An Algorithmic Approach to Analyzing Rhetorical Structures

Andrew Potter
Computer Science & Information Systems Department
University of North Alabama
Florence, Alabama, USA
apotter1@una.edu

Abstract

Although diagrams are fundamental to Rhetorical Structure Theory, their interpretation has received little in-depth exploration. This paper presents an algorithmic approach to accessing the meaning of these diagrams. Three algorithms are presented. The first of these, called *Reenactment*, recreates the abstract process whereby structures are created, following the dynamic of coherence development, starting from simple relational propositions, and combing these to form complex expressions which are in turn integrated to define the comprehensive discourse organization. The second algorithm, called *Composition*, implements Marcu's strong nuclearity assumption. It uses a simple inference mechanism to demonstrate the reducibility of complex structures to simple relational propositions. The third algorithm, called *Compression*, picks up where Marcu's assumption leaves off, providing a generalized fully scalable procedure for progressive reduction of relational propositions to their simplest accessible forms. These inferred reductions may then be recycled to produce RST diagrams of abridged texts. The algorithms described here are useful in positioning computational descriptions of rhetorical structures as discursive processes, allowing researchers to go beyond static diagrams and look into their formative and interpretative significance.

1 Introduction

It has been shown that rhetorical structures and relational propositions are interchangeable (Potter, 2023a). The structure of an RST diagram can be restated as a relational proposition and relational propositions can be returned to RST diagrams. Relational propositions, as defined by (Mann & Thompson, 1986a, 1986b, 2000), are implicit assertions arising between clauses within a text and are essential to the functioning of the text. They can be considered as an alter ego of RST relations, with each assertion consisting of a predicate (or relation) and two variables (representing a satellite and nucleus). Because the predicate notation developed for relational propositions is Python conformant (Potter, 2023a, 2023b), mapping RST diagrams to relational propositions opens the possibility of exploring rhetorical structures algorithmically, presenting a range of analytic possibilities. The immediate effect of rendering RST diagrams as code is to unlock the picture: If, as the saying goes, a picture is worth a thousand words, the diagram now becomes a movie. It is a story about what is happening in a text. The objective of the research described in this paper was to investigate some of these possibilities.

Three algorithms are presented, each addressing a distinct aspect of Rhetorical Structure Theory. The first of these is called *Reenactment*. This algorithm replays the abstract process of structure formation, demonstrating the step-by-step construction of discourse formation starting with elementary relational propositions, and combining these to form complex expressions which are in turn integrated to define the comprehensive discourse organization. The second algorithm, referred to as the *Composition* algorithm, implements Marcu's strong nuclearity assumption and demonstrates the reducibility of complex structures to simple relational propositions. The third algorithm, called *Compression*, picks up where Marcu leaves off, providing a generalized scalable method for progressive reduction of relational propositions down to their simplest possible forms.

These algorithms provide the opportunity for a direct and deep look into information implicit in

Proceedings of the 5th Workshop on Computational Approaches to Discourse (CODI 2024), pages 1–11

RST diagrams. A benefit of this is that it should set aside any notion that RST diagrams are incapable of articulating in-depth aspects of discursive development, or that they are merely static specifications (Martin, 1992). On the contrary, although RST is only a partial explanation of discourse coherence, the part it plays is an important one. If we can restate RST diagrams in computational terms and allow these terms to describe what a diagram is doing, then perhaps we can begin to enjoy a deeper appreciation for what they are telling us about the text, and that these diagrams, far from static depictions of discourse structure, are actually renderings of a dynamic process, showing how a discourse germinates from its elementary units to become a whole that is greater than its parts.

2 Framework

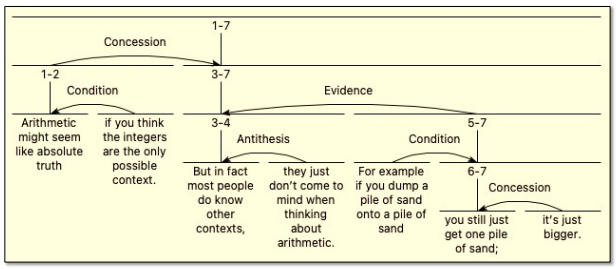
The interlocking property of rhetorical structures, where a satellite's support for a nucleus creates a span which in turn becomes the satellite for yet another nucleus, suggests that the typical rhetorical relation is rhetorically transitive, with the consequence that their intended effects develop cumulatively across complex structures, ultimately converging on an identifiable locus of effect. This abstract process is an assumption of the research described here; otherwise, the algorithms would fail to achieve produce their expected results. Potter's (2023a) algorithm for transforming RST analyses into relational propositions is used to provide the input for this framework. Throughout this process, these propositions maintain their structural isomorphism with RST diagrams.

