Polibox: Generating descriptions, comparisons, and recommendations from a database

Manfred Stede
Inst. für Linguistik / Angewandte Computerlinguistik
Universität Potsdam
P.O. Box 601553
14415 Potsdam
Germany
stede@ling.uni-potsdam.de

Abstract

We describe our ongoing work on "Polibox", a web-based text generator producing adaptive hypertext from a product database, currently one of computational linguistics textbooks. When given a specification of a desired ideal book, Polibox selects suitable candidates from the database, and presents them one-by-one to the user. Books are described, compared to one another, and, under the right circumstances, actively recommended. This project note concentrates on the stages of content selection, text planning and sentence planning.

1 Introduction

Work in automated natural language generation (NLG) has typically focused on specific types of text, such as advices (e.g., on correcting program errors), reports (e.g., of project activities), or persuasions (e.g., to stop smoking). The "Polibox" generator, whose first prototype is just being completed, combines in a single system three different discourse purposes: *describe* an object, *compare* an object to another, *evaluate* one object with respect to another. A mixture of these purposes can be found in discourses that inform about certain products and — possibly — involve recommendations on choosing one rather than the other. In a typical scenario, a client asks an expert about a particular type of product suitable for the client's needs, and the expert responds with descriptions of several candidates, compares them, and possibly singles out one candidate as especially well-suited.

While in general this amounts to a dialogue application, Polibox merely handles the text generation side. The client thus provides a description of their needs via a menu, and Polibox selects suitable products and generates descriptions. However, rather than combining them all into a single text, individual descriptions are kept distinct, and the user can select the order in which they are presented. This takes up the idea of "adaptive hypertext" realised in ILEX (Mellish et al. 1998) or Peba-II (Milosavljevic 1997).

The following section describes the overall architecture of Polibox, and subsequent sections explain in more detail how content selection, text planning, and sentence planning are being done. Finally, Polibox is compared to related earlier research.

2 System overview

2.1 Application scenario

Polibox implements the scenario of a client (e.g., a student) inquiring about a textbook suitable for his current interests and capabilities. The subject matter realized in the prototype is computational linguistics, i.e., a database with information on 25 CL texts is the basis for the generator to select information from and present it. The student specifies topics they want to see covered by the book, the language, the programming language used, the desired presence of code examples and exercises. Polibox determines a small number of books that match the requirements and generates a description of the first, accompanied by hyperlinks to further descriptions, if any. When the student selects a link, a text on the second title is generated and presented, and this description is very likely to contain explicit comparisons to the book that has been presented before. Hence, all but the first book description pay attention not only to the "target features" specified by the user, but also to the prior descriptions, as stored in the discourse history. Furthermore, the generator may decide that one book is a clear favourite and actively recommend this one, using various

In contrast to TITLE1, TITLE2 by AUTHOR2 uses Lisp as the underlying programming language. With 180 pages it is a rather short book. Besides parsing and generation, it covers machine translation as an application of CL. Unfortunately, the book is already 12 years old.

Figure 1: English translation of a Polibox paragraph

means, as explained below.

While Polibox will eventually generate both German and English text, the first prototype is restricted to German. For illustration, figure 1 shows the (anonymized) English translation of a short sample paragraph as selected by the user. In this case, the user had asked for a recent book covering parsing and generation, and this is the second book suggested by Polibox; the first was very similar but based on Prolog.

2.2 Architecture

Polibox is a web application that features elements of "adaptive hypertext" as classified by (De Bra 1999): Via a standard web browser, the user interacts with the system, which produces HTML pages on-the-fly in response to user's choices. On the welcome page, the user selects from a range of checkboxes the features she wishes to find in a CL textbook. The web server submits the request to the generation module, which implements a variant of the "standard pipeline model" of NLG (Dale, Reiter 2000):

1. Content Selection matches the target features against the book database and determines a range of suitable books (max. four[1]), a ranking of these candidates, communicative goals, and for each book those DB attributes that are to be actually verbalized. These are being mapped to propositions.

