Direct Memory Access Parsing in the Creanimate Biology Tutor

Will Fitzgerald

The Institute for the Learning Sciences
Northwestern University
Evanston, Illinois, USA
will@ils.nwu.edu

Abstract

Computer tutors that engage in dialog with students require effective parsing technology to understand the intentions of the students. This is a report of an implementation of a Direct Memory Access Parser (DMAP) for a computer based biology tutor (Creanimate). The report begins by describing the Creanimate system and giving an overview of Direct Memory Access Parsing. It then describes the parser developed for Creanimate, DMAP-C, and the extensions and enhancements compared to previous DMAP parsers. Finally, we give a qualitative evaluation of the effectiveness of the DMAP-C parser.

1: Creanimate

Creanimate is a computer-based biology tutor developed to teach elementary school age children about the functional value of the physical features of animals [1]. Creanimate teaches by telling stories. In order to find interesting stories to tell, Creanimate uses a dense, hierarchical frame-based memory representation, and various reminding strategies to retrieve appropriate stories. If a student wants to create a flying dog so that it can hunt, Creanimate might be reminded of other animals that hunt by flying. Creanimate often allows students to enter free-form text. Creanimate requires a parser to make use of these open-ended student responses.

2: Direct Memory Access Parsing

The goal of a natural language understanding system—a parser—is to recognize the connection between a text and the computer's internal representations. For example, reading "squirrels" should activate the internal representation of squirrels. Reading "Squirrels climb trees in order to pursue their mates" should activate the structure that refers to this particular combination of animal, action and behavior. DMAP is such a recognition parser [3,4]. It uses the conceptual knowledge and language information attached to that knowledge to understand language.

Creanimate uses a standard memory representation in which concepts are represented as packages of interconnections. The packaging for the concept of PURSUE-PREY, for example, includes an :OBJECT link to indicate what is being pursued; specifically, PREY. This type of link is an *attribute* link. Creanimate also uses hierarchical interconnections between concepts; PURSUE-PREY is a type of PURSUE which is a type of BEHAVIOR, and so on.

DMAP allows linguistic information, stored as *phrasal patterns*, to be attached to conceptual representations. DMAP uses these patterns to recognize concepts. There are two types of items that comprise phrasal patterns: words and attribute references. For example, attached to the KANGAROO-RAT concept might be the phrasal pattern "kangaroo rat," that is, a phrasal pattern consisting of two words. When DMAP has seen "kangaroo rat," it activates KANGAROO-RAT. Phrasal patterns can also contain attribute references; for example, attaching the phrasal pattern "pursue :OBJECT" to PURSUE-PREY tells to look for the word "pursue" followed by anything that can be the :OBJECT of PURSUE-PREY. More details about how DMAP works can be found in [4] and the appendix of [2].

The DMAP parser in Creanimate (DMAP-C) makes use of over 2,350 phrasal patterns attached to some 1100 concepts. The standard DMAP algorithm required several enhancements and modifications to work effectively within the Creanimate system.

4: Enhancements to DMAP

In addition to the standard DMAP, the DMAP-C algorithm includes:

- An expectation-based automatic spelling corrector,
- A morphological analyzer,
- Selectional type constraints and a "prefer longest parses" strategy,
- A graphical user interface for editing and testing newly defined phrasal patterns.

These enhancements are described below.

Automatic spelling correction. Creanimate is designed for use by fifth and sixth graders. Their typing skills and

1043-0989/94 $03.00

spelling skills are not fully developed and they often make mistakes. DMAP-C automatically corrects many types of spelling errors, including vowel errors, letter transpositions, extra letters, missing double letters, typing errors and sound alike errors.

Morphological analysis. Although DMAP can handle morphological analysis, this is not useful for Creanimate: "Find insects" and "finding an insect" can be parsed the same, for Creanimate's purposes. The morphological analyzer is compiled from a list of morphological rules (inspired by [5]). The compiled analyzer efficiently produces the base forms by interleaving the reverse application of the suffix rules.

