A Corpus-based Analysis for
the Ordering of Clause Aggregation Operators

James Shaw
Multimeia/Video Technology Department
755 College Road East
Siemens Corporate Research, Inc.
Princeton, NJ 08540
shaw@scr.siemens.com

Abstract

To better understand the ordering of clause aggregation operators in a text generation application, we manually annotated a small corpus. The annotated corpus supports the preferred ordering of transformations that result in shorter surface expressions, such as adjectives over relative clauses. In addition, we were able to explain why paratactic operators are applied before and after hypotactic operators.

1 Introduction

Clause aggregation, the combination of multiple clauses to formulate a sentence, is a complex process. This work focuses on the ordering of clause aggregation operators. Scott and de Souza (1990) suggested the heuristics that "syntactically simple expressions of embedding are to be preferred over more complex ones." Shaw (1998a) also concurred with such ordering preference based on a small domain specific corpus. In the current analysis, we manually annotated two larger corpora and try to find evidence to support an ordering similar to the one proposed by Scott, de Souza, and Shaw.

The type of clause aggregation operators studied in this analysis are syntactic ones – i.e., conjunction, adjective, and relative clause transformations. As a general planning task in AI, sequential ordering of applying multiple operators is an issue because aggregation operators are not commutative – applying one of the operators to the input propositions prevents application of others. For example, two clauses can be combined with either a conjunction transformation or a relative clause transformation, but not both. In addition, depending on the ordering of operators, different meanings might result. In Example (1a), the first two propositions are linked by a JOINT relation,

and the second and third propositions are linked by a CONCESSION relation. Applying subordinate clause transformation before conjunction, Sentence (1b) might be produced. In (1b), the modifying proposition only modifies the proposition "John ate oranges" and not "John drank cider." Applying the operators in the reverse ordering, Sentence (1c) results. In this case, the modifying proposition has a wide scope and modifies both propositions, (1aa) and (1ab).

(1) a. a. John drank cider.
   b. John ate oranges.
   c. (even though) John didn't like fruits.

   b. John drank cider and even though he didn't like fruits, he ate oranges.

   c. Even though John didn't like fruits, he drank cider and ate oranges.

Clearly, the ordering of operators can have an impact on the meaning of the aggregated sentences. This work explores the interactions between aggregation operators and uses a corpus-based approach to evaluate a specific ordering of these operators based on our understanding of their characteristics.

In our analysis, clause aggregation operators are categorized as either *paratactic* or *hypotactic*. Paratactic operators create conjoined constituents with equal syntactic status, i.e., simple and complex conjunctions; hypotactic operators create constructions with subordinate constituents, i.e., adjective, prepositional phrase, reduced relative clause, relative clause operators, and non-ELABORATION transformations. Early in our effort, we permuted the aggregation operators to exhaustively list all possible orderings among them and tried to identify the best ordering. But such permutation analysis was inadequate. In particular, we were intrigued by the fact that paratactic operators seem to be

applied more frequently to input propositions than hypotactic ones, i.e., paratactic, hypotactic, and then paratactic again.

In theory, for most rhetorical relations, a rhetorical relation can be realizes by either a paratactic operator or a hypotactic operator, but not both. In practice, it is difficult for a content planner to specify rhetorical relations in such a way so that the sentence planner can simply perform an one to one transformation. For example, since in general only constituents of the same syntactic category can be conjoined and a content planner lacks detailed syntactic information, a content planner cannot always correctly specify JOINT relations to all the modifying propositions which will be transformed into a conjoined constituent in the final sentence. Section 4 contains an example in which despite no JOINT relation is specified among the input propositions, the conjunctor "and," a clear surface marker of JOINT relation, appears in the final surface form.

Section 2 describes the corpus-based methodology used to evaluate the effectiveness of our proposed sequential ordering of the aggregation operators. Section 3 provides a brief description of related work. The markup language used for this annotation is described in Section 4. Section 5 presents the result of the analysis and evaluates our proposed ordering based on the annotated corpus. Section 6 provides a rationale for our ordering preference.