2. Text Planning maps a communicative goal and a set of propositions (corresponding to a single book description) to a 'rhetorical tree' that specifies coherence relations between propositions and sub-trees, as well as a partial order for presenting the information.

3. Sentence Planning maps the partially-ordered rhetorical tree to a fully linearized sequence of sentence specifications linked by 'conjunctive relations' (Martin 1992).

4. Surface realization maps each sentence specification to a German sentence.

Content selection, text planning, and sentence planning are discussed in more detail below. As for surface realization, we currently employ two different approaches: A template-based generation using the TG/2 module (Busemann 1998), and a "deep" generation using the KPML multilingual generation environment (Bateman 1997). The goal is to eventually evaluate the relative merits of both approaches for the kind of application realized in Polibox; for the purposes of this paper, we describe only the "deep" variant. Hence, the sentence planner produces expressions in the Sentence Plan Language (Kasper 1989), which KPML maps to linguistic output.

2.3 Implementation

Polibox is being implemented in Common Lisp. While both alternative surface realizers run within Allegro CL runtime images, the remaining modules are built with the freely available CMU Common Lisp (CMUCL URL). Throughout the prototype, the description logic LOOM and its reasoning services (LOOM URL) are utilized where possible. Non-deterministic algorithms (in text planning) are implemented with Screamer (Siskind, McAllister 1993), which is embedded in Common Lisp. The web server is the Common Lisp Hypermedia Server (CL-HTTP URL).

3 Content selection and text planning

3.1 Retrieve candidate books

The "database" of book entries is stored in the LOOM KR system, where the terminological component (TBox) corresponds to a DB schema, and the assertional component (ABox) corresponds to DB entries. In fact, the TBox, as usual in description logics, holds knowledge about subsumption relationships, in our application a hierarchy of CL topics (so that the user can ask for 'parsing' and will also find books about 'LR-parsing', etc.). When the user has

[1] This limit is an arbitrary choice and can be changed without affecting the program.

specified the attributes of the "ideal" book, they are translated into a LOOM query. If this query succeeds, the user's requirements are matched perfectly — a non-typical case. Should the number of books retrieved be less than four (including zero), the search attributes are relaxed step-by-step in order to produce more matches, up to the point where four books have been retrieved. (If more than four result, a random selection is made.) This occurs according to a pre-defined relaxation sequence: First look for "similar" topics covered by the book, then relax the requirements on code examples and exercises, then on the programming language; the least flexible attribute is the language of the book, for if the user has explicitly specified one language, they probably will not welcome a book in another.[2] The number and weight of relaxed constraints define a ranking of the books, which leads to the establishment of communicative goals.

3.2 Determine communicative goals and attributes to be verbalized

The book database has some 15 different attributes (author, title, publisher, year, pages, topics covered, exercises, etc.), not all of which are to be included in a book's description. The list of attributes to include is not pre-stored, but arises from four different sources:

- Identificational attributes: trivially, author name and book title are always included.
- Atypical attributes: book is very long or very short; very old or very recent; covers very many or very few topics, etc. (determined by comparing to "typicality threshold", which is computed over the whole range of entries)
- Un-/desired attributes: those that match the user's target values very well, and those that are in conflict
- Comparative attributes: similarities and differences between this book and the one described previously

Those attributes that relate to the user's target feature specification receive an evaluation that reflects the degree of the mis/match combined with the importance of the feature; values are arranged on a numerical scale from +3 to -3. Next, for each book a communicative goal is computed, which abstratcs over the candidate ranking and the factors used for evaluating the suitability. Depending on the closeness of the match and the distance to the competitors, for the first book presented the goal is one of *describe-and-highly-recommend*, *describe-and-recommend*, and *describe*; for subsequent books, *compare* is added to the labels.

