Proceedings of the 41st Annual Meeting of the Association
for Computational Linguistics, July 2003, pp. 264-271.

Integrating Discourse Markers into a Pipelined Natural Language Generation Architecture

Charles B. Callaway
ITC-irst, Trento, Italy
via Sommarive, 18
Povo (Trento), Italy, I-38050
callaway@itc.it

Abstract

Pipelined Natural Language Generation (NLG) systems have grown increasingly complex as architectural modules were added to support language functionalities such as referring expressions, lexical choice, and revision. This has given rise to discussions about the relative placement of these new modules in the overall architecture. Recent work on another aspect of multi-paragraph text, discourse markers, indicates it is time to consider where a discourse marker insertion algorithm fits in. We present examples which suggest that in a pipelined NLG architecture, the best approach is to strongly tie it to a revision component. Finally, we evaluate the approach in a working multi-page system.

1 Introduction

Historically, work on NLG architecture has focused on integrating major disparate architectural modules such as discourse and sentence planners and surface realizers. More recently, as it was discovered that these components by themselves did not create highly readable prose, new types of architectural modules were introduced to deal with newly desired linguistic phenomena such as referring expressions, lexical choice, revision, and pronominalization.

Adding each new module typically entailed that an NLG system designer would justify not only the reason for including the new module (*i.e.*, what linguistic phenomena it produced that had been previously unattainable) but how it was integrated into their architecture and why its placement was reasonably optimal (*cf.*, (Elhadad et al., 1997), pp. 4–7).

At the same time, (Reiter, 1994) argued that implemented NLG systems were converging toward a *de facto* pipelined architecture (Figure 1) with minimal-to-nonexistent feedback between modules. Although several NLG architectures were proposed in opposition to such a linear arrangement (Kantrowitz and Bates, 1992; Cline, 1994), these research projects have not continued while pipelined architectures are still actively being pursued.

In addition, Reiter concludes that although complete integration of architectural components is theoretically a good idea, in practical engineering terms such a system would be too inefficient to operate and too complex to actually implement. Significantly, Reiter states that fully interconnecting every module would entail constructing $N(N-1)$ interfaces between them. As the number of modules rises (*i.e.*, as the number of large-scale *features* an NLG engineer wants to implement rises) the implementation cost rises exponentially. Moreover, this cost does not include modifications that are not component specific, such as multilingualism.

As text planners scale up to produce ever larger texts, the switch to multi-page prose will introduce new features, and consequentially the number of architectural modules will increase. For example, Mooney's EEG system (Mooney, 1994), which created a full-page description of the Three-Mile Island nuclear plant disaster, contains components for discourse knowledge, discourse organization, rhetori-

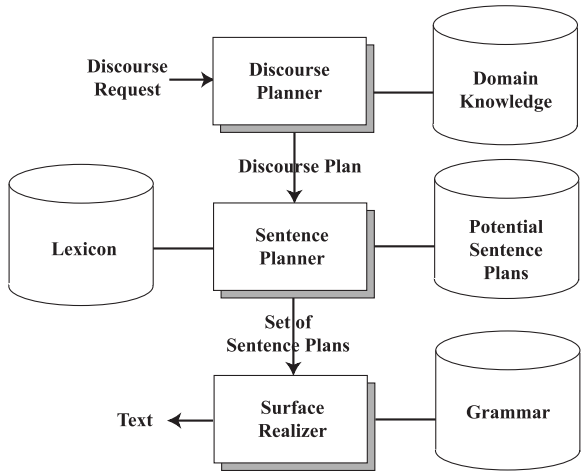

Figure 1: A Typical Pipelined NLG Architecture

cal relation structuring, sentence planning, and surface realization. Similarly, the STORYBOOK system (Callaway and Lester, 2002), which generated 2 to 3 pages of narrative prose in the Little Red Riding Hood fairy tale domain, contained seven separate components.