Marcu's strong nuclearity assumption, also known as the strong compositionality criterion, says that when two complex text spans are connected through a rhetorical relation, the same rhetorical relation holds between the nuclei of the constituent spans (Marcu, 1996, 2000). This means that from relations between spans, simple structures may be inferred. The algorithmic implementation of this supports its application to RST analyses of any size. The reenactment algorithm implements a bottom-up perspective on RST structures by enacting the dynamic process of structure development, starting with elementary relational propositions, and combining these to form a complex expression ultimately of the comprehensive discourse organization. The Compression algorithm implements a technique previously proposed by Potter (2023b). As a generalization of strong nuclearity, it progressively eliminates the precedent satellite within the RST nuclear path to reduce the relational proposition to its simplest possible expression. The technique specifies delimited transitivity for handling multinuclears and unrealized relations. Taken together the three algorithms provide a foundational set of capabilities for analyzing rhetorical structures and exploring various features of the theory, such as inference, transitivity, reducibility, intentionality, and structural dynamics. In short, the algorithms can be used for investigating a range of discourse characteristics following a well-defined algorithmic approach. These algorithms are neither large nor complex. They are of interest more for what they do rather than for how they do it. What they do is offer insights into the nature of discourse. How they do this is largely reliant on the representation of RST structures as Pythonic relational propositions. I believe their simplicity is a by-product of the alignment of the theory with the discursive organizations it describes.

3 Related Work

While the literature on Rhetorical Structure Theory is vast, only a rather narrow strand of that research is relevant to this study. This naturally encompasses the founding RST documents, including but not limited to Mann and Thompson (1988) and Mann and Thompson (1987). These publications define Rhetorical Structure Theory (RST) as a descriptive theory of text organization, as a tool for describing and characterizing texts in terms of the relations that hold among the clauses within a text. A detailed exemplification of the theory can be found in Mann, et al.'s (1992) analysis of a fund-raising text. Matthiessen and Thompson (1987) provide an in-depth discussion of the theoretical foundations of RST.

Of continuing research interest in RST has been the possibility that it could be used as a text summarization technology. Most prominent in this area has been the works of (Marcu, 1997, 1998a, 1998b, 1998c, 1999, 2000). There has also been ongoing work in extending and refining the RST relation set. Generally this has been aimed at enhancing the ability of parsers to correctly identify relations while at the same time increasing the




```
condition(2,1)
concession(7,6)
condition(5,concession(7,6))
antithesis(4,3)
evidence(condition(5,concession(7,6)),antithesis(4,3))
concession(condition(2,1),evidence(condition(5,concession(7,6)),antithesis(4,3)))
```

Figure 1: Reenacting a Rhetorical Structure (text from Cheng, 2022)

specificity of relations (Carlson & Marcu, 2001; Zeldes, 2017).

Other research has been aimed at enriching the theory. In particular, Marcu is known for articulating the aforementioned strong nuclearity assumption. Stede (2008) explored the problems of nuclearity. In his investigation of different types of salience phenomena, he found that nuclearity as defined in RST tends to conflate information from different realms of description within a single structure. He proposed a multilevel analysis approach that would reconcile these issues. A variety of formalisms have been developed that would address limitations in RST (e.g., Asher & Lascarides, 2003; Webber & Prasad, 2009; Wolf & Gibson, 2005). An assumption made for this paper is that the theory and practice of RST is sufficiently well developed as to produce useful and interesting analyses.

In a parallel but lesser-known universe is the theory of relational propositions. This theory is an antecedent to the conceptualization of RST. With relational propositions, relations between satellites and nuclei are treated as implicit coherence-producing assertions (Mann & Thompson, 1986b). A relational proposition consists of a predicate and a pair of arguments. The predicate corresponds to the RST relation, and the arguments correspond to its satellite and nucleus. A shortcoming in the early work in relational propositions was its limitation to elementary expressions. There were no provisions for complex structures. Mann and Thompson (2000) attempted to address this but without success. That leaves off where this research begins. Potter (2019a, 2023b) devised a functional notation to support representation of complex relational propositions. The original objective was to develop a deductive interpretation of RST, one that would support investigation of logical operations such as transitive implication in discourse. That work provided an initial proof of concept for the algorithms described in this paper. However, rather than rely on propositional logic, the discourse features of interest were accessed directly.

This was expedited by using Potter's (2023a) program for mapping of RST diagrams to relational propositions. Automating this step enables scalability, reduces the likelihood of error, and eliminates a lot of tedium. Because the notation used for these relational propositions is conformant with the Python programming language, the algorithm effectively converts a diagram into machine processable code. An RST analysis like the *Arithmetic* analysis shown in Figure 1 can be automatically converted to its relational proposition:

```
concession(
   condition(
      2,1),
   evidence(
      condition(
         5,
         concession(
            7,6)),
      antithesis(
         4,3)))
```

These encoded relational propositions are the drivers for the algorithms described here. Each relation has a corresponding function within the

code, called a relation handler, so that performance of the relational proposition causes execution of the defined functions.

