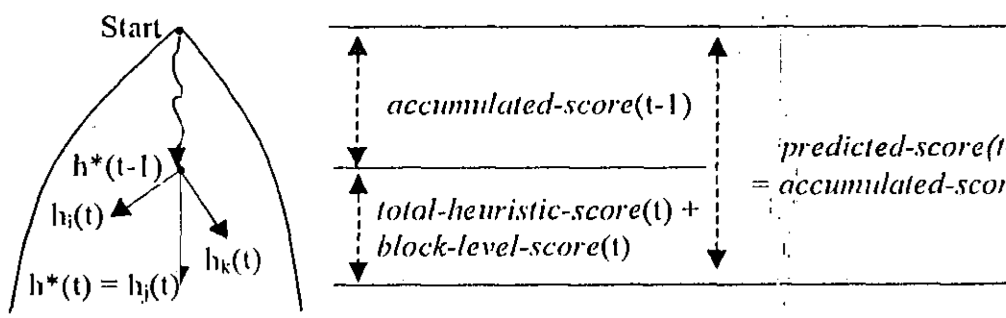


Figure 5.10. Calculating the *accumulated-score* at Time *t*

At each step of the beam search, the most promising node from *PotentialH* is selected. If a hypothesis involving two spans <Ti, Tj> is used, the new sub-tree created by joining the two sub-trees corresponding to spans <Ti> and <Tj> is added to *Subtrees*. The set *Subtrees* is now updated so that it does not contain overlapping discourse trees. The set *PotentialH* is also changed according to the change in *Subtrees*. The relations between the new sub-tree and its adjacent sub-trees in *Subtrees* are created and added to *PotentialH*.

All hypotheses computed by DAS are stored in a hypothesis set called *StoredH*. The use of this set guarantees that a discourse tree will not be created twice. When detecting a relation between two spans, the analyser first looks for this relation in *StoredH* to check whether it has already been created or not. If it is not, it will be generated by a discourse recogniser (see Chapter 4).

DAS limits the branches that the search algorithm can switch to by a constant M. This number is chosen to be 10 since through experiments it was found to be large enough to derive good discourse trees. If at a later stage it was found that this value is insufficient, the only thing DAS needs to do is to increase this value. All other values are updated accordingly. If *Subtrees* contains only one tree, this tree is added to the tree set, *Trees*.[19] This set is used to store the discourse trees that cover the entire text. The searching algorithm terminates when the number of discourse trees in *Trees* is equal to the number of trees required by the user.

Figure 5.11 outlines the main steps of the algorithm to construct rhetorical structures of a text. A detailed description of this algorithm is presented in Appendix 2.

Input:

- Discourse trees of all sentences
- Information about positions of sentences
- The value of N (the number of discourse trees required by the user).

Output:

- Discourse trees that cover the entire text.

Algorithm:

1. *Trees* = {}
2. *Subtrees* = {sentential discourse trees}
3. *accumulated-score* = 0
4. *NewH* = {hypotheses between adjacent sentential discourse trees}
5. *PotentialH* = {M highest total-score hypotheses from NewH}
6. Create M set of *hypoSet[i]* = {*appliedH, accumulated-score, Subtrees, NewH, PotentialH*} by applying each hypothesis of *PotentialH* created by Step 3.
7. Select M highest *predicted-score* hypotheses from M sets of *PotentialH* to be applied (*appliedH*). Create M new set of *hypoSet[i]* = {*appliedH, accumulated-score, Subtrees, NewH, PotentialH*} by applying each of these hypotheses. If a set *Subtrees* contains only one tree, this tree is moved to the set *Trees*.
8. Repeat Step 7 until the number of discourse trees in *Trees* is equal to N or when all *PotentialHs* are empty.

Figure 5.11. Outline of Algorithm for Deriving Text-level Discourse Trees

[19] If no relation is recognised between two discourse sub-trees, a *Joint* relation is assigned. Thus, a discourse tree that covers the entire text can always be found.

In the algorithm given in Figure 5.11, the set *Subtrees* is updated by adding to *Subtrees* the hypothetical relation that has been applied and removing from *Subtrees* the relations whose text spans overlap with the text span of the applying relation. A set *NewH* is used to store the new hypotheses that are created due to the modification of *Subtrees*. The set *PotentialH* is updated by selecting M highest *predicted-score* hypotheses among hypotheses from the old *PotentialH* and the new created set *NewH*.

To demonstrate the working process of the algorithm given in Figure 5.11, let us consider the following example:

(5.7) [In an age of specialization, the federal judiciary is one of the last bastions of the generalist.$_1$][A judge must jump from murder to antitrust cases, from arson to securities fraud, without missing a beat.$_2$][But even on the federal bench, specialization is creeping in, and it has become a subject of sharp controversy on the newest federal appeals court.$_3$]

[The Court of Appeals for the Federal Circuit was created in 1982 to serve, among other things, as the court of last resort for most patent disputes.$_4$][Previously, patent cases moved through the court system to one of the 12 circuit appeals courts.$_5$][There, judges who saw few such cases and had no experience in the field grappled with some of the most technical and complex disputes imaginable.$_6$]

For the convenience of discussion, each tree node is represented by a set of five properties:

- *From*: the begin position of the span of the tree node, represented by a sentence number.
- *To*: the end position of the span of the tree node, represented by a sentence number.
- *Relationname*: the name of the rhetorical relation.
- *Total-score*: the total score of the relation.

- *Predicted-score*: this score is used for the node in *PotentialH* only. It is used to choose the hypothesis for the next round.

The input to the text-level discourse analyser is the rhetorical structure of all sentences from the text, and information about the positions of sentences in the text. The text-level discourse analyser has to find rhetorical relations between these sentences. In the rest of this section, we will describe the process of the analyser in deriving the text-level rhetorical structures of Example (5.7).

In describing the process of the discourse analyser for Example (5.7), we use the following simplifications:

- The sentential rhetorical structures are not mentioned here.
- Each sentence in Example (5.7) is labelled as a number.
- The value M (the number of the branches that the beam search can switch to) is set to 5. The value N (the number of discourse trees) is set to 4.
- The information of the tree nodes that are created in previous steps are simplified by displaying only the name of its left and right nodes.
- The relations *Contrast*, *Circumstance*, and *Elaboration* are abbreviated to *Cons*, *Cir*, and *Ela*.

At the beginning:

- Trees = {} (Step 1).
- Subtrees = {1,2,3,4,5,6} (Step 2).
- accumulated-score = 0 (Step 3).

DAS detects all relations between adjacent sentences and puts it in *NewH*, which are shown in Table 5.1 (Step 4). In this table, the indexes of spans that participate in a relation (the 1st column), relation names (the 2nd column), the heuristic rules that have been applied to posit a relation (the 3rd column), and scores of relations (the 4th column to the 7th column) are present.

Pair	Relation name	Heuristic rule	Cue phrase and cue phrase's score	heuristics-score	block-level-score	total-score
1,2	Ela	5		30	0	30
2,3	Cont	1	but(1)	100	0	100
3,4	Ela	6		30	-1000	-970
4,5	Cir	1	previously(1)	100	0	100
5,6	Ela	5		30	0	30

Table 5.1. Rhetorical Relations in *NewH*

The heuristic rule 5 of the *Elaboration* relation exists between sentences 1 and 2 since the noun phrases of these sentences, "*the federal judiciary*" and "*a judge*" are related to each other by their semantic meanings. The heuristics-score of this pair is 30. Since these sentences are in the same paragraph, they have a *block-level-score* of 0. The *total-score* of this relation is 30 + 0 = 30. An *Elaboration* relation is assigned between these sentences with the score 30. A *Contrast* relation with a score 100 is posited between sentences 2 and 3 based on the appearance of the cue phrase "*but*" at the beginning of sentence 3. Sentences 3 and 4 are in different paragraphs, thus the *block-level-score* of the relation between them is -1000. Both sentences "talk about" the court, thus the heuristic rule 6 of the *Elaboration* relation is satisfied in this case. Their total score is the sum of their *heuristics-score* (30) and their *block-level-score* (-1000), which is equal to -970. Similarly, DAS posits a *Circumstance*[20] relation between sentences 4 and 5, with a score 100 of the cue phrase "*previously*"; and an *Elaboration* relation between sentences 5 and 6 with a score 30 of the heuristic rule 5 of the *Elaboration* relation.

[20] The *Background* relation is merged with the *Circumstance* relation in DAS.

For the convenience of the reader, each hypothesis of *NewH* is represented by a set of four properties: left span, right span, relation name, and total-score. Each hypothesis of *PotentialH* is represented by a set of five properties: left span, right span, relation name, total-score, and predicted-score.

The set *NewH* now contains the following relations:

(2,3,Cont,100), (4,5,Cir,100), (1,2,Ela,30), (5,6, Ela,30), (3,4,Ela,-970).

These relations are put into *PotentialH* (Step 5). Since the accumulated-score is now 0, the *predicted-score* of each hypothesis in *PotentialH* is equal to its total-score.

PotentialH = {(2,3,Cont,100,100), (4,5,Cir,100,100), (1,2,Ela,30,30), (5,6,Ela,30,30), (3,4,Ela,-970,-970)}

Each hypothesis in *PotentialH* is now used to create a *hypoSet*, which is shown in Table 5.2 (Step 6).

hypo Set	**appliedH**	**Subtrees**	**NewH**	**PotentialH**
1	(2,3,Cont, 100,100)	1,(2,3,Cont,100,100), 4,5,6	(1,(2,3),Ela,30), ((2,3),4,Ela,-970)	(4,5,Cir,100,200),(1,(2,3),Ela,30,130),(5,6,Ela,30,130),((2,3),4,Ela,-970,-870)
2	(4,5,Cir,100,100)	1,2,3,(4,5,Cir,100,100),6	(3,(4,5),Ela,-970), ((4,5),6,Ela,30)	(1,2,Ela,30,130),((4,5),6,Ela,30,130),(3,(4,5),Ela,-970,-870)
3	(1,2,Ela,30,30)	(1,2,Ela,30,30),3,4,5,6	((1,2),3,Cont,100)	((1,2),3,Cont,100,130),(5,6,Ela,30,60),(3,4,Ela,-970,-940)
4	(5,6,Ela,30,30)	1,2,3,4,(5,6,Ela,30,30)	(4,(5,6),Cir,100)	(4,(5,6),Cir,100,130),(3,4,Ela,-970,-940),
5	(3,4,Ela,-970,-970)	1,2,(3,4,Ela,-970,-970),5,6	(2,(3,4),Cont,100) ((3,4),5,Cir,100)	(2,(3,4),Cont,100,-870), ((3,4),5,Cir,100,-870)

Table 5.2. Analysing Process – Round 1

When the hypothesis (2,3,Cont,100,100) of the initial *PotentialH* created in Step 5 is selected, this hypothesis is added to *Subtrees* of *hypoSet[1]*. The overlapping sub-trees 2 and 3 are removed from the *Subtrees* (see *Subtrees* in line 1 of Table 5.2). The *NewH* of *hypoSet[1]* in Table 5.2 now contains two new hypotheses (1,(2,3),Ela,30) and ((2,3),4,Ela,-970) (Step 7 in Figure 5.11 or Step 9.2 in Figure A2.3 of Appendix 2). They are added to the *PotentialH* of *hypoSet[1]* in Table 5.2. The *predicted-score* of all hypotheses in the *PotentialH* of *hypoSet[1]* in Table 5.2 is calculated. For example, the new *predicted-score* of the old hypothesis (4,5,Cir,100,100) in the initial *PotentialH* created in Step 5 is now equal to

$$\begin{aligned}\text{predicted-score(hypothesis)} &= \text{accumulated-score} + \text{total-score(hypothesis)} \\ &= 100 + 100 = 200,\end{aligned}$$

since the *accumulated-score* after applying the hypothesis (2,3,Cont,100,100) is 100 (the *accumulated-score* here is equal to the *predicted-score* of the *appliedH*), and the *total-score* of the hypothesis (4,5,Cir,100,100) is 100.

hypo Set	appliedH	Subtrees	NewH	PotentialH
1	(4,5,Cir,100,200)	1,(2,3,Cont,100,100),(4,5,Cir,100,200),6	((2,3),(4,5),Ela,-970),((4,5),6,Ela,30)	(1,(2,3),Ela,30,230),((2,3),(4,5),Ela,-970,-770),((4,5),6,Ela,30,230)
2	(1,(2,3),Ela,30,130)	(1,(2,3,Cont,100,100),Ela,30,130),4,5,6	(1,(2,3)),4,Ela,-970)	(5,6,Ela,30,160),(1,(2,3)),4,Ela,-970,-840)
3	(5,6,Ela,30,130)	1,(2,3,Cont,100,100),4,(5,6,Ela,30,130)	(4,(5,6),Cir,100)	(4,(5,6),Cir,100,230),((2,3),4,Ela,-970,-840)
4	(1,2,Ela,30,130)	(1,2,Ela,30,130),3,(4,5,Cir,100,100),6	((1,2),3,Cont,100)	((1,2),3,Cont,100,230),((4,5),6,Ela,30,160),(3,(4,5),Ela,-970,-840)
5	((1,2),3,Cont,100,130)	((1,2,Ela,30,30),3,Cont,100,130),4,5,6	(((1,2,Ela,30),3,Cont,100,130),4,Ela,-970)	(5,6,Ela,30,160),(((1,2,Ela,30),3,Cont,100,130),4,Ela,-970,-840)

Table 5.3. Analysing Process – Round 2.

Since *PotentialHs* in Table 5.2 are not empty, DAS repeats Step 7 of the analysing algorithm. DAS selects the five highest *predicted-score* hypotheses from all *PotentialHs* in Table 5.2 to create five new *hypoSets* (which are given in Table 5.3). For each hypothesis selected in Table 5.2, a new *hypoSet* is created by updating the *Subtrees*, the *NewH*, and the *PotentialH* corresponding to that hypothesis. The score of the remaining hypotheses in the *PotentialH* of the *hypoSet* concerned is then updated. The new *hypoSets* are used for the next round.

When the hypothesis (4,5,Cir,100,200) of the *PotentialH* of *hypoSet[1]* in Table 5.2 is selected, this hypothesis is added to the *Subtrees* of *hypoSet[1]*. The sub-trees 4 and 5 are removed from the *Subtrees* (see *Subtrees* in line 1 of Table 5.3). The *NewH* of *hypoSet[1]* in Table 5.3 now contains two new hypotheses ((2,3),(4,5),Ela,-970) and ((4,5),6, Ela,30) (Step 7 in Figure 5.11 or Step 9.2 in Figure A2.3 of Appendix 2). These hypotheses are added to the *PotentialH* of *hypoSet[1]* in Table 5.3. The *predicted-score* of the remaining hypotheses in the *PotentialH* of *hypoSet[1]* in Table 5.2 is updated. For example, the new *predicted-score* of the old hypothesis (1,(2,3),Ela,30,130) in the *PotentialH* of *hypoSet[1]* in Table 5.2 is now equal to

$$\begin{aligned}\text{predicted-score(hypothesis)} &= \text{accumulated-score} + \text{total-score(hypothesis)}\\ &= 200 + 30 = 230,\end{aligned}$$

since the *accumulated-score* after applying the hypothesis (4,5,Cir,100,200) is 200 and the *total-score* of the hypothesis (1,(2,3),Ela,30,130) is 30.

After all *hypoSets* have been updated, DAS starts a new round by repeating Step 7 until all *PotentialHs* are empty, or when four discourse trees have been generated by the discourse analyser. The five highest *predicted-score* hypotheses from all *PotentialHs* in Table 5.3 are now selected. The *hypoSets* that correspond to these hypotheses are shown in Table 5.4.

hypoSet	appliedH	Subtrees	NewH	PotentialH
1	(1,(2,3),Ela,30,230)	(1,(2,3,Cont,100,100),Ela,30,230),(4,5,Cir,100,200),6	((1,(2,3)),(4,5),Ela,-970)	((4,5),6,Ela,30,260),((1,(2,3)),(4,5),Ela,-970,-740)
2	((4,5),6,Ela,30,230)	1,(2,3,Cont,100,100),((4,5,Cir,100,200),6,Ela,30,230)	((2,3),((4,5),6),Ela,-970)	((2,3),((4,5),6),Ela,-970,-740)
3	(4,(5,6),Cir,100,230)	1,(2,3,Cont,100,100),(4,(5,6,Ela,30,130),Cir,100,230)	((2,3),(4,(5,6)),Ela,-970)	((2,3),(4,(5,6)),Ela,-970,-740)
4	((1,2),3,Cont,100,230)	((1,2,Ela,30,130),3,Cont,100,230),(4,5,Cir,100,100),6	(((1,2),3),(4,5),Ela,-970)	((4,5),6,Ela,30,260),(((1,2),3),(4,5),Ela,-970,-740)
5	(5,6,Ela,30,160)	((1,2,Ela,30,30),3,Cont,100,130),4,(5,6,Ela,30,160)	(4,(5,6,Ela,30,160),Cir,100)	(4,(5,6),Cir,100,260),(((1,2),3),4,Ela,-970,-810)

Table 5.4. Analysing Process – Round 3

Again, the five highest *predicted-score* hypotheses are selected from all *PotentialHs* in Table 5.4. New *hypoSets* generated by applying these hypotheses are shown in Table 5.5. Line 4 of Table 5.5 shows that the analyser creates a new hypothesis that connects span (1-5) (covering sentences 1 to 5) and span (6) (see the *NewH* of *hypoSet[4]*). In the left node of the tree that corresponds to this hypothesis, span (1-3) is the nucleus; span (4-5) is the satellite in an *Elaboration* relation. In the relation between span (1) and span (2-3), span (1) is the nucleus, span (2-3) is the satellite. Therefore,

rhet_rels((<1-3><4-5>|NS),<6>) = rhet_rels(<1-3>,<6>)
= rhet_rels((<1>,<2-3>|NS), <6>) = rhet_rels(<1>,<6>)
= {*Elaboration*, 30, NN}.

Spans (4-5) and (6) are in the same paragraph, thus the *block-level-score* of rhet_rels(<4-5>,<6>) is 0. Rhet_rels(<1-3>,<6>) have the *block-level-score* of -1000 since spans (1-3) and (6) are in different paragraphs. The *block-level-score* of rhet_rels((<1-3><4-5>|NS),<6>) is the minimum value of these *block-level-score* (0 and -1000), which is -1000. The *total-score* of the new hypothesis of *hypoSet[4]* in Table 5.5 is

total-score(hypothesis) = 30 + (-1000) = -970.

The predicted-score of the hypothesis (((1,(2,3)),(4,5)),6,Ela,-970) is:

predicted-score(hypothesis) = accumulated-score + total-score(hypothesis)
= (-740) + (-970) = -1710,

since the *accumulated-score* after applying the hypothesis ((1,(2,3)),(4,5),Ela,-970,-740) is -740 and the *total-score* of the hypothesis (((1,(2,3)),(4,5)),6,Ela,-970) is -970.

hypoSet	**appliedH**	**Subtrees**	**NewH**	**PotentialH**
1	((4,5),6,Ela,30,260)	(1,(2,3,Cont,100,100),Ela,30,230),((4,5,Cir,100,200),6,Ela,30,260)	((1,(2,3)),((4,5),6),Ela,-970)	((1,(2,3)),((4,5),6),Ela,-970,-710)
2	((4,5),6,Ela,30,260)	((1,2,Ela,30,130),3,Cont,100,230),((4,5,Cir,100,100),6,Ela,30,260)	(((1,2),3),((4,5),6),Ela,-970)	(((1,2),3),((4,5),6),Ela,-970,-710)
3	(4,(5,6),Cir,100,260)	((1,2,Ela,30,30),3,Cont,100,130),(4,(5,6,Ela,30,160),Cir,100,260)	(((1,2),3),(4,(5,6)),Ela,-970)	(((1,2),3),(4,(5,6)),Ela,-970,-710)
4	((1,(2,3)),(4,5),Ela,-970,-740)	((1,(2,3,Cont,100,100),Ela,30,230),(4,5,Cir,100,200),Ela,-970,-740),6	(((1,(2,3)),(4,5)),6,Ela,-970)	(((1,(2,3)),(4,5)),6,Ela,-970,-1710)
5	((2,3),((4,5),6),Ela,-970,-740)	1,((2,3,Cont,100,100),((4,5,Cir,100,200),6,Ela,230),Ela,-970,-740)	(1,((2,3),((4,5),6)),Ela,-970)	(1,((2,3),((4,5),6)),Ela,-970,-1710)

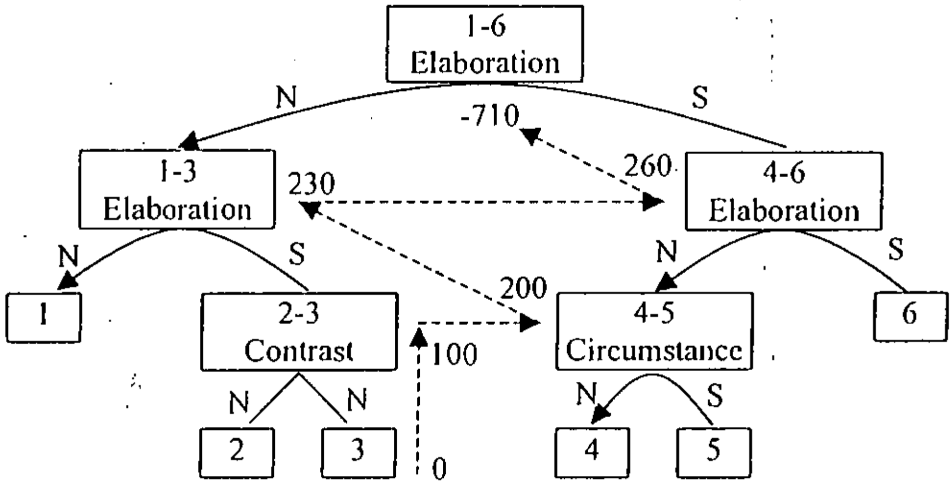
Table 5.5. Analysing Process – Round 4

The five highest *predicted-score* hypotheses are selected from all *PotentialHs* in Table 5.5. By using these hypotheses, the *hypoSets* in Table 5.6 are generated.

hypo Set	appliedH	Subtrees	NewH	PotentialH
1	((1,(2,3)),((4,5),6) ,Ela,-970,-710)	((1,(2,3,Cont,100,100),Ela,30,230),((4,5, Cir,100,200),6,Ela,30,260),Ela,-970,-710)		
2	(((1,2),3),((4,5),6) ,Ela,-970,-710)	(((1,2,Ela,30,130),3,Cont,100,230),((4,5, Cir,100,100),6,Ela,30,260),Ela,-970,-710)		
3	(((1,2),3),(4,(5,6)) ,Ela,-970,-710)	(((1,2,Ela,30,30),3,Cont,100,130),(4,(5,6, Ela,30,160),Cir,100,260),Ela,-970,-710)		
4	(((1,(2,3)),(4,5)),6 ,Ela,-970,-1710)	(((1,(2,3,Cont,100,100),Ela,30,230),(4,5, Cir,100,200),Ela,-970,-740),6,Ela,-970,- 1710)		
5	(1,((2,3),((4,5),6)) ,Ela,-970,-1710)	<stop here>		

Table 5.6. Analysing Process – Round 5

After the fourth discourse tree that covers the entire text has been derived by applying the *appliedH* of *hypoSet[4]* in Table 5.6, the analyser ends its process. No further action is done with Set 5. The four discourse trees generated by the analyser are shown in Figure 5.12. These trees are derived when the *appliedHs* of *hypoSet[1]*, *hypoSet[2]*, *hypoSet[3]*, and *hypoSet[4]* of Table 5.6 are used.



(a)

Figure 5.12. Discourse Trees Generated by the Discourse Analyser

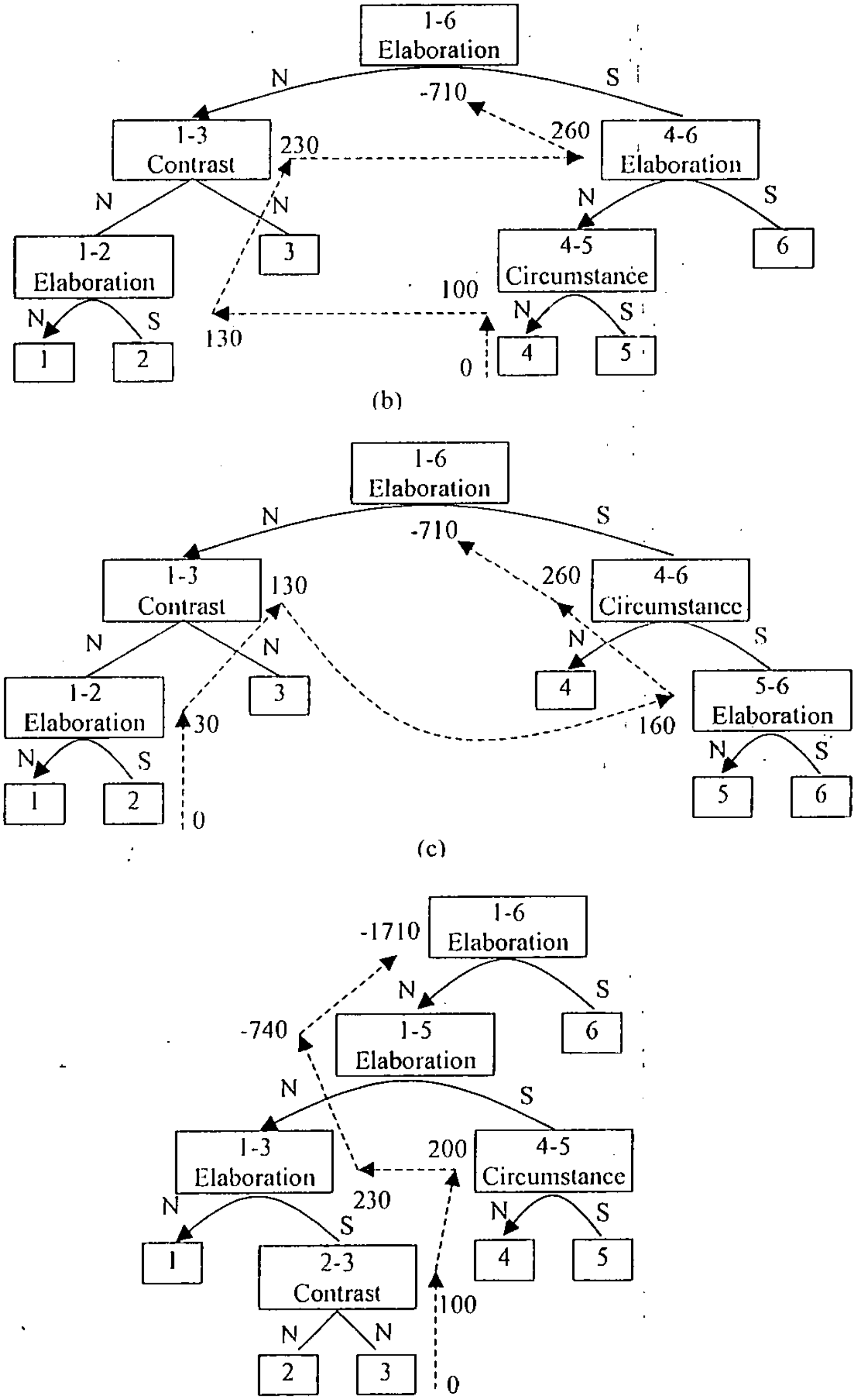


Figure 5.12. Discourse Trees Generated by the Discourse Analyser (con't)

In Figure 5.12, the dotted lines represent the order in which tree nodes are created. The *accumulated-scores* during the process are shown at the end points of these lines. The discourse tree (d) is the least preferred tree among the four trees in Figure 5.12 since it has the lowest *accumulated-score*. This is because sentences in different paragraphs are connected before sentences in the same paragraph. As such it is not correct from a linguistics point of view. DAS generates three trees (a), (b), and (c) with the same score. The text-level rhetorical structure of Example (5.7) from the RST-DT corpus is the tree in Figure 5.12.c.

The complete discourse trees of the text in Example (5.7) are created by replacing each leaf of the text-level discourse tree by the corresponding sentence-level rhetorical structure.

5.4 Summary

The discourse analyser presented in this chapter is divided into two levels: sentence-level and discourse-level, which are processed in different manners. In order to take advantage of syntactic structures, the sentence-level discourse analyser takes as its input discourse segments generated by a method presented in Chapter 3 and initial information about rhetorical relations between clauses. The syntactic information and cue phrases help the process of generating sentential discourse trees to be done simply and accurately. The main draw back of this approach is that it depends on a set of predefined rules, which may create incorrect discourse trees in some exceptional cases. A training method for learning discourse rules from a corpus is a solution to this problem.

Generating text-level rhetorical structures is more complicated than the sentence-level ones. The text-level discourse analyser involves many more discourse segments than the sentence-level one; most of them do not have an explicit signal of relation. We extended Marcu's (2000) rule set, which is used to posit relations between large spans, so that recognition factors from the satellite can contribute to the relation. Based on the rules mentioned above, the hypothetical relations between large spans are created and combined to form discourse trees. The computational explosion problem in searching for well-formed discourse trees is solved by applying a beam search and constraints about textual organisation and text adjacency. Scores are assigned to each discourse tree, so that the analyser can choose the best ones that represent the input text.

The next chapter describes our experiments and evaluates experimental results. We also compare DAS performance with the performance of existing discourse systems.

6 Evaluation

We propose a method to evaluate the output of a discourse system using precision, recall, and F-score on seven levels of processing (LeThanh et al., 2004b). This method is introduced in Section 6.1. Section 6.1 also describes the experiments carried out and presents the results achieved so far. Section 6.2 analyses the performance of DAS at different tasks and compares them with existing discourse systems. A summary of this chapter is given in Section 6.3.

6.1 Description of the Experiments

The standard information retrieval measurements (precision and recall) were used for evaluation. Precision is the proportion of assignments made that were correct. Recall is the proportion of possible assignments that were actually assigned. We also used F-score, which is a measure combining precision and recall into a single figure. We used the version in which they are weighted equally:

$$F - score = 2 * \frac{precision * recall}{precision + recall}$$

DAS performance is based on the Human Assignments (HA), the System Assignments (SA), and the overlap between them (HSA). This is demonstrated in Table 6.1.

		Human assignments		Total
		Yes	No	
System assignments	Yes	HSA	SA - HSA	SA
	No	HA - HSA		
Total		HA		

Table 6.1. Performance's Measurements

The number of assignments that the analyst considers as correct, but the system does not, is HA – HSA. The number of assignments that the system considers as correct, but the analyst does not, is SA – HSA. Precision and recall are calculated as follows.

$$precision = \frac{HSA}{SA} \qquad recall = \frac{HSA}{HA}$$

We manually trained the system by using 20 documents from the RST Discourse Treebank (RST-DT, 2002), which included ten short documents and ten long ones. The lengths of the documents varied from 30 words to 1284 words. Most sentences in those documents are long and complex. The syntactic information of these documents was taken from the Penn Treebank, which was used as the input to the discourse segmenter. To evaluate the effect of the relation set on the system's performance, we used two sets of relations. The original one consists of 22 rhetorical relations mentioned in Section 4.1. The second set consists of 14 relations, formed by grouping similar relations in the set of 22 into one. The RST-DT corpus, which was created by humans, was used as the standard discourse trees for the evaluation. The n-ary relations in the corpus are converted to binary relations during the evaluation. The accuracy of the output of DAS is measured at seven levels. The output of one process was used as input to the process following it.