This paper examines the interactions of two linguistic phenomena at the paragraph level: revision (specifically, clause aggregation, migration and demotion) and discourse markers. Clause aggregation involves the syntactic joining of two simple sentences into a more complex sentence. Discourse markers link two sentences semantically without necessarily joining them syntactically. Because both of these phenomena produce changes in the text at the clause-level, a lack of coordination between them can produce interference effects.

We thus hypothesize that the architectural modules corresponding to revision and discourse marker selection should be tightly coupled. We then first summarize current work in discourse markers and revision, provide examples where these phenomena interfere with each other, describe an implemented technique for integrating the two, and report on a preliminary system evaluation.

2 Discourse Markers in NLG

Discourse markers, or cue words, are single words or small phrases which mark specific semantic relations between adjacent sentences or small groups of sentences in a text. Typical examples include words like *however*, *next*, and *because*. Discourse markers pose a problem for both the parsing and generation of clauses in a way similar to the problems that referring expressions pose to noun phrases: changing the lexicalization of a discourse marker can change the semantic interpretation of the clauses affected.

Recent work in the analysis of both the distribution and role of discourse markers has greatly extended our knowledge over even the most expansive previous accounts of discourse connectives (Quirk et al., 1985) from previous decades. For example, using a large scale corpus analysis and human subjects employing a *substitution test* over the corpus sentences containing discourse markers, Knott and Mellish (1996) distilled a taxonomy of individual lexical discourse markers and 8 binary-valued features that could be used to drive a discourse marker selection algorithm.

Other work often focuses on particular semantic categories, such as temporal discourse markers. For instance, Grote (1998) attempted to create declarative lexicons that contain applicability conditions and other constraints to aid in the process of discourse marker selection. Other theoretical research consists, for example, of adapting existing grammatical formalisms such as TAGs (Webber and Joshi, 1998) for discourse-level phenomena.

Alternatively, there are several implemented systems that automatically insert discourse markers into multi-sentential text. In an early instance, Elhadad and McKeown (1990) followed Quirk's pre-existing non-computational account of discourse connectives to produce single argumentative discourse markers inside a functional unification surface realizer (and thereby postponing lexicalization till the last possible moment).

More recent approaches have tended to move the decision time for marker lexicalization higher up the pipelined architecture. For example, the MOOSE system (Stede and Umbach, 1998; Grote and Stede, 1999) lexicalized discourse markers at the sentence planning level by pushing them directly into the lexicon. Similarly, Power *et al.* (1999) produce multiple discourse markers for Patient Information Leaflets using a constraint-based method applied to RST trees during sentence planning.

Finally, in the CIRC-SIM intelligent tutoring system (Yang et al., 2000) that generates connected di-

alogues for students studying heart ailments, discourse marker lexicalization has been pushed all the way up to the discourse planning level. In this case, CIRC-SIM lexicalizes discourse markers inside of the discourse schema templates themselves.

Given that these different implemented discourse marker insertion algorithms lexicalize their markers at three distinct places in a pipelined NLG architecture, it is not clear if lexicalization can occur at any point without restriction, or if it is in fact tied to the particular architectural modules that a system designer chooses to include.

The answer becomes clearer after noting that none of the implemented discourse marker algorithms described above have been incorporated into a comprehensive NLG architecture containing additional significant components such as revision (with the exception of MOOSE's lexical choice component, which Stede considers to be a submodule of the sentence planner).

3 Current Implemented Revision Systems

Revision (or clause aggregation) is principally concerned with taking sets of small, single-proposition sentences and finding ways to combine them into more fluent, multiple-proposition sentences. Sentences can be combined using a wide range of different syntactic forms, such as conjunction with "and", making relative clauses with noun phrases common to both sentences, and introducing ellipsis.

Typically, revision modules arise because of dissatisfaction with the quality of text produced by a simple pipelined NLG system. As noted by Reape and Mellish (1999), there is a wide variety in revision definitions, objectives, operating level, and type. Similarly, Dalianis and Hovy (1993) tried to distinguish between different revision parameters by having users perform revision thought experiments and proposing rules in RST form which mimic the behavior they observed.