4 Algorithmic Analyses of Rhetorical Structures

As introduced earlier, this paper describes three algorithms for analyzing rhetorical structures. *Reenactment* models the bottom-up production of discourse organization. *Composition* implements Marcu's (2000) strong nuclearity. And *Compression* leverages the asymmetry of RST relations to implement transitive inference directly into relational propositions.

Each of these algorithms uses Pythonized relational propositions as input. For each algorithm there is a set of functions called *relation handlers*, one handler per relation. Typically, these functions return a tuple-formatted relational proposition, i.e., the name of the relation and a nested tuple containing satellite and nucleus identifiers, including the relation names and tuple information for any relational propositions nested within them. At runtime the handlers are invoked in order of precedence as specified by the relational proposition. Each algorithm defines a collector function that manages the values returned by the relation handlers. The output consists of one or more relational propositions, constituting the reenactments, inferences, or compressions as determined by the algorithm.

Input to each algorithm starts with RST analyses created using RSTTool or RST-Web (O'Donnell, 1997; Zeldes, 2016). These analyses are transformed into relational propositions using Potter's (2023a) conversion tool. The relational propositions are then input to the algorithms which transform them into reenacted, inferred, or compressed relational propositions. These relational propositions may be analyzed as is, or they may be used to construct new RST analyses. The following sections provide detailed descriptions of the algorithms and their applications.[1]

4.1 Reenactment Algorithm

The hierarchical appearance of RST diagrams encourages the impression of top-down tree structures. But these trees do not sprout branches as

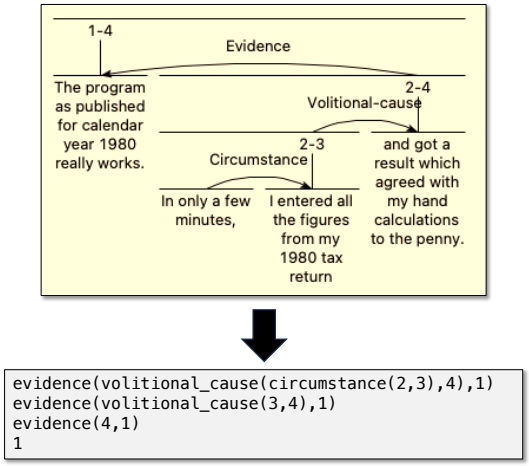


Figure 2: A Fully Compressible Analysis

it were from a root, branch, or stem. On the contrary, from a functional perspective, the diagrams are upside down: the segment nodes at the lower part of the diagram combine to form composite structures. These composite structures become increasingly complex at higher levels of the diagram. Although a completed diagram might seem to depict a static situation, what is revealed there is the end-state of a dynamic process. By modeling the abstract bottom-up process of discourse organization, the reenactment algorithm provides guidance for reading RST diagrams. The replay of a rhetorical structure shows how elementary discourse units combine logically to form relational propositions and how these propositions combine with other relational propositions to create increasingly complex expressions until a comprehensive analysis emerges. It is this comprehensive analysis that is modeled in an RST analysis.

The *reenactment algorithm* performs a bottom-up evaluation of a nested relational proposition. The design of the algorithm is simple. A relational proposition is evaluated as a Python expression. A relation handler is invoked whenever the relation occurs within an expression. These relation handlers convert a relational proposition from code to data. The function returns the name of the relation and a nested tuple containing identifiers for its satellite and nucleus. The contents of the tuple reflect the depth of the nesting of the relational proposition. The tuple representation of the relational proposition is assembled in precedence order, working from the inside out. The replay

[1] https://github.com/anpotter/aaars

manages the recursion of the expression and collects the output.

As the function makes its way through the relational proposition, it constructs the expression as it goes. In other words, it performs the relational proposition. A completed relational proposition can thus be thought of not as a static entity but as the result of an abstract process. And because relational propositions are isomorphic with their respective RST diagrams, the interpretation of the diagram can be understood as consistent with the performance of the relational proposition. As the reenactment in Figure 1 shows, RST structures define themselves from elementary relational propositions which combine to form complex expressions, enacting a logical process through which rhetorical intentionality emerges. This abstract process follows the precedence of the relational proposition.

