A Study to Improve the Efficiency of a Discourse Parsing System

Huong T. Le, 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 in (Marcu, 1999) and the unsupervised approach in (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].

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.

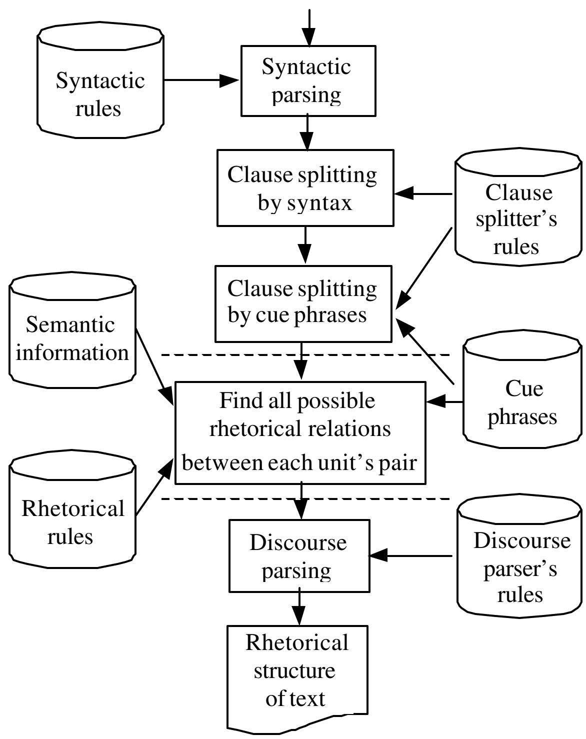

Fig. 1. The framework for a Discourse Analyzing System

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

[1] The biggest discourse corpus nowadays is the RST Discourse Treebank from LDC, with 385 Wall Street Journal articles.

[2] For further information on "edus", see (Marcu, 1997).

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% 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 "r*esult*" 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.

[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).

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 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.

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

[4] Threshold θ is selected as 0.5.

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
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 $1 * 1 = 1 > \theta$. 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]]

"A*nd*" 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) = $1 * 0.3 + 0.8 = 1.1 > \theta$.

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,*[82]] [Kidder, in typical fashion, completed a task-force study....[8.3]]]

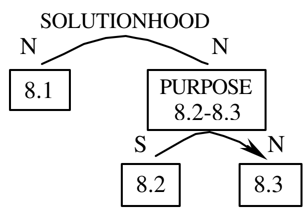

Fig. 2. The discourse tree of text (8)

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 $\{(U_1,U_2), (U_1,U_3),\dots,(U_1,U_N),(U_2,U_3),\dots,(U_{N-1},U_N)\}$ 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.

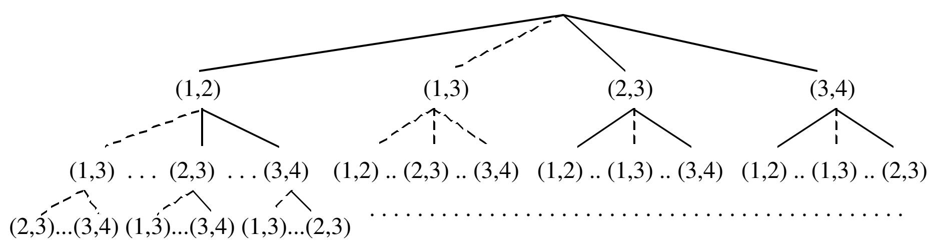

Fig. 3. The search spaces for the hypothesis set $\{(U_1,U_2), (U_1,U_3), (U_2,U_3), (U_3,U_4)\}$. RASTA visits all branches in the tree. The branches drawn by dotted lines are pruned by our proposed parser[6]

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.

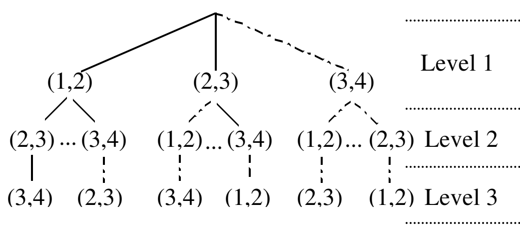

Fig. 4. Routes visit by the two parsers. RASTA visits all branches in the tree. The branches drawn by dotted lines are pruned by our proposed parser, which uses the tracing method

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
       AppliedH. 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

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

Carlson, L. and Marcu, D.: Discourse Tagging Manual. ISI Tech Report, ISI-TR-545 (2001)

Corston-Oliver, S.: Computing Representations of the Structure of Written Discourse. PhD Thesis. University of California, Santa Barbara, CA, U.S.A (1998a)

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

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)

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

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

Knott, A., Dale, R.: Using linguistic phenomena to motivate a set of coherence relations. Discourse Processes 18 (1995) 35-62

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

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

Marcu, D.: Building Up Rhetorical Structure Trees. In: Proceedings of the Thirteenth National Conference on Art ificial Intelligence (AAAI), volume 2 (1996) 1069-1074

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

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

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)

Polanyi, L.: The Linguistic Structure of Discourse (1995)

Poesio, M., Di Eugenio, D.: Discourse Structure and Anaphoric Accessibility. In: ESSLLI Workshop on Information Structure, Discourse Structure and Discourse Semantics. Helsinki (2001)

Redeker, G.: Ideational and pragmatic markers of discourse structure. Journal of Pragmatics (1990) 367-381

RST Discourse Treebank - http://www.ldc.upenn.edu/Catalog/LDC2002T07.html

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

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)

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)

WordNet - http://www.cogsci.princeton.edu/~wn/index.shtml