While neither of these were implemented revision systems, there have been several attempts to improve the quality of text from existing NLG systems. There are two approaches to the architectural position of revision systems: those that operate on semantic representations before the sentence planning level, of which a prototypical example is (Horacek, 2002), and those placed after the sentence planner, operating on syntactic/linguistic data. Here we treat mainly the second type, which have typically been conceived of as "add-on" components to existing pipelined architectures. An important implication of this architectural order is that the revision components expect to receive *lexicalized* sentence plans.

Of these systems, Robin's STREAK system (Robin, 1994) is the only one that accepts both lexicalized and non-lexicalized data. After a sentence planner produces the required lexicalized information that can form a complete and grammatical sentence, STREAK attempts to gradually aggregate that data. It then proceeds to try to opportunistically include additional optional information from a data set of statistics, performing aggregation operations at various syntactic levels. Because STREAK only produces single sentences, it does not attempt to add discourse markers. In addition, there is no *a priori* way to determine whether adjacent propositions in the input will remain adjacent in the final sentence.

The REVISOR system (Callaway and Lester, 1997) takes an entire sentence plan at once and iterates through it in paragraph-sized chunks, employing clause- and phrase-level aggregation and reordering operations before passing a revised sentence plan to the surface realizer. However, at no point does it add information that previously did not exist in the sentence plan. The RTPI system (Harvey and Carberry, 1998) takes in sets of multiple, lexicalized sentential plans over a number of medical diagnoses from different critiquing systems and produces a single, unified sentence plan which is both coherent and cohesive.

Like STREAK, Shaw's CASPER system (Shaw, 1998) produces single sentences from sets of sentences and doesn't attempt to deal with discourse markers. CASPER also delays lexicalization when aggregating by looking at the lexicon twice during the revision process. This is due mainly to the efficiency costs of the unification procedure. However, CASPER's sentence planner essentially uses the first lexicon lookup to find a "set of lexicalizations" before eventually selecting a particular one.

An important similarity of these pipelined revision systems is that they all manipulate lexicalized representations at the clause level. Given that both aggregation and reordering operators may sep-

arate clauses that were previously adjacent upon leaving the sentence planner, the inclusion of a revision component has important implications for any upstream architectural module which assumed that initially adjacent clauses would remain adjacent throughout the generation process.

4 Architectural Implications

The current state of the art in NLG can be described as small pipelined generation systems that incorporate some, but not all, of the available pipelined NLG modules. Specifically, there is no system to-date which both revises its output and inserts appropriate discourse markers. Additionally, there are no systems which utilize the latest theoretical work in discourse markers described in Section 2. But as NLG systems begin to reach toward multi-page text, combining both modules into a single architecture will quickly become a necessity if such systems are to achieve the quality of prose that is routinely achieved by human authors.

This integration will not come without constraints. For instance, discourse marker insertion algorithms assume that sentence plans are static objects. Thus any change to the static nature of sentence plans will inevitably disrupt them. On the other hand, revision systems currently do not add information not specified by the discourse planner, and do not perform true lexicalization: any new lexemes not present in the sentence plan are merely delayed lexicon entry lookups. Finally, because revision is potentially destructive, the sentence elements that lead to a particular discourse marker being chosen may be significantly altered or may not even exist in a post-revision sentence plan.

These factors lead to two partial order constraints on a system that both inserts discourse markers and revises at the clause level after sentence planning:

- **Discourse marker lexicalization cannot precede revision**
- **Revision cannot precede discourse marker lexicalization**

In the first case, assume that a sentence plan arrives at the revision module with discourse markers already lexicalized. Then the original discourse marker may not be appropraite in the revised sentence plan. For example, consider how the application of the following revision types requires different lexicalizations for the initial discourse markers:

- **Clause Aggregation**: The merging of two main clauses into one main clause and one subordinate clause:

  John had always liked to ride motorbikes. $\oplus$ **On account of this**, his wife passionately hated motorbikes. $\Rightarrow$
  John had always liked to ride motorbikes, which his wife {*** on account of this** | **thus**} passionately hated.