2 Methodology

After realizing that permutation analysis cannot be used to find the optimal ordering of clause aggregation operators, we settled on a more modest goal – to show that our proposed ordering of clause aggregation operators works well in reconstructing human-written sentences. Given a specific aggregation operator ordering, evaluation was performed using a manually annotated corpus to determine if applying the operators in the specific order will reconstruct the original sentences. The operator ordering used in the current evaluation is shown in Figure 1. If such an ordering works well for the annotated corpus, researchers can be confident that NLG systems using such an ordering will work well.

In our evaluation, a special corpus was used. To increase the chance of encountering sentences that underwent both paratactic and hy-

1. Adjective (conjunction optional)
2. Prepositional phrase (conjunction optional)
3. Reduced relative clause, including apposition (conjunction optional)
4. Relative clause (conjunction optional)
5. Transformations for other rhetorical relations (conjunction optional)
6. Simple conjunction
7. Complex conjunction

Figure 1: Our proposed ordering of clause aggregation operators.

potactic transformations, only sentences contain the conjunctor "and" were selected for annotation. Since our goal is to analyze the ordering between hypotactic and paratactic operators, these sentences are more likely to contain both types of operators than those without. By intentionally making the set of sentences to be analyzed more complex, it is more likely that evidence either supporting or negating our proposed ordering will be found. The current analysis uses a corpus from a medical domain and from Wall Street Journal. Due to the amount of effort needed to annotate complex sentences, only one hundred sentences from each domain are annotated. During the annotation stage, following tasks are performed:

- **De-aggregate original sentence**: The original sentences are broken down into smaller propositions. This is basically an ellipsis recovering process.

- **Specify rhetorical relations between propositions**: Identify the rhetorical relations between the de-aggregated propositions. This is necessary because sentences are aggregated based on the fact that they are related pragmatically or rhetorically.

- **Specify a sequence of transformation operators to combine de-aggregated propositions into the original sentence**: This is used to evaluate the applicability of our proposed ordering.

For each annotated sentence, if the sequence of the transformations does not violate our proposed ordering, it is considered as positive evidence supporting our proposed ordering. If the sequence of transformations used violates our proposed ordering, it is a negative evidence. The result from tallying the number of positive

and negative evidence will indicate how well the proposed ordering works.

The current analysis focuses on the operator ordering among different types of operators. When the same operator is applied multiple times, such as a sequence of adjective transformations, the ordering decision among the same operator is outside the scope of the current annotation effort. Shaw (1999) and Malouf (2000) addressed such linearization issue using other corpus-base approaches. Among different types of hypotactic operators, we assumed that the operators applied earlier should be closer to the head than the constituent results from operators applied later. For example, if a reduced relative clause operator is applied before a relative clause operator, the reduced relative clause will appear closer to the head than the relative clause at the surface level.

3 Related Work

The type of corpus annotation performed in this analysis is similar to discourse annotations such as RST analysis (Mann and Thompson, 1988) or cohesion analysis (Halliday and Hasan, 1976). Such effort has always been quite time consuming and laborious. To facilitate discourse annotation effort, various graphical tools have been developed (O'Donnell, 2000; Garside and Rayson, 1997). Recently, eXtensible Markup Language (XML) has been gaining popularity as the meta-language for such annotation, i.e., LT XML tool and MATE workbench from the Edinburgh Language Technology Group. Despite attempts to automate the process (Marcu, 2000), the type of annotation performed in this work must be done manually. In particular, the recovering of elided constituents during the de-aggregation process is difficult to automate. Similar to other works in clause aggregation (Scott and de Souza, 1990; Moser and Moore, 1995; Rösner and Stede, 1992), the current work uses rhetorical relations extensively. Our focus is on issues related to combining operations that transform linked clauses into sentences based on these rhetorical relations.

4 The Annotation

Section 4.1 describes the markup language used for annotation. Section 4.2 provides details on the concept of *proposition set* or *propset*, a useful device that facilitates our annotation effort.