4.2 Composition Algorithm

The *composition algorithm* is an implementation of Marcu's strong compositionality criterion. The criterion states that any relation between two spans will also hold between the nuclei of those spans (Marcu, 2000). Thus, simplified structures may be inferred from complex structures. In discussions of the criterion, it seems to be assumed that both the satellite and nucleus are themselves complex spans (e.g., Das, 2019; Demberg, Asr, & Scholman, 2019; Egg & Redeker, 2010; Marcu, 1996; Sanders et al., 2018; Stede, 2008). However, for the criterion to be delimited in this way suggests that relations between elementary units and relations between complex spans are in some way fundamentally different from one another. While there would be no difficulty in limiting the algorithm to comply with this, I have adopted a broader interpretation: nuclearity arises as a result of the relation of a unit or span to some other unit or span; hence the criterion is more broadly applicable. The only constraint is that at least one part of the relation be a span. Otherwise, any inference would be a simple repetition. Thus, the algorithm as written permits inferences in which either the satellite or the nucleus is an elementary unit, so that, for example, from the relational proposition:

```
volitional_cause(
   circumstance(
      2,3),4)
```

the algorithm makes the inference:

```
volitional_cause(3,4)
```

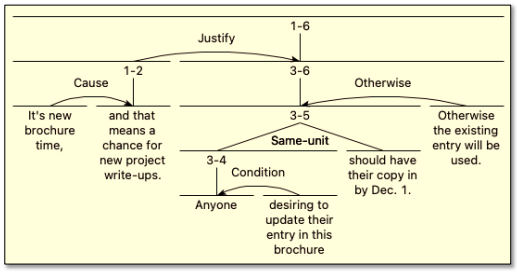
The algorithm evaluates the relation handlers for the relational proposition, collects the relational tuples, and determines which of those meet the compositionality criterion. The set of inferences generated from the RST analysis shown in Figure 1 are listed in Table 1.

Relational Proposition	Inference
concession(7,6)	6
condition(5, concession(7,6))	condition(5,6)
antithesis(4,3)	3
evidence(condition(5, concession(7,6)), antithesis(4,3))	evidence(concession(7,6),3)
concession(condition(2,1), evidence(condition(5, concession(7,6)), antithesis(4,3)))	concession(1,antithesis(4,3))

Table 1: Inferences Generated by Composition Algorithm

4.3 Compression Algorithm

The *compression algorithm* is a procedure for progressive reduction of relational propositions to their simplest accessible form. By evaluating the expression in precedence order, the expression is progressively reduced from the innermost relational propositions outward. With each iteration the relation and satellite of the precedent proposition is eliminated. In effect, the relational proposition collapses inward. Usually, but not always, the ultimate reduction will be the single elementary discourse unit identifiable as the locus of intended effect. When not, it will be the simplest accessible relational proposition containing the nucleus that would have been the locus of intended effect, were that relation realizable. In other words, the algorithm takes the compression as far as it can, and yet acknowledges that some relations are by




```
justify(cause(1,2),otherwise(6,same_unit(condition(4,3),5)))
justify(2,otherwise(6,same_unit(condition(4,3),5)))
```

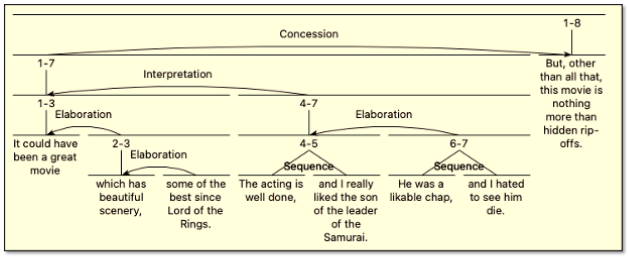
Figure 3: A Partially Compressible Analysis

definition or by position resistant to reduction. The *Tax Program* analysis (Figure 2, above) provides a simple example of a fully compressible analysis. With each step, the innermost relation and its satellite are eliminated. The CIRCUMSTANCE and its satellite are dropped first. Next VOLITIONAL-CAUSE and its satellite are dropped, followed by elimnating the satellite from the EVIDENCE relation, ultimately leaving only segment 1: *the program as published for calendar year 1980 really works*. Applying this procedure to a variety of RST analyses has yielded positive results. However, not all RST analyses are as simple as the Tax Program.

Some relations are not compressible and require special treatment. These include multinuclears, relations with unrealized satellites, and attribution relations. While multinuclears may seem syntactically and semantically simple, they present complications. The nuclei within a multinuclear relation may consist solely of elementary discourse units, but quite commonly these nuclei are complex relational propositions that must themselves be reduced. So, on one level multinuclears may be treated as unanalyzable virtual units, but on the other, it is necessary to analyze the members of the relation, subjecting each to the compression process.

Relations with unrealized satellites include CONDITION, PURPOSE, UNLESS, and OTHERWISE. Unrealized relations do not permit inference or realization of the nucleus from the satellite. With the CONDITION relation the satellite presents a hypothetical, future, or otherwise unrealized situation such that realization of the nucleus is dependent on it. Hence the nucleus remains hypothetical. Similar dependencies hold for UNLESS and OTHERWISE. With PURPOSE, the nucleus is an activity that must be performed in order for the satellite to be realized. The relation between the satellite and nucleus holds but has not been realized. The compressibility of these relations depends on their position within a relational proposition. If the relation is positioned as the satellite of a relational proposition, it may be eliminated, but if it is the nucleus, it may not. This is because the process of reduction involves the progressive elimination of satellites. This, particularly when combined with multinuclear relations, can result in structures that are resistant to compression. *The New Brochure Time* analysis shown in Figure 3 is an example of this. There the OTHERWISE relation cannot be reduced because neither the satellite nor the nucleus is realized. SAME-UNIT is a pseudo-relation used for linking discontinuous text fragments that are really a single discourse unit. It is modeled on the multinuclear schema. The compression completes after only one reduction.