- **Reordering**: Two originally adjacent main clauses no longer have the same fixed position relative to each other:

  Diesel motors are well known for emitting excessive pollutants. $\oplus$ **Furthermore**, diesel is often transported unsafely. $\oplus$ **However**, diesel motors are becoming cleaner. $\Rightarrow$
  Diesel motors are well known for emitting excessive pollutants, {*** however** | **although**} they are becoming cleaner. **Furthermore**, diesel is often transported unsafely.

- **Clause Demotion**: Two main clauses are merged where one of them no longer has a clause structure:

  The happy man went home. $\oplus$ **However**, the man was poor. $\Rightarrow$
  The happy {*** however** | **but**} poor man went home.

These examples show that if discourse marker lexicalization occurs before clause revision, the changes that the revision module makes can render those discourse markers undesirable or even grammatically incorrect. Furthermore, these effects span a wide range of potential revision types.

In the second case, assume that a sentence plan is passed to the revision component, which performs various revision operations before discourse markers are considered. In order to insert appropriate discourse markers, the insertion algorithm must access the appropriate rhetorical structure produced by the discourse planner. However, there is no guarantee

that the revision module has not altered the initial organization imposed by the discourse planner. In such a case, the underlying data used for discourse marker selection may no longer be valid.

For example, consider the following generically represented discourse plan:

C1: "John and his friends went to the party."
[temporal "before" relation, *time(C1, C2)*]
C2: "John and his friends gathered at the mall."
[causal relation, *cause(C2, C3)*]
C3: "John had been grounded."

One possible revision that preserved the discourse plan might be:

> "**Before** John and his friends went to the party, they gathered at the mall **since** he had been grounded."

In this case, the discourse marker algorithm has selected "before" and "since" as lexicalized discourse markers prior to revision. But there are other possible revisions that would destroy the ordering established by the discourse plan and make the selected discourse markers unwieldy:

> "John, {* **since** | ϵ} who had been grounded, gathered with his friends at the mall **before** going to the party."

> "{* **Since** | **Because**} he had been grounded, John and his friends gathered at the mall and {* **before** | **then**} went to the party."

Reordering sentences without updating the discourse relations in the discourse plan itself would result in many wrong or misplaced discourse marker lexicalizations. Given that discourse markers cannot be lexicalized before clause revision is enacted, and that clause revision may alter the original discourse plan upon which a later discourse marker insertion algorithm may rely, it follows that the revision algorithm should update the discourse plan as it progresses, and the discourse marker insertion algorithm should be responsive to these changes, thus delaying discourse marker lexicalization.

5 Implementation

To demonstrate the application of this problem to real world discourse, we took the STORYBOOK (Callaway and Lester, 2001; Callaway and Lester, 2002) NLG system that generates multi-page text in the form of Little Red Riding Hood stories and New York Times articles, using a pipelined architecture with a large number of modules such as revision (Callaway and Lester, 1997). But although it was capable of inserting discourse markers, it did so in an ad-hoc way, and required that the document author notice possible interferences between revision and discourse marker insertion and hard-wire the document representation accordingly.

Upon adding a principled discourse marker selection algorithm to the system, we soon noticed various unwanted interactions between revision and discourse markers of the type described in Section 4 above. Thus, in addition to the other constraints already considered during clause aggregation, we altered the revision module to also take into account the information available to our discourse marker insertion algorithm (in our case, intention and rhetorical predicates). We were thus able to incorporate the discourse marker selection algorithm into the revision module itself.