4.1 The Markup Language

The de-aggregated sentences are annotated using XML notation. Each sentence entry consists of five parts. The first part is the original sentence. The second part is a list of de-aggregated propositions after manual reconstruction of the ellided constituents. These propositions are enclosed in a propset[1], which might contain nested propsets. The third section specifies the rhetorical relations which linked the de-aggregated propositions or propsets to create cohesion. The number of rhetorical relations in a sentence entry is always one less than the number of propositions. The fourth section is a sequence of transformations that can be applied to the de-aggregated propositions to reconstruct the original sentence. The fifth section contains annotator's comments. One of them, `seqordering` tag, indicates whether the sequence of the transformations in the transformation annotation section violates or adheres to the proposed aggregation operator ordering. The `conj` tag indicates whether a conjunctor "and" in the original sentence contains a collective or distributive reading. Following annotated sentence entry is an example taken from our corpus.

```
<sentence id="s32">
  Local sports fans themselves, long known
  for their passive demeanor at games and
  propensity to leave early, don't resist
  the image.
 <propset id="pset32-1">
   <prop id="p32-1">
     Local sports fans don't resist the
     image. </prop>
   <prop id="p32-2">
     Local sports fans are long known for
     their passive demeanor at games.</prop>

   <prop id="p32-3">
     Local sports fans are long known for
     their propensity to leave early.</prop>
   </propset>

   <focus entity='local sports fans'/>
   <rst-rel id="r32-1" name="elab"
     nuc="p32-1"  sat="p32-2"  ref="no"/>
   <rst-rel id="r32-2" name="elab"
     nuc="p32-1"  sat="p32-3"  ref="no"/>

   <trans id="tx32-1" name="conj-simp"
     nuc="p32-2" sat="p32-3" />
   <trans id="tx32-2"
     name="rel-reduced-del-wh-be"
     nuc="p32-1" sat="tx32-1" />

   <seqorder valid="true" />
   <conj id="c32-1" type="dist" />
</sentence>
```

---

[1] The concept of propset will be discussed in Section 4.2.

In this example, the original sentence is broken into three propositions, with propositions p32-2 and p32-3 modifying p32-1 with ELABORATION relations, r32-1 and r32-2. Both proposition p32-2 and p32-3 can be transformed into reduced relative clauses modifying p32-1, "[who are] long known for their passive..." and "[who are] long known for their propensity..." Using the ordering specified in Figure 1, reduced relative clause operator is applied first. Because p32-2 and p32-3 are syntactically similar and can be conjoined using conjunction, the optional conjunction operator is activated before the reduced relative clause operator is applied. After the first simple conjunction transformation, the intermediate results can be expressed as the following:

```
<prop id="p32-1">
  Local sports fans don't resist the
  image. </prop>
<prop id="tx32-1">
  Local sports fans are long known for
  their passive demeanor at games and
  propensity to leave early. </prop>
```

with <prop id="tx32-1"> contains the result of applying simple conjunction transformation to p32-2 and p32-3. The combined result <prop id="tx32-1"> undergoes further transformation in <trans id="tx32-2"> as a satellite proposition to nucleus proposition <prop id="p32-1">. The reduced relative clause transformation deletes "who" and "be", and the original sentence is reproduced:

```
<prop id="tx32-2">
  Local sports fans, long known for their
  passive demeanor at games and propensity to
  leave early, don't resist the image. </prop>
```

As explained earlier in Section 1, because of lack of detailed syntactic information, it is difficult for content planners to specify JOINT relations to all the modifying propositions which might be combined and appear as a conjoined constituent in the final sentence. Instead of performing such task in content planners, in our system, the sentence planner opportunistically uses the conjunctor, "and," to combine these two syntactically similar modifying propositions in the final surface form.

4.2 The *Proposition Set* Concept

In our preliminary effort to annotate the selected sentences with rhetorical relations, we realized that simply specifying rhetorical relations among the de-aggregated propositions did not seem to provide sufficient information to reproduce the original sentence. For example, the propositions in Sentence (1a) in Section 1 can be realized as either Sentence (2a) or (2b) depending on whether a hypotactic operator or a conjunction operator is applied first.

(2) a. John drank cider and even though he didn't like fruits, he ate oranges.

   b. Even though John didn't like fruits, he drank cider and ate oranges.

