Text Revision: A Model and Its Implementation

Kentaro Inui, Takenobu Tokunaga and Hozumi Tanaka
Department of Computer Science
Tokyo Institute of Technology
2-12-1 Ōokayama Meguro Tokyo 152 Japan
{inui, take, tanaka}@cs.titech.ac.jp

Abstract
To generate good text, many kinds of decisions should be made. Many researchers have spent much time searching for the architecture that would determine a proper order for these decisions. However, even if such an architecture is found, there are still certain kinds of problems that are difficult to consider during the generation process. Those problems can be more easily detected and solved by introducing a revision process after generation. In this paper, we argue the importance of text revision with respect to natural language generation, and propose a computational model of text revision. We also discuss its implementation issues and describe an experimental Japanese text generation system, WEIVER.

1 Introduction
During the course of text generation, many kinds of decisions should be made. These decisions are generally classified into two categories: decisions on what-to-say, that is, topic selection and topic organization, and decisions on how-to-say, that is, decisions on grammatical choices and lexical choices. Many of the text generation systems proposed thus far make these decisions in a fixed sequential order. The order usually begins with decisions on what-to-say and ends with decisions on how-to-say.

These decisions, however, are interdependent, and a more versatile architecture is required to handle these interactions [4, 8, 16]. For example, the number of propositions contained in a sentence is constrained by 2 sets of decisions, the rhetorical relations among the propositions (what-to-say) and the complexity of the sentence (how-to-say). These decisions are interdependent. Furthermore, within each of the sets, there are other interactions among decisions. To account for this, some researchers have developed devices that allow interactions among decision modules [1, 8, 16, 17]. For example, Hovy proposed an architecture that can dynamically decide the order of decisions during the generation process [8].

One limitation to these approaches is that the system needs to foresee how a generation decision constrains subsequent decisions. There are, however, certain kinds of problems that are difficult to detect before the text is actually generated. Structural ambiguities, for example, are difficult to detect before lexical choice, word order, and punctuation decisions are all made. We call these kinds of problems surface problems.
Whatever decision order we adopt, there is the possibility that surface problems will still remain (see Sect. 3).

This leads us to the idea of revising text once generated. We introduce a text revision process that solves surface problems. In this paper, we argue the importance of text revision with respect to natural language generation, and propose a computational model of text revision. We also describe an experimental Japanese text generation system, WIEVER, which incorporates a revision module.

Since our target language is Japanese, we provide a brief introduction to Japanese in the next section. In Sect. 3 we give examples illustrating why the revision process is necessary. In Sect. 4 we provide a summary of our approach to solve the problems presented in Sect. 3. The implementation issues are discussed in Sect. 5.

2 Brief Introduction to Japanese

A simple Japanese sentence consists of a sequence of postpositional phrases (PPs) followed by a predicate (a verb or an adjective). A PP consists of a noun phrase (NP) followed by a postposition, which indicates the case role of the NP. We say "each PP modifies the predicate" and call PPs the modifiers and the predicate the modifiee. For example, both "John-ga"¹ and "Tokyo-ni" modify "sundeiru" in sentence (1).

1 We denote PP in “NP-postposition” form for convenience of explanation.

(1) John-ga Tokyo-ni sundeiru.
   John-NOM in Tokyo lives.

The order of PPs is not strictly fixed, so it is possible to scramble PPs without changing the meaning of the sentence². For example, the sentence "Tokyo-ni John-gaundeiru." has the same meaning as that of sentence (1). As with prepositional phrase attachments in English, one of the important constraints in Japanese is that no two modification relations cross each other³.

When a sentence has only one modifiee, the modification relation can be uniquely determined. However, this is not always the case. Sentence (2) is an example in which the noun "Mary" is modified by the verb "sat~eiku." In this example, "Poochie-to" can modify either "satteiku" or "miteita."

2 Of course, scrambling may change nuances and the naturalness of the sentence.

3 There are some exceptions but this is generally considered a reasonable constraint.

(2) John-ga Poochie-to satteiku Mary-wo miteita.
   John-NOM with Poochie departing Mary-ACC look at-PAST

Depending on which verb "Poochie-to" modifies, this sentence gives two interpretations⁴:

- John looked at [ Mary departing with Poochie ].
  (the case "Poochie-to" modifies "satteiku")

4 It is interesting that the translation in English is also ambiguous. We make the difference clear by using brackets and inversion.