- Level 1 - The accuracy of discourse segments. This was calculated by comparing the segment boundaries assigned by the discourse segmenter with the segment boundaries assigned by a human.
- Level 2 - The accuracy of the combination of text spans at the sentence-level. DAS generates a correct combination if it connects the same spans as the human does.
- Level 3 - The accuracy of the nuclearity role of spans at the sentence-level.
- Level 4a - The accuracy of rhetorical relations at the sentence-level (with the set of 22 relations).
- Level 4b - The accuracy of rhetorical relations at the sentence-level (with the set of 14 relations).
- Level 5 - The accuracy of the combination of text spans for the entire text.

- Level 6 - The accuracy of the nuclearity role of spans for the entire text.
- Level 7a - The accuracy of rhetorical relations for the entire text (with the set of 22 relations).
- Level 7b - The accuracy of rhetorical relations for the entire text (with the set of 14 relations).

In order to have an accurate evaluation of the system's performance, we tested the system by carrying out two more experiments, each of which consists of a different set of 20 documents from the RST Discourse Treebank (RST-DT, 2002). The lengths of those documents vary from 29 words to 1432 words. In these experiments, we did not do any modification to the system by observing the RST structures from the corpus that correspond to the input documents. The performances of DAS in all three experiments are shown in Table 6.2. In this table, DAS1, DAS2, and DAS3 represent the system's performance in the first, second, and third experiment respectively.

Level		1	2	3	4a	4b	5	6	7a	7b
DAS1	Precision	88.2	68.4	61.9	53.9	54.6	54.5	47.8	39.6	40.5
	Recall	85.6	64.4	58.3	50.7	51.4	52.9	46.4	38.5	39.3
	F-score	**86.9**	**66.3**	**60.0**	**52.2**	**53.0**	**53.7**	**47.1**	**39.1**	**39.9**
DAS2	Precision	92.2	72.2	63.2	55.1	56.4	56.5	47.9	40.2	41.0
	Recall	90.3	71.0	62.2	54.2	55.5	55.1	46.8	39.2	40.0
	F-score	**91.2**	**71.6**	**62.7**	**54.7**	**55.9**	**55.8**	**47.3**	**39.7**	**40.5**
DAS3	Precision	91.7	68.5	60.7	51.3	52.9	53.8	44.8	36.1	37.4
	Recall	88.5	66.3	58.8	49.6	51.2	52.2	43.4	35	36.3
	F-score	**90.1**	**67.4**	**59.7**	**50.4**	**52.0**	**53.0**	**44.1**	**35.5**	**36.8**

Table 6.2. DAS Performances in Three Experiments

Table 6.2 shows that the performance of DAS is quite stable. Therefore, it is reasonable to take the average of these performances as the real performance of DAS, which can be used to compare with the performance of other discourse systems. The average value of DAS performances is shown in the upper part of Table 6.3.

The performance of the human was considered as the upper bound for DAS performance. This value was obtained by evaluating the inter-agreement in the corpus. That is, we compared the discourse structures of each document annotated by two different human analysers using 53 double-annotated documents from the RST corpus. The differences of these double-annotated documents were used to calculate precision, recall, and F-score. This performance is labelled "Human", it is shown in the lower part of Table 6.3. An evaluation of these performances is presented in Section 6.2.

Level		1	2	3	4a	4b	5	6	7a	7b
DAS	Precision	90.7	69.7	61.9	53.4	54.6	54.9	46.8	38.6	39.6
	Recall	88.1	67.2	59.8	51.5	52.7	53.4	45.5	37.6	38.5
	F-score	**89.4**	**68.4**	**60.8**	**52.4**	**53.6**	**54.2**	**46.2**	**38.1**	**39.1**
Human	Precision	98.7	88.4	82.6	69.2	74.7	73.0	65.9	53.0	57.1
	Recall	98.8	88.1	82.3	68.9	74.4	72.4	65.3	52.5	56.6
	F-score	**98.7**	**88.3**	**82.4**	**69.0**	**74.5**	**72.7**	**65.6**	**52.7**	**56.9**
$\frac{F-score(DAS)}{F-score(Human)} * 100\%$		90.6	77.5	73.8	75.9	71.9	74.6	70.4	72.3	68.7

Table 6.3. DAS Performance Vs. Human Performance

6.2 Discussion

In the experiments carried out in this research, the output of one process was used as input to the process following it. The error of one process is, therefore, the accumulation of the error of the process itself and the error from the previous process. As a result, the accuracy of DAS and that of humans decline as the processing level increases. DAS provides a reliable result at the discourse segmentation level (90.7% precision and 88.1% recall). The system's performance at the sentence-level is acceptable when compared with humans. The low accuracies of DAS for the entire text (46.2% F-score at Level 6 and 38.1% F-score at Level 7a) indicate that the discourse trees generated by DAS are quite different from those in the corpus. The final error of DAS (Levels 7a and 7b) is the accumulation of errors from all processes, starting from the discourse segmenter.

The rest of this section analyses the performance of each module in detail. Section 6.2.1 discusses the factors that reduce the segmenter's performance and compares this performance with the performances of existing discourse segmenters. Section 6.2.2 analyses the impacts of previous processes to the performance of the sentence-level discourse analyser. Section 6.2.2 also compares the performance of DAS' sentence-level analyser with the performance of the best sentence-level analysers that we know of. Finally, Section 6.2.3 discusses the factors that affect the performance of the text-level discourse analyser and compares the accuracy of the final output of DAS with those of the discourse analyser created by Marcu (2000).

6.2.1 Performance of the Discourse Segmenter

A discourse segmenter with high accuracy is very important to the performance of a discourse analyser, since its accuracy affects the performance of all processes that occur afterwards. The discourse segmenter's performance depends on three factors. The first factor is the accuracy of syntactic information. Although most syntactic documents from the Penn Treebank are well-structured, this corpus sometime contains inaccurate analysis, which reduces the performance of this module. For example, consider Example (6.1) below:

(6.1) In the Lilly case, the appeals court broadly construed a federal statute to grant Medtronic, a medical device manufacturer, an exemption to infringe a patent under certain circumstances.

The syntactic structure of this sentence from the Penn Treebank is:

```
( (S (PP-LOC In
          (NP the Lilly case))
    ,
    (NP-SBJ the appeals court)
    (VP (ADVP-MNR broadly)
```

```
        construed
        (S (NP-SBJ a federal statute)
           (VP to
                 (VP grant
                     (NP (NP Medtronic)
                         ,
                         (NP a medical device manufacturer)
                         ,)
                     (NP (NP an exemption)
                         (SBAR (WHADVP-1 0)
                       (S (NP-SBJ *)
                        (VP to
                            (VP infringe
                              (NP a patent)
                              (PP-LOC under
                                   (NP certain
circumstances))
                              (ADVP-MNR *T*-1))))))))))
      .))
```

DAS splits this sentence into two elementary discourse units (see Example 6.2), whereas the RST-DT corpus splits it into three units (see Example 6.3). The elementary discourse units generated by the RST-DT corpus are correct in this case.

(6.2) [In the Lilly case, the appeals court broadly construed a federal statute to grant Medtronic, a medical device manufacturer, an exemption][to infringe a patent under certain circumstances.]

(6.3) [In the Lilly case, the appeals court broadly construed a federal statute][to grant Medtronic, a medical device manufacturer, an exemption][to infringe a patent under certain circumstances.]

The missing segment boundary in Example (6.2), which is between "*a federal statute*" and "*to grant*", relates to the syntactic structure of the sentence from the Penn Treebank. As we see from the syntactic structure of Example (6.1), the text "*a federal statute to grant Medtronic, a medical device manufacturer, an exemption to infringe a patent under certain circumstances*" is tagged as a sentence in the Penn Treebank. This analysis does not follow the normal concept of a sentence: the main verb phrase of a sentence cannot start with "*to*". Because of this syntactic information, DAS does not generate a segment boundary before "*to grant*" since it is not allowed to split the subject and the verb phrase of a sentence. The syntactic information of Example (6.1) indicates that "*to infringe a patent under certain circumstances*" is a clause that belongs to the noun phrase "*an exemption to infringe a patent under certain circumstances*". Therefore, DAS puts a segment boundary between "*an exemption*" and "*to infringe a patent under certain circumstances*". Since incorrect syntactic structures in the Penn Treebank are rare, this factor does not reduce significantly the system's performance.

The second factor that reduces the segmentation performance is the over-segmentation of the RST-DT corpus. Texts in the RST corpus are sometime analysed into very small spans such as the words "*and*" and "*or*" as in Example (6.4), which are not clauses with independent functional integrity.

(6.4) [Every order shall be presented to the President of the United States;]$_{6.4.1}$ [and]$_{6.4.2}$ [before the same shall take effect,]$_{6.4.3}$ [shall be approved by him,]$_{6.4.4}$ [or]$_{6.4.5}$ [being disapproved by him,]$_{6.4.6}$ [shall be repassed by two-thirds of the Senate and House of Representatives.]$_{6.4.7}$

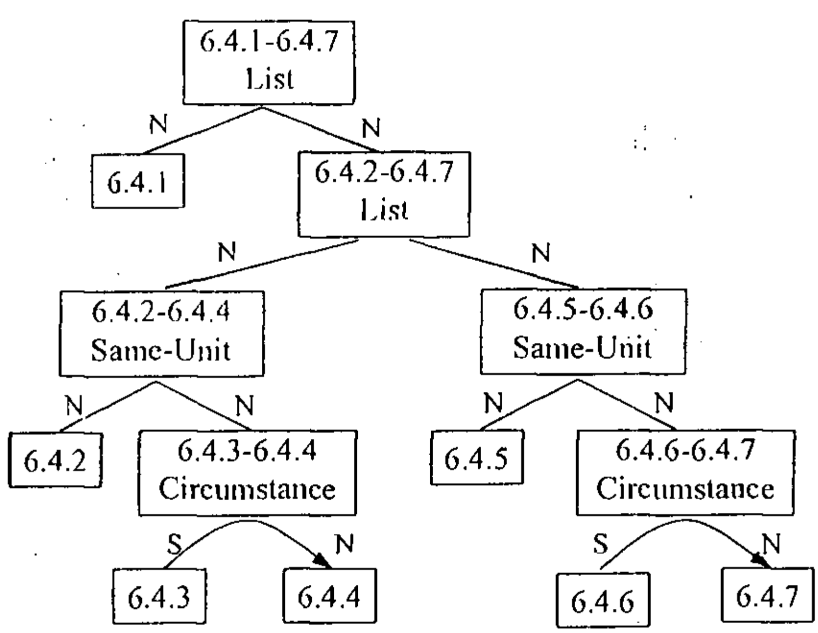


Figure 6.1. Discourse Tree of Example (6.4) Taken From the RST-DT Corpus

The discourse tree of Example (6.4) from the RST-DT corpus is presented in Figure 6.1 above. The cue phrase "*or*" is split from "*being disapproved by him*" by the RST-DT corpus since "*or*" is not semantically involved in the relation between the two elementary discourse units "*being disapproved by him*" and "*shall be repassed by two-thirds of the Senate and House of Representatives*". However, "*or*" belongs to the span "*or being disapproved by him, shall be repassed by two-thirds of the Senate and House of Representatives*" in the relation between it and its left span "*and before the same shall take effect, shall be approved by him*". That is why "*or*", which is span (6.4.5), is reconnected with spans (6.4.6-6.4.7) by a *Same-Unit* relation. The same explanation is applied for the segmentation of the word "*and*" from the span "*before the same shall take effect*" by the RST-DT corpus. We consider this analysis as over-segmentation since it creates many spans smaller than elementary discourse units. DAS prevents this situation by not separating cue phrases from the elementary discourse units that go after them. It is illustrated in Example (6.5) shown below.

(6.5) [Every order shall be presented to the President of the United States; $_{6.5.1}$][and before the same shall take effect, $_{6.5.2}$][shall be approved by him, $_{6.5.3}$][or being disapproved by him, $_{6.5.4}$][shall be repassed by two-thirds of the Senate and House of Representatives. $_{6.5.5}$]

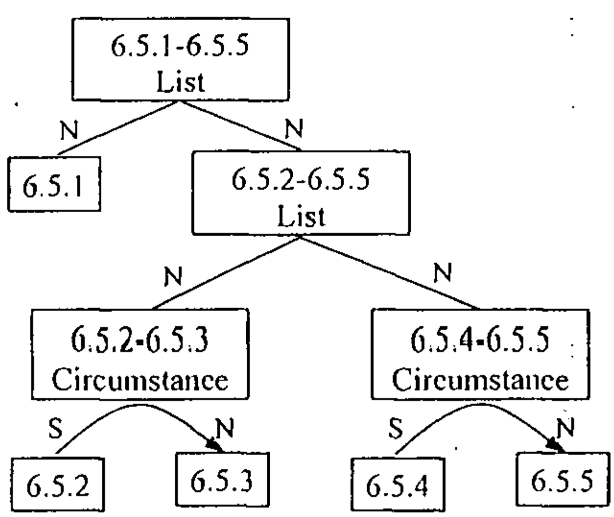


Figure 6.2. Discourse Tree of Example (6.5) Generated by DAS

Figure 6.2 represents the discourse tree of Example (6.5) generated by DAS. This tree is preferred by DAS because its discourse segments are closer to the definition of elementary discourse units (Mann and Thompson, 1988) than the discourse segments of the tree presented in Figure 6.1. Furthermore, this approach reduces the complexity of the discourse tree while rhetorical relations between elementary discourse units are still correct. This treatment results in some differences between the output of DAS and the RST-DT corpus, which means the system's performance is reduced by such cases. We accept this reduction since we do not want DAS to make the text too fragmented by creating many phrases that do not have independent functional integrity. The behaviour of DAS in such cases is supported by other human analysts. An example of such a case is shown in (6.6). This example is received from Mann's (2003) website.

(6.6) [Using thumbs is not the problem $_{6.6.1}$][but heredity is $_{6.6.2}$][and the end result is no use of thumbs $_{6.6.3}$][if I don't do something now. $_{6.6.4}$]

The discourse tree of Example (6.6) is represented in Figure 6.3.

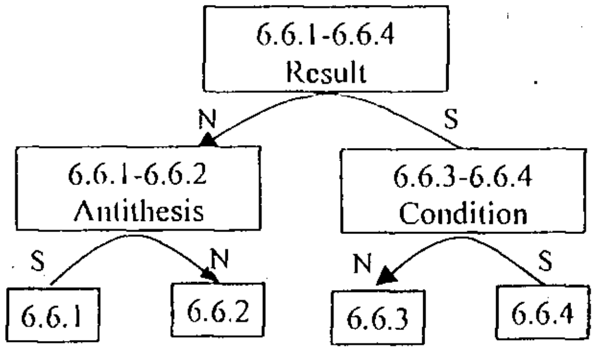


Figure 6.3. Discourse Tree of Example (6.6)

In this example, although the cue phrase "*and*" is not involved in the rhetorical relation between the two elementary discourse units "*the end result is no use of thumbs*" and "*if I don't do something now*", it is not segmented from the elementary discourse unit, "*the end result is no use of thumbs*".

The third factor that affects the system's performance is the segmentation rules. The current rule set used in DAS was created manually based on the segmentation principles proposed by Carlson et al. (2002). In order to get a high performance, a flexible rule set that can adapt to new situations is preferred. This could be achieved by using a machine learning algorithm to learn segmentation rules. This process can be integrated with DAS in future work.

Since the work in this field has not achieved any certainty about the criteria to indicate the exact segment boundaries, and there is no standard benchmark, it is difficult to compare one research result with others. Nonetheless, Okumura and Honda (1994) have carried out experiments on three texts, which are from exam questions in Japanese. The average precision and recall rates of that experiment were 25% and 52% respectively.

Passonneau and Litman (1997) have proposed a series of algorithms for identifying segment boundaries based on various combinations of referential noun phrases, cue phrases, and pauses. Their experiments were carried out on a corpus of spontaneous, narrative monologues. The best algorithm in their series, which combines all the three features, achieved 52% precision and 39% recall. The performance of the discourse segmenter of DAS (90.7% precision and 88.1% recall) is better than this system.

Soricut and Marcu (2003) carried out their experiments on the RST-DT corpus, in which 347 articles were used as the training set and 38 ones were used as the test set. The precision, recall, and F-score of their system when the syntactic information from the Penn Treebank was used as the input are 84.1%, 85.4%, and 84.7% respectively. The discourse segmenter of DAS has a better performance than the one in Soricut and Marcu (2003).

The performance of the discourse segmenter of DAS is promising when compared with other discourse segmenters known to us. It has proved that the combination of sentential syntactic structures and cue phrases are reliable enough for discourse segmentation. However, since the output of the discourse segmenter is used as the input to the later process, this module needs to be as exact as possible. To improve the accuracy of this module, future work includes developing a method for learning segmentation rules, and studying the impact of other factors such as semantic information on discourse segmentation.

6.2.2 Performance of the Sentence-level Discourse Analyser

The discourse segmentation rules discussed in Chapter 3 split text into discourse segments and connect these segments to create rhetorical relations. For this reason, the accuracy of span combinations at the sentence-level (Level 2) depends on the segmentation rules and the post segmenting process. A missing segment boundary can cause several misplaced rhetorical relations. For example, let us reconsider the text in Example (6.1) in the previous section. For the convenience of the reader, we repeat below the discourse segments derived by DAS and the RST-DT corpus for this example as Examples (6.7) and (6.8), respectively.

(6.7) [In the Lilly case, the appeals court broadly construed a federal statute to grant Medtronic, a medical device manufacturer, an exemption $_{6.7.1}$][to infringe a patent under certain circumstances. $_{6.7.2}$]

(6.8) [In the Lilly case, the appeals court broadly construed a federal statute $_{6.8.1}$][to grant Medtronic, a medical device manufacturer, an exemption $_{6.8.2}$][to infringe a patent under certain circumstances. $_{6.8.3}$]

DAS creates a nuclear-satellite relation between the two spans (6.7.1) and (6.7.2) (see Figure 6.4.a). Meanwhile, the corpus assigns a nuclear-satellite relation between the two spans (6.8.2) and (6.8.3), and a nuclear-satellite relation between the span (6.8.1) and the span that covers the two spans (6.8.2) and (6.8.3) (see Figure 6.4.b).

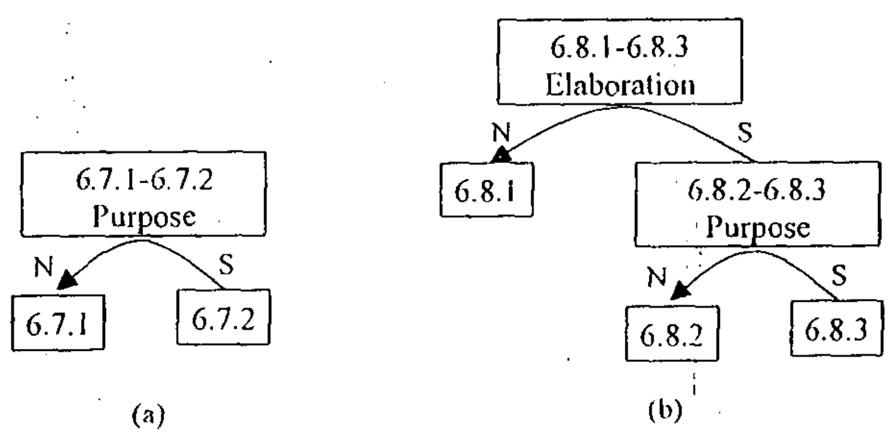


Figure 6.4. Discourse Tree of Examples (6.7) and (6.8)

The corpus does not contain a relation between the two spans "*In the Lilly case, the appeals court broadly construed a federal statute to grant Medtronic, a medical device manufacturer, an exemption*" and "*to infringe a patent under certain circumstances*". Instead, it contains a relation between "*In the Lilly case,* *the appeals court broadly construed a federal statute*" and "*to grant Medtronic, a medical device manufacturer, an exemption to infringe a patent under certain circumstances*". As a result, there is no rhetorical relation shared by DAS and the corpus in this case.

Table 6.3 shows a performance fall from Level 1 to Level 2 (reduction of 21% F-score) which mainly caused by the segmentation disagreement between DAS and the RST-DT corpus. The nuclearity role of spans and the accuracy of rhetorical relations reduce 7.6% F-score from Level 2 to Level 3, 8.4% F-score from Level 3 to Level 4a, and 7.2% F-score from Level 3 to Level 4b. This proves that the factors to recognise rhetorical relations (Chapter 4) are good enough to posit sentential relation properties. Thus, the largest problem in future work at the sentence-level is to improve the accuracy of combination of text spans.

Soricut and Marcu (2003) developed a probabilistic model for the sentence-level discourse parser called SPADE, using the same training set and test set as in their discourse segmentation module. This system provides the best performance among existing sentence-level discourse analysers that we know of. Although SPADE and DAS use the same corpus, it is still difficult to compare the performances of these two systems since the SPADE evaluation uses slightly different criteria than DAS's. Soricut and Marcu compute the accuracy of the sentence-level discourse trees without labels, with 18 labels and with 110 labels. It is not clear how the sentence-level discourse trees are considered as correct. Due to this reason, the performance given by the human annotation agreement reported by them is different than the calculation used in this research. We compared the performance of the two systems using the percentages of the F-scores between the systems and the humans. SPADE performance and human performance calculated by Soricut and Marcu when syntactic trees from the Penn Treebank are used as the input is presented in Table 6.4. For the convenience of the reader, DAS performance is repeated in the lower part of Table 6.4. "Human" and "Human*" in Table 6.4 refer to the human performance of the RST-DT corpus calculated by Soricut and Marcu (2003) and by us respectively.

We consider the evaluation of the "*Unlabeled*" case in Soricut and Marcu's experiment as the evaluation of Level 2 in our experiments. The values shown in Table 6.3 and Table 6.4 imply that the F-score's percentage of DAS performance and the performance of human analysts can be considered as approximate to that of SPADE.

	Unlabeled	110 labels	18 labels	22 labels	14 labels
SPADE	73.0	52.6	56.4		
Human	92.8	71.9	77.0		
$\frac{F-score(SPADE)}{F-score(Human)}*100\%$	78.7	73.2	73.2		
DAS	68.4			52.4	53.6
Human*	88.3			69.0	74.5
$\frac{F-score(DAS)}{F-score(Human^*)}*100\%$	77.5			75.9	71.9

Table 6.4. SPADE Performance, DAS Performance, and Human Performance

6.2.3 Performance of DAS

The text-level discourse analyser constructs discourse trees from sentences. It is independent of the accuracy of elementary discourse units. Instead, it depends on the hypothetical rhetorical relations generated by the relation recognising process (see Section 4.2) and the organisation of the text, which is used for the textual organisational constraint. Some documents from the RST corpus, which are used in the experiments carried out in this research, contain incorrect paragraph boundaries. The textual organisational constraint may create incorrect segment boundaries here since it relies on a well-organised text structure. This problem contributes to the error of the discourse analyser at the text-level. To solve the problem of incorrect paragraph boundaries, we propose to apply a text segmentation approach (e.g., Choi, 2000).

To our knowledge, there is only one report about a discourse analysing system for the entire text that measures accuracy (Marcu, 2000). When training on 20 WSJ documents and testing on 3 WSJ documents from the Penn Treebank, Marcu's decision-tree-based discourse parser receives 21.6% recall and 54.0% precision for the nuclearity; 13.0% recall and 34.3% precision for rhetorical relations. The recall is more important than the precision since we want rhetorical relations that are as correct as possible. Therefore, the discourse analysing system presented in this research shows a significantly better performance. However, more work needs to be done to improve the reliability of the system.

6.3 Summary

In this chapter, we have evaluated the performance of DAS based on different processing levels. The experimental results showed that DAS has a good performance when compared with current discourse analysing systems. Syntactic information and cue phrases are efficient in constructing discourse structures at the sentence-level, especially in discourse segmentation. The performance of the entire DAS system is better than the one created by Marcu (2000), which is the best discourse system that we know of. This chapter also analysed different factors that affect the accuracy of the system, including the errors of the RST-DT corpus and the Penn Treebank corpus, the segmentation rules, the relation recognition rules, and the method of using the textual organisational constraint.

The following chapter summaries the content of this thesis, emphasises the contributions, and delineates possible future work for this thesis.

7 Conclusions

This thesis has concentrated on constructing a system for automatically deriving the rhetorical structure of written texts. While the rhetorical structure has been proved to be useful in many fields of text processing such as text summarisation and information extraction, such discourse systems are difficult to find because discourse analysis is one of the vast and least defined areas in linguistics. Different approaches have been proposed for the linguistic analysis of discourse, from interaction sociolinguistics, and pragmatics to conversation analysis. However, none of these approaches can define a rule set that can automatically derive rhetorical structures. An agreement among researchers about rhetorical structures has not yet been found. For example, Marcu (2000), Forber et al (2003), and Polanyi et al. (2004) have different ways to analyse a text, which result in different rhetorical structures. For this reason, a discourse corpus that is accepted by all researchers does not exist at the time of writing this thesis.

This research follows the Rhetorical Structure Theory (Mann and Thompson, 1988), which has inspired many studies in discourse analysis. The RST-DT corpus (RST-DT, 2002), which was annotated with rhetorical structures in the framework of the RST, was used in the experiments of this research. The system implemented in this research (DAS) takes as its input a written text and its syntactic parsed document and produces as its output binary discourse trees in the framework of RST.

The text is first segmented into elementary discourse units by using sentential syntactic structures and cue phrases. These discourse units are then used to construct the rhetorical structures of the text. The discourse constructer is divided into two levels: sentence-level and text-level. The former constructs the discourse tree for each sentence. Only one rhetorical structure is generated for each sentence, and is based on the segmentation rules. The latter posits rhetorical relations between sentences. The rhetorical structure at the text-level is derived by selecting rhetorical relations to connect adjacent and non-overlapping spans to form a tree that covers the entire text. The constraints of textual organisation and textual adjacency are used in a beam search to generate such rhetorical structures from a set of all possible rhetorical relations in the text.

To posit a rhetorical relation, different recognition factors are used, including cue phrases, NP cues, VP cues, reiterative devices, reference words, time references, substitution words, ellipses, and syntactic information. Each heuristic rule, which corresponds to some recognition factors, is assigned a score, depending on its weight, to decide a relation. The relation with a high total-heuristic-score is preferred over a lower one.

The experimental evaluation on seven levels of analysis showed that this approach provides good performance when compared with current research in discourse analysis. Yet, undoubtedly there are still spaces in this research that need to be improved in order to achieve better results.

A problem rising from the experimental evaluation is the disagreement among researchers on the principles of discourse analysis. There are several trends in discourse analysis, each of which processes a text in a different way (Marcu, 2000; Forbes et al., 2003; Polanyi et al., 2004). Different RST corpora are created accordingly in order to fit with the theory that they have established. Experiments reported in this research used the RST-DT corpus created by Carlson et al. (2002), which is not without problems. Since this corpus is the only available discourse corpus known to us that is created based on the framework of the Rhetorical Structure Theory, it is used in our experiments. However, we believe that our approach can also be adapted to other discourse analysis theories. For example, in order to generate discourse trees following the D-LTAG proposed by Forbes et al. (2003), all processes of DAS can still remain the same, only the node structure of a discourse tree needs to be modified to fit with new method of discourse representation (i.e., the cue phrases and the punctuation marks that are used as anchors to connect spans are stored separately from the spans).

This work has focused on the use of syntax and relatively shallow semantics to construct discourse structure. There is a continuum between syntax and semantics. We have used cue phrases which are towards the syntactic end of the continuum. NP and VP cue phrases are more semantic, and cohesive devices are even more semantic. The system could be improved by the use of even richer semantics, but it is striking how effective this relatively shallow analysis is.

The main contributions of the thesis are summarised in Section 7.1. Section 7.2 addresses future work and proposes some directions to solve the problems in future work.

7.1 Contributions of the Thesis

The approach to discourse analysing presented in this thesis is inspired by Marcu (2000) and Corston (1998). The contributions of this thesis can be summarised as follows:

1. **Proposing new factors for signalling relations between elementary discourse units.** These factors are NP cues and VP cues. The ellipses, which are not used in Marcu (2000) and Corston (1998), are integrated in DAS. Although VP-ellipsis has been investigated in Kehler and Shieber (1997), a discourse system that uses VP-ellipses has not been reported by them. Besides VP-ellipses, other types of ellipses (NP-ellipses, clause-ellipses) are also used in DAS.

2. **Improving the rules to posit relations between large spans.** The rules to posit relations between large spans were extended from Marcu (2000) so that cue phrases from the satellites can contribute to the recognition process.

3. **A new discourse segmentation method.** This discourse segmentation approach uses syntactic information and cue phrases. A post segmenting process is used in this approach to refine segment boundaries after being generated by the above segmentation factors.

4. **A new method for deriving sentential discourse trees.** This method produces trees fast and accurately. As described in Section 3.1, these trees are created by the post segmenting process of discourse segmentation. based on the sentential syntactic structure and cue phrases. If we do not count the relation names of each discourse tree, only one discourse tree is generated for each sentence. After the segmentation process, DAS only

needs to posit nuclearity roles and relation names for existing discourse trees. DAS does not have to combine random spans to check whether a relation exists or not. Meanwhile, the systems of Marcu (2000) and Corston (1998) examine all combinations of spans, irrespective of the adjacency property. If a sentence has N elementary discourse units, in order to find possible rhetorical relations, N(N+1) pairs of elementary discourse units have to be checked by Marcu's system as well as Corston's. The sentence-level search space in DAS is much smaller than that of Marcu (2000) and Corston (1998).

5. **Improving the efficiency of the discourse analyser.** Unlike Marcu's (2000) system, which generates all combinations of discourse trees and then filters out the inappropriate ones, DAS takes efficiency issues seriously, and tries to avoid combinatorial explosion by using a beam search with constraints about textual organisation and textual adjacency. The search space of DAS is also smaller than Corston's (1998), as discussed in Section 5.3.1.

6. **Evaluation methods.** To evaluate the tree quality, Marcu (2000) computes a weight for each valid discourse tree and retains only the trees that are maximal. This weight function gives high priority for right-branching trees. However, this priority does not apply for all genres. Corston (1998) uses heuristic scores associated with heuristic rules to form rhetorical structure. The heuristic score associated with a tree is computed from the heuristic scores of the relations used in constructing the tree. Unlike Corston (1998), DAS calculates the score for each tree by summing up the total-score of every relation contributing to it. The relation score is computed from the *total-heuristic-score* of recognition factors and the *block-level-score*, which relates to the position relation of its left and right spans. The *block-level-score* is used to ensure that spans in the same textual block are connected before spans in different ones. The left-branching trees and the right-branching ones are considered equally in Corston (1998) and DAS.

To evaluate the discourse system, the accuracy at seven processing levels was calculated based on experimental results. By this detailed evaluation, one will know which process needs to be improved most.