Selectional restrictions and the *Prefer longest parses strategy.* DMAP-C allows the system designer to specify the type or types of object to look for. For example, we could tell the parser to look for BEHAVIORs or ACTIONs, and it would ignore other types. Selectional restrictions are not enough, however. Consider Creanimate asking a student for an animal, and the student enters "kangaroo rat." DMAP-C has been asked to restrict its parses, but three animal concepts are referenced: KANGAROO, RAT and KANGAROO-RAT. Creanimate prefers the longest covering phrase.

Graphical user interface for editing phrasal patterns. The Creanimate system uses a GUI to allow adding and editing conceptual representations, and index stories attached to those representations. An add-on module for entering and testing phrasal patterns attached to concepts in the memory representation was created for the workstation.

5: Evaluating the DMAP-C Parser

How successful is DMAP-C? Does it support effective tutorial dialogs? Using responses from student transcripts, interactions between Creanimate and students which required parsing were analyzed a number of different ways. The analysis is based on responses from 49 students in the first weeks of the on-site testing.

The Creanimate system automatically creates transcripts of student interactions. From the transcripts, we extracted all instances of the use of the DMAP-C parser. We categorized these instances, some of which were discarded because of faults in the logging facility. There were 146 instances from 49 sessions. These were tagged as parsed Correctly, Parsed Noise as Noise, or Parsed Incorrectly. Parsed Correctly meant DMAP-C correctly parsed the student's input, so that the tutorial dialog successfully continued and was not interrupted. Parsed Noise as Noise meant that DMAP-C did not parse ("recognized as noise") something that was nonsensical or ill-formed. Parsed Incorrectly meant DMAP-C failed in one of two ways: it didn't recognize something it should have; or it parsed to something it should not have. The success rate is the number of Parsed Correctly + number of Parsed Noise as Noise divided by the total number of instances, considered as a percentage. The success rate for this preliminary system was 83% (N=121). Extending the ontology of Creanimate and adding other minimal changes to DMAP-C should boost this rate.

6: Summary

Direct Memory Access Parsing integrates parsing technology with a conceptual memory representation. Creanimate's dense, hierarchical memory representation provides a perfect opportunity for successful deployment of such a parser. DMAP-C successfully combined this technology with extensions such as automatic correction of spelling errors, morphological analysis and a graphical user interface for attaching phrasal patterns to memory representations. The interface affordances of Creanimate also provided constraints on the input to the parser which also added to the success. Other systems which have similar characteristics may also benefit from DMAP-style parsing.

Acknowledgments

The compiler for the morphological analyzer was written by Chris Riesbeck. Chip Cleary, Danny Edelson and Chris Riesbeck provided helpful comments on earlier drafts of this paper. This work was supported in part by the Defense Advanced Research Projects Agency, monitored by the Office of Naval Research under contracts N00014-91-J-4092 and N00014-90-J-4117. The Institute for the Learning Sciences was established in 1989 with the support of Andersen Consulting, part of The Arthur Andersen Worldwide Organization. The Institute receives additional support from Ameritech and North West Water Group plc, Institute Partners, and from IBM.

References

[1] Edelson, D. C. (1993). Learning from stories: Indexing and reminding in a Socratic case-based teaching system for elementary school biology. The Institute for the Learning Sciences, Northwestern University, Technical Report #43, July 1993.

[2] Fitzgerald, W. (1994). Using natural language processing to construct large-scale hypertext systems. To appear in Proceedings *Eighth Knowledge Acquisition for Knowledge-Based Systems Workshop*, Banff, Canada, January 30–February 4, 1994.

[3] Martin, C. (1990) *Direct Memory Access Parsing.* Ph.D. Thesis, Yale University.

[4] Riesbeck, C. K., & Schank, R. C. (1989). *Inside Case-Based Reasoning.* Hillsdale, New Jersey: Lawrence Erlbaum Associates.

[5] Winograd, T. (1983). *Language as a Cognitive Process: Syntax.* Reading, Massachusetts: Addison-Wesley.