This is contrary to most NLG systems where discourse marker lexicalization is performed as late as possible using the modified discourse plan leaves after the revision rules have reorganized all the original clauses. In an architecture that doesn't consider discourse markers, a generic revision rule without access to the original discourse plan might appear like this (where *type* refers to the main clause syntax, and *rhetorical type* refers to its intention):

If type(clause1) = <*type*>
type(clause2) = <*type*>
subject(clause1) = subject(clause2)
then make-subject-relative-clause(clause1, clause2)

But by making available the intentional and rhetorical information from the discourse plan, our modified revision rules instead have this form:

If rhetorical-type(clause1) = <*type*>
rhetorical-type(clause2) = <*type*>
subject(clause1) = subject(clause2)
rhetorical-relation(clause1, clause2) $\asymp$ *set-of-features*
then make-subject-relative-clause(clause1, clause2)
lexicalize-discourse-marker(clause1, *set-of-features*)
update-rhetorical-relation(clause1, *current-relations*)

where the function *lexicalize-discourse-marker* determines the appropriate discourse marker lexicalization given a set of features such as those described in (Knott and Mellish, 1996) or (Grote and Stede, 1999), and *update-rhetorical-relation* causes the appropriate changes to be made to the running discourse plan so that future revision rules can take those alterations into account.

STORYBOOK takes a discourse plan augmented with appropriate low-level (*i.e.*, unlexicalized, or conceptual) rhetorical features and produces a sentence plan without discarding rhetorical information. It then revises and lexicalizes discourse markers concurrently before passing the results to the surface realization module for production of the surface text.

Consider the following sentences in a short text plan produced by the generation system:

1. "In this case, Mr. Curtis could no longer be tried for the shooting of his former girlfriend's companion." <agent-action>

   [causal relation]

2. "There is a five-year statute of limitations on that crime." <existential>

   [opposition relation]

3. "There is no statute of limitations in murder cases." <existential>

Without revision, a discourse marker insertion algorithm is only capable of adding discourse markers before or after a clause boundary:

"In this case, Mr. Curtis could no longer be tried for the shooting of his former girlfriend's companion. **This is because** there is a five-year statute of limitations on that crime. **However**, there is no statute of limitations in murder cases."

But a revised version with access to the discourse plan and integrating discourse markers that our system generates is:

"In this case, Mr. Curtis could no longer be tried for the shooting of his former girlfriend's companion, **because** there is a five-year statute of limitations on that crime **even though** there is no statue of limitations in murder cases."

A revision module without access to the discourse plan and a method for lexicalizing discourse markers will be unable to generate the second, improved version. Furthermore, a discourse marker insertion algorithm that lexicalizes before the revision algorithm begins will not have enough basis to decide and frequently produce wrong lexicalizations. The actual implemented rules in our system (which generate the example above) are consistent with the abstract rule presented earlier.

Revising sentence 1 with 2:

If rhetorical-type(clause1) = agent-action
  rhetorical-type(clause2) = existential
  rhetorical-relation(clause1, clause2)
    $\asymp$ {causation, simple, ...}
then make-subordinate-bound-clause(clause2, clause1)
  lexicalize-discourse-marker(clause2, {causation, simple})
  update-rhetorical-relation(clause1, clause2, *agent-action*, *existential*, *causation*)

Revising sentence 2 with 3:

If rhetorical-type(clause2) = existential
  rhetorical-type(clause3) = existential
  rhetorical-relation(clause2, clause3)
    $\asymp$ {opposition, simple, ...}
then make-subject-relative-clause(clause2, clause3)
  lexicalize-discourse-marker(clause1, {opposition, simple})
  update-rhetorical-relation(clause1, clause2, *existential*, *existential*, *current-relations*)

Given these parameters, the discourse markers will be lexicalized as *because* and *even though* respectively, and the revision component will be able to combine all three base sentences plus the discourse markers into the single sentence shown above.