7.2 Future Work

Generating an automatic discourse analysing system is a difficult task. Although many efforts have been put into different issues of discourse analysis such as discourse segmentation and relation recognition, there is room to improve the performance of the discourse system. In this research, we proposed an approach to generate a discourse system, concentrating on the system's performance and on the problem of combinatorial explosion in searching for a discourse tree representing a text. An implementation has been made based on this approach. Due to the time limitation, several tasks that have been proposed to improve the system's performance are left for future work, including:

1. **Integrating a learning algorithm to learn the score of cue phrases and scores of heuristic rules.** The score of a cue phrase is assigned between 0 and 1, depending on its certainty in signalling a relation. For example, "*in contrast*" explicitly signals a *Contrast* relation, meanwhile "*however*" can indicate a *Contrast*, or an *Antithesis* relation. The score of a heuristic rule is between 0 and 100, and also depends on its certainty in signalling a relation. At present, these scores are assigned using human intuition. The heuristics rules and their scores can be modified when new examples are provided. However, these tasks are currently done manually. A learning algorithm is necessary to improve the accuracy in recognising relations and to adapt to new data and genres.

2. **Integrating a learning algorithm to learn syntactic-based rules and cue-phrase-based rules that are used to segment text and posit rhetorical relations.** The basic segmentation rules in DAS are based on the segmentation principles in Carlson et al. (2002). We have manually improved this rule set based on different linguistic sources. In order to make this set more adaptable with other genres of data, a training algorithm to learn new instances and to optimise the result should be integrated into

DAS. New heuristic rules are added, and existing heuristic scores are adjusted until DAS can derive the closest rhetorical structures to the ones created by human analysts.

3. **Investigating a method to segment text into semantic-related paragraphs if there is no information about the organisation of text.** As discussed in Sections 5.3.1 and 6.2.3, the method of using a *block-level-score* to reduce the search space has a problem when there is no information about the organisation of text or when a text contains incorrect paragraph boundaries. In order to solve this problem, an aspect that is worth investigating is semantic relations in a text. For example, if some sentences refers to the same area or domain (e.g., law), and other sentences involve another domain (e.g., computer science), it is likely that the former sentences and the latter ones belong to two different paragraphs, each of which fulfils a communication goal. We propose a method using the text segmentation approach based on word term frequencies and semantic relations as a potential method to determine correct paragraph boundaries and linkages among paragraphs.

4. **Evaluating the system in more detail by calculating the performance at each level with a correct input or with the input from the previous process.** By this evaluation, the real performance of each module and the affects of the previous modules on the next ones are computed. This information is important to find which module needs to be improved in order to improve the system's performance.

In addition to improving the system's performance, we would also like to integrate a syntactic parser into DAS. The current version of DAS depends on a corpus that contains sentential syntactic structures. An available syntactic parser with high performance will be chosen to be integrated with DAS, so that DAS can generate sentential syntactic structures by itself.

Since our motivation in carrying out this research on discourse analysis is to improve the performance of a text processing application, DAS will be integrated into a more practical text processing system such as text summarisation, text translation, or information retrieval. Moreover, DAS performance can also be evaluated by its impact on the accuracy of other text processing tasks.

Last but not least, we would like to apply this research to other languages especially to the Vietnamese language. The purpose of this is twofold. First, we would like to investigate the impact of our approach to a language that has different characteristics than English. Second, since only a few studies on text processing have been carried out for Vietnamese language, this research will be a good contribution to this area.

Bibliography

Batliner, A., Kompe, R., Kießling, A., Niemann, H., and Nöth, E. (1996). Syntactic-Prosodic Labeling Of Large Spontaneous Speech Data-Bases. In *Proceedings of ICSLP-96*, USA, pp.1720-1723.

Beeferman, D., Berger, A., and Lafferty J. (1999). Statistical models for text segmentation. *Machine Learning*. In Special Issue on Natural Language Learning, edited by C. Cardie and R. Mooney, 34:177-210.

Bies, A., Ferguson, M., Katz, K., and MacIntyre, R. (1995). Bracketing Guidelines for Treebank II Style, Penn Treebank Project.

Boguraev, B.K. and Neff, M.S. (2000). Discourse Segmentation in Aid of Document Summarisation. In *Proceedings of the 33rd Hawaii International Conference on System Sciences*-Volume 3, pp.3004.

Bouchachia, A., Mittermeir, R., and Pozewaunig, H. (2000). *Document Identification by Shallow Semantic Analysis*. NLDB 190-202.

Carlson, L., Marcu, D., and Okurowski, M.E. (2002). Building a Discourse-Tagged Corpus in the Framework of Rhetorical Structure Theory. In *Current Directions in Discourse and Dialogue*, Jan van Kuppevelt and Ronnie Smith eds., Kluwer Academic Publishers, pp.85-112.

Choi, F. (2000). Advances in domain independent linear text segmentation. In *Proceedings of NAACL'00*, Seattle, USA, pp.26-33.

Cristea, D. (2000). An Incremental Discourse Parser Architecture. In *Proceedings of the Second International Conference Natural Language Processing - NLP 2000*, Patras, Greece. Lecture Notes in Artificial Intelligence 1835, Springer, pp.162-175.

Cristea, D. and Dima, G.E. (2001). An Integrating Framework for Anaphora Resolution. In *Information Science and Technology*, Romanian Academy Publishing House, Bucharest, 4(3-4):273-292.

Corston, S.O. (1998). *Computing Representations of the Structure of Written Discourse*. Ph.D. Thesis. University of California, Santa Barbara, CA, U.S.A.

Dalrymple, M. and Kehler, A. (1995). On the Constraints Imposed by Respectively. *Linguistic Inquiry* 26(3):531-536 (Squibs and Discussion).

Elhadad, N. and McKeown, K. (2001). Towards Generating Patient Specific Summaries of Medical Articles. In *Proceedings of the NAACL Workshop on Automatic Summarisation*, Pittsburgh, PA, pp.31-39.

Forbes, K. and Webber, B. (2002). A Semantic Account of Adverbials as Discourse Connectives. In *Proceedings of the 3rd SIGDial Workshop on Discourse and Dialogue*, Philadelphia PA, pp. 27-36.

Forbes, K. and Miltsakaki, E. (2002). Empirical Studies of Centering Shifts and Cue Phrases as Embedded Segment Boundary Markers. In E. Kaiser (ed.). *Penn Working Papers in Linguistics. Current work in linguistics*. 7(2):39-57.

Forbes, K., Miltsakaki, E., Prasad, R., Sarkar, A., Joshi, A., and Webber, B. (2003). D-LTAG System: Discourse Parsing with a Lexicalized Tree-Adjoining Grammar. *Journal of Logic, Language and Information*, 12(3):261-279.

Gardent, C. (1997). *Discourse tree adjoining grammars*. Claus report no.89, University of Saarland, Saarbücken.

GATE (2004). *General Architecture for Text Engineering*. University of Shelfield, UK.

Grosz, B.J. and Sydner C.L. (1986). Attention, intentions and the structure of discourse. *Computational Linguistics*, 12:175-204.

Gundel, J., Hegarty, M., and Borthen, K. (2003). Cognitive status, information structure and pronominal reference to clausally introduced entities. *Journal of Logic, Language and Information*, 12:281-299.

Hahn, U., and Strube, M. (1997). Centering in-the-Large: Computing Referential Discourse Segments. In *Proceedings of ACL'97/EACL'97*. Madrid, Spain, pp.104-111.

Harabagiu, S. and Maiorano, S. (1999). Knowledge-lean coreference resolution and its relation to textual cohesion and coreference. In *Proceedings of the ACL Workshop on Discourse/Dialogue Structure and Reference*, pp.29-38.

Halliday, M.A.K. and Hasan, R. (1976). *Cohesion in English*. London, England: Longman.

Heusinger, K. (2001). Intonational Phrasing and Discourse Segmentation. In *Proceedings of the ESSLLI Workshop on Information Structure, Rhetorical structure and Discourse Semantics*. Helsinki, pp.189-200.

Hirschberg, J. and Litman, D. (1993). Empirical Studies on the Disambiguation of Cue Phrases. *Computational Linguistics*, 19(3):501-530.

Hobbs, J. (1979). Coherence and coreference. *Cognitive Science* 3:67-90.

Hobbs, J. (1985). On the Coherence and Structure of Discourse. Technical Report CSLI-85-37, Center for the Study of Language and Information.

Hovy, E. H. (1990). Parsimonious and profligate approaches to the question of rhetorical structure relation. In *Proceedings of the 5th International Workshop on Natural Language Generation*. Pittsburgh, pp.128-136.

Hovy, E. (1993). Automated Discourse Generation Using Rhetorical structure Relations. *Artificial Intelligence*, 63, pp.341-386.

Hovy E. and Lin, C. (1999). Automatic Text Summarisation in SUMMARIST. In *Advanced in automatic text summarisation*, edited by Inderjeet Mani and Mark T.Maybury. The MIT Press.

Jones, K.S. (1993). What might be in a summary? In *Proceedings of Information Retrieval*: Von der Modellierung zur Anwendung (eds.), pp.9-26, Universitatsverlag Knstanz.

Joshi, A. and Kuhn, S. (1979). Centered Logic: The Role of Entity Centered Sentence Representation in Natural Language Inferencing. In *Proceedings of the 6th International Joint Conference on Artificial Intelligence*, pp 435-439.

Joshi, A. and Weinstein, S. (1981). Control of Inference: Role of Some Aspects of Rhetorical structure: Centering. In *Proceedings of the 7th Internotionol Joint Conference on Artificial Intelligence*, pp.385-387.

Kameyama, M. (1997). Recognizing Referential Links: an Information Extraction Perspective. In *Proceedings of the Workshop "Operational Factors in Practical, Robust Anaphora Resolution for Unrestricted Texts"*, Madrid, pp.46-53.

Kehler, A. (1994). Common Topics and Coherent Situations: Interpreting Ellipsis in the Context of Discourse Inference. In *Proceedings of the 32nd Annual Conference of the Assaciation for Camputational Linguistics (ACL-94)*, pp.50-57.

Kehler, A. (1996). Coherenee and the Coordinate Structure Constraint. In *Proceedings of the 22nd Annual Meeting of the Berkeley Linguistics Society (BLS 22)*, Berkeley, CA, pp.220-231.

Kehler, A. and Shieber, S. (1997). Anaphoric Dependencies in Ellipsis. *Computatianal Linguistics*, 23(3):457-466.

Knott, A. (1996). *A Data-Driven Methodology for Motivating a Set of Coherence Relations.* Ph.D. Thesis, University of Edinburgh, UK.

Knott, A. and Dale, R. (1994). Using Linguistic Phenomena to Motivate a Set of Rhetorical Relations. *Discourse Processes* 18(1):35-62.

Komagata, N. (2001). Entangled Information Structure: Analysis of Complex Sentence Structures. In *Proceedings of the ESSLLI 2001 Workshop on Information Structure, Rhetorical structure and Discourse Semantics*. Helsinki, pp.53-66.

Korbayov, I.K. and Webber, B. (2000). Information Structure and the Interpretation of "otherwise". In *Proceedings of the 2nd International Conference in Contrastive Semantics and Pragmatics (SIC-CSP 2000)*, Cambridge, UK, pp.67-83.

Kozima, H. (1994). *Computing lexical cohesion as a tool for text analysis*. Ph.D. Thesis: Graduate School of Electro-Communications, University of Electro-Communications.

Kurohashi, S. and Nagao, M. (1994). Automatic detection of rhetorical structure by checking surface information in sentences. In *Proceedings of the 15th International Conference on Computotional Linguistics*, 2:1123-1127.

LeThanh. H., Abeysinghe. G., and Huyck. C. (2003a). Using Cohesive Devices to Recognize Rhetorical Relations in Text. In *Proceedings of 4th Computational Linguistics UK Research Colloquium (CLUK-4)*, pp.123-128.

LeThanh, H. and Abeysinghe, G. (2003b). A Study to Improve the Efficiency of a Discourse Parsing System. In *Proceedings of the 4th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing'03)*, pp.104-117.

LeThanh, H., Abeysinghe, G., and Huyck, C. (2004a). Automated Discourse Segmentation by Syntactic Information and Cue Phrases. In *Proceedings of the IASTED International Conference on Artificiol Intelligence and Applications (AIA 2004)*, pp.293-298.

LeThanh, H., Abeysinghe, G., and Huyck, C. (2004b). Generating Discourse Structures for Written Texts. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004)*, pp.329-335.

Litman, D.J. (1996). Cue Phrase Classification Using Machine Learning. *Journal of Artificial Intelligent Research*, 5:53-94.

Litman, D.J. and Passonneau, R.J. (1993). Empirical evidence for intention-based discourse segmentation. In *Proceedings of the ACL Workshop on Intentionality and Structure in Rhetorical relations*, pp.60-63.

Maier, E. and Hovy, E.H. (1991). A Metafunctionally Motivated Taxonomy for Discourse Structure Relations. In *Proceedings of the 3rd European Workshop on Language Generation*. Innsbruck, Austria, pp.38-45.

Mann, W. C. and Thompson, S. A. (1988). Rhetorical Structure Theory: Toward a Functional Theory of Text Organization. *Text*, 8:243-281.

Mann, W.C. (2003). A View of Rhetorical Structure Theory. http://www.sil.org/%7Emannb/rst/index.htm

Marcu, D. (1997). The Rhetorical Parsing, Summarization, and Generation of Natural Language Texts. PhD Thesis, Department of Computer Science, University of Toronto.

Marcu, D. (1999). A decision-based approach to rhetorical parsing. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL)*. Maryland, pp.365-372.

Marcu, D. (2000). *The theory and practice of discourse parsing and summarisation*. MIT Press, Cambridge, Massachusetts. London, England.

Marcu, D., Amorrortu, E., and Romera, M. (1999). Experiments in Constructing a Corpus of Discourse Trees. *The ACL'99 Workshop on Standards and Tools for Discourse Tagging*, pp.48-57.

Marcu, D. and Echihabi, A. (2002). An Unsupervised Approach to Recognising Rhetorical relations. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp.368-375.

Matthiessen, C. and Thompson, S.A. (1988). *The structure of discourse and 'subordination'*. In Haiman and Thompson (eds.), pp.275-329.

McKeown, K.R. (1985). *Text Generation: Using Discourse Strategies and Focus Constraints to Generate Natural Language Text*. Cambridge University Press.

Mellish, C., Knott, A., Oberlander, J., and O'Donnell, M. (1998). Experiments Using Stochastic Search for Text Planning. In *Proceedings of IWNLG*-98, Niagara-on-the-Lake, Canada. Association for Computational Linguistics, pp.98-107.

Miike, S., Itoh, E., Ono, K., and Sumita, K. (1994). A full-text retrieval system with a dynamic abstract generation function. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information*. Dublin, Ireland, pp.152-161.

Mitkov, R. (2002). *Anaphora Resolution*. Longman.

Mitra, M., Singhal, A., and Buckley, C. (1997). Automatic text summarisation by paragraph extraction. In *Proceedings of the ACL/EACL-97 Workshop on Intelligent Scalable Text Summarisation*, pp.31-36, Madrid, Spain.

Morato, J., Llorens, J., Genova, G., and Moreiro, J. A. (2003). Experiments in discourse analysis impact on information classification and retrieval algorithms. *Information Processing and Management*. 39(6):825-851.

Morris, J. and Hirst, G. (1991). Lexical Cohesion Computed by Thesaural Relations as an Indicator of the Structure of the Text. *Computational Linguistics*, 17:21-28.

Nomoto, T. and Matsumoto, Y. (2001). A New Approach to Unsupervised Text Summarisation. In *Proceedings of SIGIR'01*, New Orleans, Louisiana, USA, pp.26-34.

O'Donnell, M. (2002). RSTTool - an RST Markup Tool. http://www.wagsoft.com/RSTTool/index.html

Okumura, M. and Honda, T. (1994). Word Sense Disambiguation and Text Segmentation Based on Lexical Cohesion. In *Proceedings of the 15th Conference on Computational Linguistics (COLING-94)*, 2:755-761.

Passonneau, R. J. and Litman, D. J. (1997). Discourse Segmentation by Human and Automated Means. *Computational Linguistics* 23(1):103-139.

Penn Treebank (1999). Linguistic Data Consortium. http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC99T42

Polanyi, L. (1988). A formal model of the structure of discourse. *Journal of Pragmatics* 12:601-638.

Polanyi, L. (1996). *The Linguistic Structure of Discourse*. Technical Report CSLI-96-200. Center for the Study of Language and Information.

Polanyi, L., Culy, C., Thione, G.L., and Ahn, D. (2004). A Rule Based Approach to Discourse Parsing. In *Proceedings of SigDial2004*, pp.108-117.

Poesio, M. and Di Eugenio, D. (2001). Discourse structure and Anaphoric Accessibility. In *Proceedings of the ESSLLI Workshop on Information Structure, Rhetorical structure and Discourse Semantics*. Helsinki, pp.129-143.

Power, R. (2000). Mapping Rhetorical Structures to Text Structures by Constraint Satisfactions. Technical Report. ITRI-00-01. ITRI, University of Brighton, UK.

Power, R., Scott, D., and Bouayad, N.A. (2003). Document Structure. *Computational Linguistics* 29(2) :211-260.

Quirk, R., Greenbaum, S., Leech, G., and Svartvik, J. (1972). *A Grammar of Contemporary English*. Longman.

Rau, L.F., Brandow, R., and Mitze, K. (1994). Domain-Independent Summarisation of News. In *Summarizing Text for Intelligent Communication*, pp.71-75, Dagstuhl, Germany.

Redeker, G. (1990). Ideational and pragmatic markers of rhetorical structure. *Journal of Pragmatics*, pp.367-381.

Rino, L.H.M. and Scott, D. (1994). Automatic Generation of Draft Summaries: Heuristics for Content Selection. In *Proceedings of the Third International Conference of the Cognitive Science of Natural Language Processing*. Dublin City University, Ireland.

RST-DT (2002). *RST Discourse Treebank*. Linguistic Data Consortium. http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalog Id=LDC2002T07.

Rutledge, L., Bailey, B., Ossenbruggen, J.V., Hardman, L., and Geurts, J. (2000). Generating Presentation Constraints from Rhetorical Structure. In *Proceedings of the 11th ACM conference on Hypertext and Hypermedia*. San Antonio, Texas, USA. pp. 19-28.

Salton, G., Singhal, A., Mitra, M., and Buckley, C. (1999). Automatic Text Structuring and Summarisation. In *Advances in Automatic Text Summarisation*, pp. 341-356.

Salkie, R. (1995). *Text and discourse analysis*. London, Routledge.

Schiffrin, D. (1987). *Discourse markers*. Cambridge: Cambridge University Press.

Schiffrin, D. (1994). *Approaches to discourse*. Oxford: Blackwell.

Siegel, E.V. and McKeown, K.R. (1994). Emergent linguistic rules from inducing decision trees: disambiguating discourse clue words. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pp.820-826, Seattle, WA.

Scott, D.R. and de Souza, C.S. (1990). Getting the message across in RST-based text generation. In *Current Research in Natural Language Generation*. Academic Press, pp.47-73.

Soricut, R. and Marcu, D. (2003). Sentence Level Discourse Parsing using Syntactic and Lexical Information. In *Proceedings of HLT-NAACL 2003*, pp.149-156.

Tablan, M., Barbu, C., Popescu, H., Hamza, R., Nita, C.I., Bocaniala, C.D., Ciobanu, C., and Cristea, D. (1988). Co-operation and Detachment in Discourse Understanding. In *Proceedings of the Workshop on Lexical Semantics and Rhetorical structure, ESSLLI'98*, Saarbruecken.

Torrance, M. and Bouayad-Agha, N. (2001). Rhetorical structure analysis as a method for understanding writing processes. In *Proceedings of the International Workshop on Multi-disciplinary Approaches of discourse (MAD 2001)*, pp. 51-59, Amsterdam & Nodus Publications.

Utiyama, M. and Isahara, H. (2001). A Statistical Model for Domain-Independent Text Segmentation. In *Proceedings of ACL/EACL-2001*, pp. 491-498.

Webber, B. L. (1991). Structure and ostension in the interpretation of discourse deixis. *Language and Cognitive Processes*, 6(2), pp.107-135.

Webber, B., Knott, A., Stone, M., and Joshi, A. (1999a). Multiple Discourse Connectives in a Lexicalized Grammar for Discourse. In *Proceedings of the Third International Workshop on Computational Semantics*, Tilburg, The Netherlands, pp.309-325.

Webber, B., Knott, A., Stone, M., and Joshi, A. (1999b). Discourse relations: A Structural and Presuppositional Account Using Lexicalised TAG. In

Proceedings of the Meeting of the Association far Computational Linguistics, College Park MD, pp.41-48.

Webber, B., Stone, M., Joshi, A., and Knott, A. (2003). Anaphora and Rhetorical structure. *Computational Linguistics*, 29(4):545-587.

WordNet (2004). http://www.cogsci.princeton.edu/~wn/index.shtml (last visited 9/2004)

Appendix 1

Architecture of DAS

Appendix 1 presents the architecture of DAS implemented in this thesis, which is created based on our proposed solution discussed in Chapters 3, 4, and 5.

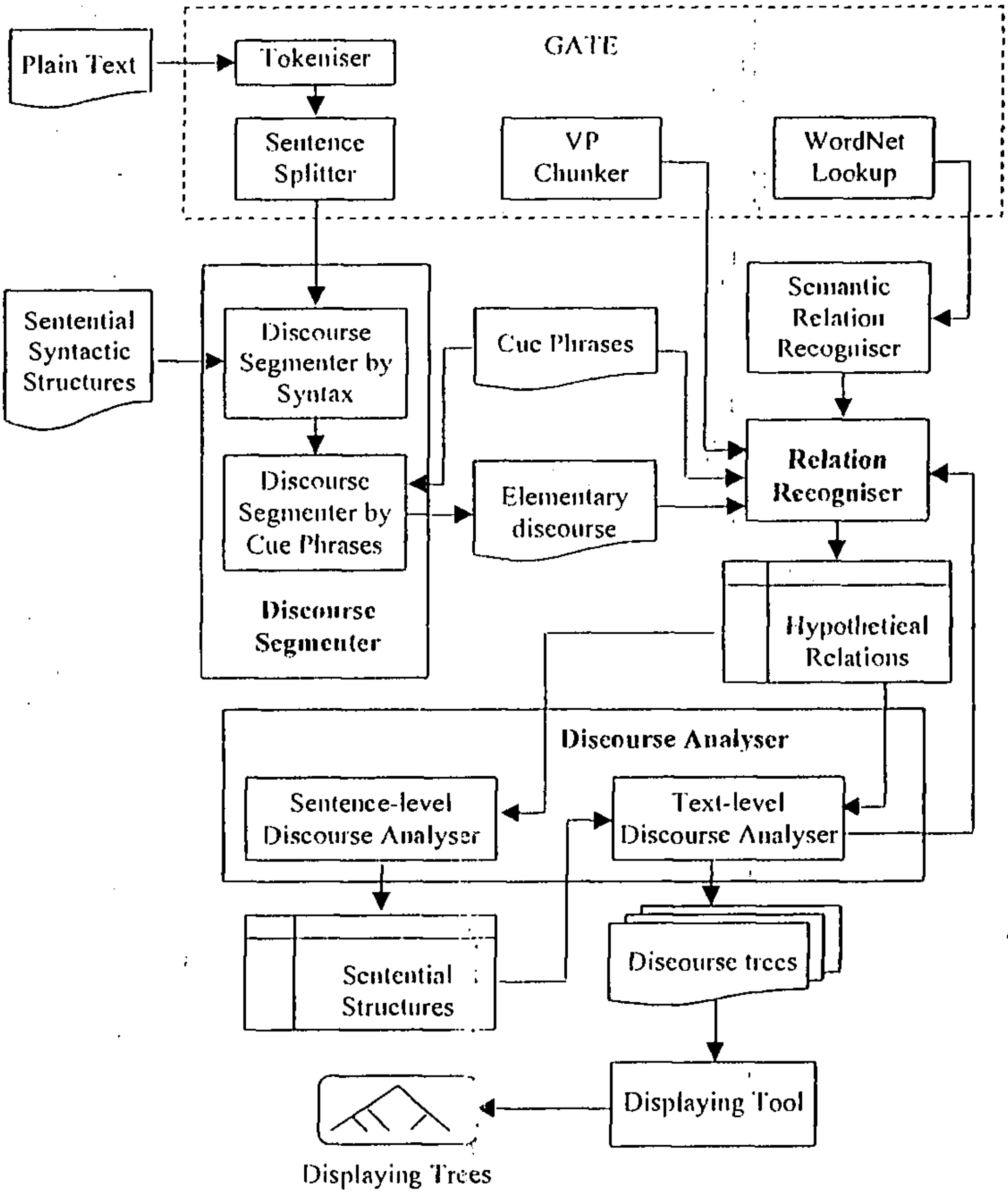


Figure A1.1. The Architecture of DAS

DAS takes articles from the RST Discourse Treebank (2002) as the input and derives RST trees. In order to easy integrate with other language processing modules and DAS. GATE (2004) - a software developed by the computational linguistics team at the Sheffield University, United Kingdom – is used as the infrastructure of DAS. GATE is an architecture in which text processing tools can be created and used. It has a Collection of Reusable Objects for Language Engineering (CREOLE) that enables language processing components to be loaded into GATE. A series of ProcessingResources (PRs) is available in CREOLE, which can be reused in constructing new text processing systems. The PRs used in DAS are:

- **Tokeniser** tokenises the input text into words.
- **Sentence Splitter** finds and marks sentence boundaries.
- **VPChunker** gets the original form of the verb.
- **WordNet Lookup** (WordNet, 2004) gets the meaning of a word and the semantic relation between words (e.g., synonyms, antonyms).

The main processing modules of DAS, which are created by us, include:

- **Discourse Segmenter** segments text into elementary discourse units. One sentence is processed at a time. Two components of this module, **Discourse Segmenter by Syntax** and **Discourse Segmenter by Cue Phrases,** split a sentence into elementary discourse units by using syntactic information and cue phrases, respectively.
- **Relation Recogniser** finds all possible rhetorical relations between elementary discourse units.
- **Semantic Recogniser** computes a semantic relation between words.
- **Discourse Analyser** derives rhetorical structures from text. It is divided into two parts: **Sentence-level Discourse Analyser** and **Text-level Discourse Analyser**. The former constructs discourse trees for each sentence. Starting with sentence as its smallest spans, the latter derives

rhetorical relations between sentences to produce discourse trees for the entire text.

DAS was implemented using Java language. The working process of DAS is briefly described below.

The plain text of an article from the RST-DT (2003) is tokenised by the Tokeniser. Based on the output of the Tokeniser and the rules that identify sentence boundaries, the Sentence Splitter segments the text into paragraphs and sentences. The plain text of a sentence and its syntactic structure is used as the input to the Discourse Segmenter by Syntax. This syntactic information is taken from the syntactic document of the Penn Treebank that corresponds to the input text of DAS. The Discourse Segmenter by Syntax segments a sentence into clauses by a rule set, based on syntactic information. The output of this process is further segmented by the Discourse Segmenter by Cue Phrases. The set of cue phrases used in DAS is inherited from those in Grosz and Sidner (1986), Hirschberg and Litman (1993), Knott and Dale (1994), and Marcu (2000). The reader is referred to Chapter 3 for a detailed description of the discourse segmentation process.

The output of the Discourse Segmenter by Cue Phrases, which is stored in a text file, is now used as the input to the Relation Recogniser. This module posits all possible relations between elementary discourse units by using the syntactic information of a sentence, cue phrases, time relation, and semantic relations between discourse units (see Section 4.2). The semantic relations are computed by the Semantic Recogniser, which computes the semantic relations between words by using information from the WordNet Lookup. The original form of the verb, which is obtained by the VP Chunker of GATE, assists the Relation Recogniser to detect VP cues. All relations generated by the Relation Recogniser are stored in a relation set, which will be used by the Discourse Analyser. The task of recognising rhetorical relation was discussed in Chapter 4.

Next, the Sentence-level Discourse Analyser generates a discourse tree for each sentence. Starting with sentences as its smallest spans, the Text-level Discourse Analyser derives rhetorical relations between sentences to produce discourse trees for the entire text. During the analysing process, the Text-level Discourse Analyser may need to go back to the relation recognition process, as the analyser may generate new combinations of spans, which have not been created before. Each discourse tree generated by the Discourse Analyser is stored in a file, which is then displayed by a Displaying Tool called RSTTool (O'Donnell, 2002). The Discourse Analyser was discussed in Chapter 5.

Appendix 2

Algorithms

This appendix represents an extended version of some main algorithms implemented in this thesis.

Input:

- *start* and *end* position of the phrase needed to be processed. The first time this algorithm is called, *start* and *end* are assigned as integer values 0 and the length of the executed sentence, respectively.
- A list of rhetorical relations *sentNodes* created by the segmentation procedure presented in Figure 3.1.

Output:

- Rhetorical relations after refining boundaries, each of which contains adjacent and non-overlapping spans. These rhetorical relations should cover the entire input span.

Defragment(start, end)

1. if(*start* >= *end*) Return;
2. *minsta* = the left most position among all relations within the input phrase that satisfies (*minsta* >= *start*).
3. *maxend* = the end position of the right span of the tree node starting at *minsta*. If two or more tree nodes start at *minsta*, *maxend* = the maximum value of these end positions that satisfies (*maxend* <= *end*).
4. *middle* = the position of the segment boundary between the left and right node of the tree node that starts at *minsta* and ends at *maxend*. If two or more tree nodes start at *minsta* and end at *maxend*, *middle* = the maximum value of these middle positions that satisfies (*middle* > *minsta* and *middle* < *maxend*).
5. if not found (*minsta*, *maxend*, *middle*) Return;
6. *changenode* = the tree node that has the start position *minsta*, the end position *maxend*, and the end position of its left node *middle*.
7. if(*minsta* > *start*)

   7.1 if(*changenode.leftrole* = 'N'):

   7.1.1 Expand the left node of the *changenode* to the start position:

```
        changenode.from = start; changenode.leftnode.from = start;
  7.1.2 Update sentNodes;
 7.2    else[21]:
  7.2.1 Create a new node, whose left node corresponds to the
        remaining span, and the right node is the changenode:
        newnode.leftnode.from = start; newnode.leftnode.to = minsta;
        newnode.rightnode = changenode;
        newnode.from = start; nawnode.to = changenode.to;
        newnode.leftrole = 'N';  newnode.rightrole = 'N';
        if(changanode. leftrole = 'S')
                newnode.relationname = "Same-Unit";
  7.2.2 Update sentNodes;
8. if(maxend < end)
 8.1    if(changenode. leftrole = 'S'):
  8.1.1 Expand the right node of the changenode to the end position:
        changenode.to = end; changenode.rightnode.to = end;
  8.1.2 Update sentNodes;
 8.2    else:
  8.2.1 Create a new node, whose left node is the changenode, and
        the right node corresponds the remaining span:
        newnode.rightnode.from =maxend; newnode.rightnode.to =end;
        newnode.leftnode = changenode;
        newnode.from = changenode.from; newnode.to = end;
        newnode.leftrole = 'N';  newnode.rightrole = 'N';
        if(changenode.leftnode = 'N')
                newnode.relationname = "Same-Unit";
  8.2.2 Update sentNodes;
9. if(middle < end) Defragment(start, middle);
10. if(middle > start) Defragment(middle, end);
11. Return.
```

Figure A2.1. Pseudo-code for the *Defragment* Process of Discourse Segmentation by Syntax

[21] The *leftrole* and *rightrole* of a tree node is empty ('') when the nuclearity roles have not been assigned.

Input:

- Two non-overlapping spans $Unit_1$, $Unit_2$. (These spans do not need to be adjacent.)
- The syntactic rule has been used to segment text (when the input spans are clauses).
- Lists of cue phrases, VP cues, and NP cues.