In Sentence (2a), the third proposition (1ac) only modifies second proposition (1ab), not the first (1aa). The JOINT relation between the event (1aa) and (1ab) describes merely events and one of them is in conflict with the fact "John didn't like fruits." While in Sentence (2b), the proposition (1ac) has a wide scope and modifying both propositions (1aa) and (1ab). To clarify the scope of such modifying construction, we came up with the concept of *proposition set*, or *propset* which facilitates the specification of the scope of modifying proposition, as shown below.

```
<sentence id="s1">
  <propset id="pset1-1">
    <prop id="p1-1">
      John drank cider.  </prop>
    <propset id="pset1-2">
      <prop id="p1-2">
        John ate oranges.  </prop>
      <prop id="p1-3">
        (even though) John didn't like
        fruits. </prop>
    </propset>
  </propset>

  <focus entity='John'/>
  <rst-rel id="r1-1" name="joint"
    nuc="p1-1" sat="pset1-2" ref="no"/>
  <rst-rel id="r1-2" name="concession"
    nuc="p1-2" sat="p1-3" ref="no"/>
</sentence>
```

The annotation for sentence s1 specified a narrow scope for the modifying proposition (p1-3), as in Sentence (2a). In rhetorical relation, r1-2, the second and third propositions are linked by a CONCESSION relation. Together, they are linked to p1-1 through a JOINT relation in r1-1.

```
<sentence id="s2">
  <propset id="pset1-1">
    <propset id="pset1-2">
      <prop id="p1-1">
        John drank cider.</prop>
      <prop id="p1-2">
```

```
        John ate oranges.  </prop>
    </propset>
    <prop id="p1-3">
      (even though) John didn't like
      fruits. </prop>
  </propset>

    <focus entity='John'/>
    <rst-rel id="r1-1" name="joint"
      nuc="p1-1" sat="p1-2" ref="no"/>
    <rst-rel id="r1-2" name="concession"
      nuc="pset1-2" sat="p1-3" ref="no"/>
</sentence>
```

In the second annotation for sentence s2, the modifying proposition, p1-3, has a wide scope. To specify that a modifying proposition modifies both p1-1 and p1-2, *propset* is used to group the two propositions before they are jointly modified by p1-3. As a result of making the scope clear in the de-aggregated proposition, our system can present either Sentence (2a) or (2b) and ensures correct scopings of modifying propositions are conveyed.

Incorporating the concept of propset into annotation provided several benefits:

- **Specify certain propositions are more tightly related**. Tightly related events are grouped together in a propset, such as events related to a patient's smoking habit, "he was a smoker" and "he quit 10 years ago." The system treats propositions in a propset as one proposition and will combine them first before aggregating the combined proposition with others.
- **Simplify the annotation process for certain constructions**. Information contained in the embedded S-structure of verbs like "said" or "believe" can be extracted and analyzed as a propset. For example, "John believed Tim invested in stock and real estate." Without using propset, the subject and verb of the main clause would appear multiple times in the de-aggregated propositions; i.e., "John believed Tim invested in stock" and "John believed Tim invested in real estate." By eliminating such recurrences of the same main subjects and verbs, aggregation analysis is simplified. The transformations which combine all the propositions in the propset of an embedded S-structure are annotated as "ARG" transformations. They are just cosmetic artifacts and unlikely to have any impact on the analysis of the ordering of the operators.
- **Minimize scope ambiguity**. The earlier examples, (2a) and (2b), illustrate this point

as well. By using propset, the scope of the modifying proposition can be made explicit.
- **Minimize redundant specification of multiple modifying rhetorical relations**. When a proposition modifies multiple propositions at the same time, the propositions being modified can be grouped under a propset so that only one rhetorical relation need be specified between the modifying proposition and the propset being modified. If in such case, multiple rhetorical relations are specified between each modifying proposition and propositions being modified, the number of rhetorical relations could be greater than the number of propositions. Since transformation operators are directly related to rhetorical relations, extra or redundant specifications of rhetorical relations would introduce complications to the implementation of the aggregation operators.