6 Preliminary Evaluation

Evaluation of multi-paragraph text generation is exceedingly difficult, as empirically-driven methods are not sufficiently sophisticated, and subjective human evaluations that require multiple comparisons of large quantities of text is both difficult to control for and time-consuming. Evaluating our approach is even more difficult in that the interference between discourse markers and revision is not a highly fre-

	# Sentences	# Revisions	# DMs	# Co-occurring	DM/Rev Separate	Integrated
Article 1	112	90	29	14	17 (56.8%)	26 (89.7%)
Article 2	54	93	50	30	24 (48.0%)	45 (90.0%)
Article 3	72	117	46	26	21 (45.7%)	42 (91.3%)

Table 1: Interactions between revision and discourse markers

quent occurrence in multi-page text. For instance, in our corpora we found that these interference effects occurred 23% of the time for revised clauses and 56% of the time with discourse markers. In other words, almost one of every four clause revisions potentially forces a change in discourse marker lexicalizations and one in every two discourse markers occur near a clause revision boundary.

However, the "penalty" associated with incorrectly selecting discourse markers is fairly high leading to confusing sentences, although there is no cognitive science evidence that states exactly how high for a typical reader, despite recent work in this direction (Tree and Schrock, 1999). Furthermore, there is little agreement on exactly what constitutes a discourse marker, especially between the spoken and written dialogue communities (*e.g.*, many members of the latter consider "uh" to be a discourse marker).

We thus present an analysis of the frequencies of various features from three separate New York Times articles generated by the STORYBOOK system. We then describe the results of running our combined revision and discourse marker module with the discourse plans used to generate them. While three NYT articles is not a substantial enough evaluation in ideal terms, the cost of evaluation in such a knowledge-intensive undertaking will continue to be prohibitive until large-scale automatic or semiautomatic techniques are developed.

The left side of table 1 presents an analysis of the frequencies of revisions and discourse markers as found in each of the three NYT articles. In addition, we have indicated the number of times in our opinion that revisions and discourse markers co-occurred (*i.e.*, a discourse marker was present at the junction site of the clauses being aggregated).

The right side of the table indicates the difference between the accuracy of two different versions of the system: *separate* signifies the initial configuration of the STORYBOOK system where discourse marker insertion and revision were performed as separate process, while *integrated* signifies that discourse markers were lexicalized during revision as described in this paper. The difference between these two numbers thus represents the number of times per article that the integrated clause aggregation and discourse marker module was able to improve the resulting text.

7 Conclusion

Efficiency and software engineering considerations dictate that current large-scale NLG systems must be constructed in a pipeline fashion that minimizes backtracking and communication between modules. Yet discourse markers and revision both operate at the clause level, which leads to the potential of interference effects if they are not resolved at the same location in a pipelined architecture. We have analyzed recent theoretical and applied work in both discourse markers and revision, showing that although no previous NLG system has yet integrated both components into a single architecture, an architecture for multi-paragraph generation which separated the two into distinct, unlinked modules would not be able to guarantee that the final text contained appropriately lexicalized discourse markers. Instead, our combined revision and discourse marker module in an implemented pipelined NLG system is able to correctly insert appropriate discourse markers despite changes made by the revision system. A corpus analysis indicated that significant interference effects between revision and discourse marker lexicalization are possible. Future work may show that similar interference effects are possible as successive modules are added to pipelined NLG systems.

References

Charles B. Callaway and James C. Lester. 1997. Dynamically improving explanations: A revision-based

approach to explanation generation. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence*, pages 952–58, Nagoya, Japan.

Charles B. Callaway and James C. Lester. 2001. Narrative prose generation. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, pages 1241–1248, Seattle, WA.

Charles B. Callaway and James C. Lester. 2002. Narrative prose generation. *Artificial Intelligence*, 139(2):213–252.

Ben E. Cline. 1994. *Knowledge Intensive Natural Language Generation with Revision*. Ph.D. thesis, Virginia Polytechnic and State University, Blacksburg, Virginia.