Output: Rhetorical relations between $Unit_1$ and $Unit_2$.

Algorithm:

1. If the input spans are clauses, use the syntactic rule from the input to posit relations. Otherwise, go to Step 2.
   1.1 If a relation name is found, Stop.
   1.2 Otherwise, go to Step 2.
2. Find all cue phrases in Unit1 and Unit2.
3. If cue phrases are found, compute the actual score of each cue phrase and the total score of these actual scores.
   3.1 If the total score >= 0, check the necessary conditions of the relations corresponding to these cue phrases.
      3.1.1 If one relation satisfies, compute the total score of all heuristic rules of this relation. Posit these relations between $Unit_1$ and $Unit_2$. Stop.
      3.1.2 If no relation is satisfied, go to Step 4.
   3.2 If the total score < 0, go to Step 4.
4. Find the main verb phrases of the two units and stem these VPs.
5. Check whether the stemmed VPs contain VP cues or not.
   5.1 If VP cues are found, compute the actual score of each VP cue and the total score of the actual score of these VP cues.
      5.1.1 If the total score >= 0, check the necessary conditions of the relations corresponding to these VP cues.
         5.1.1.1 If at least one relation is satisfied, compute the total score of all heuristic rules of these relations. Posit these relations between $Unit_1$ and $Unit_2$. Stop.
         5.1.1.2 If no relation is satisfied, go to Step 6.
      5.1.2 If the total score < 0, go to Step 6.

5.2 If VP cues are not found, go to Step 6.

6. Find the main noun phrases from the subjects and objects of the two units and stem these NPs.

7. Check whether the stemmed NPs contain NP cues or not.

    7.1 If NP cues are found, compute the actual score of each NP cue and the total score of the actual score of NP cues.

        7.1.1 If the total score >= θ, check the necessary conditions of the relations corresponding to these NP cues.

            7.1.1.1 If at least one relation is satisfied, compute the total score of all heuristic rules of these relations. Posit these relations between Unit1 and Unit2. Stop.

            7.1.1.2 If no relation is satisfied, go to Step 8.

        7.1.2 If the total score < θ, go to Step 8.

    7.2 If NP cues are not found, go to Step 8.

8. Check other heuristic rules of each relation.

    8.1 If several relations are signalled, compute the total score of these relations.

        8.1.1 If the total score >= θ, check the necessary conditions of these relations.

            8.1.1.1 If at least one relation is satisfied, posit these relations between Unit1 and Unit2. Stop.

            8.1.1.2 If no relation is satisfied, go to Step 9.

        8.1.2 If the total score < θ, go to Step 9.

    8.2 If no relation is signalled, go to Step 9.

9. If there is a signal indicating that $Unit_1$ and $Unit_2$ has a semantic relation, posit an *Elaboration* relation. Otherwise, posit a *Joint* relation. Stop.

Figure A2.2. Outline of the Algorithm to Posit Relations Between Spans

Input:

- Discourse trees of all sentences from the input text
- Information about positions of sentences in the text
- The value of N (the number of discourse trees required by the user).

Output:

- Discourse trees that cover the entire text.

Algorithm:

1. *Trees* = { }.
2. *Subtrees* = {all sentential discourse trees}.
3. If *Subtrees* contains only one tree, add this tree to Trees. Stop. Otherwise, go to Step 4.
4. *accumulated-score = 0*
5. Find hypothesis between adjacent sentential discourse trees. With each hypothesis:
   - *total-score(hypothesis) = total-heuristic-score(hypothesis) + block-level-score(hypothesis)*
   - *predicted-score(hypothesis) = total-score(hypothesis)*
   - Sort the hypotheses by their *predicted-score*.
   - Store the hypotheses in a set called *NewH*.
6. Select M highest *total-score* hypotheses from *NewH* and put them into *PotentialH*.
7. For each hypothesis in *PotentialH* (called *appliedH*), create a hypothesis set *hypoSet[i]* (i = 1÷M). For each *hypoSet[i]*, compute:

   7.1. *accumulated-score = predicted-score(appliedH)*.

   7.2. *Subtrees:*
   - *Subtrees = Subtrees* ∪ {*appliedH*} \ {trees that overlap with *appliedH*}
   - If *Subtrees* has only one tree, add that tree to *Trees*.
   - If the number of discourse trees in *Trees* is equal to N, Stop. Otherwise, continue.

   7.3. *NewH:* This set stores new hypotheses that are created due to the modification of *Subtrees*. They are relations between the node created by *appliedH* and its adjacent nodes.

With each hypothesis in *NewH*:

total-score(hypothesis) = total-heuristic-score(hypothesis)
+ block-level-score(hypothesis).

7.4. *PotentialH*: This set stores all the potential hypotheses which can be used after *appliedH* has been used.

- *PotentialH = PotentialH* \ {*appliedH*} \ {hypotheses that overlap with *appliedH*} ∪ *NewH*
- With each hypothesis in *PotentialH*:
  predicted-score(hypothesis) = accumulated-score
  + total-score(hypothesis)
- Sort hypotheses in *PotentialH* by *predicted-score*.
- If there are more than M hypotheses in *PotentialH*, keep M highest *predicted-score* hypotheses in *PotentialH*. Otherwise, keep all hypotheses in *PotentialH*.

8. If all *PotentialHs* are empty, Stop. Otherwise, go to step 9.
9. Select M highest *predicted-score* hypotheses from M sets of *PotentialH* to be applied (*appliedH*). Let us say *hypoSet[p]* is the set that *appliedH* belongs to. With each *appliedH*:

9.1 If *appliedH* appears in the *PotentialH* of other *hypoSets* that are at the same level with *hypoSet[p]*, delete *appliedH* from those *hypoSets*.

9.2 Update all sets and variables in *hypoSet[p]* (Steps 7.1 to 7.4).

9.3 Repeat Step 9 until the number of discourse trees in *Trees* is equal to N or all *PotentialHs* are empty.

Figure A2.3. Outline of Algorithm for Deriving Text-level Discourse Trees

Appendix 3

List of Cue Phrases

In this table, the information about each cue phrase is encoded by

cue_phrase(position_in_text, side, scope, score), in which

- **position_in_text** is the position of the cue phrase in the span where the cue phrase can be used to signal relation. position_in_text can be 'B' (beginning), 'M' (middle), 'E' (end), or 'A' (any position).
- side can be 'L' (left), 'R' (right), or 'A' (any side).
- scope can be 'C' (clause), 'S' (sentence), or 'P' (paragraph).
- score is between 0 and 1.

Index	Relation name	Cue phrase
1	List	also(A,R,S,1), alternatively(A,R,S,0.8), but also(B,R,C,1), and also(B,R,C,1), not only(A,L,C,1), and(B,R,S,0.8), and another(B,R,S,1), neither(A,A,S,1), nor(B,R,C,1), or(B,R,S,0.8), too(E,R,C,0.8), in addition(B,R,P,0.8)
2	Sequence	and(B,R,S,0.8), and then(B,R,S,1), at first(B,L,P,1), in the beginning(A,L,P,0.8), at the beginning(A,L,P,0.8), at last(A,R,P,1), at the end(A,R,P,1), in the end(A,R,P,0.8), eventually(A,R,P,1), formerly(B,L,S,0.5), in turn (M,R,S,1), initially(A,L,P,1), last(A,R,P,0.8), lastly(A,R,P,0.8), latter(A,R,P,1), next(B,R,P,1), subsequently(A,R,P,1), then(A,R,P,1), then again(A,R,S,0.8), thereafter(A,R,S,1), thereupon(A,R,S,0.5), ultimately(A,R,S,1), whereupon(B,R,C,1), after that(A,R,S,1), following (A,R,C,0.5)

3	Condition	as long as(B,A,C,1), as soon as(B,A,C,1), as far as(B,A,C,1), given(B,A,C,1), given that(B,A,C,1), if(B,A,C,1), only(B,A,C,1), only after(B,A,C,1), only if(B,A,C,1), only when(B,A,C,1), provided(B,R,C,1), provided that(B,A,C,1), providing that(B,A,C,1), unless(B,A,C,1), until(B,R,C,0.5), until then(B,R,S,1)
4	Otherwise	alternatively(B,R,S,0.8), else(A,R,C,0.7), elsewhere(A,R,C,0.7), in place of(B,A,C,1), otherwise(A,R,S,0.8), in other respects(B,R,S,1), in other ways(B,R,S,1), if not(B,R,S,0.8)
5	Hypothetical	arguably(A,A,S,1), it may seem that(B,R,S,1), on the assumption(B,A,C,1), perhaps(A,R,S,1), possibly(A,R,S,1), presumably(A,A,S,1), quite likely(M,A,C,1), suppose(A,L,C,1), suppose that(B,L,P,1), it may be the ease that(B,R,P,1), it is possible that(B,R,P,1), supposing(B,L,C,1)
6	Antithesis	although(B,A,C,0.8), apart from(B,L,C,1), aside from(B,L,C,1), but(B,R,P,0.8), despite(B,A,C,0.8), except(B,A,C,1), however(A,R,P,0.8), instead(A,R,S,1), instead of(B,R,C,1), whereas(B,R,C,0.8), while(B,R,C,0.5), yet(B,R,P,0.5)
7	Contrast	as against(B,R,S,1), by contrast(B,R,S,1), but(B,R,S,1), contrariwise(A,R,P,1), conversely(A,R,P,1), however(A,R,P,0.8), in a contrary(B,R,P,1), in contrast(B,R,P,1), on another(B,R,P,0.5), on one side(B,L,S,1), on the contrary(B,R,P,1), on the other hand(B,R,P,1), on the other side(B,R,P,1), yet(B,R,P,0.5), in a different point of view(B,R,P,1), in the opposite(B,R,P,1), unlike(B,L,C,1), still(B,R,S,0.5), while(B,R,C,0.3)
8	Concession	although(B,A,C,0.8), anyhow(E,R,S,1), anyway(A,R,S,1), despite(B,A,C,0.8), even(B,R,C,0.8), in spite of(B,A,C,1),

		in despite of(B,A,C,1), notwithstanding(A,R,C,1), nevertheless(E,R,C,1), nonetheless(E,R,C,1), though(A,L,C,1), still(B,R,P,0.8), yet(B,R,P,0.5)
9	Cause	because(B,A,C,1), because of(B,A,C,1), because of this(B,R,S,1), it is because(B,R,S,1), merely because(B,R,C,1), only because(A,R,P,1), simply because(A,R,P,1), since(B,A,C,0.8), so(B,R,C,1), as(B,A,C,0.5), due to(B,A,C,1)
10	Result	as a consequence(B,R,P,1), as a corollary(B,R,P,1), as a logical conclusion(B,R,P,1), as a result(B,R,P,1), as it turned out(B,R,P,1), consequently(B,R,P,1), in consequence(B,R,P,1), thereby(A,R,P,1), therefore(A,R,P,1), thereupon(A,R,P,0.8), thus(A,R,P,1), so(B,R,P,1), whereby(B,R,P,1)
11	Cause-Result	
12	Purpose	for(B,A,C,1), for the matter(B,A,P,1), for the reason(B,A,C,1), for this(B,A,P,1), for this reason(B,A,P,1), in order to(B,A,C,1), in the hope that(B,A,C,1), so as(B,R,C,1), so that(B,R,C,1), to(B,A,C,1), aim at(B,R,C,1), aiming at(B,R,C,1)
13	Solutionhood	
14	Circumstance	actually(A,R,S,0.8), after(B,A,C,1), after a time(A,A,P,1), after all(B,R,P,1), after that(B,R,P,1), after this(A,A,P,1), afterwards(A,R,P,1), again(A,R,P,0.5), all this time(B,R,P,1), already(A,R,P,0.5), another time(A,R,P,0.6), as(B,A,C,0.4), as for(B,A,C,1), as to (B,A,C,1), at that moment(B,R,P,1), at that time(B,R,P,1), at the moment(B,R,P,1), at the outset(B,L,P,1), at the same time(A,R,P,1), at this date(A,R,P,1), at this moment(A,A,P,1), at this point(A,A,P,0.5), at this stage(A,R,P,1), at which(B,R,C,0.8), before(A,A,P,1), by that time(A,R,P,1), by then(A,R,P,1), each time(A,A,C,1),

		earlier(A,R,P,1), either case(A,R,S,0.4), either event(A,R,S,0.4), either way(A,R,S,0.4), every time(A,A,P,1), everywhere(A,A,P,1), from now on(A,A,P,1), from then on(A,A,P,1), here(A,A,P,0.6), heretofore(A,A,P,0.8), hitherto(A,A,P,0.8), in any case(A,A,P,1), in case(A,A,C,1), in doing(A,A,C,1), in doing so(B,R,C,1), in such a(B,R,C,1), in such an(B,R,C,1), in that(B,R,C,1), in that case(B,R,C,1), in the beginning(B,L,P,0.8), in the case of(B,A,C,1), in the end(B,R,P,0.8), in the event(A,A,P,1), in the first place(A,R,P,1), in the meantime(A,R,P,1), in this case(A,R,P,1), in this connection(A,R,P,1), in this respect(A,R,P,1), in this way(A,R,P,1), in which(B,R,C,1), in which case(B,R,C,1), instantly(A,A,S,0.5), just as(B,R,C,1), just before(B,R,C,1), just then(A,R,P,1), meanwhile(B,R,P,1),never again(A,R,P,1), now(A,A,P,1), now that(A,A,P,1), on the bases(A,A,P,1), on the basis(A,A,P,1), on this basis(A,R,P,1), on which(B,R,C,1), once(A,A,C,0.5), once again(A,R,S,0.7), once more(A,R,S,.7), particularly when(B,R,C,1), presently(A,A,S,1), previously(A,R,P,1), since(B,A,P,0.8), some time(A,A,P,0.4), the moment(A,A,P,0.7), this time(A,R,P,0.8), thus far(A,R,P,0.2), to the degree that(B,R,C,1), to the extent(B,R,C,1), to this end(A,R,P,1), under the circumstances(A,A,P,1), under these circumstances(A,A,P,1), until(B,A,C,0.6), up to now(B,A,P,1), up to this(B,R,P,1), when(B,A,C,1), whenever(B,A,C,1), where(B,A,C,1), wherein(B,R,C,1), wherever(B,A,C,1), while(B,A,C,0.8), with regard to(B,A,C,1), with respect to(B,A,C,1), without(B,A,C,0.5)

15	Manner	as(B,A,C,0.5), as if(B,R,C,1), as though(B,R,C,1), decidedly(A,A,S,0.5), definitely(A,A,S,0.5), doubtless(A,A,S,0.3), in the same way(A,A,C,1), more accurately(A,A,S,0.5), more precisely(A,A,S,0.5), more specifically(A,A,S,0.5), parenthetically(A,A,S,0.5), regardless(B,A,C,1), simultaneously(A,A,S,0.5), with(B,A,C,0.3), without(B,A,C,0.3)
16	Means	by(B,A,C,1), by means of(B,A,C,1), using(B,A,C,1)
17	Interpretation	in other words(B,R,P,1), according to(B,A,C,0.5), that is how(B,R,P,1), that is to say(B,R,P,1), that is why(B,R,P,1), to wit(B,R,P,1)
18	Evaluation	by comparison(A,R,P,1), certainly(A,R,P,0.5), clearly(A,R,P,0.5), conceivably(A,R,P,0.5), doubtless(A,R,P,0.5), equally(A,R,P,0.5), in comparison(B,R,P,1), most likely(A,R,P,1), more accurately(A,R,P,0.5), more importantly(A,R,P,1), more precisely(A,R,P,0.5), the more(A,R,P,1), very likely(0.5)
19	Summary	briefly speaking(B,R,P,1), in conclusion(B,R,P,1), in short(B,R,P,1), in summarisation(B,R,P,1), it can be concluded that(B,R,P,1), summarising(A,R,P,0.6), summing up(A,R,P,0.8), to summary(B,R,P,1), in brief(B,R,P,1), to conclusion(B,R,P,1), succinctly(A,R,P,0.5), compendiously(A,R,P,0.5), compactly(A,R,P,0.5)
20	Elaboration	above all(B,R,P,1), add to this(B,R,P,1), additionally(A,R,P,1), and(B,R,S,0.7), as well(A,R,S,0.5), at least(B,R,S,1), besides(B,R,P,1), besides that(B,R,P,1), for example(A,R,S,1), for instance(A,R,S,1), formerly(A,R,S,0.3), furthermore(B,R,P,1), in addition(B,R,P,1), in fact(B,R,P,0.8), in particular(A,R,P,1), in general(A,R,P,0.8), including(B,R,C,1), moreover(B,R,P,1), more to the

		point(B,R,P,1), on a different note(A,R,P,0.5), or(B,R,C,0.5), or again(B,R,C,0.5), similarity(A,R,S,1), speaking of(B,A,C,1)
21	Explanation	clearly(A,R,P,0.7), conceivably(A,R,P,0.8), in fact(B,R,P,0.8), let us assume(B,R,P,1), let us consider(B,R,P,1), the fact is(B,R,P,1), to explain(B,R,P,1), it is clear that(B,R,P,1), it is explained that(B,R,P,1), it stands to reason that(B,R,P,1), it is true that(B,R,P,1), it is easy to understand that(B,R,P,1), we can understand that(B,R,P,1), in point of fact(B,R,P,1)
22	Joint	

Appendix 4

List of NP Cues and VP Cues

In this table, the information about each NP or VP cue is encoded by cue_phrase(score), with the score ranging from 0 to 1.

Index	Relation name	NP cue	VP cue
1	List		
2	Sequence	following(1)	come after(1), succeed (0.5), follow(0.2)
3	Condition	condition(0.7), necessary(0.7), important(0.7), essential(0.7), requirement(1), requisite(1)	be necessary(1), be important(1), be essential(1), be requisite(1), require(1), have to(1), must(1)
4	Otherwise		
5	Hypothetical	possibility(1), hypothezis/zes(0.5), hypothesis/ses(0.5), guess(0.5), conjecture(0.5), supposition(0.5), assumption(0.5), reckoning(0.5), speculation(0.5)	guess(1), assume(1), suppose(1), suspect(1), reckon(1), think(1), opine(1), imagine(0.5), speculate(0.5), conjecture(0.5), hypothesize(0.5), hypothesise(0.5)
6	Antithesis		
7	Contrast	turning point(0.5), opposite(0.5)	
8	Concession		
9	Cause	cause(0.5), effect(0.5),	result from(1), be why(1), be because(1)

10	Result	result(0.5), outcome(1)	
11	Cause-Result		affect(1), cause(0.5), make(0.2), induce(1), create(0.2), bring(0.2), effectuate(1), raise(0.2)
12	Purpose		to (+ verb) (1), aim(0.5), purpose(1)
13	Solutionhood	solution(1)	answer(1), solve(1), resolve(0.5), respond(0.5), reply(0.2), react(0.2)
14	Circumstance	situation(0.5)	
15	Manner		
16	Means		
17	Interpretation	meaning(1)	mean(1), can be understand(1), stand for(1), translate(0.2)
18	Evaluation		succeed(0.5), fail(1), increase(1), fall(0.5), drop(0.5), decrease(1)
19	Summary	summarisation(0.2), brief(0.2), outline(0.2), abstract(0.5), main idea(1)	summary(0.5), conclude(0.5), brief(0.5)
20	Elaboration		include(1), consist(1),
21	Explanation	goal(1), target(1), purpose(1), reason(1), fact(0.5), aim(1), objective(1), intent(1), intention(1)	
22	Joint		

Appendix 5

Syntax-Based Segmentable Chains

This appendix presents the *syntactic chains* that are used in DAS to segment a sentence into elementary discourse units. For simplicity, the parts <text> inside a syntactic chain, such as <text1> and <text2> in (i-a) (Section 3.1.1), are removed from the representation of the chain. The following abbreviations are used in the syntactic chains:

NP – noun phrase

VP – verb phrase

SBJ – subject

S – sentence

SBAR – subordinate clause and relative clause

RRC – reduced relative clause

PRN - parenthetical

PP - prepositional phrase

PRS – parenthetical -sentence (see the syntactic chain (i-d) in Section 3.1.1)

PS - prepositional-sentence (see the syntactic chain (i-e) in Section 3.1.1)

Sx - basic clause types such as subordinate clause (SBAR) and participial clause (S-ADV)

ADx – adverb phrase or adjective phrase

ADS – a clause starts with an adverb. (ADS) is an abbreviation of the syntactic chain (ADx (S))

WHx – any phrase starts with WH-question (e.g., who, what, why)

ADVP – adverb phrase

<conjunction> - a conjunction such as "*and*", "*or*", comma, and semicolon.

"|" means "or"

"..." stands for any text or syntactic chain.

1. (NP (NP) ... (RRC|VP|Sx|PS))

The clause with the syntactic role (RRC|VP|Sx|PS) is split from the noun phrase (NP (NP) ... (RRC|VP|Sx|PS)). If Sx is a subordinate clause, the clause that has this syntactic role must have more than 1 word.

2. (VP (VP) <conjunction> (VP|Sx|RRC|PS|ADS|SBAR))

The clause (VP|Sx|RRC|PS|ADS|SBAR) is split from the verb phrase (VP (VP) <conjunction> (VP|Sx|RRC|PS|ADS|SBAR)). If Sx starts with to, VP must be an attribution verb.

3. (VP ... (Sx|RRC|PS|ADS) <conjunction> (Sx|RRC|PS|ADS))

The clauses (Sx|RRC|PS|ADS) are split from the verb phrase (VP ... (Sx|RRC|PS|ADS) <conjunction> (Sx|RRC|PS|ADS)).

4. (S (NP-SBJ) (VP .... (SBAR)) ...)

The clause (SBAR) is split from the sentence (S (NP-SBJ) (VP .... (SBAR)) ...) when the subject of the sentence (NP-SBJ) is not "*It*" and the subordinate clause starts with wh|that|empty, then (S).

5. (S (NP-SBJ) (VP .... (SBAR) <conjunction> (Sx|RRC|PS|ADS|SBAR)) ...)

The clauses (SBAR) and (Sx|RRC|PS|ADS|SBAR) are split from the sentence (S (NP-SBJ) (VP ..... (SBAR) <conjunction> (Sx|RRC|PS|ADS|SBAR)) ...).

6. (Sx ... (Sx) <conjunction> (Sx))

Two clauses (Sx) inside the sentence (Sx ...(Sx) <conjunction> (Sx)) are split from this sentence.

7\. (Sx ...(Sx) (WHx) (Sx))

Two clauses (Sx) and (WHx) (Sx) inside the sentence (Sx ...(Sx) (WHx) (Sx)) are split from this sentence.

8\. (Sx ...(Sx), (NP-SBJ) (VP))

The clause (Sx) inside the sentence (Sx ...(Sx), (NP-SBJ) (VP)) , in which VP is an attribution verb, is split from this sentence.

9\. (Sx ...(Sx) , (VP) (NP-SBJ))

The clause (Sx) inside the sentence (Sx ...(Sx) , (VP) (NP-SBJ)) , in which VP is an attribution verb, is split from this sentence.

10\. (Sx (NP-SBJ) , (Sx), (VP))

The clause (Sx) inside the sentence (Sx (NP-SBJ) , (Sx), (VP)) , in which VP is an attribution verb, is split from this sentence.

11\. (Sx (ADVP) , (Sx) , (NP-SBJ) (VP))

The clause (Sx) inside the sentence (Sx (ADVP) , (Sx) , (NP-SBJ) (VP)) , in which VP is an attribution verb, is split from this sentence.

12\. (Sx (VP) (NP-SBJ) , (Sx))

The clause (Sx) inside the sentence (Sx (VP) (NP-SBJ) , (Sx)) , in which VP is an attribution verb, is split from this sentence.

13\. (S (NP-SBJ) (VP .... (SBAR) <conjunction> (SBAR))...)

The clauses (SBAR) are split from the sentence (S (NP-SBJ) (VP .... (SBAR) <conjunction> (SBAR))...).

14\. (Sx ... (PS), (NP-SBJ) (VP))

The clause (PS) is split from the sentence (Sx ... (PS), (NP-SBJ) (VP)).

15. (VP (ADS))

The clause (ADS) is split from the verb phrase (VP (ADS)).

16. (VP ... (SBAR) (SBAR))

The clauses (SBAR) are split from the verb phrase (VP ... (SBAR) (SBAR)).

17. (VP (NP|PP) (SBAR|RRC))

The clause (SBAR|RRC) is split from the verb phrase (VP (NP|PP) (SBAR|RRC)).

18. (VP (NP|PP) (VP|Sx|PS))

The clause (VP|Sx|PS) is split from the verb phrase (VP (NP|PP) (VP|Sx|PS)).

Appendix 6

Conditions to Posit Rhetorical Relations[22]

1 – Sequence (multi-nuclear)

A *Sequence* is a list of events presented in chronological order. For example, the span "*The president could call*" has a *Sequence* relation with the span "*and declare that he would single-handedly kill the BART funds unless the congressman "shapes up" on the foreign-policy issue*" in Example (1).

(1) [The president could call][and declare that he would single-handedly kill the BART funds unless the congressman "shapes up" on the foreign-policy issue.]

Index	Necessary Condition
1	Two units are two co-ordinate clauses or two sentences.
2	If both units have subjects and do not contain attribution verbs, then these subjects need to meet the following requirement: they must either be the same, identical, synonyms, co-hyponyms, or hypernyms/hyponyms, or the subject of $Unit_2$ is a pronoun or a noun phrase that can replace the subject of $Unit_1$.
3	There is an explicit indication that the event expressed by $Unit_1$ temporally precedes the event expressed by $Unit_2$.
4	The *Contrast* relation is not satisfied.

Table A6.1. Necessary Conditions for the *Sequence* Relation

Index	Heuristic Rule	Score
1	$Unit_2$ contains a *Sequence* cue phrase.	100

[22] The definitions of rhetorical relations in this appendix are from Mann and Thompson (1988) and Carlson et al. (2002).

2	Both units contain enumeration conjunctions (*first, second, third...*).	100
3	Both subjects of $Unit_1$ and $Unit_2$ contain NP cues.	90
4	$Unit_2$ contains a VP cue.	90
5	Both units are clauses in which verb phrases agree in tense.	20

Table A6.2. Heuristic Rules for the *Sequence* Relation

2 – Contrast (multi-nuclear)

In a *Contrast* relation, two nuclei come in contrast with each other along some dimension. The contrast may happen in only one or a few respects, while everything else can remain the same in other respects.

For example:

(2) [In an age of specialisation, the federal judiciary is one of the last bastions of the generalist. A judge must jump from murder to antitrust cases, from arson to securities fraud, without missing a beat.][*But* even on the federal bench, specialisation is creeping in, and it has become a subject of sharp controversy on the newest federal appeals court.]

Index	Necessary Condition
1	Two units are coordinate.
2	The subject of $Unit_2$ is not a demonstrative pronoun, nor it is modified by a demonstrative.

Table A6.3. Necessary Conditions for the *Contrast* Relation

Index	Heuristic Rule	Score
1	$Unit_2$ contains a *Contrast* cue phrase.	100
2	The VP of $Unit_2$ contains a VP cue.	90

3	The main NP of $Unit_2$ contains a NP cue.	90
4	The main subjects of two units are co-hyponyms, or some/other.	50
5	$Unit_2$ has one of the structures: be incorrectly + attribution verb, be wrongly + attribution verb, attribution verb + by mistake.	40

Table A6.4. Heuristic Rules for the *Contrast* Relation

3 - Antithesis (mononuclear)

In an *Antithesis* relation, the situations presented in N and S are in contrast (see the *Contrast* relation); because of an incompatibility that arises from the contrast, one cannot have positive regard for both situations presented in N and S; comprehending S and the incompatibility between the situations presented in N and S increases R's positive regard for the situation presented in N. The *Antithesis* relation differs from the *Concession* relation, which is characterised by a violated expectation.

For example:

(3) [Kidder competitors aren't outwardly hostile to the firm, as many are to a tough competitor like Drexel Burnham Lambert Inc. that doesn't have Kidder's long history.] [*__However,__ competitors say that Kidder's hiring binge involving executive-level staffers, some with multiple-year contract guarantees, could backfire unless there are results.*]

There is no necessary condition for the *Antithesis* relation.

Index	Heuristic Rule	Score
1	$Unit_2$ contains an *Antithesis* cue phrase.	100
2	$Unit_2$ contains the cue phrase *but* or *however*, and the VP of $Unit_2$ contains the phrase *not* + a frequency adverb (e.g., *always, frequently, usually*).	80
3	$Unit_2$ contains *but* or *however*, and the VP of $Unit_2$ contains the phrase *not* + a degree adverb (e.g., *absolutely*, *quite*).	80
4	$Unit_2$ contains *but* or *however*, and the VP of $Unit_2$ contains one	80

	of the following words: *can, could, may, might, be able to.*	

Table A6.5. Heuristic Rules for the *Antithesis* Relation

4 – Concession (mononuclear)

This is a nuclear-satellite relation. In a *Concession* relation, the situation indicated in the nucleus is contrary to *expectation* in the light of the information presented in the satellite. In other words, a *Concession* relation is always characterised by *a violated expectation.* (Compare to *Antithesis.*) In some cases, the nuclearity role of spans in a *Concession* relation does not depend on the semantics of the spans, but rather on the intention of the writer.

For example:

(4) • [Its 1,400-member brokerage operation reported an estimated $5 million loss last year,] [*although Kidder expects it to turn a profit this year.*]

Index	Necessary Condition
1	The *Contrast* relation is not satisfied.

Table A6.6. Necessary Conditions for the *Concession* Relation

Index	Heuristic Rule	Score
1	$Unit_1$ or $Unit_2$ contains a *Concession* cue phrase. If two units are sentences and the cue phrase of $Unit_2$ is *yet, still, even*, $Unit_2$ is N. Otherwise, $Unit_2$ is S.	100

Table A6.7. Heuristic Rules for the *Concession* Relation

5 - Condition (mononuclear)

This is a nuclear-satellite relation. In a *Condition* relation, the truth of the proposition associated with the nucleus is a consequence of the fulfilment of the condition in the satellite. The satellite presents a situation that is not realised.

For example:

(5) [Kidder's hiring binge involving executive-level staffers, some with multiple-year contract guarantees, could backfire][*unless there are results.*]

Index	**Necessary Condition**
1	$Unit_2$ does not have a *Cause* cue phrase at the beginning or right after the *Condition* cue phrase.

Table A6.8. Necessary Conditions for the *Condition* Relation

Index	**Heuristic Rule**	**Score**
1	$Unit_2$ contains a *Condition* cue phrase.	100
2	The verb of $Unit_2$ contains a VP cue.	90
3	The subject of $Unit_2$ contains a NP cue and the main verb is *to be*.	90

Table A6.9. Heuristic Rules for the *Condition* Relation

6 – Otherwise (mononuclear or multi-nuclear)

An *Otherwise* relation is a mutually exclusive relation between two elements of equal importance. The situations presented by both the satellite and the nucleus are unrealised. Realising the situation associated with the nucleus will prevent the realisation of the consequences associated with the satellite.

For example:

(6) [The executive close to Saatchi and Saatchi said that "if a bidder came up with a ludicrously high offer, a crazy offer which Saatchi knew it couldn't beat, it would have no choice but to recommend it to shareholders.] [*But otherwise it would undoubtedly come back" with an offer by management.*]

There is no necessary condition for the *Otherwise* relation.