3.3 Map DB attributes to propositions

Before we can construct a text plan that represents the propositional contents of the text, the individual DB attributes are mapped to a level of propositions; this often involves aggregation of attributes. Examples:

- The *author* and *title* attributes can be combined into a "(WRITTEN-BY title author)" proposition.
- Binary features like *has-exercises* and *has-code-examples* can be combined into a proposition "(CONTAINS book (and exercises code))".
- A feature that is relevant to the user can be re-formulated by a computation. In our sample text in fig. 1, this applied to (YEAR 1990), which was in conflict with the user's wish for a recent book, and hence was mapped to the proposition "(OLD book 12yrs)".
- If the book is evaluated very positively, a distinct "(RECOMMEND book)" proposition can be added.

Notice that propositions are still "pre-verbal"; in the sentence planning stage, lexicalization will choose from a range of lexemes that can express the given content.

3.4 Build a rhetorical tree

As in many current generation systems, a text plan amounts to a tree of propositions and "rhetorical relations", as inspired by (Mann, Thompson 1987). Unlike many current systems, however, we distinguish between the rhetorical relations in the text plan and the more surface-oriented "conjunctive relations" (Martin 1992)

[2] In principle, the relaxation sequence could also be specified by the user, but for now it is hard-wired.

established in sentence planning (see the next section).

Building the text plan proceeds in two steps: First, a set of rules tries, bottom-up, to establish rhetorical relations between individual propositions and then recursively between the resulted trees of depth one; then, another set of rules link the set of small trees together into a complete tree (top-down). In both stages, we first try to establish "informative" relations, and only if this fails, the relatively "weak" relations ELABORATION or, worse, JOINT are established. Altogether, we currently use these rhetorical relations: CAUSE, CONCESSION, CONTRAST, ELABORATION, JOINT, LIST, RESULT. Examples of the bottom-up stage:

- If there is exactly one proposition with positive evaluation and exactly one with a negative evaluation, they are combined into a CONCESSION tree. E.g.: *Although the book is rather old, it covers all the topics you are interested in.* If there are more positive or negative propositions, these are combined first, in order to avoid multiple concessions within the same paragraph.
- When a "(RECOMMEND book)" proposition is present, it is linked to positively-evaluated propositions via a RESULT relation. E.g.: *The book covers all the topics you are interested in. Also, it is quite recent. Therefore, it would be a good choice for you.*

The operation of the top-down rules depends on the communicative goal. For describing and comparing, little more than arranging similar propositions into LISTs or ELABORATIONs is done. For recommending a book, however, the rules are responsible for achieving the overall rhetorical effect. How this is done depends on the configuration produced by the bottom-up rules; in general, the top-down rules have to make sure that, e.g., a "(RECOMMEND book)" proposition appears either at the beginning or the end of the paragraph, and that the paragraph does not end with a negative proposition.

4 Sentence planning

An architectural innovation of Polibox is its separating the tree of deep rhetorical relations from a structure involving surface-oriented 'conjunctive relations'. The latter is constructed in sentence planning and represents a level of representation at which sentence scope and structure have been fixed, but lexicalisation is still ahead.

Conjunctive relations (henceforth CR) are an approach to characterizing text connectedness developed by (Martin 1992). Martin offers a fine-grained classification of relations and the range of their realizations in English, characterized by sets of semantic features, which Martin organizes as system networks.

The groups of relations distinguished by Martin are the following: additive (addition, alternation), comparative (similarity, contrast), temporal (simultaneous, successive), consequential (purpose, condition, consequence, concession, manner). Each group is analyzed in great detail and systematically linked to realization through connectives and clause structure. Notice that there are some similarities to the relations proposed by RST (Mann, Thompson 1987); however, Martin's approach is decidedly oriented to linguistic realization, whereas RST relations were considered "pre-realizational". Polibox indeed implements this distinction by employing RST in text planning and CR in sentence planning. The ordering step proceeds as follows: Propositions are treated as nodes in an (initially disconnected) graph, and the ordering information already present in the rhetorical tree provides the first directed edges. By computing similarity/contrastiveness among propositions, the remaining edges are inserted. Next, the proposition stream is "chunked" into clauses and sentences in accordance with the relations present in the rhetorical tree; the graph edges are labelled with appropriate CRs.