Hercules Dalianis and Eduard Hovy. 1993. Aggregation in natural language generation. In *Proceedings of the Fourth European Workshop on Natural Language Generation*, Pisa, Italy.

Michael Elhadad and Kathy McKeown. 1990. Generating connectives. In *COLING '90: Proceedings of the Thirteenth International Conference on Computational Linguistics*, pages 97–101, Helsinki, Finland.

Michael Elhadad, Kathleen McKeown, and Jacques Robin. 1997. Floating constraints in lexical choice. *Computational Linguistics*, 23(2):195–240.

Brigitte Grote. 1998. Representing temporal discourse markers for generation purposes. In *Proceedings of the Discourse Relations and Discourse Markers Workshop*, pages 22–28, Montréal, Canada.

Brigitte Grote and Manfred Stede. 1999. Ontology and lexical semantics for generating temporal discourse markers. In *Proceedings of the 7th European Workshop on Natural Language Generation*, Toulouse, France, May.

Terrence Harvey and Sandra Carberry. 1998. Integrating text plans for conciseness and coherence. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics*, pages 512–518, August.

Helmut Horacek. 2002. Aggregation with strong regularities and alternatives. In *Second International Natural Language Generation Conference*, pages 105–112, Harriman, NY, July.

M. Kantrowitz and J. Bates. 1992. Integrated natural language generation systems. In R. Dale, E. Hovy, D. Rosner, and O. Stock, editors, *Aspects of Automated Natural Language Generation*, pages 247–262. Springer-Verlag, Berlin.

Alistair Knott and Chris Mellish. 1996. A data-driven method for classifying connective phrases. *Journal of Language and Speech*, 39.

David J. Mooney. 1994. *Generating High-Level Structure for Extended Explanations*. Ph.D. thesis, The University of Delaware, Newark, Delaware.

Richard Power, Christine Doran, and Donia Scott. 1999. Generating embedded discourse markers from rhetorical structure. In *Proceedings of the Seventh European Workshop on Natural Language Generation*, Toulouse, France.

R. Quirk, S. Greenbaum, G. Leech, and J. Svartvik. 1985. *A Comprehensive Grammar of the English Language*. Longman Publishers.

Mike Reape and Chris Mellish. 1999. Just what is aggregation anyway? In *Proceedings of the 7th European Workshop on Natural Language Generation*, Toulouse, France, May.

Ehud Reiter. 1994. Has a consensus NL generation architecture appeared, and is it psycholinguistically plausible? In *Proceedings of the Seventh International Workshop on Natural Language Generation*, pages 163–170, Kennebunkport, ME.

Jacques Robin. 1994. *Revision-Based Generation of Natural Language Summaries Providing Historical Background*. Ph.D. thesis, Columbia University, December.

James Shaw. 1998. Clause aggregation using linguistic knowledge. In *Proceedings of the 9th International Workshop on Natural Language Generation*, pages 138–147, Niagara-on-the-Lake, Canada.

Manfred Stede and Carla Umbach. 1998. DiM-Lex: A lexicon of discourse markers for text generation and understanding. In *Proceedings of the Joint 36th Meeting of the ACL and the 17th Meeting of COLING*, pages 1238–1242, Montréal, Canada, August.

J. E. Fox Tree and J. C. Schrock. 1999. Discourse markers in spontaneous speech. *Journal of Memory and Language*, 27:35–53.

Bonnie Webber and Aravind Joshi. 1998. Anchoring a lexicalized tree-adjoining grammar for discourse. In *Proceedings of the COLING-ACL '96 Discourse Relations and Discourse Markers Workshop*, pages 86–92, Montréal, Canada, August.

Feng-Jen Yang, Jung Hee Kim, Michael Glass, and Martha Evens. 2000. Lexical usage in the tutoring schemata of Circsim-Tutor: Analysis of variable references and discourse markers. In *The Fifth Annual Conference on Human Interaction and Complex Systems*, pages 27–31, Urbana, IL.