Index	Heuristic Rules	Score
1	$Unit_2$ contains an *Otherwise* cue phrase.	100
2	$Unit_2$ has the structure: if ... not ...	50

Table A6.10. Heuristic Rules for the *Otherwise* Relation

7 – Hypothetical (mononuclear)

In a *Hypothetical* relation, the satellite presents a situation that is not factual, but that one supposes or conjectures to be true. The nucleus presents the consequences that would arise should the situation come true. A *Hypothetical* relation presents a more abstract scenario than a *Condition* relation does.

For example:

(7) ["For some of these companies, this will be the first quarter with year-to-year negative comparisons," says Leonard Bogner, a chemical industry analyst at Prudential Bache Research.][*"This could be the first of five or six down quarters."*]

There is no necessary condition for the *Hypothetical* relation. The heuristic rules for the *Hypothetical* relation are shown in Table A6.11 below.

Index	Heuristic Rules	Score
1	$Unit_2$ contains a *Hypothetical* cue phrase.	100
2	The NP of $Unit_2$ is "*it*" or a demonstrative pronoun, and the main verb phrase of $Unit_2$ is *can|could|may|might + be*.	100
3	The VP of $Unit_2$ contains a VP cue.	90
4	The NP of $Unit_2$ contains a NP cue, and the main verb is *to be*.	90

Table A6.11. Heuristic Rules for the *Hypothetical* Relation

8 – Result (mononuclear)

The situation presented in the satellite is the result of the situation presented in the nucleus. The cause, which is the nucleus, is the most important part. The satellite represents the result of the action. When it is not clear whether the cause or result is more important, select the multi-nuclear relation *Cause-Result*.

For example:

(8) ["Those that pulled out (of stocks) regretted it," he said,][*"so I doubt you'll see any significant changes" in institutional portfolios as a result of Friday's decline.*]

There is no necessary condition for the *Result* relation.

Index	Heuristic Rule	Score
1	$Unit_2$ contains a *Result* cue phrase.	100
2	The VP of $Unit_2$ contains a VP cue.	90
3	The subject of $Unit_2$ contains a NP cue, and the verb is *to be*.	90
4	$Unit_2$ is subordinate $Unit_1$; $Unit_2$ is a detached *–ing* participial clause.	60

Table A6.13. Heuristic Rules for the *Result* Relation

9 – Cause (mononuclear)

The situation presented in the satellite is the cause of the situation presented in the nucleus. The result, which is the nucleus, is the most important part. In contrast to a *Purpose* relation, the situation presented in the nucleus of a *Cause* relation is factual, i.e., it is achieved.

For example:

(9) [A year earlier, operating profit in telephone operations was reduced by a similar amount][*as a result of a provision for a reorganization.*]

There is no necessary condition for the *Cause* relation.

Index	Heuristic Rule	Score
1	$Unit_1$ or $Unit_2$ contains a *Cause* cue phrase.	100
2	The VP of $Unit_2$ contains a VP cue.	90
3	The subject of $Unit_2$ contains a NP cue, and the verb is *to be*.	90
4	The object of $Unit_2$ contains a NP cue of *Result*, and the verb is *to be*.	60

Table A6.12. Heuristic Rules for the *Cause* Relation

10 - Cause-Result (multi-nuclear)

This is a causal relation in which two elementary discourse units, one representing the cause and the other representing the result, are of equal importance or weight. When either the cause or the result is more important, select the corresponding mononuclear relation *Cause* or *Result*, respectively.

For example:

(10) [And Judge Newman, a former patent lawyer, wrote in her dissent when the court denied a motion for a rehearing of the case by the full court,]["The panel's judicial legislation **has affected** an important high-technological industry, without regard to the consequences for research and innovation or the public interest."]

Index	Necessary Condition
1	Two units are coordinate.

Table A6.14. Necessary Conditions for the *Cause-Result* Relation

Index	Heuristic Rule	Score
1	$Unit_2$ contains a VP cue.	100

Table A6.15. Heuristic Rules for the *Cause-Result* Relation

11 – Purpose (mononuclear)

In contrast to a *Result* relation, the situation presented in the satellite of a *Purpose* relation is only putative, i.e., it is yet to be achieved. Most often it can be paraphrased as "*nucleus* in order to *satellite*." The purpose clause is the satellite. For example:

(11) [*To answer the brokerage question,*][Kidder, in typical fashion, completed a task-force study.]

Index	Necessary Condition
1	Two units are coordinate.
2	$Unit_2$ is not dominated by and does not contain cue phrases compatible with the *Condition* relation.

Table A6.16. Necessary Conditions for the *Purpose* Relation

Index	Heuristic Rule	Score
1	$Unit_2$ starts with a *Purpose* cue phrase.	100
2	The subject of $Unit_2$ contains a NP cue, and the verb is *to be*.	90
3	The VP of $Unit_2$ contains a VP cue.	90
4	The syntactic role of one unit has S–PRP (purpose or reason).	90
5	$Unit_1$ or $Unit_2$ starts with *To* + V.	90

Table A6.17. Heuristic Rules for the *Purpose* Relation

12 - Solutionhood (mononuclear or multi-nuclear)

The *Problem-Solution, Question-Answer,* and *Statement-Response* in (Carlson et al., 2002) are grouped into the *Solutionhood* relation in this thesis. In a *Solutionhood* relation, one span presents a problem/question/statement, and the other span presents a solution/answer/response. The relation may be mononuclear, depending on the context. For example, both spans in Example (12) below are nuclei of a *Solutionhood* relation.

(12) [With investment banking as Kidder's "lead business," where do Kidder's 42-branch brokerage network and its 1,400 brokers fit in? Mr. Carpenter this month sold off Kidder's eight brokerage offices in Florida and Puerto Rico to Merrill Lynch & Co., refuelling speculation that Kidder is getting out of the brokerage business entirely. Mr. Carpenter denies the speculation.][**To answer** the brokerage question, Kidder, in typical fashion, completed a task-force study...]

Index	**Necessary Condition**
1	Two units are coordinate.

Table A6.18. Necessary Conditions for the *Solutionhood* Relation

Index	**Heuristic Rule**	**Score**
1	$Unit_1$ contains a *Solutionhood* cue.	100
2	The subject of $Unit_2$ contains a NP cue, and the verb is *to be*.	90
3	The VP of $Unit_2$ contains a VP cue.	90
4	$Unit_2$ is in ellipsis situation and containing one of the words *how, why, what, who, whom, whose, which*	90

Table A6.19. Heuristic Rules for the *Solutionhood* Relation

13 – Manner (mononuclear)

A manner satellite explains the way in which something is done. (Sometimes it also expresses some sort of similarity/comparison.) The satellite answers the question "*in what manner?*" or "*in what way?*". A *Manner* relation is less "goal-oriented" than a *Means* relation, and often is more of a description of the style of an action. For example:

(13) [A judge must jump from murder to antitrust cases, from arson to securities fraud.] [*without missing a beat.*]

There is no necessary condition for the *Manner* relation.

Index	Heuristic Rule	Score
1	One unit starts with a *Manner* cue phrase.	100
2	The syntactic role of one unit contains –MNR.	90
3	$Unit_1$ is Ving + ADV.	70

Table A6.20. Heuristic Rules for the *Manner* Relation

14 – Means (mononuclear)

A means satellite specifies a method, mechanism, instrument, channel or conduit for accomplishing some goals. It should tell you how something was or is to be accomplished. In other words, the satellite answers a "*by which means?*" or "*how?*" question that can be assigned to the nucleus. It is often indicated by the preposition by. For example:

(14) [*By blocking this enzyme,*][the new compound, dubbed GS 4104, prevents the infection from spreading.]

There is no necessary condition for the *Means* relation.

Index	Heuristic Rule	Score
1	One unit starts with a *Means* cue phrase.	100

Table A6.21. Heuristic Rules for the *Manner* Relation

15 – Interpretation (mononuclear)

In an *Interpretation* relation, one side of the relation gives a different perspective on the situation presented in the other side. It is subjective, presenting the personal opinion of the writer or of a third party. An interpretation can be: (1) an explanation of what is not immediately plain or explicit; (2) an explanation of actions, events, or statements by pointing out or suggesting inner relationships, motives, or by relating particulars to general principles; or (3) an understanding or appreciation of a situation in light of individual belief, judgment, interest, or circumstance.

The interpretation may be mononuclear, with the interpretation occurring in the satellite or in the nucleus; or it may be multi-nuclear, with the interpretation occurring in one of the nucleus.

For example:

(15) [By the end of this year, 63-year-old Chairman Silas Cathcart -- the former chairman of Illinois Tool Works who was derided as a "tool-and-die man" when GE brought him in to clean up Kidder in 1987 -- retires to his Lake Forest, Ill., home, possibly to build a shopping mall on some land he owns. "I've done what I came to do" at Kidder, he says.][*And that means 42-year-old Michael Carpenter, president and chief executive since January, will for the first time take complete control of Kidder and try to make good on some grandiose plans. Mr. Carpenter says he will return Kidder to prominence as a great investment bank.*]

There is no necessary condition for the *Interpretation* relation.

Index	Heuristic Rule	Score
1	$Unit_2$ contains an *Interpretation* cue phrase.	100
2	$Unit_2$ has a subordinate clause and the main VP of $Unit_2$ contains a VP cue.	90
3	The subject of $Unit_2$ contains a NP cue.	90
4	$Unit_2$ has a subordinate clause and the main VP of $Unit_2$ is in report style.	80

Table A6.22. Heuristic Rules for the *Interpretation* Relation

16 – Evaluation (mononuclear or multi-nuclear)

In an *Evaluation* relation, one span assesses the situation presented in the other span of the relationship on a scale of good to bad. An evaluation can be an appraisal, estimation, rating, interpretation, or assessment of a situation. The evaluation can be the viewpoint of the writer or another agent in the text. The assessment may occur in a multi-nuclear relationship (*Evaluation*), when the spans representing the situation and the assessment are of equal weight. Example (16) is nucleus – satellite; whereas Example (17) is a multi-nuclear relation.

(16) [But racial gerrymandering is not the best way to accomplish that essential goal.] [*It is a quick fix for a complex problem.*]

(17) [Employers must deposit withholding taxes exceeding $3,000 within three days after payroll -- or pay stiff penalties --] [and that's a big problem for small businesses.]

Index	Necessary Condition
1	Two units are coordinate.
2	The subject of $Unit_2$ is a pronoun or a NP, which replaces the object of $Unit_1$.
3	There is an adjective after the main verb of $Unit_2$.
4	$Unit_2$ does not have a *Circumstance* cue phrase at the beginning or right after the *Evaluate* cue phrase (e.g., "*especially when*").

Table A6.23. Necessary Conditions for the *Evaluation* Relation

Index	Heuristic Rule	Score
1	$Unit_2$ contains an *Evaluation* cue phrase.	100
2	The verb of $Unit_2$ contains a VP cue.	90
3	The subject of $Unit_2$ contains a NP cue.	90
4	The main verb of $Unit_2$ is *to be* and the object contains a NP cue.	60
5	The VP of $Unit_2$ has the structure: verb + (adj+er/est).	50

Table A6.24. Heuristic Rules for the *Evaluation* Relation

17 – Summary (mononuclear)

In a *Summary* relation, one span summarises the information presented in another span. The former is shorter than the latter.

For example:

(18) [*In what could prove a major addition to the Philippines' foreign-investment portfolio, a Taiwanese company signed a $180 million construction contract to build the centerpiece of a planned petrochemical complex.*][Taiwan's USI Far East Corp., a petrochemical company, initialed the agreement with an unidentified Japanese contractor to build a naphtha cracker, according to Alson Lee, who heads the Philippine company set up to build and operate the complex. Mr. Lee, president of Luzon Petrochemical Corp., said the contract was signed Wednesday in Tokyo with USI Far East officials. Contract details, however, haven't been made public.]

There is no necessary condition for the *Summary* relation.

Index	Heuristic Rule	Score
1	$Unit_2$ starts with a *Summary* cue phrase.	100
2	The VP of $Unit_2$ contains a VP cue.	90
3	The subject of $Unit_2$ contains a NP cue and the verb of $Unit_2$ is *to be* or an attribution verb.	90

Table A6.25. Heuristic Rules for the *Summary* Relation

18 – Explanation (mononuclear)

The *Evidence*, *Justify* and the *Explanation-Argumentative* in (Carlson et al., 2002) are grouped into the *Explanation* relation in this thesis. In an *Explanation* relation, the satellite provides a factual explanation or justification for the situation presented in the nucleus.

For example:

(19) [Mr. Carpenter says that when he assumes full control, Kidder will finally tap the resources of GE.][*One of GE's goals when it bought 80% of Kidder in 1986 was to take advantage of "syngeries" between Kidder and General Electric Capital Corp., GE's corporate-finance unit, which has $42 billion in assets. The leveraged buy-out group of GE Capital now reports to Mr. Carpenter.*]

Index	Necessary Condition
1	Two units are coordinate.

Table A6.26. Necessary Conditions for the *Explanation* Relation

Index	Heuristic Rule	Score
1	$Unit_2$ contains an *Explanation* cue phrase.	100
2	The verb of $Unit_2$ contains a VP cue.	90
3	The subject of $Unit_2$ contains a NP cue and the main verb is *to be*.	90

Table A6.27. Heuristic Rules for the *Explanation* Relation

19 – Joint (multi-nuclear)

A *Joint* is not a rhetorical relation, but a pseudo-relation. By convention, *Joint* is a multi-nuclear relation. It is used when DAS cannot recognise any other relation between spans. There is no necessary condition and no heuristic rule for this relation.

Using Cohesive Devices to Recognize Rhetorical Relations in Text

Huong Le Thanh, Geetha Abeysinghe, and Christian Huyck
School of Computing Science, Middlesex University,
The Burroughs, London NW4 4BT, UK.
{H.Le, G.Abeysinghe, C.Huyck}@mdx.ac.uk

Abstract

This paper investigates factors that can be used in discourse analysis, specifically, cohesive devices. The paper shows that cohesive devices such as cue phrases can provide information about the linkages inside a text. We propose three types of cue phrases (the ordinary cue phrases, noun-phrase cues, and verb-phrase cues). An algorithm to compute rhetorical relations between two elementary discourse units is also presented.

1 Introduction

Rhetorical Structure Theory (RST) (Mann and Thompson, 1988) offers an explanation of the coherence of texts. It models the discourse structure of a text by a hierarchical tree diagram that labels relations between text spans (typically clauses or larger linguistic units). There are two kinds of relations: nucleus-satellite relation and multinuclear relations. A nucleus-satellite relation involves two nodes in which one node has a specific role relative to the other. The more important node between them in realising the writer's communicative goals is called a *nucleus*; the less important one is called a *satellite*. A multinuclear relation involve two or more nodes, each of which is equally important in realising the writer's communicative goals. RST can be applied in many fields, such as automatic summarisation, text generation, and text indexing.

Analysing textual rhetorical structures is difficult because discourse can be complex and vague. Many approaches in this area use cue phrases (such as "but", "however") to recognise rhetorical relations (e.g. Marcu, 1997; Corston-Oliver, 1998; Webber, 2001) because of their efficiency and simplicity. Cue phrases show a great potential in discourse analysis because most cue phrases have a specific discourse role. They indicate a rhetorical relation between different parts of a text. However, these approaches have problems when no cue phrases are found, which frequently happens.

This research carries out a study on textual coherence devices in order to solve this problem. The discourse parser we proposed involves the following three steps. Firstly, we split text into elementary discourse units (EDUs)[1]. Secondly, after defining EDUs, all potential rhetorical relations between these units are discovered. Finally, based on this relation set, all rhetorical structures will be produced using a discourse parser to combine small texts into larger ones.

This paper discusses step 2 of our proposed system. Different factors that can be used in identifying relations among discourse units are analyzed in Section 2. Section 3 describes the relation set and the method for recognizing relations. We present our conclusions in Section 4.

2 Factors Used in Recognizing Relations

2.1 Cohesive Devices

Cohesive devices are not the unique way to make text coherence. However, they are chosen in our research because of their efficiency and simplicity. Salkie (1995) presented different types of cohesive devices. We have considered a few of them to be implemented in our system. They are categorised into four groups: reiterative devices, reference words, ellipsis, and cue phrases.

The reiterative devices include synonyms (employer/boss), superordinates/hyponyms (country/Mexico), co-hyponyms (United Kingdom/Mexico), and antonyms (simple/complex). These devices are good factors in recognizing rhetorical relations. For example, antonyms often express a CONTRAST relation.

Reference words include personal pronouns (he,

[1] For further information on "EDUs", see (Marcu, 1997).

she, it, etc.), demonstratives (this, that, these, those), and comparative constructions (the same thing; a different person, etc.). Reference words need help from their environment to determine their full meaning. Thus, they create links between texts.

To benefit from reiterative and reference words, we extract the main noun phrases from the actor and the object of sentences. A thesaurus is then used to find the semantic relation between these noun phrases.[2] This cohension participates in deciding rhetorical relations of text.

Another important cohesive device is ellipsis. This is a special form of substitution, where only a part of a sentence is omitted. Ellipsis can be found by analyzing the syntax of the sentence. Ellipsis often occurs in question/answer sequences. Therefore, ellipsis can be used to recognize a SOLUTIONHOOD relation (see Section 3).

2.2 Cue Phrases

Cue phrases (such as "however", "as a result"), sometime called connectives or conjunctions, are used to indicate a specific connection between different parts of a text. This is the strongest cohesive device due to two reasons. Firstly, most cue phrases have a rhetorical meaning. If two text spans are connected by a cue phrase, their relation will be determined by the cue phrase's rhetorical meaning. Secondly, identifying cue phrases is quite simple because it is essentially based on pattern matching. Meanwhile, syntactic information is needed in order to explore other text devices such as synonyms and antonyms. Because of its strength and simplicity, there are many approaches that use cue phrases to recognize rhetorical relations (e.g., Knott and Dale, 1995; Marcu, 1997). However, as mentioned before, these approaches have problems when no cue phrase is found.

Our solution to this problem is to further expand the cue phrase concept. We propose three kinds of cue phrases:

1. Ordinary cue phrase (called cue phrase).
2. Special words or phrases in the main noun phrase (subject or object) of a sentence (called noun-phrase cue or NP cue).
3. Special words or phrases in the verb phrase of a sentence (called verb-phrase cue or VP cue).

Cue phrases must match exactly, whereas noun phrases and verb phrases are stemmed before being compared with NP/VP cue. Examples of NP and VP cues are shown in example (1) and example (2) respectively, below.

(1) [New York style pizza meets California ingredients,] [and the *result* is the pizza from this Church Street pizzeria.]

(2) [Chairman Silas Cathcart retires to his Lake Forest.] [That ***means*** Michael Carpenter will take complete control of Kidder.]

The noun "*result*" indicates a RESULT relation in example (1); meanwhile the verb "*means*" determines an INTERPRETATION relation in example (2).

A word/phrase can be a cue word/phrase in some cases, but not in the others. For example, the word "*and*" is a cue word in example (3), but not in example (4) as shown below.

(3) [Mary borrowed that book from our library last Monday,] [***and*** she returned it this morning.]

SEQUENCE

(4) Mary has a cat ***and*** a dog.

Some phrases (e.g., "*in spite of*") have a discourse meaning in all of their occurrences. Thus, each cue phrase has a different effect in deciding rhetorical relations. To control their strength, scores are assigned to different cue phrases.

If a word/phrase always has a discourse meaning and represents only one rhetorical relation, it will get the highest score, 1. If a word/phrase always has a discourse meaning and represents N relations (e.g., the cue phrase "*although*" expresses an ANTITHESIS relation or a CONCESSION relation), the score of that cue phrase for each type of relation will be 1/N. If a cue phrase only has a discourse meaning in some cases (e.g., "*and*"), its maximum score will be lower than 1.

Examples (3) and (4) show that the word's position is also important in deciding the word's discourse role. Therefore, if a word or a phrase has a discourse meaning in only some special positions inside a sentence, the information about its position will be given to the word/ phrase. If a word/phrase has a discourse role irrespective of its position in the sentence, no information will be provided about its position.

For example, the word "*second*" only has a discourse meaning when it stands at the beginning of a clause/sentence (indicated by the letter "B"). It has 50% certainty to be a LIST relation (hence given a

[2] We have chosen WordNet (WordNet, 2002), a machine-readable thesaurus and semantic network, for this purpose.

score of 0.5). Then it will be stored in the cue phrases' set for the LIST relation as "*second*(B, 0.5)".

Similarly, NP cues and VP cues also have scores depending on their strength in deciding rhetorical relations. Information involving ordinary cue phrases, NP cues, and VP cues (such as the relations that the cue represents for, and relation's score) are stored in text files for further use.

3 Relation Set and Relation Recognition

To generate a rhetorical structure from text, we need to decide which rhetorical relations,[3] and how many relations are enough. If we define just a few relations, the rhetorical trees will be easy to construct, but they will not be very informative. On the other hand, if we have a large relation set, the trees will be very informative; but they will be difficult to construct.

The RST discourse corpus consists of 78 rhetorical relation types. I is difficult to automatically construct RST trees based on such a large relation set. Therefore, we define a smaller set but sufficient to characterize relations by grouping similar relations into one. Based on the rhetorical relations that have been proposed in the literature, e.g., (Mann and Thompson, 1988), and (Hovy, 1990), the following set of 22 relations has been chosen to be used in our system:

LIST, SEQUENCE, CONDITION, OTHERWISE, HYPOTHETICAL, ANTITHESIS, CONTRAST, CONCESSION, CAUSE, RESULT, CAUSE-RESULT, PURPOSE, SOLUTIONHOOD, CIRCUMSTANCE, MANNER, MEANS, INTERPRETATION, EVALUATION, SUMMARY, ELABORATION, EXPLANATION, and JOINT.

3.1 Relation Recognition

Similar to (Corston-Oliver, 1998), we divide the features that help us to recognize a rhetorical relation into two parts:

(1) the conditions that two text spans must satisfy in order *to accept* a specific relation between them;

(2) and, the tokens used for *predicting* a relation.

We call the features in part (1) the necessary conditions and the features in part (2) the cue set. A cue set consists of heuristic rules involving cue phrases, NP cues, VP cues, and cohesive devices. The necessary conditions ensure that the two text spans have no conflict with the definition of the relation being tested. The necessary conditions may not consist of any token to realise a specific relation. The system can only recognise a rhetorical relation between two units if all necessary conditions and at least one cue are satisfied.

3.2 Scoring Heuristic Rules

Cue phrases, NP cues, VP cues, and cohesive devices have different effects in deciding rhetorical relations. Therefore, it is necessary to assign a score to each heuristic rule. The cue phrase's rule has the highest score of 1, as cue phrases are the strongest signal. NP cues and VP cues are the extension cases of cue phrases. They are also strong cues, but weaker than normal cue phrases. Thus, the heuristic rules involving NP cues and VP cues have the score of 0.9. The cohesive devices have lower scores than NP cues and VP cues. Depending on their certainty, the heuristic rules corresponding to these devices receive the scores of 0.2 to 0.8. It is of interest to note that each score can be understood as the percentage of cases in which the cue recognises a correct rhetorical relation. These scores are first assigned to heuristic rules according to human linguistic intuitions. After building the whole system, different sets of scores will be tested in order to find the optimal scores for the system.

As mentioned is Section 2.2, each cue phrase, NP cue or VP cue has its own score. It follows that the actual score for those cues is:

Actual Score = Score(heuristic rule) * Score(cue phrase, or NP cue, or VP cue).

The final score of a relation is equal to the sum of all heuristic rules contributing to that relation. The system will test the necessary conditions of that relation if its final score is more than or equal to a threshold θ.[4]

In the following section, we analyze the LIST relation to illustrate the usage of necessary conditions, cue set, and scores in recognizing relations between two EDUs.

3.3 Algorithm for recognising relations between two EDUs

As mentioned in Section 3.1, the heuristics rules in the cue set provide a suggestion of relations between

[3] For further information on "rhetorical relation", see (Mann and Thompson, 1988).

[4] Threshold θ is selected as 0.5.

two text spans. Thus, we start detecting relations between two text spans by testing the cue set, from the highest score rule to the lowest one. If several relations are recommended, the necessary conditions of these relations are checked in order to find the appropriate relations. Due to lack of space, a detailed description of this process is not presented in this paper. The pseudo-code for recognising relations between two EDUs is shown below:

```
Input: Two EDUs U1 and U2, list of
ordinary cue phrases (CPs), list of VP
cues, and list of NP cues.
Output: Relation set (R) between U1 and
U2.
1. Find all CPs of U1 and U2.
2. If CPs are found, compute actual score
   of the relations suggested by CPs.
3. Check necessary conditions (NCs) of the
   relations suggested by CPs whose actual
   score > θ.
4. Add the relations that satisfy NCs to
   (R).
5. If no relation satisfies, go to step 6.
   Otherwise, Return.
6. Find the main VP of each unit and stem
   them.
7. If one of these stemmed VPs consists of
   a VP cue, compute actual score of the
   stemmed VP and total score.[5]
8. Check NCs of the relations correspon-
   ding to the VP cue whose total score >
   θ.
9. Add the relations that satisfy NCs to
   (R).
10. If no relation satisfies, go to step
    11. Otherwise, Return.
11. Find the subject of each unit and
    stem these NPs.
12. If one of these stemmed NPs consists
    of a NP cue, compute actual score of
    the stemmed NP and total score.
13. Check NCs of the relations corres-
    ponding to the NP cue whose total
    score > θ.
14. Add the relations that satisfy NCs to
    (R).
15. If no relation satisfies, go to step
    16. Otherwise, Return.
16. For each of 22 relations in the
    proposed relation set:
    16.1. Check the remaining cues of the
          current relation (the cues that
          do not involve ordinary CP, VP
          cues, and NP cues).
    16.2. Compute total score of the
          relations suggested by cues.
    16.3. Check NCs of the relations whose
          total score > θ.
    16.4. Add the relations that satisfy NCs
          to (R).
17. Return.
```

[5] Total score is the accumulated scores of heuristic rules up to the current time.

In the following section, we analyze the LIST relation to illustrate the usage of necessary conditions, cue set, scores, and the algorithm for recognizing relations between two EDUs.

3.4 LIST Relation

A LIST is a multinuclear relation whose elements can be listed, but not in a CONTRAST or other stronger types of multinuclear relation (Carlson and Marcu, 2001). A LIST relation is often considered as a SEQUENCE relation if there is an explicit indication of temporal sequence.

The necessary conditions for a LIST relation between two units, $Unit_1$ and $Unit_2$, are shown below:

1. Two units are syntactically co-ordinates.
2. If both units have subjects and do not follow the reported style, then these subjects need to meet the following requirement: they must either be identical or be synonym, co-hyponym, or superordinate/hyponym; or the subject of $Unit_2$ is a pronoun or a noun phrase that can replace the subject of $Unit_1$.
3. There is no explicit indication that the event expressed by $Unit_1$ temporally precedes the event expressed by $Unit_2$.
4. The CONTRAST relation is not satisfied.

The first condition is based on syntactic information to guarantee that the two units are syntactically independent. The second condition checks the linkage between the two units by using reiterative and co-reference devices. The third condition distinguishes a LIST relation from a SEQUENCE relation. The last condition ensures that the stronger relation, CONTRAST, is not present in that context. In order to check this condition, the CONTRAST relation is always examined before the LIST relation.

The cue set of the LIST relation is shown below:

1. $Unit_2$ contains a LIST cue phrase. Score: 1
2. Both units contain enumeration conjunctions (*first, second, third...*). Score: 1
3. Both subjects of $Unit_1$ and $Unit_2$ contain NP cues. Score: 0.9
4. If both units are reported sentences, they mention the same object. Score: 0.8
5. If the subjects of two units are co-hyponyms, then the verb phrase of $Unit_2$ must be the same as

the verb phrase of $Unit_1$, or $Unit_2$ should have the structure "*so + auxiliary + sbj*". Score: 0.8

6. Both units are clauses in which verb phrases agree in tense (e.g., past, present). Score: 0.5

For example, the cue "*also*" in sentence (5.2) suggests a LIST relation between unit (5.1) and unit (5.2) in the following case: [6]

(5) [Mr. Cathcart is credited with bringing some basic budgeting to traditionally freewheeling Kidder.[5.1]] [He *also* improved the firm's compliance procedures for trading.[5.2]]

The actual score of cue 1, with the cue word "*also*", is equal to Score(cue 1) * Score("*also*"). The cue word "*also*" has the score of 1 for the LIST relation, so the actual score is $1*1=1>\theta$. Therefore, the necessary conditions of the LIST relation are checked. Text spans (5.1) and (5.2) are two sentences, thus they syntactically coordinate (condition 1). In addition, the subject of the text span (5.2), "*he*", is a pronoun, which replaces the subject of the text span (5.1), "*Mr. Cathcart*" (condition 2). There is no evidence of an increasingly temporal sequence (condition 3), and also no signal of a CONTRAST relation (condition 4). Therefore, a LIST relation is recognized between text spans (5.1) and (5.2).

The cue word "*and*" is found in example (6):

(6)[But the Reagan administration thought otherwise,[6.1]] [*and* so may the Bush administration.[6.2]]

"*And*" is considered as a cue word because it stands at the beginning of the clause (6.2) (cue 1). It can be used in a LIST relation, a SEQUENCE relation, or an ELABORATION relation. With the score of 0.3 for the cue word "*and*" in the LIST relation, the actual score of cue 1 = Score(cue 1)*Score("*and*") = $1*0.3 = 0.3 < \theta$. Also, another cue of the LIST relation is found between clause (6.1) and clause (6.2). The subjects of two text spans, "*the Reagan administration*" and "*the Bush administration*", are co-hyponyms. In addition, clause (6.2) has the structure "*so + auxiliary + sbj*". With the satisfaction of cue 1 and cue 5, the total score is:

Total score = Actual Score(cue 1) + Score(cue 5) $= 0.3 + 0.8 = 1.1 > \theta$.

As in the previous example, the necessary conditions of the LIST relation are checked and then a LIST relation is recognized between clause (6.1) and clause (6.2).

4 Conclusion

In this paper, we have explored several variants of cue phrases, and exploring combining with other feasible cohesive devices to recognise relations between two text spans. It was shown that NP cues, and VP cues are good predictors for discovering rhetorical relations. In the case where cue phrases are not available, other text cohesive devices (e.g., synonyms, and antonyms) can be a reasonable substitution.

The algorithm for recognising relations between two text spans is being implemented. The evaluation will be done by using documents from the RST Discourse Treebank after the completion of the implementation. Future work will focus on improving this algorithm's performance by refining the conditions to recognise relations mentioned in Section 3.1.

References

Carlson, L. and Marcu, D. (2001) *Discourse Tagging Manual.* ISI Tech Report ISI-TR-545.

Corston-Oliver, S. (1998). *Computing Representations of the Structure of Written Discourse.* PhD Thesis, University of California, U.S.A.

Hovy, E. H. (1990) *Parsimonious and profligate approaches to the question of discourse structure relations.* Proceedings of the 5th International Workshop on Natural Language Generation, Pittsburgh, 128-136.

Knott, A., Dale, R. (1995) *Using linguistic phenomena to motivate a set of coherence relations.* Discourse Processes 18:35-62.

Mann, W. C. and Thompson, S. A. (1988) *Rhetorical Structure Theory: Toward a Functional Theory of Text Organization.* Text, vol. 8(3), 243-281.

Marcu, D. (1997) *The Rhetorical Parsing, Summarization, and Generation of Natural Language Texts.* PhD Thesis, Department of Computer Science, University of Toronto.