The elimination of specifying multiple rhetorical relations for a single proposition is particularly important because it makes one transformation operator corresponds to a single rhetorical relation. This simplification makes the aggregation task more manageable.

5   The Results

The 200-sentence corpus was de-aggregated into 763 clauses, about 3.8 clauses per sentence. After specifying the transformation operators for the sentences according to our proposed sequential ordering, the majority of the sentences can be resynthesized from the de-aggregated propositions using our ordering of aggregation operators (195 out of 200). The percentage is quite high because the incorporation of *propset* in the annotation takes care of many cases which would have violated our proposed ordering. This result provides evidence supporting our claim that the proposed ordering shown in Figure 1 is effective.

Excluding the 40 "ARG" relations in our analysis, there are 20 different types of rhetorical relations identified in the corpus, with a total of 523 rhetorical relations. The interesting one for our analysis are ELABORATION, JOINT, and SEQUENCE. Together, these three rhetorical relations made up of 440 of 523 rhetorical relations. Except for a few of them (e.g., joint-collective, alternative, and comparative), the other rhetorical relations are hypotactic in nature.

The full annotated corpus is available through the Web (www.cs.columbia.edu/~shaw/col02). In the annotated corpus, excluding "ARG" transformations which are not involved in the ordering of aggregation operators, there are 523 transformations used, roughly 2.6 transformations for each sentence. Of the 523 transformations, 417 (80%) of them are transformations related to JOINT, ELABORATION, and SEQUENCE, while 106 (20%) of them are not implemented at all. The transformations which were not implemented in our system include "or", parenthesis, using "with" for paratactic operation, or any transformation which involves extraction. In the analysis, we did not remove sentences containing transformations which our system does not handle because doing so would eliminate many complex sentences appropriate for our analysis. Instead, unhandled transformations are categorized as either hypotactic and paratactic, and they are mapped to the closest type of transformations during evaluation. Since the sentences selected for analysis are not random because they all contain the word "and," this bias might create a tendency to select sentences with transformations our system handles well, such as paratactic transformation (62% contains conjunctor "and"). Given that our goal was to find as much interactions between hypotactic and paratactic operators as possible, this bias is reasonable.

Of the five unsupported cases, two of them involved application of relative clause transformation before reduced relative clause. One such sentence is shown below:

```
The patient was a 38-year-old woman from
the Dominican Republic [who presented to
the Cardiology Clinic in 11/90] [complaining
of dyspnea on exertion and palpitations].
```

In this example, the relative clause ("who presented...") is closer to its head, "woman," than the reduced relative clause ("complaining of ..."). Based on their surface ordering, the constituents in the original sentences indicated that a relative clause transformation is applied before the reduced relative clause transformation, and violates our ordering preference shown in Figure 1. The other unsupported cases involved realizing JOINT relations using hypotactic constructions or realizing ELABO-RATION relations using paratactic construction. Since these transformations are not the expected transformation operators for the rhetorical relations, they violated our proposed ordering. Overall, unsupported cases are rare.

6   Rationale for the Ordering

Before the current annotation effort was underway, the ordering of operators used in our system was paratactic operators, hypotactic operators, and then paratactic operators again. It was not clear why paratactic operators are applied multiple times while hypotactic operators only once. It was not clear if there were different types of JOINT relations connecting the propositions which resulted in multiple applications of paratactic operators. After preliminary annotation of the corpus using propset, the answer became clear – the first application of paratactic operators is a sub-step of the hypotactic operations which combine satellite propositions with subordinate rhetorical relations (i.e., ELABORATION) that have similar structures and modify the same entity in their nucleus proposition. The ordering of clause aggregation operators should be hypotactic operators followed by paratactic operators, but inside the hypotactic operators, paratactic operators are also applied to specific configurations of satellite propositions as an optimization.