Alternatively, it can be useful to relax the reducibility constraint in order to focus on intentional development. For example, this can be of interest when the unrealized relations involve actions that might be taken by the reader. This is the case for the CONDITION and OTHERWISE relations for the *New Brochure Time* analysis shown above in Figure 3, presumably the writer of the text expected that these conditions would hold for to




```
concession(interpretation(elaboration(sequence(6,7),sequence(4,5)),elaboration(elaboration(3,2),1)),8)
concession(interpretation(sequence(4,5),elaboration(elaboration(3,2),1)),8)
concession(interpretation(sequence(4,5),elaboration(2,1)),8)
concession(interpretation(sequence(4,5),1),8)
concession(1,8)
8
```

Figure 4: Reduction of Multinuclear Relations (Adapted from Lu et al., 2019)

some readers. With the constraints removed, the analysis reduces to `same_unit(3,5)`, or *Anyone…should have their copy in by December 1*.

Sometimes, as a compression proceeds, a non-compressible relation will be shifted from a nuclear to a satellite position. When this occurs, the relation can be eliminated. This can be observed in the process shown in Figure 4. There are two SEQUENCE relations in the analysis, one as satellite and the other as nucleus of an ELABORATION relation. When the ELABORATION is eliminated, it takes with it its satellite, thus eliminating the first of the SEQUENCE relations. The remaining SEQUENCE is now satellite to the INTERPRETATION relation, making it eligible for elimination, which occurs when it becomes the precedent relational proposition. The status of the ATTRIBUTION relation has been debated from time immemorial, so perhaps it is fitting that it should require special attention here. Mann and Thompson (1987) rejected it as a legitimate relation, but it was subsequently instated and refined by Carlson and Marcu (2001), as well as by Zeldes (2023), and yet provisionally rejected by Stede, Taboada, and Das (2017) and reduced to alternative relations by (Potter, 2019b). For the present research, ours is not to reason why, but rather to process any and all analyses as they presented. ATTRIBUTION is treated (at least optionally) as irreducible in part because sourcing of information is often part of the intended effect, particularly when the intention of the attributed material differs from that of the writer.

In order to assess the algorithm's applicability over larger texts, the compression algorithm was tested on several analyses from the GUM corpus (Zeldes, 2017). Because these analyses make frequent use of multinuclear relations, this resulted in reduced compressibility, so that the results are sometimes lengthy in their own right. Code was added to the algorithm to enable recovery of compressed texts. The results of this suggest coherence is preserved, albeit with some irregularities in surface cohesion and punctuation. For the *GUM Academic Thrones* analysis, the original contains 87 segments, and compression reduced this to 17 segments. The compressed text was mapped to its relational proposition to create an RST analysis relationally consistent with the source. The compressed text generated by the compression is shown in Figure 5. For readability, line breaks were inserted for each of the ORGANIZATION-HEADING relations. This text, along with the relational proposition, was used to create the RST analysis shown in Figure 6. The original segment identifiers are preserved for

A Comparative Discourse Analysis of Fan Responses to Game of Thrones

For us , as digital humanists , defining the " transmedia fan " is of particular relevance

Methodology

As a first step the current project undertakes a comparative discourse analysis of online conversations of Game of Thrones fans . As a pilot project , the current work takes the content of both comment threads and analyzes each thread separately Through this analysis , a categorization of themes emerges A comparison of categories and sub-categories between both groups provides preliminary findings to support an emergent model , or models , of the " transmedia fan " .

Conclusion

The present research represents a first step The question is , fundamentally , an examination Future research should explore the negotiation tactics The current study will contribute to the development of further qualitative and quantitative research This project is of relevance to researchers in media studies , fan studies , information studies and digital humanities.