RST Discourse Treebank - http://www.ldc.upenn.edu/Catalog/LDC2002T07.html

Salkie, R. (1995) *Text and discourse analysis.* London: Routledge.

Webber, B. et al. (2001) *D-LTAG System - Discourse Parsing with a Lexicalized Tree Adjoining Grammar.* In ESSLLI 2001 Workshop on Information structure, Discourse structure and Discourse Semantics.

WordNet (2002) - http://www.cogsci.princeton.edu/~wn/index.shtml

[6] The superscripts such as 5.1 and 5.2 are used to distinguish different discourse units focussed on in each example.

A Study to Improve the Efficiency of a Discourse Parsing System

Huong T. Le and Geetha Abeysinghe

School of Computing Science, Middlesex University,
The Burroughs, London NW4 4BT, UK.
{H.Le, G.Abeysinghe}@mdx.ac.uk

Abstract. This paper presents a study of the implementation of a discourse parsing system, where only significant features are considered. Rhetorical relations are recognized based on three types of cue phrases (the normal cue phrases, Noun-Phrase cues and Verb-Phrase cues), and different textual coherence devices. The parsing algorithm and its rule set are developed in order to create a system with high accuracy and low complexity. The data used in this system are taken from the RST Discourse Treebank of the Linguistic Data Consortium (LDC).

1 Introduction

Rhetorical Structure Theory (RST) (Mann and Thompson, 1988) is a method of structured description of text. It provides a general way to describe the relations among clauses in a text, whether or not they are grammatically or lexically signaled. RST can be applied in many fields, such as automatic text summarization, text generation and text indexing.

Recognizing textual rhetorical structures still remains a hard problem because discourse is complex and vague. Literature shows that a considerable amount of work has been carried out in this area. However, only a few algorithms for implementing rhetorical structures have been proposed so far.

One of the pioneering works has been proposed by Marcu (1997). His advanced discourse parser is based on cue phrases, and therefore faces problems when cue phrases are not present in the text. Corston-Oliver (1998a) improved Marcu's system by integrating cue phrases with anaphora, deixis and referential continuity. Webber (2001) started from a different approach by implementing a discourse parsing system for a Lexicalized Tree Adjoining Grammar (LTAG). Webber developed a grammar that uses discourse cue as an anchor to connect textual trees. Like Marcu's system, Webber's parser too cannot recognize relations when there is no cue phrase present in the text.

Another trend in discourse analysis is learning-based, such as the decision-based approach i(Marcu, 1999) and the unsupervised one (Marcu and Echihabi, 2002). This approach produces an impressive result but requires a large enough corpus for training purpose to be available. Such a sufficient discourse corpus is difficult to find[1].

[1] The biggest discourse corpus nowadays is the RST Discourse Treebank from LDC, with 385 Wall Street Journal articles.

A. Gelbukh (Ed.): CICLing 2003, LNCS 2588, pp. 101–114, 2003.

We take the approach proposed by Marcu (1997) and extended by Corston-Oliver (1998a), and concentrate on improving the efficiency of the discourse parser. We proposed to do this by several ways: improving the correctness of dividing text into elementary discourse units (edus)[2] by combining syntactic-based method with cue-phrase-based method; using cohesive devices as relation's predictors; refining rules for the discourse parser; and improving Corston-Oliver's parser to reduce its complexity. The data used in the experiment are the discourse documents from The RST Discourse Treebank.

Our discourse analysis involves the following three computational steps. Firstly, we split text into elementary discourse units. Secondly, after defining edus, all potential rhetorical relations between these units are discovered. Finally, based on this relation set, all rhetorical structures will be produced using a discourse parser to combine small texts into larger ones. The basic framework for our discourse analysis system is depicted in Figure 1.

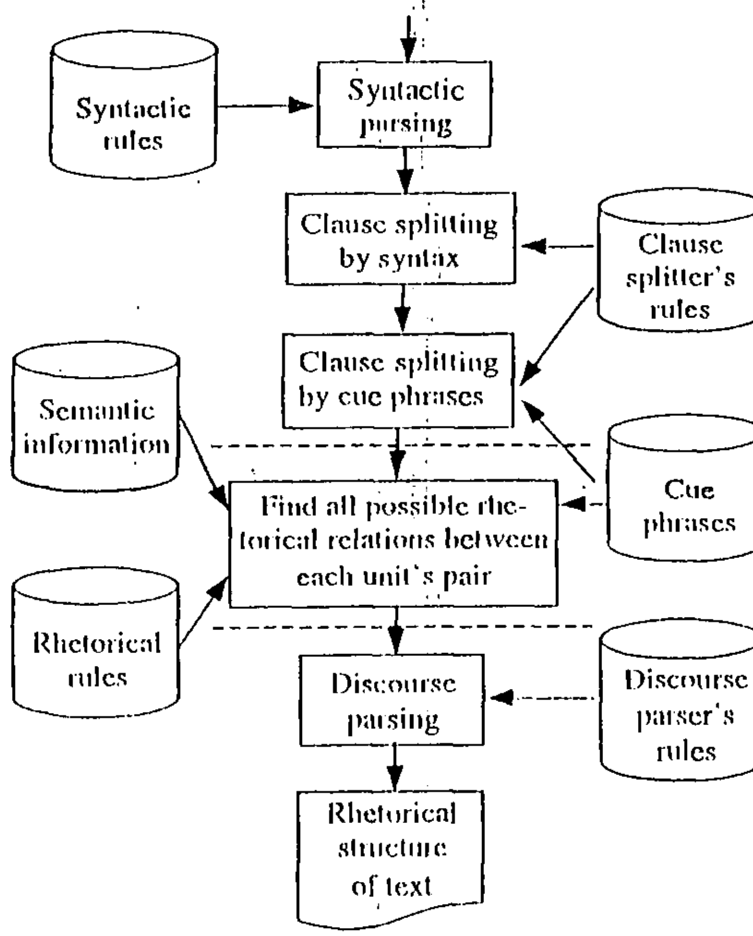


Fig. 1. The framework for a Discourse Analyzing System

The way of dividing text into elementary discourse units is discussed in Section 2. Section 3 analyzes different factors that can be used in deciding rhetorical relations among discourse units. The relation set and the method for recognizing relations are described in Section 4. The discourse parser and its rule set are discussed in Section 5. We present our conclusions in Section 6.

2 Identifying Elementary Discourse Units

According to Mann and Thompson (1988), each discourse unit should have an independent functional integrity. Thus, a discourse unit can be a clause in a sentence or a single sentence. Marcu (1997) identifies edus based on regular expressions of cue phrases. If all edus contain cue phrases, this method is simple and very efficient since only a shallow parsing is required. However, Redeker (1990) has found that only 50%

[2] For further information on "edus", see (Marcu, 1997).

of clauses contain cue phrases. Marcu has not provided any solution to deal with the non-cue phrase cases, and his system fails in this situation. In addition, the use of cue phrases in Marcu's system does not guarantee to produce correct edus. Cue phrases do not provide any syntactic information; hence the edus generated by his system might not have an independent functional integrity.

Instead of using cue phrases, Corston-Oliver (1998a) implemented a syntactic parser and then used syntactic information to identify edus. This method suffers from high complexity, but can solve the problems faced by Marcu's system (Marcu, 1997). Corston-Oliver's parser did not process correctly the case where strong cue phrases make noun phrases become a separate edu. Two edus shown below in example (1) are considered as one edu in Corston-Oliver's parser:

(1) [*According to a Kidder World story about Mr. Megargel,*] [all the firm has to do is "position ourselves more in the deal flow."]

To deal with this problem, we divide the task of identifying edus into two processes. First, the system uses syntactic information to split text. In order to get the syntactic information, a syntactic parser is integrated to the system. Then, the system seeks strong cue phrases from the splitted text to make a further splitting when cue phrases are found, as in example (1). Due to lack of space, a detail description of this process is not presented in this paper.

3 Factors Used for Recognizing Relations

3.1 Text Cohesion as Relation's Predictors

Syntax provides us with information about how words combine to form sentences. What it does not show is how sentences combine to form an understandable and informative text. This is the role of text and discourse analysis. Cohesion can fill up this gap. They seek linguistic features and analyze their occurrence. Text can therefore be evaluated according to how cohesive they are. Cohesive devices are not the unique factor to make text coherence. However, they are chosen here because of their efficiency and simplicity. Salkie (1995) presented different types of cohesive devices. We have considered a few of them to be implemented in our system. They are synonyms, superordinates/hyponyms, opposite words, ellipsis, reference words and connectives. These cohesive devices are categorized into four groups: reiterative devices, reference words, ellipsis and cue phrases.

The Reiterative devices include synonyms (employer/boss), superordinates/hyponyms (country/Mexico), co-hyponyms (United Kingdom/Mexico), and antonyms (simple/complex). They are important features to define relations. For example, co-hyponyms (or multiple opposites), binary opposites (male/female) and antonyms often express a CONTRAST relation.

The Reference words include personal pronouns (I, you, he, she, it, we, they), their object forms (me, him, etc.) and their possessive forms (my, mine, your, yours, etc.), demonstratives (this, that, these, those) or comparative constructions (the same thing, a different person, etc.). Reference words need help from their environment to determine their full meaning. Thus, they create links between texts.

Another important cohesive device is ellipsis. This is a special form of substitution, where only a part of a sentence is omitted. Ellipsis can be found by analyzing syntax of the sentence. The ellipsis situation often occurs in question/answer sequences. Therefore, ellipsis can be used to recognize the SOLUTIONHOOD relation (see Section 4).

In order to recognize the reiteration and reference words from text, a lexical database is required. We have chosen WordNet for this purpose. It is a machine-readable thesaurus and semantic network developed and maintained by the Cognitive Science Laboratory at Princeton University. Two kinds of relations are represented in this database: lexical and semantic. Lexical relations hold between word forms, whereas semantic relations hold between word meanings. These relations include hypernymy/hyponymy, antonymy, entailment, and meronymy/holonymy.

3.2 Cue Phrases

Cue phrases (e.g., however, as a result), sometime called connectives or conjunctions, are used to indicate a specific connection between different parts of a text. This is the strongest cohesive device due to two reasons. Firstly, most cue phrases have a rhetorical meaning. If two text spans are connected by a cue phrase, their relation will be determined by the cue phrase's rhetorical meaning. Secondly, identifying cue phrases is quite simple because it is essentially based on pattern matching. Syntactic information is needed in order to explore other text devices such as synonyms and antonyms. Because of its strength and simplicity, there are many approaches which use cue phrases to recognize rhetorical relations (Knott and Dale, 1995; Marcu, 1997). However, these approaches have problems when no cue phrase is found.

One solution to this problem is to further expand the cue phrase's definition. We propose three kinds of cue phrases:

1. Normal cue phrase (called cue phrase) ;
2. Special words or phrases in a main noun phrase of a sentence (called Noun-Phrase cue or NP cue);
3. Special words or phrases in a verb phrase of a sentence (called Verb-Phrase cue or VP cue).

Cue phrases must match exactly, whereas noun phrases and verb phrases are simplified or stemmed before being compared with NP/VP cue. Examples of NP and VP cues are shown in (2) and (3), respectively, below.

(2) [New York style pizza meets California ingredients,] [and the *result* is the pizza from this Church Street pizzeria.]

(3) [By the end of this year, 63-year-old Chairman Silas Cathcart retires to his Lake Forest, Ill., home, possibly to build a shopping mall on some land he owns. "I've done what I came to do" at Kidder, he says.] [And that *means* 42-year-old Michael Carpenter, president and chief executive since January, will for the first time take complete control of Kidder and try to make good on some grandiose plans. Mr. Carpenter says he will return Kidder to prominence as a great investment bank.]

The noun "*result*" indicates a RESULT relation in example (2); meanwhile the VP cue "*means*" determines an INTERPRETATION relation in example (3).

A word/phrase can be a cue word/phrase in some cases, but this may not be in the others. For example, the word "*and*" is a cue word in example (4), but not so in example (5) as shown below.

(4) [Mary borrowed that book from our library last Monday,] [*and* she returned it this morning.]

SEQUENCE

(5) Mary has a cat *and* a dog.

In contrast, some phrases (e.g., "*in spite of*") have a discourse meaning in all of their occurrences. Thus, each cue phrase has a different effect in deciding rhetorical relations. To control their strength, scores are assigned to different cue phrases.

If a word/phrase always has a discourse meaning and represents only one rhetorical relation, it will get the highest score, 1. If a word/phrase always has a discourse meaning and represents N relations (e.g., cue phrase "*although*" can express an ANTITHESIS relation or a CONCESSION relation), the score of that cue phrase for each type of relation will be 1/N. If a cue phrase only has a discourse meaning in some cases (e.g., "*and*"), its maximum score will be lower than 1.

Examples (4) and (5) show that the word's position is also important in deciding the word's discourse role. Therefore, if a word or a phrase has a discourse meaning in only some special positions inside a sentence, the information about its position will be given to the word/phrase. If a word/phrase has a discourse role irrespective of its position in the sentence, no information will be provided about its position.

For example, the word "*second*" only has a discourse meaning when it stands at the beginning of a clause/sentence (indicated by the letter "B"). It has 50% certainty to be a LIST relation (hence given a score of 0.5). Then it will be stored in the cue phrases' set for the LIST relation as "*second*(B, 0.5)".

Similarly, NP cues and VP cues also have scores depending on their strength in deciding rhetorical relations.

4 Relation Set and Relation Recognition

To generate a rhetorical structure from text, we need to decide which rhetorical relations,[3] and how many relations are enough. If we define just a few relations, the rhetorical trees will be easy to construct; but they will not be very informative. On the other hand, if we have a large relation set, the trees will be very informative; but they will be difficult to construct.

The RST discourse corpus consists of 78 rhetorical relation types. It is difficult to automatically construct RST trees based on such a large relation set. Therefore, we define a smaller set but sufficient to characterize relations by grouping similar relations into one. Based on the rhetorical relations that have been proposed in the litera-

---

3 A rhetorical relation involves two or more text spans (typically clauses or larger linguistic units) related such that one of them has a specific role relative to the other. For further information on "rhetorical relation", see (Mann and Thompson, 1988).

ture, e.g., (Mann and Thompson, 1988), and (Hovy, 1990), the following set of 22 relations has been chosen to be used in our system:

LIST, SEQUENCE, CONDITION, OTHERWISE, HYPOTHETICAL, ANTITHESIS, CONTRAST, CONCESSION, CAUSE, RESULT, CAUSE-RESULT, PURPOSE, SOLUTIONHOOD, CIRCUMSTANCE, MANNER, MEANS, INTERPRETATION, EVALUATION, SUMMARY, ELABORATION, EXPLANATION, and JOINT.

4.1 Relation Recognition

Similar to Corston-Oliver (1998a), we divide the features, which help us to recognize a rhetorical relation, into two parts:

(1) the conditions that two text spans must satisfy in order to *accept* a specific relation between them;
(2) and, the tokens used for *predicting* a relation.

We call the features in part (1) as the necessary conditions and the features in part (2) as the Cue set. A Cue set consists of heuristic rules which involve cue phrases, NP cues, VP cues and cohesive devices. The necessary conditions ensure that the two text spans has no conflict with the concept of the relation being tested. The necessary conditions may not consist of any token to realize a specific relation. The system can only recognize a rhetorical relation between two units if all necessary conditions and at least one cue are satisfied.

Corston-Oliver tests the Cue set after the necessary conditions are satisfied. Thus, all rhetorical relations have to be checked sequentially one by one (thirteen relations are checked in his system).

The system that we propose detects relations in a different order. It first extracts cues from the two edus. When several relations are suggested by cues, the necessary conditions of these relations are checked in order to find the appropriate one. Since each cue represents one or two rhetorical relations in average, there are much less relations that need to be checked by our system. The definition of LIST relation discussed in Section 4.3 will further illustrate this idea.

4.2 Scoring Heuristic Rules

Cue phrases, NP cues, VP cues and cohesive devices have different effects in deciding rhetorical relations. Therefore, it is necessary to assign a score to each heuristic rule. The cue phrase's rule has the highest score of 1, as cue phrases are the strongest signal. NP cues and VP cues are the extension cases of cue phrases. They are also strong cues, but weaker than normal cue phrases. Thus, the heuristic rules involving NP cues and VP cues have the score of 0.9. The cohesive devices have lower scores than NP cues and VP cues. Depending on their certainty, the heuristic rules corresponding to these devices receive the scores of 0.2 to 0.8. It is of interest to notice that each score can be understood as the percentage of cases in which the cue recognizes a correct rhetorical relation.

Heuristic scores can be trained by evaluating the output of the discourse parser with RST trees in an existing discourse corpus. Unfortunately, no discourse corpus large enough for training purposes currently exists. For this reason, scores are first assigned to heuristic rules according to human linguistic intuitions. After building the whole system, different sets of scores will be tested in order to find the optimal scores for the system.

As mentioned is Section 3.2, each cue phrase, NP cue or VP cue has its own score. It follows that the actual score for those cues is:

Actual Score = Score(heuristic rule) * Score(cue phrase, or NP cue, or VP cue).

The final score of a relation is equal to the sum of all heuristic rules contributing to that relation. The system will test the necessary conditions of that relation if its final score is more than or equal to a threshold θ. [4]

In the following section, we analyze the LIST relation to illustrate the usage of necessary conditions, Cue set and scores in recognizing rhetorical relations between two edus.

4.3 LIST Relation

A LIST is a multinuclear relation whose elements can be listed, but not in a CONTRAST or other stronger type of multinuclear relation. A LIST exhibits some sort of parallel structure between the units involved in the relation (Carlson and Marcu, 2001). A LIST relation is often considered as a SEQUENCE relation if there is an explicit indication of temporal sequence.

The necessary conditions for a LIST relation between two units, $Unit_1$ and $Unit_2$, are shown below:

1. Two units are syntactically co-ordinates.
2. If both units have subjects and do not follow the reported style, then these subjects need to meet the following requirement: they must either be identical or be synonym, co-hyponym, or superordinate/hyponym; or the subject of $Unit_2$ is a pronoun or a noun phrase that can replace the subject of $Unit_1$.
3. There is no explicit indication that the event expressed by $Unit_1$ temporally precedes the event expressed by $Unit_2$.
4. The CONTRAST relation is not satisfied.

The first condition is based on syntactic information to guarantee that the two units are syntactically independent. The second condition checks the linkage between the two units by using reiterative and co-reference devices. Syntactic and semantic information are used to determine these units' subjects and their relations. The third condition distinguishes a LIST relation from a SEQUENCE relation. The last condition ensures that the stronger relation, CONTRAST, is not present in that context. In order to check this condition, the CONTRAST relation is always examined before the LIST relation.

The cue set of the LIST relation is shown below:

---

[4] Threshold θ is selected as 0.5.

1. $Unit_2$ contains a LIST cue phrase. Score: 1
2. Both units contain enumeration conjunctions (*first, second, third...*). Score: 1
3. Both subjects of $Unit_1$ and $Unit_2$ contain NP cues. Score: 0.9
4. If both units are reported sentences, they mention the same object. Score: 0.8
5. If the subjects of two units are co-hyponyms, then the verb phrase of $Unit_2$ must be the same as the verb phrase of $Unit_1$, or $Unit_2$ should have the structure "*so + auxiliary + sbj*". Score: 0.8
6. Both units are clauses in which verb phrases agree in tense (e.g., past, present). Score: 0.5
7. Both units are sentences in which verb phrases agree in tense (e.g., past, present). Score: 0.2

For example, the cue word "*also*" in the sentence "He *also* improved the firm's compliance procedures for trading" suggests a LIST relation between two discourse units (6.1) and (6.2) in the following case [5]:

(6) [Mr. Cathcart is credited with bringing some basic budgeting to traditionally free-wheeling Kidder.[6.1]] [He *also* improved the firm's compliance procedures for trading.[6.2]]

Since only cue 1 is satisfied in this case, the final score is:

Final score = Actual score(cue 1) = Score(cue 1) * Score("*also*"). The cue word "*also*" has the score of 1 for the LIST relation, so the final score is 1 * 1 = 1 > 0. Therefore, the necessary conditions of the LIST relation are checked. Text spans (6.1) and (6.2) are two sentences, thus they are syntactically coordinate (condition 1). In addition, the subject of text span (6.2), "*he*", is a pronoun, which replaces for the subject of text span (6.1), "Mr. Cathcart" (condition 2). There is no evidence of an increasingly temporal sequence (condition 3), and also no signal of a CONTRAST relation (condition 4). Therefore, a LIST relation is recognized between text spans (6.1) and (6.2).

The cue word "*and*" is found in example (7):

(7) [But the Reagan administration thought otherwise,[7.1]] [*and* so may the Bush administration.[7.2]]

"*And*" is considered as a cue word because it stands at the beginning of clause (7.2) (cue 1). The subjects of two text span, "the Reagan administration" and "the Bush administration", are co-hyponyms. In addition, clause (7.2) has the structure "*so + auxiliary + sbj*". With the score of 0.3 for the cue word "*and*" in the LIST relation, and with the satisfaction of cue 5, the final score is:

Final score = Score(cue 1) * Score("*and*") + Score(cue 5) = 1 * 0.3 + 0.8 = 1.1 > 0.

As in the previous example, the necessary conditions of the LIST relation are checked and then a LIST relation is recognized between clause (7.1) and clause (7.2).

---

[5] The superscripts such as 6.1 and 6.2 are used to distinguish different discourse units focussed on in each example.

5 Rhetorical Parser

5.1 Rules for the Rhetorical Parser

Rhetorical rules are constraints of text spans in a RST tree. They are used in a discourse parser to find rhetorical relations between non-elementary discourse units. To formalize these rules, the following definitions are applied:

- $<T>$ is a text span that can be presented by a RST tree, a RST subtree, or a leaf.
- $<T_i\ T_j>$ is a text span in which a rhetorical relation exists between two adjacent text spans $<T_i>$ and $<T_j>$. The possible roles of $<T_i>$ and $<T_j>$ in a rhetorical relation are Nucleus – Nucleus, Nucleus – Satellite, and Satellite – Nucleus. These cases are coded as $<T_i\ T_j \mid NN>$, $<T_i\ T_j \mid NS>$, and $<T_i\ T_j \mid SN>$, respectively.
- rhet_rels($<T_i>$,$<T_j>$) is the rhetorical relations between two adjacent text spans $<T_i>$ and $<T_j>$, each of which has a corresponding RST tree.

The paradigm rules in our proposed system are shown below:

Rule 1:
rhet_rels($<T_1\ T_2 \mid NN>$, $<T>$) $\equiv$ rhet_rels($<T_1>$, $<T>$) $\cap$ rhet_rels($<T_2>$, $<T>$).

<u>If:</u> there is a relation between two text spans $<T_1>$ and $<T_2>$, in which both of them play the nucleus roles,
<u>Then:</u> the rhetorical relations between the text span $<T_1\ T_2>$ and its right-adjacent text span T hold only when they hold between $<T_1>$ and $<T>$, and between $<T_2>$ and $<T>$.

Rule 2: rhet_rels($<T_1\ T_2 \mid NS>$, $<T>$) $\equiv$ rhet_rels($<T_1>$, $<T>$).
Rule 3: rhet_rels($<T_1\ T_2 \mid SN>$, $<T>$) $\equiv$ rhet_rels($<T_2>$, $<T>$).
Rule 4: rhet_rels($<T>$, $<T_1\ T_2 \mid NS>$) $\equiv$ rhet_rels($<T>$, $<T_1>$).

Rules 1-4 are based on the proposal of Marcu (1997) which states, "*If a rhetorical relation R holds between two text spans of the tree structure of a text, that relation also holds between the most important units of the constituent spans*". From this point of view, Marcu (1997) and Corston-Oliver (1998a) analyzed relations between two text spans by considering only their nuclei.

However, the rule with the left side rhet_rels($<T>$, $<T_1\ T_2 \mid SN>$), is not formalized in the same way as rules 1-4. This is a special case which has not been solved in (Marcu, 1997) and (Corston-Oliver, 1998a). This case is illustrated by example (8) below:

(8) [With investment banking as Kidder's "lead business," where do Kidder's 42-branch brokerage network and its 1,400 brokers fit in? Mr. Carpenter this month sold off Kidder's eight brokerage offices in Florida and Puerto Rico to Merrill Lynch & Co., refueling speculation that Kidder is getting out of the brokerage business entirely. Mr. Carpenter denies the speculation.[8.1]] [[*To answer the brokerage question,*[8.2]] [Kidder, in typical fashion, completed a task-force study....[8.3]]]

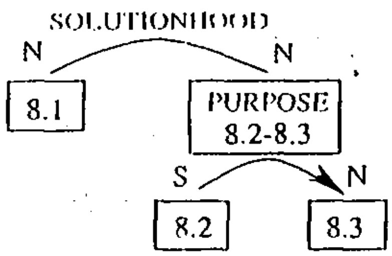


Fig. 2. The discourse tree of text (8)

The cue "*To (+Verb)*" in text span (8.2) indicates a PURPOSE relation between two text spans (8.2) and (8.3), while the VP cue "*answer*" in text span (8.2) indicates a SOLUTIONHOOD relation between two larger text spans (8.1) and (8.2-8.3).

Example (8) shows that although the content of the satellite does not determine rhetorical relations of its parent text span, special cue phrases inside the satellite are still a valuable source. We apply a different treatment in this situation than the rules proposed by Marcu (1997), as shown below.

To recognize the relations rhet_rels(<T>, <T_1 T_2 | SN>), we firstly find all cue phrases *restCPs* in text span <T_1> which have not been used to create the relation between <T_1> and <T_2>, then check rhet_rels(<T>, <T_1>) by using *restCPs*. If a relation is found, it is assigned to rhet_rels(<T>, <T_1 T_2| SN>). Otherwise, rhet_rels(<T>, <T_1 T_2| SN>) ≡ rhet_rels(<T> <T_2>).

Applying this rule to example (8) with two text spans (8.1) and (8.2-8.3), we have *restCPs* = "*answer*" since the cue "*To*" is used for the relation between (8.2) and (8.3). The relation between (8.1) and (8.2-8.3) is recognized as SOLUTIONHOOD by using the cue "*answer*" in *restCPs*. In contrast, if we use the Marcu's rules, rhet_rels((8.1), (8.2 8.3 | SN)) = rhet_rels((8.1), (8.3)). That means the cue "*answer*" is not considered in this case.

5.2 Algorithm for Rhetorical Parser

The idea for this algorithm was first introduced by Marcu (1996) and then further developed by Corston-Oliver (1998a). Marcu proposed a shallow, cue-phrase-based approach to discourse parsing. Marcu's system splits text into edus and hypothesizes theirs rhetorical relations based on the appearance of cue phrases. Then, all the RST trees compatible with the hypothesized relations are generated. Although Marcu's discourse parser was considerably advanced at that time, it still had weaknesses. When the number of hypothesized relations increases, the number of possible RST trees increase exponentially. Marcu's parser creates all possible pairs of text spans by permutation operations without considering of their usefulness. As a result, a huge amount of ill-formed trees are created.

The improved algorithm in RASTA, proposed by Corston-Oliver (1998a), solves this problem by using a recursive, backtracking algorithm that produces only well-formed trees. If RASTA finds a combination of two text spans leading to an ill-formed tree, it will backtrack and go to another direction, thus reducing the search space. By applying the higher score hypotheses before the lower ones, RASTA tend to produce the most reliable RST trees first. Thus, RASTA can stop after a number of RST trees are built.

Although a lot of improvement had been made, RASTA's search space is still not optimal. Given the set of edus, RASTA checks each pair of edus to determine rhetorical relations. With N edus $\{U_1, U_2, ..., U_N\}$, N(N-1) pairs of edus $\{(U_1,U_2),$

$(U_1,U_3),\ldots,(U_1,U_N),(U_2,U_3),\ldots,(U_{N-1},U_N)\}$ are examined. Then, all possible relations are tested in order to build RST trees.

The search space in our system is much less than that in RASTA. Since only two adjacent text spans can be combined to a larger text span, only N-1 pairs of edus (U_1,U_2), (U_2,U_3), ..., (U_{N-1},U_N) are selected. Instead of checking every pair of edus as in RASTA, only N-1 pairs of adjacent edus are examined by our system. The relations recognized by this examination are called hypothesis relations (or hypotheses). They are stored in a hypothesis set. Relations in this set will be called from the highest score to the lowest score ones.

To illustrate this idea, we consider a text with four edus U_1, U_2, U_3, U_4, and the hypothesis set H of these edus, H= $\{(U_1,U_2), (U_1,U_3), (U_2,U_3), (U_3,U_4)\}$. The set H consists of all possible relations between every pair of edus. (U_i,U_j) refers to the hypotheses that involve two edus U_i and U_j. Since two edus U_1 and U_3 are not adjacent, the hypothesis (U_1,U_3) is not selected by our proposed parser. Figure 3 shown below displays the search space for the set H. In this figure, each edu U_i is replaced by the corresponding number i.

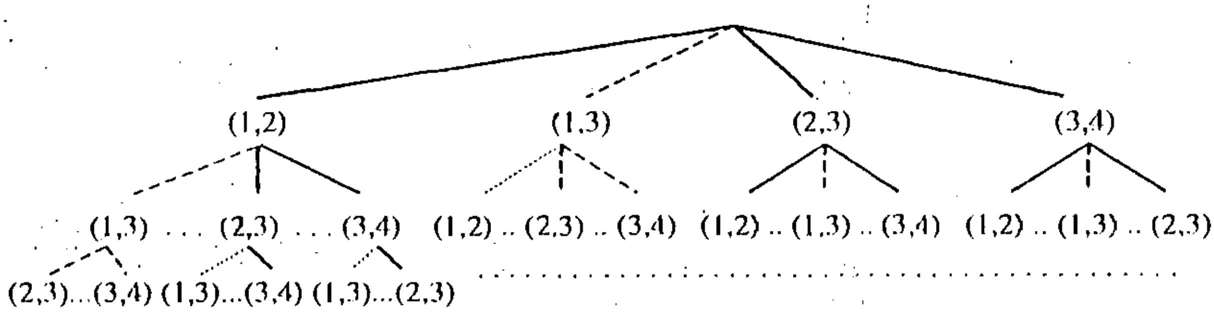


Fig. 3. The search spaces for the hypothesis set $\{(U_1,U_2), (U_1,U_3), (U_2,U_3), (U_3,U_4)\}$. RASTA visits all branches in the tree. The branches drawn by dotted lines are pruned by our proposed parser [6]

Another problem with RASTA is that one RST tree can be created twice by grouping the same text spans in different orders. If derived hypotheses of the set H contain $\{(U_1,U_2),(U_3,U_4)\}$, RASTA will generate two different combinations which create the same tree as shown below:

Join U_1 and U_2 -> Join U_3 and U_4 -> Join (U_1,U_2) and (U_3,U_4).
Join U_3 and U_4 -> Join U_1 and U_2 -> Join (U_1,U_2) and (U_3,U_4).

To deal with this redundancy problem faced by RASTA, our algorithm uses a tracing method. The hypothesis set is updated every time a new branch on the search tree is visited. When the parser visits a new branch, all nodes previously visited in the same level as that branch are removed from the hypothesis set. This action ensures that the algorithm does not recreate the same RST tree again.

Let's assume that both RASTA and our proposed parser start from the search space drawn by solid lines in Figure 3. Our proposed tracing method is explained in more detailed using Figure 4 below.

[6] Due to lack of space, all nodes of this tree cannot be presented together in this figure.