We believe the ordering of hypotactic and paratactic operators might be related to the locality of their operations. The operations to insert a modifying constituent into a sentence is a local operation because such insertion can be done without considering other constituents in the sentence which are not being modified. For example, attaching a prepositional phrase "with deep pocket" to the sentence "Bob Morgan is a reputable stock-broker who is interested in dot coms." can be performed without considering how "with deep pocket" interacts with adjectives or the relative clause, or where the entity being modified, "stock-broker", appears in the sentence. In contrast, paratactic operators are global in nature because they are very sensitive to constituents that are identical across all the propositions being combine. Due to *directional constraint* (Ross, 1970; Shaw, 1998b), the deletion of identical constituents cannot be made locally but must wait until the surface ordering of the identical constituents is known. In

comparison, hypotactic operations have fewer constraints and should be applied earlier.

The proposed sequence in Figure 1 applies all hypotactic operations before paratactic operations. The ordering for intra-hypotactic operators is chosen to produce the most concise sentence by applying operators producing shortest transformed constituents first. Similarly, simple conjunction operator is applied before complex conjunction operator because the simple conjunction operator produces more concise expressions. Other hypotactic transformations for non-ELABORATION relation are treated similarly as relative clause transformations. Since there is little or no deletion result from such constructions ("Because John likes fruit, he ate oranges." has no deletion), they are low on the priority and thus become the last one of the hypotactic transformations.

7   Conclusion

Current work made two signification observations. First, in Section 6 we explained why some paratactic operators are applied before the hypotactic operators while others are applied later. Secondly, *propset* was used for annotating propositions during the de-aggregation process. The importance of rhetorical relations in clause aggregation operations were noted by many researchers (Scott and de Souza, 1990; Moser and Moore, 1995; Rösner and Stede, 1992), but the concept of propset was not mentioned in previous literature related to rhetorical relations. It facilitates annotation during the de-aggregation process and allows annotator to ensure that both the number of transformations and rhetorical relations is always one smaller than the number of propositions. This kept both the de-aggregation and aggregation process manageable.

The goal of this work is to find evidence to support the proposed ordering of the aggregation operators for synthesizing grammatical and concise sentences. By imposing our proposed ordering onto de-aggregated propositions and trying to re-synthesize the original sentences, we determined the proposed ordering works well based on a human-written corpus. Using such ordering information and ensuring the content planner can specify propositions, propset, and rhetorical relations as the markup in the annotated corpus, the research community can incor-porate clause aggregation operations into natural language generation systems and expect grammatical, concise sentences to be automatically generated.

References

Roger Garside and Paul Rayson. 1997. Higher-level annotation tools. In R. Garside, G. Leech, and A. McEnery, editors, *Corpus Annotation: Linguistic Information from Computer Text Corpora*.

Michael A. K. Halliday and R. Hasan. 1976. *Cohesion in English*.

Robert Malouf. 2000. The order of prenominal adjectives in natural language generation. In *Proc. of the 38th ACL*.

William C. Mann and Sandra A. Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text*, 8(3):243–281.

Daniel Marcu. 2000. The rhetorical parsing of unrestricted texts: A surface-based approach. *Computational Linguistics*, 26(3):395–448.

Megan Moser and Johanna D. Moore. 1995. Investigating cue selection and placement in tutorial discourse. In *Proc. of the 33rd ACL*.

Michael O'Donnell. 2000. RSTTool 2.4 – a markup tool for rhetorical structure theory. In *Proc. of the 1st INLG Conference*.

Dietmar Rösner and Manfred Stede. 1992. Customizing RST for the automatic production of technical manuals. In R. Dale, E. Hovy, D. Rösner, and O. Stock, editors, *Aspects of Automated Natural Language Generation*.

John Robert Ross. 1970. Gapping and the order of constituents. In M. Bierwisch and K. Heidolph, editors, *Progress in Linguistics*.

Donia R. Scott and Clarisse S. de Souza. 1990. Getting the message across in RST-based text generation. In R. Dale, C. Mellish, and M. Zock, editors, *Current Research in Natural Language Generation*.

James Shaw and Vasileios Hatzivassiloglou. 1999. Ordering among premodifiers. In *Proc. of the 37th ACL*.

James Shaw. 1998a. Clause aggregation using linguistic knowledge. In *Proc. of the 9th INLG*.

James Shaw. 1998b. Segregatory coordination and ellipsis in text generation. In *Proc. of the 17th COLING and the 36th ACL*.