Figure 5: Academic Thrones Compressed Text

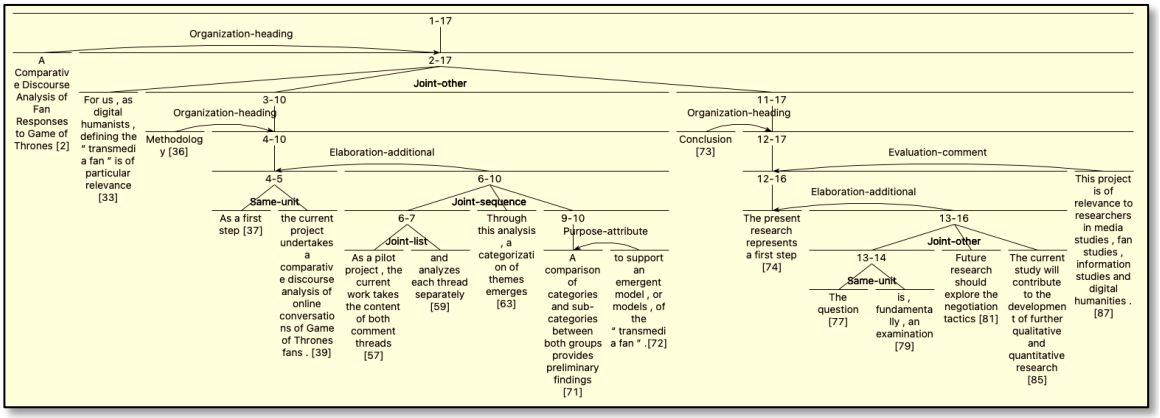


Figure 6: Compressed GUM Academic Thrones RST Analysis

reference. The rhetorical structure as well as the text survived the compression process. For the complete original text, see Forcier (2017).

The compression algorithm supports a longstanding view about nuclearity: simple summarizations should be possible merely by lopping off satellites. Moreover, this is reflected in a limitation that surfaced during testing. In analyses of longer documents where the JOINT relation and its variants are necessary to hold the structure together, guideposts such as ORGANIZATIONAL-HEADING become helpful for assuring readability. This is as true for the compressions as it is for the original texts.

In compressions of longer texts, such as the GUM analysis of Nancy Pelosi's speech on George Floyd, where such guideposts are lacking, minor digressions which work well in the original spoken medium become difficult in the transcript, and these difficulties are apparent in the compressions. That these reflect the features of the original should be understood as an affirmation of RST as an explanation of discourse coherence. The features of the document are carried forward through multiple layers of analysis.

As to whether the compression algorithm's contribution provides anything new or unique, I would argue that it affirms claims often left to intuition, and that it does so in a systematic and repeatable manner. The code is freely available to anyone who cares to take it for a test drive. Moreover, the approach is generalizable to other RST problems – once their solutions can be stated algorithmically, they can be readily evaluated and applied to a wide range of cases.

5 How it Works

The algorithms described here all share a common design. Each consists of two parts: a set of *relation handlers* and a *core algorithm*. A handler is provided for each relation in the RST relation set. These handlers are functions evaluated in response to each occurrence of their corresponding relation in a relational proposition. They are simple one-liners. Each handler returns a tuple containing the function's name and a nested tuple containing its satellite and nuclear identifiers. The functions obtain their names at runtime using a system call. Thus, in the reenactment and composition algorithms, an occurrence of the relational proposition `concession(1,2)` will return the tuple: `('concession',(1,2))`, and an occurrence of the relational proposition `evidence(3, concession(1,2))` will return the tuple: `('evidence',(3,('concession',(1,2)))`

When a relational proposition is evaluated, each handler is called in precedence order, with each function returning its name and arguments to the calling function. In this way, the program essentially *performs* the relational proposition, starting with the innermost (hence higher precedence) functions, working outward to the edges of the expression. The reenactment algorithm exploits that process.

The compress algorithm is only slightly more complicated. Each of its relation handlers makes a call to the core compression algorithm, passing it its relation name and arguments. Special handling for nonreducible relations is specified syntactically in the handler functions. The evaluation of the