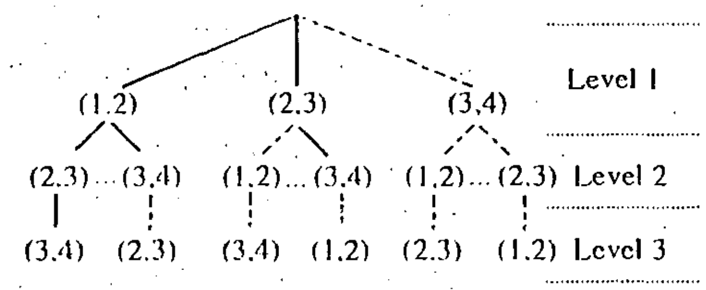


Fig. 4. Routes visit by the two parsers. RASTA visits all branches in the tree. The branches drawn by dotted lines are pruned by our proposed parser, which uses the tracing method

Our proposed parser first visits the branches which start with node (1,2) in level 1. After visiting these branches, the parser continues to the branches which start with node (2,3) in level 1. Since all RST trees or subtrees involving the node (1,2) are already visited, this node does not need to be revisited in the future. The branch that connects node (2,3) in level 1 with node (1,2) in level 2 is pruned from the search tree. As a result, the route (2,3) $\rightarrow$ (1,2) $\rightarrow$ (3,4) is not visited by our algorithm.

The discourse parser for our system is explained below.

A set called *Subtrees* is used in our parser to store the temporal subtrees created during the process. This set is initiated with all edus $\{U_1, U_2, ..., U_N\}$.

All possible relations that can be used to construct bigger trees at a time t form a hypothesis set *PotentialH*. If a hypothesis involving two text spans $<T_i, T_j>$ is used, the new subtree, created by joining $<T_i>$ and $<T_j>$, is added to the set *Subtrees*. The two small trees corresponding to the two text spans $<T_i>$ and $<T_j>$ are removed from *Subtrees*. Thus, all members of the set *Subtrees* are disjoined and their combination covers the entire text.

Each time the *Subtrees* changes, the hypothesis set *PotentialH* becomes obsolete. The hypotheses in the *PotentialH* relating to the subtrees that are removed in the previous step cannot be used. For that reason, the hypotheses, which do not fit with the new *Subtrees*, are removed from the *PotentialH*. Although some hypotheses are not considered as candidates to construct RST trees at one round of the parser, they may be needed later when the parser follows a different searching branch. All hypotheses computed by the discourse parsing system are stored in a hypothesis set called *StoredH*.

The *PotentialH* has not got any hypothesis to process the new subtree after the *Subtrees* changes. These relations will be added to the *PotentialH* after the relations between the new subtree and its adjacent trees are checked by using rules of the rule set.

When checking for a relation, the parser searches for that relation in the set of all hypotheses *StoredH*. If it is not found, the new hypothesis will be created by applying rules shown in Section 5.1. The hypotheses involving two unadjacent edus may be created during this process when the algorithm tries to create a rhetorical relation between two larger-adjacent text spans containing these edus.

The following algorithm briefly describes the steps in our discourse parser.

```
Function PARSER(Subtrees, PotentialH, <T1,T2>) {
/* <T1,T2> is created in the previous step by the two text spans
   T1 and T2 */
   If the number of final RST trees reaches a required limit,
   Exit.
```

```
  If Subtrees has only one tree, store it in the set of final
  RST trees and Return.
  If <T1,T2> = null (this is the first call to PARSER),
     NewH = all rhetorical relations between pairs of adjacent
     edus.
  Else, NewH = all rhetorical relations between <T1,T2> with its
     left adjacent text span LT and its right adjacent text
     span RT.
  Add all members of NewH to PotentialH.
  Remove all obsolete hypotheses from PotentialH.
  While PotentialH is not empty {
    - AppliedH = the highest score hypothesis in PotentialH.
    - Remove AppliedH from PotentialH.
    - Find two subtrees ST1 and ST2 in Subtrees satisfying Ap-
      pliedH. The text spans corresponding to ST1 and ST2 are
      <T1> and <T2>, respectively.
    - Remove ST1 and ST2 from Subtrees.
    - Add the new subtree created by ST1 and ST2 to Subtrees.
    - Call PARSER(Subtrees, PotentialH, <T1,T2>).
  }
  Return
}
```

6 Conclusion

In this paper, we have presented a discourse parsing system, in which syntactic information, cue phrases and other cohesive devices are investigated in order to define elementary discourse units and hypothesize relations.

To determine relations between texts, we explored all variants of cue phrases, combining with other feasible cohesive devices. It was shown that the position of cue phrases in a sentence, Noun-Phrase cues, and Verb-Phrase cues are good predictors for discovering rhetorical relations. In the case where cue phrases are not available, other text cohesive devices (e.g., synonyms, and antonyms) can be a reasonable substitution.

The construction of a discourse parser from the set of elementary discourse units was further analyzed. We have proved that the satellite in a rhetorical relation sometimes can provide good relation indications. This notation is implemented in creating the rule set for the parser. Based on the adjacency constraint of discourse analysis adapted from (Mann and Thompson, 1988), several improvements have been made to reduce the algorithm's complexity and at the same time improve its efficiency.

References

1. Bouchachia, A., Mittermeir, R., Pozewaunig, H.: Document Identification by Shallow Semantic Analysis. NLDB (2000) 190–202
2. Carlson, L. and Marcu, D.: Discourse Tagging Manual. ISI Tech Report, ISI-TR-545 (2001)

3. Corston-Oliver, S.: Computing Representations of the Structure of Written Discourse. PhD Thesis. University of California, Santa Barbara, CA, U.S.A (1998a)
4. Corston-Oliver, S.: Beyond string matching and cue phrases: Improving efficiency and coverage in discourse analysis. In: Eduard Hovy and Dragomir Radev: The Spring Symposium. AAAI Technical Report SS-98-06. AAAI Press (1998b) 9–15
5. Gundel, J., Hegarty, M., Borthen, K.: Information structure and pronominal reference to clausally introduced entities. In: ESSLLI Workshop on Information Structure: Discourse Structure and Discourse Semantics. Helsinki (2001)
6. Hobbs, J.: On the Coherence and Structure of Discourse. Technical Report CSLI-85-37, Center for the Study of Language and Information (1985)
7. Hovy, E. H.: Parsimonious and profligate approaches to the question of discourse structure relation. In: Proceedings of the 5th International Workshop on Natural Language Generation. Pittsburgh (1990) 128–136
8. Knott, A., Dale, R.: Using linguistic phenomena to motivate a set of coherence relations. Discourse Processes 18 (1995) 35–62
9. Komagata, N.: Entangled Information Structure: Analysis of Complex Sentence Structures. In: ESSLLI 2001 Workshop on Information Structure, Discourse Structure and Discourse Semantics. Helsinki (2001) 53–66
10. Mann, W. C. and Thompson, S. A.: Rhetorical Structure Theory: Toward a Functional Theory of Text Organization. Text, vol. 8 (1988) 243–281
11. Marcu, D.: Building Up Rhetorical Structure Trees. In: Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI), volume 2 (1996) 1069–1074
12. Marcu, D.: The Rhetorical Parsing, Summarization, and Generation of Natural Language Texts. PhD Thesis, Department of Computer Science, University of Toronto (1997)
13. Marcu, D.: A decision-based approach to rhetorical parsing. The 37th Annual Meeting of the Association for Computational Linguistics (ACL). Maryland (1999) 365–372
14. Marcu, D., Echihabi, A.: An Unsupervised Approach to Recognizing Discourse Relations. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL). Philadelphia, PA (2002)
15. Polanyi, L.: The Linguistic Structure of Discourse (1995)
16. Poesio, M., Di Eugenio, B.: Discourse Structure and Anaphoric Accessibility. In: ESSLLI Workshop on Information Structure, Discourse Structure and Discourse Semantics. Helsinki (2001)
17. Redeker, G.: Ideational and pragmatic markers of discourse structure. Journal of Pragmatics (1990) 367–381
18. RST Discourse Treebank - http://www.ldc.upenn.edu/Catalog/LDC2002T07.html
19. Salkie, R.: Text and discourse analysis. London, Routledge (1995)
20. Webber, B. et al.: D-LTAG System – Discourse Parsing with a Lexicalized Tree Adjoining Grammar. In: ESSLLI Workshop on Information structure, Discourse structure and Discourse Semantics (2001)
21. Webber, B., Knott, A., Stone, M., Joshi, A.: Discourse Relations: A Structural and Presuppositional Account Using Lexicalised TAG. Meeting of the Association for Computational Linguistics, College Park MD (1999)
22. WordNet. http://www.cogsci.princeton.edu/~wn/index.shtml

Proceedings of the IASTED International Conference
ARTIFICIAL INTELLIGENCE AND APPLICATIONS
February 16-18, 2004, Innsbruck, Austria

AUTOMATED DISCOURSE SEGMENTATION BY SYNTACTIC INFORMATION AND CUE PHRASES

Huong Le Thanh, Geetha Abeysinghe, and Christian Huyck
School of Computing Science, Middlesex University,
The Burroughs, London NW4 4BT, UK.
{H.Le, G.Abeysinghe, C.Huyck}@mdx.ac.uk

Abstract

This paper presents an approach to automatic segmentation of text written in English into Elementary Discourse Units (EDUs)[1] using syntactic information and cue phrases. The system takes documents with syntactic information as the input and generates EDUs as well as their nucleus/satellite roles. The experiment shows that this approach gives promising results in comparison with some of the prominent research relevant to our approach.

Key Words: Natural Language Processing, Discourse Segmentation, Syntactic Information, Cue Phrases.

1 Introduction

Previous research in discourse has shown that the discourse structure of a text is constructed from smaller discourse segments ([1], [2]). According to Mann and Thompson [1], all discourse units should have independent functional integrity, such as independent clauses. The smallest discourse unit is called an Elementary Discourse Unit (EDU) [3].

Discourse has been automatically segmented using disparate phenomena: lexical cohesion ([4], [5], [6]), discourse cues ([2], [3], [7], [8]), and syntactic information ([9], [10]). However, the criteria to indicate the exact discourse segment boundaries are still not certain.

The weakness of the lexical cohesion approach is that it cannot guarantee independent discourse units, which is the essential condition for discourse segmentation. Discourse cues, such as cue phrases, pauses, and referential identities ([3], [11]) can be a solution for this problem. Marcu's shallow analyser [3] splits text into EDUs by mapping cue phrases and punctuation marks. However, this approach cannot correctly identify boundaries in complex sentences, which do not have any lexical discourse cues.

Passonneau and Litman [7] proposed two sets of algorithms for linear segmentation based on the linguistic features of discourse. The first set is based on referential pronoun phrases, cue words and pauses. The second set uses error analysis and machine learning. The machine learning method requires training, which is heavily dependent on the manually annotated corpora. A large discourse corpus for such a training purpose is difficult to find.[2]

One of the well-organised system, which used syntactic approach, was done by Corston-Oliver [10]. He defined a rule set for discourse segmentation basing on grammatical information. However, the computational algorithm used by him to segment text is not mentioned in his thesis. In addition, Corston-Oliver's system does not detect the cases when strong cue phrases make noun phrases become EDUs.

Considering the problems mentioned above, we propose a new method that combines the syntactic approach with the discourse cue approach. Since a typical discourse unit is an independent clause or a simple sentence [1], the text is first split into EDUs using syntactic information. To deal with the case where strong cue phrases make a noun phrase become a separate EDU, a further segmentation process is undertaken after segmenting by syntax. The purpose of this process is to detect strong cue phrases. These processes will be discussed in more detail in the following sections.

The rest of this paper is organised as follows. The first step of our system (Step 1), discourse segmentation by syntax, is described in Section 2. Discourse segmentation by cue phrases (Step 2) is represented in Section 3. In Section 4, we describe our experiment and discuss the result we have achieved so far. Section 5 concludes the paper and delineates the possible future work of this approach.

2 Discourse Segmentation by Syntax – Step 1

The discourse segmentation by syntax module takes parsed documents from the Penn Treebank [13] as its input. One sentence is analyzed at each iteration of the segmentation process.[3] This module not only splits sentences into clauses, but also provides primary information about discourse relations among EDUs, such as which EDUs should have a discourse connection, and the status assigned to them (nuclei and satellites).

[1] For further information on "EDU", see [3].

[2] The biggest discourse corpus that we know of is the RST Discourse Treebank [12], with 385 Wall Street Journal articles.

[3] The sentence's pauses can be recognised by a syntactic parser. In this experiment, information about sentence's pauses is in parsed documents of the Penn Treebank.

2.1 Segmentation Principles

In Step 1, the principles for segmenting sentences into discourse units are based on the syntactic relations between words. These principles are based on previous research on discourse segmentation ([10], [14]). The main principles used in our system are shown below:

(i) *The clause that is attached to a noun phrase (NP) can be recognised as an embedded unit. If the clause is a subordinate clause, it must contain more than one word.*

For example:

(1) [Mr. Silas Cathcart built a shopping mall on some land][he owns.]

(ii) *Coordinate clauses and coordinate sentences of a complex sentence are EDUs.*

For example:

(2) [The firm's brokerage force has been trimmed][and its mergers-and-acquisitions staff increased to a record 55 people.]

(iii) *Coordinate clauses and coordinate elliptical clauses of verb phrases (VPs) are EDUs. Coordinate VPs that share a direct object with the main VP are not considered as a separate discourse segment.*

For example:

(3) [The firm seemed to be on the verge of a meltdown,][racked by internal squabbles and defections.]

(iv) *Clausal complements of reported verbs and cognitive verbs are EDUs.*

For example:

(4) [Mr. Carpenter says][that Kidder will finally tap the resources of GE.]

Using the Penn Treebank's syntactic assignments [15], principle (i) corresponds to syntactic chains (i-a) and (i-b) as shown below:

(i-a) (NP|NP-SBJ <text1> (SBAR|RRC <text2>))

(i-b) (NP|NP-SBJ <text1> (PRN <text2> (S <text3>)))

SBJ, SBAR, RRC, PRN, and S stand for subject (SBJ), subordinate clause and relative clause (SBAR), reduce relative clause (RRC), parenthetical (PRN), and sentence (S) respectively. Syntactic chain (i-a) means a subordinate clause or a reduced relative clause is inside a noun phrase. <text1>, <text2>, and <text3> are the context of a noun phrase. For example, consider the sentence "*The land he owns is very valuable.*" The syntactic chain which represents the noun phrase "*The land he owns*" in the above sentence can be written as *(NP The land (SBAR he owns))*.

If a clause, which is attached to a noun phrase, is headed by a preposition, then the syntactic chain of the noun phrase that corresponds to principle (i) is:

(i-c) (NP|NP-SBJ <text1> (PP <text2> (S|VP <text3>)))

In chain (i-c), PP stands for prepositional phrase. According to principle (i), <text2> in syntactic chain (i-a), and <text2> combining with <text3> in syntactic chains (i-b) and (i-c) are recognised as embedded units. To simplify syntactic chains (i-b) and (i-c), the system creates two labels named PRS (parenthetical-sentence) and PS (prepositional-sentence). These two labels are described respectively in (i-d) and (i-e) below:

(i-d) (PRN <text2> (S <text3>)) → (PRS <text2-3>)

(i-e) (PP <text2> (S|VP <text3>)) → (PS <text2-3>)

"→" can be interpreted as "convert to". <text2-3> is the concatenated string of <text2> and <text3>. By using syntactic chains (i-d) and (i-e), syntactic chains (i-a) to (i-c) can be grouped into one syntactic chain as follow:

(i-a') (NP|NP-SBJ <text1> (SBAR|RRC|PS|PRS <text2'>))

It should be noted that <text2'> in (i-a') is <text2-3> in (i-d) and (i-e). Due to space constraint, we only represent syntactic chains of the segmentation principles (ii), (iii), and (iv). In the syntactic chains corresponding to principles (ii), (iii), and (iv) as shown below, Sx stands for basic clause types such as subordinate clause and relative clause (SBAR), participial clause (S-ADV),... "And|but|or..." stands for a conjunction such as "and", or "but", or "or".

The syntactic chain of principle (ii) is:

(ii-a) (Sx <text1> (Sx <text2>) and|but|or... (Sx <text3>))

The syntactic chain of principle (iii) is:

(iii-a) (VP (VP <text1>) and|but|or... (VP|Sx|RRC|PPS <text2>))

The syntactic chains of principle (iv) is:

(iv-a) (S (NP-SBJ <text1>) (VP <text2> (SBAR <text3>)))

(iv-b) (S (NP-SBJ <text1>) (VP <text2> (SBAR <text3>) and|but|or... (SBAR <text4>)))

<text1> in (iv-a) and (iv-b) are not the pronoun "it".

(iv-c) (Sx (Sx <text1>), (NP-SBJ <text2>) (VP <text3>))

(iv-d) (Sx (Sx <text1>) , (VP <text2>) (NP-SBJ <text3>))

(iv-e) (Sx (NP-SBJ <text1>) , (Sx <text2>), (VP <text3>))

(iv-f) (Sx (VP <text1>) (NP-SBJ <text2>) , (Sx <text3>))

<text3> in (iv-c), <text2> in (iv-d), <text3> in (iv-e), and <text1> in (iv-f) are reported verbs or cognitive verbs.

2.2 Segmentation Algorithm

The input to this algorithm is the syntactic string of a sentence, in which <text> is replaced by a token #x,y# (where x,y is the begin and end position of <text> in the sentence being analysed). Each token of the syntactic string of the sentence is separated by a space. For example, the syntactic string of the sentence

(5) "The book I read yesterday is interesting."

is:

(5a) ((S (NP-SBJ (NP The book) (SBAR I read yesterday)) (VP is (ADJP interesting))).)

The input to the segmentation algorithm in this case is:

(5b) ((S (NP-SBJ (NP #0,7#) (SBAR #9,24#)) (VP #26,27# (ADJP #29,39#))) .)

The segmentation algorithm uses a stack to store tokens of the syntactic string during the reading process. It pushes and pops tokens onto and off the stack in order to analyse them. The algorithm ends when the syntactic string is reduced to the string "((S #x,y#) .)". The steps of the algorithm are described below:

Stack (Top of stack)	Input string	Compared string	Operations
	((S (NP-SBJ (NP #0,7#) (SBAR #9,24#)) (VP #26,27# (ADJP #29,39#))) .)		Pushing "(" onto the stack
(	(S (NP-SBJ (NP #0,7#) (SBAR #9,24#)) (VP #26,27# (ADJP #29,39#))) .)		Pushing "(" onto the stack
((	S (NP-SBJ (NP #0,7#) (SBAR #9,24#)) (VP #26,27# (ADJP #29,39#))) .)		Pushing "S" onto the stack
((S (NP-SBJ (NP #0,7#) (SBAR #9,24#))	(VP #26,27# (ADJP #29,39#))) .)		Popping off the strings on top of the stack, generating a compared string
((S	(VP #26,27# (ADJP #29,39#))) .)	(NP-SBJ (NP #0,7#) (SBAR #9,24#))	Mapping principle 1, splitting text (creating discourse segments), encoding the compared string, pushing it back onto the stack
((S (NP-SBJ #0,24#)	(VP #26,27# (ADJP #29,39#))) .)		Pushing "(" onto the stack
((S (NP-SBJ #0,24#) (	VP #26,27# (ADJP #29,39#))) .)		Pushing "VP" onto the stack
((S (NP-SBJ #0,24#) (VP	#26,27# (ADJP #29,39#))) .)		Pushing "#26,27#" onto the stack
((S (NP-SBJ #0,24#) (VP #26,27# (ADJP #29,39#))	) .)		Popping off the strings on top of the stack, generating a compared string
((S (NP-SBJ #0,24#)	) .)	(VP #26,27# (ADJP #29,39#))	No principle satisfies, encoding the compared string, pushing it back onto the stack
((S (NP-SBJ #0,24#) (VP #26,39#)	) .)		Pushing ")" onto the stack
((S (NP-SBJ #0,24#) (VP #26,39#))	.)		Popping off the strings on top of the stack, generating a compared string
(	.)	(S (NP-SBJ #0,24#) (VP #26,39#))	No principle satisfies, encoding the compared string, pushing it back onto the stack
((S #0,39#)	.)		Pushing "." onto the stack
((S #0,39#) .	)		Pushing ")" onto the stack
((S #0,39#) .)			STOP

Table 1. Progress of Segmenting Sentence (5) Using Syntactic Information

1. Read characters in the input string from left to right and put them onto a stack, until a space is found.
2. Repeat Step 1 until two consecutive close brackets are found on the top of the stack.
3. Pop off strings from the top of the stack into a separate string called "*compared string*" until the number of open brackets and the number of close brackets in the *compared string* are equal.
4. Compare the *compared string* with the sample syntactic strings (e.g., the syntactic string (a')) to check whether they match or not.
   4a. If they match, split the text corresponding to the *compared string* based on the segmentation principles. Store the information about the split text in the system. Go to Step 5.
   4b. If they do not match, go to Step 5.
5. Encode the *compared string* as a position tag #x,y# and push it back onto the stack with its syntactic information.
6. Repeat Step 1 to Step 5 until the input string is empty and the stack contains the following tokens, considering from the bottom of the stack: "(", "(", "S", "#x,y#", ")", ".", ")".

Table 1 represents the segmentation progress of sentence (5). Due to space constraints, some steps of the segmentation process are skipped.

The output of the segmentation algorithm for sentence (5) is two segments, "*The book*" and "*I read yesterday*", which contribute to one relation. The text "*is interesting*" is not in any text spans of the output. Another procedure, which is called the post process, will be called after the segmentation algorithm in order to deal with this problem. This procedure is described in Section 2.3.

2.3 Post Process

The purpose of this post process is to refine the output of the segmentation algorithm described in Section 2.2. There are two situations which need the post process. The first situation is that the segmentation of embedded units makes the text fragmented. For example, sentence (5) after being processed by Step 1 will have the structure as follows:

(6) The book | I read yesterday | is interesting.
N | S | UNKNOWN[4]

The text "*is interesting*" cannot be a single EDU because it does not have independent functional integrity. Meanwhile, the embedded clause "*I read yesterday*" provides additional information for the noun phrase "*the book*". "*The book*" is the nucleus (N) (the most important part in the relation); "*I read yesterday*" is the satellite (S) in the relation. In this case, a relation called SAME-UNIT[5] is

[4] UNKNOWN text span specifies the text fragment after syntactically segmenting the sentence. It is not a discourse relation.

[5] SAME-UNIT is a special relation, in which two text spans are on the same discourse unit [3]. SAME-UNIT is not a discourse relation.

created between "*The book I read yesterday*" and "*is interesting*". Both text spans "*The book I read yesterday*" and "*is interesting*" have an equally important role in contributing to the sentence "*The book I read yesterday is interesting*". Therefore, both of them are nucleus in the SAME-UNIT relation.

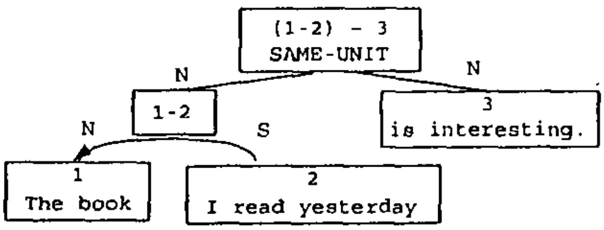


Fig. 1. Discourse Structure of Example (6)

The post process's operation depends on the position of the embedded unit. When the satellite of a relation is near an UNKNOWN text span, a SAME-UNIT relation is assigned between the UNKNOWN text span and the text span that contains the nucleus and satellite. Otherwise, when the nucleus of a relation is adjacent to an UNKNOWN text span, the UNKNOWN text span is merged with the nucleus, as in example (7) below.

(7) Mr. Silas Cathcart built a shopping mall on some land *he owns.*
(UNKNOWN: Mr. Silas Cathcart built a shopping mall on; N: some land; S: *he owns.*)

The segmentation by syntax algorithm finds two segments in the sentence (7), "*some land*" and "*he owns*", but the actual segments should be "*Mr. Silas Cathcart built a shopping mall on some land*" and "*he owns*".

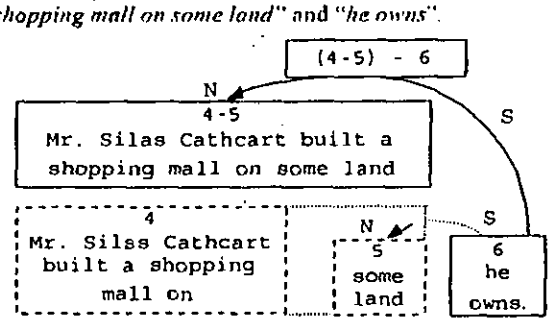


Fig. 2. Discourse Structure of Example (7)

Fig. 2 presents the discourse structure of the sentence in example (7). "*Mr. Silas Cathcart built a shopping mall on some land*" is the nucleus; "*he owns*" is the satellite in the relation. The dotted line shows the syntactic relation between "*some land*" and "*he owns*". The solid line shows a discourse relation between the two actual discourse units, after the sentence has been processed by Step 2.

The second situation needing post processing involves the placement of adverbs in EDUs. Some adverbs, which should stand at the beginning of the right clause, are put at the end of the left clause by the process in Step 1. This situation is detected and corrected by the procedure in Step 2. Examples (8) and (9) show such a situation. The clause "*they did not have enough people*" is split from the sentence "*They had to give up that campaign, mainly because they did not have enough people*" by syntactic information in Step 1. However, the correct segmentation in this case should be "*mainly because they did not have enough people*", not "*they did not have enough people*". After undergoing the process in Step 2, the boundary created by Step 1 is moved backward to the position between the comma and the two adverbs "*mainly because*", as shown in example (9).

(8) [They had to give up that campaign, *mainly because*][they did not have enough people.]
(9) [They had to give up that campaign,] [mainly because they did not have enough people.]

The input to the post processing procedure is the output of the segmentation algorithm in Section 2.2. The output of the post processing procedure is the discourse segments after refining boundaries.

3 Discourse Segmentation by Cue Phrases – Step 2

Several noun phrases are considered as EDUs when they are accompanied by strong cue phrases. These cases cannot be recognised by syntactic information. Therefore, another segmentation process is integrated into the system to deal with such cases. This process finds strong cue phrases from the output of Step 1. When a strong cue phrase is found, the algorithm seeks the end boundary of the noun phrase. These end boundaries can be punctuations such as a comma, semicolon, or full stop. Normally, a new EDU is created from the beginning position of the cue phrase to the end boundary of the noun phrase. However, this action may create incorrect results:

(10) [In 1988, Kidder eked out a $46 million profit, *mainly*][because of severe cost cutting.]

The correct segmentation for the sentence given in example (10) is generated by Step 2, and is given in example (11) below:

(11) [In 1988, Kidder eked out a $46 million Profit,][*mainly* because of severe cost cutting.]

Such a situation happens when an adverb stands before the cue phrases. Step 2 deals with such cases, by first detecting the noun phrase, which will be an EDU, and then checking for the appearance of adverbs before a strong cue phrase. If an adverb is found, the new EDU is recognized from the beginning position of the adverb to the end boundary of the noun phrase. Otherwise, the new EDU is split from the beginning position of the cue phrase to the end boundary of the noun phrase, for example:

(12) [According to a Kidder World story about Mr. Megargel,] [all the firm has to do is "position ourselves more in the deal flow."]

4 Evaluation

Eight documents of the RST Discourse Treebank [16] are used in the experiment. These documents are Wall Street Journal articles from the LDC Treebank [13], which have been annotated with discourse structure by human. The system's input is the corresponding syntactically parsed

documents taken from the Penn Treebank. The documents used in this experiment consists of 166 sentences with 3810 words. Most of the sentences are long and complex. The evaluation is done by comparing the EDUs assigned by the system with the EDUs from the eight RST documents mentioned above. Two EDUs are considered as similar if they have the same boundaries. There are 474 EDUs assigned by the system and 487 EDUs created by human, in which 386 EDUs of these two EDU sets are similar. Thus, there are 88 EDUs created by the system, which are not assigned by human. There are 101 EDUs created by human, which are not assigned by the system.
The standard information retrieval measurements (precision and recall) are used for evaluation. The precision is the proportion of assignments made that were correct. The recall is the proportion of possible assignments that were actually assigned. The precision and the recall of our experiment are:

$$Precision=\frac{386}{386+88}=81.4\% \qquad Recall=\frac{386}{386+101}=79.3\%$$

These measurements depend on several factors. The primary factor is the accuracy of syntactic information. The incorrectness syntactic information will decrease the accuracy of the segmentation's result. The syntactic documents from the Penn Treebank, which are used as the input of our system, also contain analytical errors. Since these errors in the Penn Treebank are rare, this factor does not have a great effect on our system's performance.
The second factor is the difference in human judgements. One person does not always agree with on segmentation [17]. The text in the RST corpus is analysed into very small text spans, which is not how our system segments. For example, consider the segmentation of the following sentence in the RST corpus:

(13) [Every order shall be presented to the President of the United States;]$_7$ [and]$_8$]before the same shall take effect,]$_9$ [shall be approved by him,]$_{10}$ [or]$_{11}$ [being disapproved by him,]$_{12}$ [shall be repassed by two-thirds of the Senate and House of Representatives.]$_{13}$

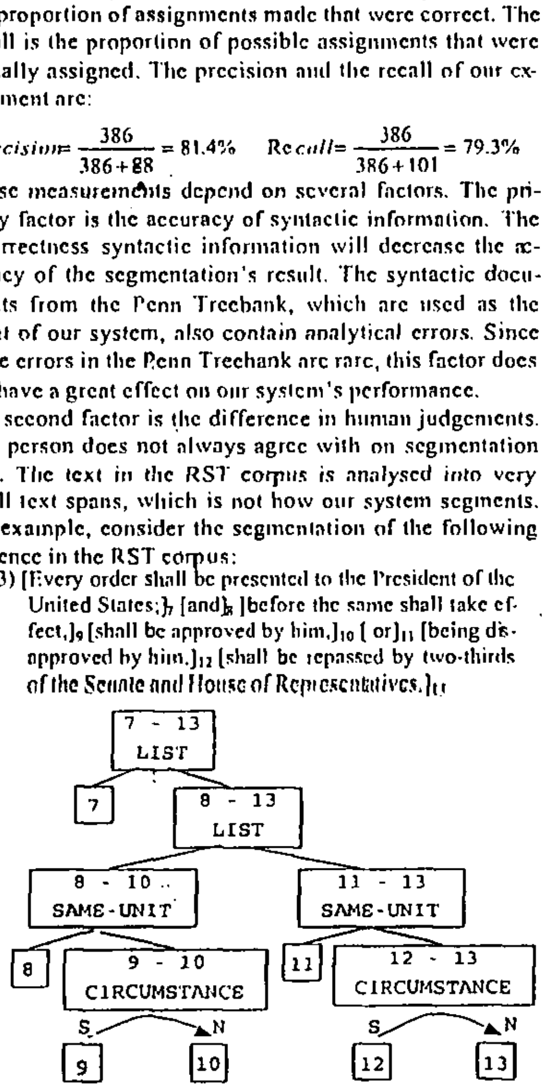


Fig. 3. Discourse Structure of Example (13), Getting from The RST Discourse Corpus[6]

The sentence in example (13) is treated differently by our system, which is shown in example (14):

(14) [Every order shall be presented to the President of the United States;]$_{14}$ [and before the same shall take effect,]$_{15}$ [shall be approved by him,]$_{16}$ [or being disapproved by him,]$_{17}$ [shall be repassed by two-thirds of the Senate and House of Representatives.]$_{18}$

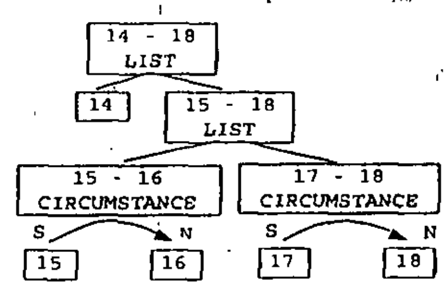


Fig. 4. Discourse Structure of Example (14), Generating by Our System

Over-segmentation is prevented as much as possible in our system because it makes discourse analysis more complicated. The appearance of new discourse units not only affects the EDUs next to them, but also the EDUs in other parts of the text. Since the merging of discourse conjunctions with their clauses does not change the general meaning of this discourse structure, we analyse the sentence in a different way than that in the RST corpus. This treatment causes some difference between the output of our system with the data from the RST corpus.
As discussed above, incorrect syntactic information and the disagreement in human judgements reduce the system's performance. We accept this reduction because not all discourse structures in the RST corpus are absolutely correct. Several discourse segments in the RST corpus are not accepted by other researchers.
Since researchers are still not certain about the criteria to indicate the exact discourse segment boundaries, and there is no standard benchmark, it is difficult to compare one researcher's result with others. Nonetheless, Okumura and Honda [6] carried out experiments on three texts, which were from exam questions in Japanese. The average precision and recall rates of that experiment were 25% and 52% respectively. The best precision and recall in the series of Passonneau and Litman's experiment [7], which used machine learning approach, were 95% and 53% respectively. Marcu [18] carried out experiments on a corpus of 90 discourse trees, which were built manually from the text in the Message Understanding Conference (MUC) coreference corpus, the Wall Street Journal (WSJ) corpus, and Brown corpus. If the system was trained in all corpora, the precision and recall for testing on WSJ corpus were 79.6% and 25.1%. These values are lower than our system. The precision and recall for MUC corpus were 96.9% and 75.4%; those of Brown corpus were 80.3% and 44.2% respectively. Although several results reported in [7] and [18] are higher than our result, the efficiency of these systems should not be judged purely on theses numbers since they depend on other factors

[6] All relation names mentioned in this paper are aiming at making discourse structures clearer. Recognising discourse relations is not in this paper's scope.

such as the size of training corpora, the corpora's domains, and the accuracy of human annotation. Meanwhile, the performance of our system is acceptable because our system does not need any training.