Introducing the CR level leads us to split the sentence planning stage in two parts: First, propositions are packaged into clause- and sentence-chunks, decisions between paratactic and hypotactic connection are made, and the class of connectives to be used is fixed. All these decisions are influenced by the communicative goal and by the discourse memory, which records the linearization in order to support subsequent ordering decisions as well as choice of clause themes.

The remaining tasks in moving from CR to sentence plans (Kasper 1989) are choosing

open-class words and connectives, and mapping proposition participants to thematic roles. Here, interactions between choosing connectives and determining the verb complex can be accounted for. Once again, the communicative goal and information from the discourse memory enter these decisions as parameters, e.g. for choosing a connective that marks a proposition as *given* (cf. the difference between *since* and *because*). For the lexicalisation step, we use an extended version of the 'Moose' module (Stede 1999).

5 Related work

The merge between NLG and adaptive hypertext was introduced by the ILEX (Mellish et al. 1998) and Peba-II (Milosavljevic 1997) systems, and with respect to user interface, Polibox adopts the same general approach as these two systems. However, we have put an emphasis on the variety of communicative goals that can be pursued in Polibox, resulting in a broader range of text types.

Rhetorical trees have been employed by many generation systems (including ILEX) as the output of text planning, but in the earlier systems, a single layer of coherence relations was used. A proposal for distinguishing "deep" rhetorical relations from conjunctive relations operating close to the surface was made by (Bateman 1999) for the task of handling spoken language, but to our knowledge this approach has so far not been implemented. Polibox, in contrast to earlier systems, treats the text plan tree as only partially ordered and leaves remaining ordering decisions to sentence planning. Controlling information structure is one advantage of using two distinct layers; eventually, the method will be applied to bilingual generation, where the different linguistic means for attaching clauses and sentences can be systematically related by a (language-specific) layer of conjunctive relations.

References

J. Bateman. "Enabling technology for multilingual natural language generation: the KPML development environment." In: Journal of Natural Language Engineering 3(1), 1997.

J. Bateman. "The dynamics of surfacing." Proc. of the workshop on levels of representation in discourse, Edinburgh Univ., 1999

S. Busemann. "Best-first surface realization." Proc. of the Eighth Int'l Workshop on Natural Language Generation, Herstmonceux Castle, 1998.

CMU Common Lisp. http://www.cons.org/cmucl/

Common Lisp Hypermedia Server. http://www.ai.mit.edu/projects/iiip/doc/cl-http/home-page.html

R. Dale, E. Reiter. *Building natural language generation systems.* Cambridge University Press, 2000.

P. De Bra. "Design Issues in Adaptive Hypermedia Application Development." Proc. of the Second Workshop on Adaptive Systems and User Modeling on the World Wide Web, Toronto and Banff, 1999.

R. Kasper. "A flexible interface for linking applications to the PENMAN sentence generator." Proc. of the DARPA workshop on speech and natural language processing. Univ. of Pennsylvania, 1989.

Loom Knowledge Representation Language. http://www.isi.edu/projects/loom

W. Mann, S. Thompson. "Rhetorical structure theory: a theory of text organization." In: L. Polyani (ed.): *The structure of discourse.* Norwood: Ablex, 1987.

J. Martin. English Text: System and Structure. Amsterdam: John Benjamins, 1992.

C. Mellish, M. O'Donnell, J. Oberlander, A. Knott. "An architecture for opportunistic text generation." Proc. of the Ninth Int'l Workshop on Natural Language Generation, Niagara-on-the-Lake, Ontario, 1998.

M. Milosavljevic. "Content Selection in Comparison Generation." Proc. of the 6th European Workshop on Natural language Generation. Duisburg, 1997.

J. Siskind, D. McAllister. "Nondeterministic Lisp as a Substrate for Constraint Logic Programming." Proc. of AAAI-93, 1993.

M. Stede. *Lexical semantics and knowledge representation in multilingual text generation.* Dordrecht/Boston: Kluwer, 1999.