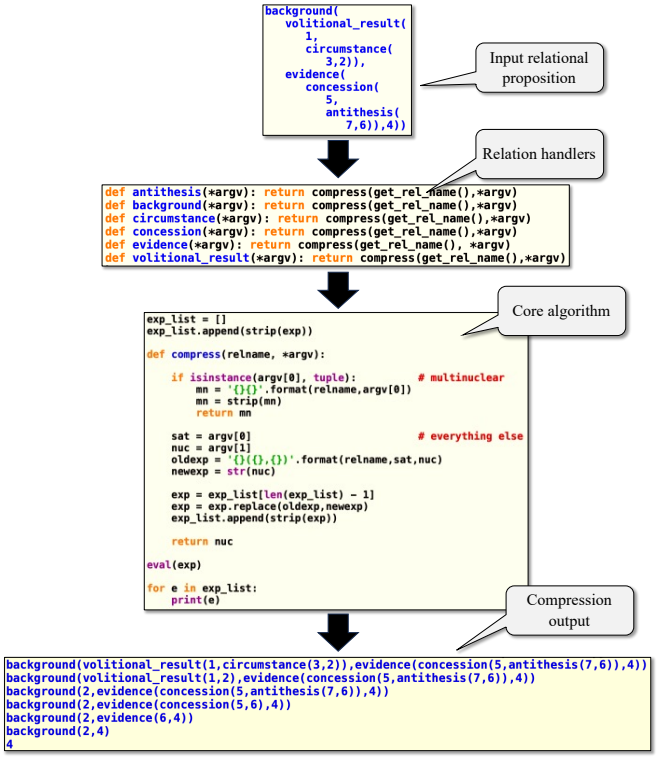


Figure 7: How it Works

relational proposition shown at the top of Figure 7 invokes each of the cited relation handlers and each of these call the `compress` function, first `circumstance`, followed by `volitional_result`, `antithesis`, `concession`, `evidence`, and finally the outermost relation, `background`. This leaves little for the core algorithm to do. Since multinuclears are non-compressible, the algorithm simply formats them and returns the formatted expression. For compressible relations, the algorithm simply replaces the current relational proposition with its nucleus, thus for each step eliminating the relation and satellite. Functionally, it infers the nucleus from the relational proposition. This is consistent with Marcu's strong nuclearity assumption. Because this process is implicit within the relational proposition, we can say it is also implicit within the RST diagram from which the proposition is derived, and therefore inferable from within the text itself. Figure 7 shows the complete code for the compress algorithm. For space reasons, the list of relation handlers has been limited to what is required for the example.

6 Conclusion

An RST analysis can be understood as an explanation of the organizational composition of a text. By identifying the text structure, by showing how its elements come together, an RST analysis explains how the text accomplishes what it is intended to do. The algorithms described in this paper contribute to that explanation. Reenactment is a step-by-step articulation of coherence development. The composition algorithm identifies relational propositions implicit within the text. The compress algorithm performs a deconstruction of the structure from its totality down to its intentional essence. These algorithms show that rhetorical structures can be studied in terms of their relational propositions. The relational propositions generated by the algorithms are inferences which follow directly from the source rhetorical structure. For each inference there is an isomorphic RST analysis and a corresponding text, that is, a structure within the structure and a text within the text. Thus, these simple algorithms provide interpretations of rhetorical structures as discursive processes, enabling the analyst to move beyond static diagrams and study formative and interpretative features of rhetorical structure. By positioning the algorithms within the framework of relational propositions, considerable simplicity is achieved. The algorithms extend the scope of RST as a tool for explaining discourse organization.

References

Nicholas Asher, & Alex Lascarides. 2003. *Logics of conversation*. Cambridge, UK: Cambridge University Press.

Lynn Carlson, & Daniel Marcu. 2001. *Discourse tagging reference manual* (TR-2001-545). Retrieved from Marina del Rey, CA: ftp://ftp.isi.edu/isi-pubs/tr-545.pdf

Eugenia Cheng. 2022. *The Joy of Abstraction*. Cambridge: Cambridge University Press.

Debopam Das. 2019. Nuclearity in RST and signals of coherence relations. In Amir Zeldes, Debopam Das, Erick Maziero Galani, Juliano Desiderato Antonio, & Mikel Iruskieta (Eds.), *Proceedings of the Workshop on Discourse Relation Parsing and Treebanking 2019* (pp. 30-37). Minneapolis, Minnesota: Association for Computational Linguistics.

Vera Demberg, Fatemeh Torabi Asr, & Merel Scholman. 2019. How compatible are our discourse annotations? Insights from mapping RST-DT and PDTB annotations.

Markus Egg, & Gisela Redeker. 2010. How complex is discourse structure? In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)* (pp. 1619-1623). Valletta, Malta: European Languages Resources Association (ELRA).

Eric Forcier. 2017. Re(a)d wedding: A comparative discourse analysis of fan responses to Game of Thrones. *Digital Humanities*.

Ruqian Lu, Shengluan Hou, Chuanqing Wang, Yu Huang, Chaoqun Fei, & Songmao Zhang. 2019. Attributed Rhetorical Structure Grammar for domain text summarization. *arxiv.org.*

William C. Mann, Christian M. I. M. Matthiessen, & Sandra A. Thompson. 1992. Rhetorical structure theory and text analysis. In William C. Mann & Sandra A. Thompson (Eds.), *Discourse description: Diverse linguistic analyses of a fund-raising text* (pp. 39-78). Amsterdam: John Benjamins.

William C. Mann, & Sandra A. Thompson. 1986a. Assertions from discourse structure. In *HLT '86: Proceedings of the workshop on strategic computing natural language* (pp. 257-270). Morristown, NJ: Association for Computational Linguistics.

William C. Mann, & Sandra A. Thompson. 1986b. Relational propositions in discourse. *Discourse Processes, 9*(1), 57-90.

William C. Mann, & Sandra A. Thompson. 1987. *Rhetorical structure theory: A theory of text organization* (ISI/RS-87-190). Retrieved from Marina del Rey, CA:

William C. Mann, & Sandra A. Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text - Interdisciplinary Journal for the Study of Discourse, 8*(3), 243-281.

William C. Mann, & Sandra A. Thompson. 2000. *Toward a theory of reading between the lines: An exploration in discourse structure and implicit communication.* Paper presented at the Seventh International Pragmatics Conference, Budapest, Hungary.