Our system's performance is promising when compared with the systems mentioned above and with other discourse segmentation systems known to us. However, more experimenting using a larger corpus is needed in order to get a more reliable evaluation.

5 Conclusion and Future Work

In this paper, we have presented a discourse segmentation method based on syntax and cue phrases. The discourse segmenter consists of two modules. Firstly, text is split based on syntactic information, aiming at receiving discourse units with independent functional integrity. Secondly, noun phrases that have the role of EDUs are recognised by detecting strong cue phrases from text.

Our preliminary experiment shows that this method attains promising results without any training. The experimental result is encouraging in comparison with existing segmentation methods. However, the system's performance can still be improved by the following ways: investigating a method to reduce the effect of syntactic information; and refining the rules for segmentation by syntax and for post processing. We leave these tasks for future work. Future work also includes integrating a syntactic parser with the discourse segmenter. Since there are many advanced syntactic parsers currently available, this problem can be easily solved.

A discourse parser cannot provide good results without accurate discourse segmentation. Therefore, this research is important in building discourse analysing systems, which have a wide range of applications including text summarisation.

References

[1] Mann, W. C. and Thompson, S. A., Rhetorical Structure Theory: Toward a Functional Theory of Text Organization, *Text, 8*, 1988, 243-281.

[2] Grosz, B.J. and Sidner C.L., Attention, intentions and the structure of discourse. *Computational Linguistics, 12*, 1986, 175-204.

[3] Marcu, D., *The Rhetorical Parsing, Summarisation, and Generation of Natural Language Texts* (Ph.D. Thesis: Department of Computer Science, University of Toronto, 1997).

[4] Morris, J., & Hirst, G., Lexical Cohesion Computed by Thesaural Relations as an Indicator of the Structure of the Text, *Computational Linguistics, 17*, 1991, 21-28.

[5] Kozima, H., *Computing lexical cohesion as a tool for text analysis* (Ph.D. Thesis: Graduate School of Electro-Communications, University of Electro-Communications, 1994).

[6] Okumura, M. and Honda, T., Word Sense Disambiguation and Text Segmentation Based on Lexical Cohesion, *Proc. of the 15th Conf. on Computational Linguistics (COLING-94), 2*, 1994, 755-761.

[7] Passonneau, R. J. and Litman, D. J., Discourse Segmentation by Human and Automated Means, *Computational Linguistics, 23*(1), 1997, 103-139.

[8] Forbes, K. and Miltsakaki, E., Empirical Studies of Centering Shifts and Cue Phrases as Embedded Segment Boundary Markers, *Penn Working Papers in Linguistics, 7*(2), 2002, 39-57.

[9] Batliner, A., Kompe, R., Kießling, A., Niemann, H., and Nöth, E., Syntactic-Prosodic Labeling Of Large Spontaneous Speech Data-Bases, *Proc. of ICSLP*, USA, 1996.

[10] Corston-Oliver, S., *Computing Representations of the Structure of Written Discourse* (Ph.D. Thesis: University of California, Santa Barbara, CA, U.S.A, 1998).

[11] Webber, B. L., Structure and ostension in the interpretation of discourse deixis, *Language and Cognitive Processes, 6*(2), 1991, 107-135.

[12] Carlson, L., Marcu, D., and Okurowski, M. E., RST Discourse Treebank, *LDC*, 2002.

[13] Marcus, M. P., Santorini, B. and Marcinkiewicz, M.A., Penn Treebank II, *LDC*, 1995.

[14] Carlson, L. and Marcu, D., Discourse Tagging Manual, *ISI Tech Report*, ISI-TR-545, 2001.

[15] Bies, A. et al., Bracketing Guidelines for Treebank II Style, *Penn Treebank Project*, 1995.

[16] Carlson, L., Marcu, D., and Okurowski, M. E., *Eight documents of the RST Discourse Treebank* (from http://www.isi.edu/%7Emarcu/, 2002)

[17] Litman, D. J. and Passonneau, R. J., Intention-based segmentation: Human reliability and correlation with linguistic cues, *Proc. of the 31st Annual Meeting of the Association for Computational Linguistics*, 1993, 148-155.

[18] Marcu, D., A Decision-Based Approach to Rhetorical Parsing, *The 37th Annual Meeting of the Association for Computational Linguistics (ACL)*, Maryland, 1999, 365-372.

Generating Discourse Structures for Written Texts

Huong LeThanh, Geetha Abeysinghe, and Christian Huyck
School of Computing Science, Middlesex University
The Burroughs, London, NW4 4BT, United Kingdom
{H.Le, G.Abeysinghe, C.Huyck}@mdx.ac.uk

Abstract

This paper presents a system for automatically generating discourse structures from written text. The system is divided into two levels: sentence-level and text-level. The sentence-level discourse parser uses syntactic information and cue phrases to segment sentences into elementary discourse units and to generate discourse structures of sentences. At the text-level, constraints about textual adjacency and textual organization are integrated in a beam search in order to generate best discourse structures. The experiments were done with documents from the RST Discourse Treebank. It shows promising results in a reasonable search space compared to the discourse trees generated by human analysts.

1 Introduction

Many recent studies in Natural Language Processing have paid attention to Rhetorical Structure Theory (RST) (Mann and Thompson 1988; Hovy 1993; Marcu 2000; Forbes et al. 2003), a method of structured description of text. Although rhetorical structure has been found to be useful in many fields of text processing (Rutledge et al. 2000; Torrance and Bouayad-Agha 2001), only a few algorithms for implementing discourse analyzers have been proposed so far. Most research in this field concentrates on specific discourse phenomena (Schiffrin 1987; Litman and Hirschberg 1990). The amount of research available in discourse segmentation is considered small; in discourse parsing it is even smaller.

The difficulties in developing a discourse parser are (i) recognizing discourse relations between text spans and (ii) deriving discourse structures from these relations. Marcu (2000)'s parser is based on cue phrases, and therefore faces problems when cue phrases are not present in the text. This system can apply to unrestricted texts, but faces combinatorial explosion. The disadvantage of Marcu's approach is that it produces a great number of trees during its process, which is the essential redundancy in computation. As the number of relations increases, the number of possible discourse trees increases exponentially.

Forbes et al. (2003) have a different approach of implementing a discourse parser for a Lexicalized Tree Adjoining Grammar (LTAG). They simplify discourse analysis by developing a grammar that uses cue phrases as anchors to connect discourse trees. Despite the potential of this approach for discourse analysis, the case of no cue phrase present in the text has not been fully investigated in their research. Polanyi et al. (2004) propose a far more complicated discourse system than that of Forbes et al. (2003), which uses syntactic, semantic and lexical rules. Polanyi et al. have proved that their approach can provide promising results, especially in text summarization.

In this paper, different factors were investigated to achieve a better discourse parser, including syntactic information, constraints about textual adjacency and textual organization. With a given text and its syntactic information, the search space in which well-structured discourse trees of a text are produced is minimized.

The rest of this paper is organized as follows. The discourse analyzer at the sentence-level is presented in Section 2. A detailed description of our text-level discourse parser is given in Section 3. In Section 4, we describe our experiments and discuss the results we have achieved so far. Section 5 concludes the paper and proposes possible future work on this approach.

2 Sentence-level Discourse Analyzing

The sentence-level discourse analyzer constructs discourse trees for each sentence. In doing so, two main tasks need to be accomplished: discourse segmentation and discourse parsing, which will be presented in Section 2.1 and Section 2.2.

2.1 Discourse Segmentation

The purpose of discourse segmentation is to split a text into elementary discourse units (edus)[1]. This task is done using syntactic information and cue phrases, as discussed in Section 2.1.1 and Section 2.1.2 below.

2.1.1 Segmentation by Syntax – Step 1

Since an edu can be a clause or a simple sentence, syntactic information is useful for the segmentation process. One may argue that using syntactic information is complicated since a syntactic parser is needed to generate this information. Since there are many advanced syntactic parsers currently available, the above problem can be solved. Some studies in this area were based on regular expressions of cue phrases to identify edus (e.g., Marcu 2000). However, Redeker (1990) found that only 50% of clauses contain cue phrases. Segmentation based on cue phrases alone is, therefore, insufficient by itself.

In this study, the segmenter's input is a sentence and its syntactic structure; documents from the Penn Treebank were used to get the syntactic information. A syntactic parser is going to be integrated into our system (see future work).

Based on the sentential syntactic structure, the discourse segmenter checks segmentation rules to split sentences into edus. These rules were created based on previous research in discourse segmentation (Carlson et al. 2002). The segmentation process also provides initial information about the discourse relation between edus. For example, the sentence "*Mr. Silas Cathcart built a shopping mall on some land he owns*" maps with the segmentation rule

(NP|NP-SBJ <text1> (SBAR|RRC <text2>))

In which, NP, SBJ, SBAR, and RRC stand for noun phrase, subject, subordinate clause, and reduce relative clause respectively. This rule can be stated as, "*The clause attached to a noun phrase can be recognized as an embedded unit.*"

The system searches for the rule that maps with the syntactic structure of the sentence, and then generates edus. After that, a post process is called to check the correctness of discourse boundaries. In the above example, the system derives an edu "*he owns*" from the noun phrase "*some land he owns*". The post process detects that "*Mr. Silas Cathcart built a shopping mall on*" is not a complete clause without the noun phrase "*some land*". Therefore, these two text spans are combined into one. The sentence is now split into two edus "*Mr. Silas Cathcart built a shopping mall on some land*" and "*he owns.*" A discourse relation between these two edus is then initiated. Its relation's name and the nuclearity roles of its text spans are determined later on in a relation recognition-process (see Section 2.2).

2.1.2 Segmentation by Cue Phrase–Step 2

Several NPs are considered as edus when they are accompanied by a strong cue phrase. These cases cannot be recognized by syntactic information; another segmentation process is, therefore, integrated into the system. This process seeks strong cue phrases from the output of Step 1. When a strong cue phrase is found, this process detects the end boundary of the NP. This end boundary can be punctuation such as a semicolon, or a full stop. Normally, a new edu is created from the begin position of the cue phrase to the end boundary of the NP. However, this procedure may create incorrect results as shown in the example below:

(1) [In 1988, Kidder eked out a $46 million profit, *mainly*][because of severe cost cutting.]

The correct segmentation boundary for the sentence given in Example (1) should be the position between the comma (',') and the adverb "*mainly*". Such a situation happens when an adverb stands before a strong cue phrase. The post process deals with this case by first detecting the position of the NP. After that, it searches for the appearance of adverbs before the position of the strong cue phrase. If an adverb is found, the new edu is segmented from the start position of the adverb to the end boundary of the NP. Otherwise, the new edu is split from the start position of the cue phrase to the end boundary of the NP. This is shown in the following example:

(2) [According to a Kidder World story about Mr. Megargel,] [all the firm has to do is "position ourselves more in the deal flow."]

[1] For further information on "edus", see (Marcu 2000).

Similar to Step 1, Step 2 also initiates discourse relations between edus that it derives. The relation name and the nuclearity role of edus are posited later in a relation recognition-process.

2.2 Sentence-level Discourse Parsing

This module takes edus from the segmenter as the input and generates discourse trees for each sentence. As mentioned in Section 2.1, many edus have already been connected in an initial relation. The sentence-level discourse parser finds a relation name for the existing relations, and then connects all sub-discourse-trees within one sentence into one tree. All leaves that correspond to another sub-tree are replaced by the corresponding sub-trees, as shown in Example (3) below:

(3) [She knows$_{3.1}$] [what time you will come$_{3.2}$][because I told her yesterday.$_{3.3}$]

The discourse segmenter in Step 1 outputs two sub-trees, one with two leaves "*She knows*" and "*what time you will come*"; another with two leaves "*She knows what time you will come*" and "*because I told her yesterday*". The system combines these two sub-trees into one tree. This process is illustrated in Figure 1.

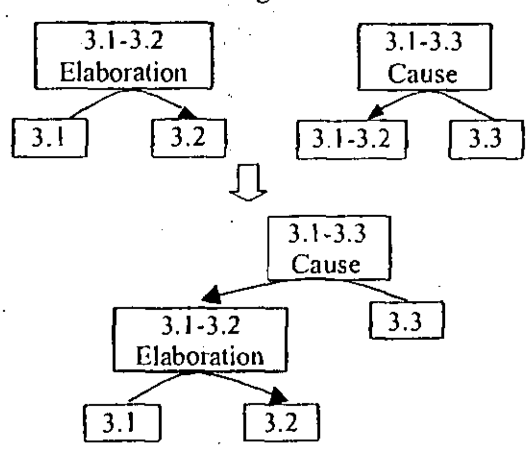


Figure 1. The discourse structure of text (3)

Syntactic information is used to figure out which discourse relation holds between text spans as well as their nuclearity roles. For example, the discourse relation between a reporting clause and a reported clause in a sentence is an Elaboration relation. The reporting clause is the nucleus; the reported clause is the satellite in this relation.

Cue phrases are also used to detect the connection between edus, as shown in (4):

(4) [He came late] [*because of* the traffic.]

The cue phrase "*because of*" signals a Cause relation between the clause containing this cue phrase and its adjacent clause. The clause containing "*because of*" is the satellite in a relation between this clause and its adjacent clause.

To posit relation names, we combine several factors, including syntactic information, cue phrases, NP-cues, VP-cues[2], and cohesive devices (e.g., synonyms and hyponyms derived from WordNet) (Le and Abeysinghe 2003). With the presented method of constructing sentential discourse trees based on syntactic information and cue phrases, combinatorial explosions can be prevented and still get accurate analyses.

3 Text-level Discourse Analyzing

3.1 Search Space

The original search space of a discourse parser is enormous (Marcu 2000). Therefore, a crucial problem in discourse parsing is search-space reduction. In this study, this problem was solved by using constraints about textual organization and textual adjacency.

Normally, each text has an organizational framework, which consists of sections, paragraphs, etc., to express a communicative goal. Each textual unit completes an argument or a topic that the writer intends to convey. Thus, a text span should have semantic links to text spans in the same textual unit before connecting with text spans in a different one. Marcu (2000) applied this constraint by generating discourse structures at each level of granularity (e.g., paragraph, section). The discourse trees at one level are used to build the discourse trees at the higher level, until the discourse tree for the entire text is generated. Although this approach is good for deriving all valid discourse structures that represent the text, it is not optimal when only some discourse trees are required. This is because the parser cannot determine how many discourse trees should be generated for each paragraph or section. In this research, we apply a different approach to control the levels of granularity. Instead of processing one textual unit at a time, we use a *block-level-score* to connect the text spans

[2] An NP-cue (VP-cue) is a special noun (verb) in the NP (VP) that signals discourse relations.

that are in the same textual unit. A detailed description of the *black-level-score* is presented in Section 3.2. The parser completes its task when the required number of discourse trees that cover the entire text is achieved.

The second factor that is used to reduce the search space is the textual adjacency constraint. This is one of the four main constraints in constructing a valid discourse structure (Mann and Thompson 1988). Based on this constraint, we only consider adjacent text spans in generating new discourse relations. This approach reduces the search space remarkably, since most of the text spans corresponding to sub-trees in the search space are not adjacent. This search space is much smaller than the one in Marcu's (2000) because Marcu's system generates all possible trees, and then uses this constraint to filter the inappropriate ones.

3.2 Algorithm

To generate discourse structures at the text-level, the constraints of textual organization and textual adjacency are used to initiate all possible connections among text spans. Then, all possible discourse relations between text spans are posited based on cue phrases, NP-cues, VP-cues and other cohesive devices (Le and Abeysinghe 2003). Based on this relation set, the system should generate the best discourse trees, each of which covers the entire text. This problem can be considered as searching for the best solution of combining discourse relations. An algorithm that minimizes the search space and maximizes the tree's quality needs to be found. We apply a beam search, which is the optimization of the best-first search where only a predetermined number of paths are kept as candidates. This algorithm is described in detail below.

A set called *Subtrees* is used to store sub-trees that have been created during the constructing process. This set starts with sentential discourse trees. As sub-trees corresponding to contiguous text spans are grouped together to form bigger trees, *Subtrees* contains fewer and fewer members. When *Subtrees* contains only one tree, this tree will represent the discourse structure of the input text. All possible relations that can be used to construct bigger trees at a time t form a hypothesis set *PotentialH*. Each relation in this set, which is called a hypothesis, is assigned a score called a *heuristic-score*, which is equal to the total score of all discourse cues contributing to this relation. A cue's score is between 0 and 100, depending on its certainty in signaling a specific relation. This score can be optimized by a training process, which evaluates the correctness of the parser's output with the discourse trees from an existing discourse corpus. At present, these scores are assigned by our empirical research.

In order to control the textual block level, each sub-tree is assigned a *block-level-score*, depending on the block levels of their children. This *block-level-score* is added to the *heuristic-score*, aiming at choosing the best combination of sub-trees to be applied in the next round. The value of a *block-level-score* is set in a different value-scale, so that the combination of sub-trees in the same textual block always has a higher priority than that in a different block.

- If two sub-trees are in the same paragraph, the tree that connects these sub-trees will have the *block-level-score* = 0.
- If two sub-trees are in different paragraphs, the *block-level-score* of their parent tree is equal to -1000 * (Li-L0), in which L0 is the paragraph level, Li is the lowest block level that two sub-trees are in the same unit. For example, if two sub-trees are in the same section but in different paragraphs; and there is no subsection in this section; then Li-L0 is equal to 1. The negative value (-1000) means the higher distance between two text spans, the lower combinatorial priority they get.

When selecting a discourse relation, the relation corresponding to the node with a higher *block-level-score* has a higher priority than the node with a lower one. If relations have the same *block-level-score*, the one with higher *heuristic-score* is chosen.

To simplify the searching process, an *accumulated-score* is used to store the value of the search path. The *accumulated-score* of a path at one step is the highest *predicted-score* of this path at the previous step. The *predicted-score* of one step is equal to the sum of the *accumulated-score*, the *heuristic-score* and the *block-level-score* of this step. The searching process now becomes the process of searching for the hypothesis with highest *predicted-score*.

At each step of the beam search, we select the most promising nodes from *PotentialH* that have been generated so far. If a hypothesis involving two text spans <Ti> and <Tj> is used, the new sub-tree created by joining the two sub-trees corresponding to these text spans is added to *Subtrees*. *Subtrees* is now updated so that it does not contain overlapping sub-trees. *PotentialH* is also updated according to the change in *Subtrees*. The relations between the new sub-tree and its adjacent sub-trees in *Subtrees* are created and added to *PotentialH*.

All hypotheses computed by the discourse parser are stored in a hypothesis set called *StoredH*. This set is used to guarantee that a discourse sub-tree will not be created twice. When detecting a relation between two text spans, the parser first looks for this relation in *StoredH* to check whether it has already been created or not. If it is not found, it will be generated by a discourse relation recognizer.

The most promising node from *PotentialH* is again selected and the process continues. A bit of depth-first searching occurs as the most promising branch is explored. If a solution is not found, the system will start looking for a less promising node in one of the higher-level branches that had been ignored. The last node of the old branch is stored in the system. The searching process returns to this node when all the others get bad enough that it is again the most promising path. In our algorithm, we limit the branches that the search algorithm can switch to by a number M. This number is chosen to be 10, as in experiments we found that it is large enough to derive good discourse trees. If *Subtrees* contains only one tree, this tree will be added to the tree's set.[3] The searching algorithm finishes when the number of discourse trees is equal to the number of trees required by the user. Since the parser searches for combinations of discourse relations that maximize the *accumulated-score*, which represents the tree's quality, the trees being generated are often the best descriptions of the text.

4 Evaluation

The experiments were done by testing 20 documents from the RST Discourse Treebank (RST-DT 2002), including ten short documents and ten long ones. The length of the documents varies from 30 words to 1284 words. The syntactic information of these documents was taken from Penn Treebank, which was used as the input of the discourse segmenter. In order to evaluate the system, a set of 22 discourse relations (list, sequence, condition, otherwise, hypothetical, antithesis, contrast, concession, cause, result, cause-result, purpose, solutionhood, circumstance, manner, means, interpretation, evaluation, summary, elaboration, explanation, and joint) was used.[4] The difference among *cause*, *result* and *cause-result* is the nuclearity role of text spans. We also carried out another evaluation with the set of 14 relations, which was created by grouping similar relations in the set of 22 relations. The RST corpus, which was created by humans, was used as the standard discourse trees for our evaluation. We computed the output's accuracy on seven levels shown below:

- Level 1 - The accuracy of discourse segments. It was calculated by comparing the segment boundaries assigned by the discourse segmenter with the boundaries assigned in the corpus.
- Level 2 - The accuracy of text spans' combination at the sentence-level. The system generates a correct combination if it connects the same text spans as the corpus.
- Level 3 - The accuracy of the nuclearity role of text spans at the sentence-level.
- Level 4 - The accuracy of discourse relations at the sentence-level, using the set of 22 relations (level 4a), and the set of 14 relations (level 4b).
- Level 5 - The accuracy of text spans' combination for the entire text.
- Level 6 - The accuracy of the nuclearity role of text spans for the entire text.
- Level 7 - The accuracy of discourse relations for the entire text, using the set of 22 relations (level 7a), and the set of 14 relations (level 7b).

The system performance when the output of a syntactic parser is used as the input of our discourse segmenter will be evaluated in the future, when a syntactic parser is integrated with our system. It is also interesting to evaluate the per-

[3] If no relation is found between two discourse sub-trees, a Joint relation is assigned. Thus, a discourse tree that covers the entire text can always be found.

[4] See (Le and Abeysinghe 2003) for a detailed description of this discourse relation set.

Level		1	2	3	4a	4b	5	6	7a	7b
System	Precision	88.2	68.4	61.9	53.9	54.6	54.5	47.8	39.6	40.5
	Recall	85.6	64.4	58.3	50.7	51.4	52.9	46.4	38.5	39.3
	F-score	86.9	66.3	60.0	52.2	53.0	53.7	47.1	39.1	39.9
Human	Precision	98.7	88.4	82.6	69.2	74.7	73.0	65.9	53.0	57.1
	Recall	98.8	88.1	82.3	68.9	74.4	72.4	65.3	52.5	56.6
	F-score	98.7	88.3	82.4	69.0	74.5	72.7	65.6	52.7	56.9
F-score(Human) – F-score(System)		11.8	22	22.4	16.8	21.5	19.0	18.5	13.7	17.0

Table 1. Our system performance vs. human performance

formance of the discourse parser when the correct discourse segments generated by an analyst are used as the input, so that we can calculate the accuracy of our system in determining discourse relations. This evaluation will be done in our future work.

In our experiment, the output of the previous process was used as the input of the process following it. Therefore, the accuracy of one level is affected by the accuracies of the previous levels. The human performance was considered as the upper bound for our discourse parser's performance. This value was obtained by evaluating the agreement between human annotators using 53 double-annotated documents from the RST corpus. The performance of our system and human agreement are represented by precision, recall, and F-score[5], which are shown in Table 1.

The F-score of our discourse segmenter is 86.9%, while the F-score of human agreement is 98.7%. The level 2's F-score of our system is 66.3%, which means the error in this case is 28.7%. This error is the accumulation of errors made by the discourse segmenter and errors in discourse combination, given correct discourse segments. With the set of 14 discourse relations, the F-score of discourse relations at the sentence-level using 14 relations (53.0%) is higher than the case of using 22 relations (52.2%).

The most recent sentence-level discourse parser providing good results is SPADE, which is reported in (Soricut and Marcu 2003). SPADE includes two probabilistic models that can be used to identify edus and build sentence-level discourse parse trees. The RST corpus was also used in Soricut and Marcu (S&M)'s experiment, in which 347 articles were used as the training set and 38 ones were used as the test set. S&M evaluated their system using slightly different criteria than those used in this research. They computed the accuracy of the discourse segments, and the accuracy of the sentence-level discourse trees without labels, with 18 labels and with 110 labels. It is not clear how the sentence-level discourse trees are considered as correct. The performance given by the human annotation agreement reported by S&M is, therefore, different than the one used in this paper. To compare the performance between our system and SPADE at the sentence-level, we calculated the difference of F-score between the system and the analyst. Table 2 presents the performance of SPADE when syntactic trees from the Penn Treebank were used as the input.

	Discourse segments	Unlabelled	110 labels	18 labels
SPADE	84.7	73.0	52.6	56.4
Human	98.3	92.8	71.9	77.0
F-score(H) - F-score(S)	13.6	19.8	19.3	20.6

Table 2. SPADE performance vs. human performance

Table 1 and Table 2 show that the discourse segmenter in our study has a better performance than SPADE. We considered the evaluation of the "*Unlabelled*" case in S&M's experiment as the evaluation of Level 2 in our experiment. The values shown in Table 1 and Table 2 imply that the error generated by our system is considered similar to the one in SPADE.

To our knowledge, there is only one report about a discourse parser at the text-level that measures accuracy (Marcu 2000). When using WSJ documents from the Penn Treebank, Marcu's decision-tree-based discourse parser received 21.6% recall and 54.0% precision for the

[5] The F-score is a measure combining into a single figure. We use the F-score version in which precision (P) and recall (R) are weighted equally, defined as 2*P*R/(P+R).

span nuclearity; 13.0% recall and 34.3% precision for discourse relations. The recall is more important than the precision since we want discourse relations that are as correct as possible. Therefore, the discourse parser presented in this paper shows a better performance. However, more work needs to be done to improve the system's reliability.

As shown in Table 1, the accuracy of the discourse trees given by human agreement is not high, 52.7% in case of 22 relations and 56.9% in case of 14 relations. This is because discourse is too complex and ill defined to easily generate rules that can automatically derive discourse structures. Different people may create different discourse trees for the same text (Mann and Thompson 1988). Because of the multiplicity of RST analyses, the discourse parser should be used as an assistant rather than a stand-alone system.

5 Conclusions

We have presented a discourse parser and evaluated it using the RST corpus. The presented discourse parser is divided into two levels: sentence-level and text-level. The experiment showed that syntactic information and cue phrases are quite effective in constructing discourse structures at the sentence-level, especially in discourse segmentation (86.9% F-score). The discourse trees at the text-level were generated by combining the hypothesized discourse relations among non-overlapped text spans. We concentrated on solving the combinatorial explosion in searching for discourse trees. The constraints of textual adjacency and textual organization, and a beam search were applied to find the best-quality trees in a search space that is much smaller than the one given by Marcu (2000). The experiment on documents from the RST corpus showed that the proposed approach could produce reasonable results compared to human annotator agreements. To improve the system performance, future work includes refining the segmentation rules and improving criteria to select optimal paths in the beam search. We would also like to integrate a syntactic parser to this system. We hope this research will aid the development of text processing such as text summarization and text generation.

References

Lynn Carlson, Daniel Marcu, and Mary Ellen Okurowski 2002. Building a Discourse-Tagged Corpus in the Framework of Rhetorical Structure Theory. In *Current Directions in Discourse and Dialogue*, Kluwer Academic Publishers.

Katherine Forbes, Eleni Miltsakaki, Rashmi Prasad, Anoop Sarkar, Aravind Joshi and Bonnie Webber 2003. D-LTAG System: Discourse Parsing with a Lexicalized Tree-Adjoining Grammar. *Journal of Logic, Language and Information*, 12(3), 261-279.

Edward Hovy 1993. Automated Discourse Generation Using Discourse Structure Relations. *Artificial Intelligence*, 63: 341-386.

Huong T. Le and Geetha Abeysinghe 2003. *A Study to Improve the Efficiency of a Discourse Parsing System*. In Proc of CICLing-03, 104-117.

Diane Litman and Julia Hirschberg 1990. *Disambiguating cue phrases in text and speech*. In Proc of COLING-90. Vol 2: 251-256.

William Mann and Sandra Thompson 1988. Rhetorical Structure Theory: Toward a Functional Theory of Text Organization. *Text*, 8(3): 243-281.

Daniel Marcu 2000. *The theory and practice of discourse parsing and summarization*. MIT Press, Cambridge, Massachusetts, London, England.

Livia Polanyi, Chris Culy, Gian Lorenzo Thione and David Ahn 2004. *A Rule Based Approach to Discourse Parsing*. In Proc of SigDial2004.

Gisela Redeker 1990. Ideational and pragmatic markers of discourse structure. *Journal of Pragmatics*, 367-381.

RST-DT 2002. *RST Discourse Treebank*. Linguistic Data Consortium. http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2002T07.

Lloyd Rutledge, Brian Bailey, Jacco van Ossenbruggen, Lynda Hardman, and Joost Geurts 2000. *Generating Presentation Constraints from Rhetorical Structure*. In Proc of HYPERTEXT 2000.

Deborah Schiffrin 1987. *Discourse markers*. Cambridge: Cambridge University Press.

Radu Soricut and Daniel Marcu 2003. *Sentence Level Discourse Parsing using Syntactic and Lexical Information*. In Proc of HLT-NAACL 2003.

Mark Torrance, and Nadjet Bouayad-Agha 2001. Rhetorical structure analysis as a method for understanding writing processes. In Proc of MAD 2001.