Daniel Marcu. 1996. Building up rhetorical structure trees. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence* (Vol. 2, pp. 1069-1074). Portland, Oregon: American Association for Artificial Intelligence.

Daniel Marcu. 1997. From discourse structures to text summaries. In *Proceedings of the ACL'97/EACL'97 Workshop on Scalable Text Summarization* (pp. 82-88). Madrid, Spain.

Daniel Marcu. 1998a. Improving summarization through rhetorical parsing tuning. In *The Sixth Workshop on Very Large Corpora* (pp. 206-215). Montreal, Canada.

Daniel Marcu. 1998b. *The rhetorical parsing, summarization, and generation of natural texts.* (Doctor of Philosophy dissertation), University of Toronto, Toronto, Canada.

Daniel Marcu. 1998c. To build text summaries of high quality, nuclearity is not sufficient. In *Working Notes of the AAAI-98 Spring Symposium on Intelligent Text Summarization.* Stanford, CA: AAAI.

Daniel Marcu. 1999. Discourse trees are good indicators of importance in text. In *Advances in automatic text summarization* (pp. 123-136).

Daniel Marcu. 2000. *The theory and practice of discourse parsing and summarization*. Cambridge, MA: MIT Press.

J. R. Martin. 1992. *English text: System and structure*. Philadelphia: John Benjamins.

Christian M.I.M. Matthiessen, & Sandra A. Thompson. 1987. The structure of discourse and 'subordination'. In John Haiman & Sandra A. Thompson (Eds.), *Clause combining in grammar and discourse* (pp. 275-329). Amsterdam: John Benjamins.

Michael O'Donnell. 1997. RST-Tool: An RST analysis tool. In *Proceedings of the 6th*

European Workshop on Natural Language Generation. Duisburg, Germany: Gerhard-Mercator University.

Andrew Potter. 2019a. Reasoning between the lines: A logic of relational propositions. *Dialogue and Discourse, 9*(2), 80-110.

Andrew Potter. 2019b. The rhetorical structure of attribution. In Amir Zeldes, Debopam Das, Erick Maziero Galani, Juliano Desiderato Antonio, & Mikel Iruskieta (Eds.), *Proceedings of the Workshop on Discourse Relation Parsing and Treebanking (DISRPT2019)* (pp. 38-49). Minneapolis, MN: Association for Computational Linguistics.

Andrew Potter. 2023a. An algorithm for Pythonizing rhetorical structures. In Sara Carvalho, Anas Fahad Khan, Ana Ostroški Anić, Blerina Spahiu, Jorge Gracia, John P. McCrae, Dagmar Gromann, Barbara Heinisch, & Ana Salgado (Eds.), *Language, data and knowledge 2023 (LDK 2023): Proceedings of the 4th Conference on Language, Data and Knowledge* (pp. 493-503). Vienna, Austria: NOVA CLUNL.

Andrew Potter. 2023b. Text as tautology: an exploration in inference, transitivity, and logical compression. *Text & Talk, 43*(4), 471-503. doi:doi:10.1515/text-2020-0230

Ted J M Sanders, Vera Demberg, Jet Hoek, Merel C.J. Scholman, Fatemeh Torabi Asr, Sandrine Zufferey, & Jacqueline Evers-Vermeul. 2018. Unifying dimensions in coherence relations: How various annotation frameworks are related. *Corpus Linguistics and Linguistic Theory, 17*(1), 1-71.

Manfred Stede. 2008. RST revisited: Disentangling nuclearity. In Cathrine Fabricius-Hansen & Wiebke Ramm (Eds.), *'Subordination' versus 'coordination' in sentence and text – from a cross-linguistic perspective* (pp. 33-58). Amsterdam: Benjamins.

Manfred Stede, Maite Taboada, & Debopam Das. 2017. *Annotation guidelines for rhetorical structure*. Retrieved from Potsdam and Burnaby: http://www.sfu.ca/~mtaboada/docs/research/RST_Annotation_Guidelines.pdf

Bonnie Webber, & Rashmi Prasad. 2009. Discourse structure: Swings and roundabouts. *Oslo Studies in Language, 1*(1), 171-190.

Florian Wolf, & Edward Gibson. 2005. Representing discourse coherence: A corpus-based analysis. *Computational Linguistics, 31*(2), 249-287.

Amir Zeldes. 2016. rstWeb – A browser-based annotation interface for Rhetorical Structure Theory and discourse relations. In *Proceedings of NAACL-HLT 2016 (Demonstrations)* (pp. 1-5). San Diego, California: Association for Computational Linguistics.

Amir Zeldes. 2017. The GUM corpus: Creating multilayer resources in the classroom. *Language Resources and Evaluation, 51*(3), 581-561.

Amir Zeldes. 2023, November 20. Rhetorical Structure Theory annotation - RST++. Retrieved from https://wiki.gucorpling.org/gum/rst