Beetle II: A System for Tutoring and Computational Linguistics Experimentation

BEETLE II: a system for tutoring and computational linguistics experimentation

Myroslava O. Dzikovska and **Johanna D. Moore**
School of Informatics, University of Edinburgh, Edinburgh, United Kingdom
{m.dzikovska,j.moore}@ed.ac.uk

Natalie Steinhauser and **Gwendolyn Campbell**
Naval Air Warfare Center Training Systems Division, Orlando, FL, USA
{gwendolyn.campbell,natalie.steihauser}@navy.mil

Elaine Farrow
Heriot-Watt University
Edinburgh, United Kingdom
e.farrow@hw.ac.uk

Charles B. Callaway
University of Haifa
Mount Carmel, Haifa, Israel
ccallawa@gmail.com

Abstract

We present BEETLE II, a tutorial dialogue system designed to accept unrestricted language input and support experimentation with different tutorial planning and dialogue strategies. Our first system evaluation used two different tutorial policies and demonstrated that the system can be successfully used to study the impact of different approaches to tutoring. In the future, the system can also be used to experiment with a variety of natural language interpretation and generation techniques.

1 Introduction

Over the last decade there has been a lot of interest in developing tutorial dialogue systems that understand student explanations (Jordan et al., 2006; Graesser et al., 1999; Aleven et al., 2001; Buckley and Wolska, 2007; Nielsen et al., 2008; VanLehn et al., 2007), because high percentages of self-explanation and student contentful talk are known to be correlated with better learning in human-human tutoring (Chi et al., 1994; Litman et al., 2009; Purandare and Litman, 2008; Steinhauser et al., 2007). However, most existing systems use pre-authored tutor responses for addressing student errors. The advantage of this approach is that tutors can devise remediation dialogues that are highly tailored to specific misconceptions many students share, providing step-by-step scaffolding and potentially suggesting additional problems. The disadvantage is a lack of adaptivity and generality: students often get the same remediation for the same error regardless of their past performance or dialogue context, as it is infeasible to author a different remediation dialogue for every possible dialogue state. It also becomes more difficult to experiment with different tutorial policies within the system due to the inherent completixites in applying tutoring strategies consistently across a large number of individual hand-authored remediations.

The BEETLE II system architecture is designed to overcome these limitations (Callaway et al., 2007). It uses a deep parser and generator, together with a domain reasoner and a diagnoser, to produce detailed analyses of student utterances and generate feedback automatically. This allows the system to consistently apply the same tutorial policy across a range of questions. To some extent, this comes at the expense of being able to address individual student misconceptions. However, the system's modular setup and extensibility make it a suitable testbed for both computational linguistics algorithms and more general questions about theories of learning.

A distinguishing feature of the system is that it is based on an introductory electricity and electronics course developed by experienced instructional designers. The course was first created for use in a human-human tutoring study, without taking into account possible limitations of computer tutoring. The exercises were then transferred into a computer system with only minor adjustments (e.g., breaking down compound questions into individual questions). This resulted in a realistic tutoring setup, which presents interesting challenges to language processing components, involving a wide variety of language phenomena.

We demonstrate a version of the system that has undergone a successful user evaluation in

Proceedings of the ACL 2010 System Demonstrations, pages 13–18,
Uppsala, Sweden, 13 July 2010.

2009. The evaluation results indicate that additional improvements to remediation strategies, and especially to strategies dealing with interpretation problems, are necessary for effective tutoring. At the same time, the successful large-scale evaluation shows that BEETLE II can be used as a platform for future experimentation.

The rest of this paper discusses the BEETLE II system architecture (Section 2), system evaluation (Section 3), and the range of computational linguistics problems that can be investigated using BEETLE II (Section 4).

2 System Architecture

The BEETLE II system delivers basic electricity and electronics tutoring to students with no prior knowledge of the subject. A screenshot of the system is shown in Figure 1. The student interface includes an area to display reading material, a circuit simulator, and a dialogue history window. All interactions with the system are typed. Students read pre-authored curriculum slides and carry out exercises which involve experimenting with the circuit simulator and explaining the observed behavior. The system also asks some high-level questions, such as "What is voltage?".

The system architecture is shown in Figure 2. The system uses a standard interpretation pipeline, with domain-independent parsing and generation components supported by domain specific reasoners for decision making. The architecture is discussed in detail in the rest of this section.

2.1 Interpretation Components

We use the TRIPS dialogue parser (Allen et al., 2007) to parse the utterances. The parser provides a domain-independent semantic representation including high-level word senses and semantic role labels. The contextual interpreter then uses a reference resolution approach similar to Byron (2002), and an ontology mapping mechanism (Dzikovska et al., 2008a) to produce a domain-specific semantic representation of the student's output. Utterance content is represented as a set of extracted objects and relations between them. Negation is supported, together with a heuristic scoping algorithm. The interpreter also performs basic ellipsis resolution. For example, it can determine that in the answer to the question "Which bulbs will be on and which bulbs will be off in this diagram?", "off" can be taken to mean "all bulbs in the diagram will be off." The resulting output is then passed on to the domain reasoning and diagnosis components.

2.2 Domain Reasoning and Diagnosis

The system uses a knowledge base implemented in the KM representation language (Clark and Porter, 1999; Dzikovska et al., 2006) to represent the state of the world. At present, the knowledge base represents 14 object types and supports the curriculum containing over 200 questions and 40 different circuits.

Student explanations are checked on two levels, verifying *factual* and *explanation* correctness. For example, for a question "Why is bulb A lit?", if the student says "it is in a closed path", the system checks two things: a) is the bulb indeed in a closed path? and b) is being in a closed path a reasonable explanation for the bulb being lit? Different remediation strategies need to be used depending on whether the student made a factual error (i.e., they misread the diagram and the bulb is not in a closed path) or produced an incorrect explanation (i.e., the bulb is indeed in a closed path, but they failed to mention that a battery needs to be in the same closed path for the bulb to light).

The knowledge base is used to check the factual correctness of the answers first, and then a diagnoser checks the explanation correctness. The diagnoser, based on Dzikovska et al. (2008b), outputs a diagnosis which consists of lists of correct, contradictory and non-mentioned objects and relations from the student's answer. At present, the system uses a heuristic matching algorithm to classify relations into the appropriate category, though in the future we may consider a classifier similar to Nielsen et al. (2008).

2.3 Tutorial Planner

The tutorial planner implements a set of generic tutoring strategies, as well as a policy to choose an appropriate strategy at each point of the interaction. It is designed so that different policies can be defined for the system. The currently implemented strategies are: acknowledging the correct part of the answer; suggesting a slide to read with background material; prompting for missing parts of the answer; hinting (low- and high- specificity); and giving away the answer. Two or more strategies can be used together if necessary.

The hint selection mechanism generates hints automatically. For a low specificity hint it selects

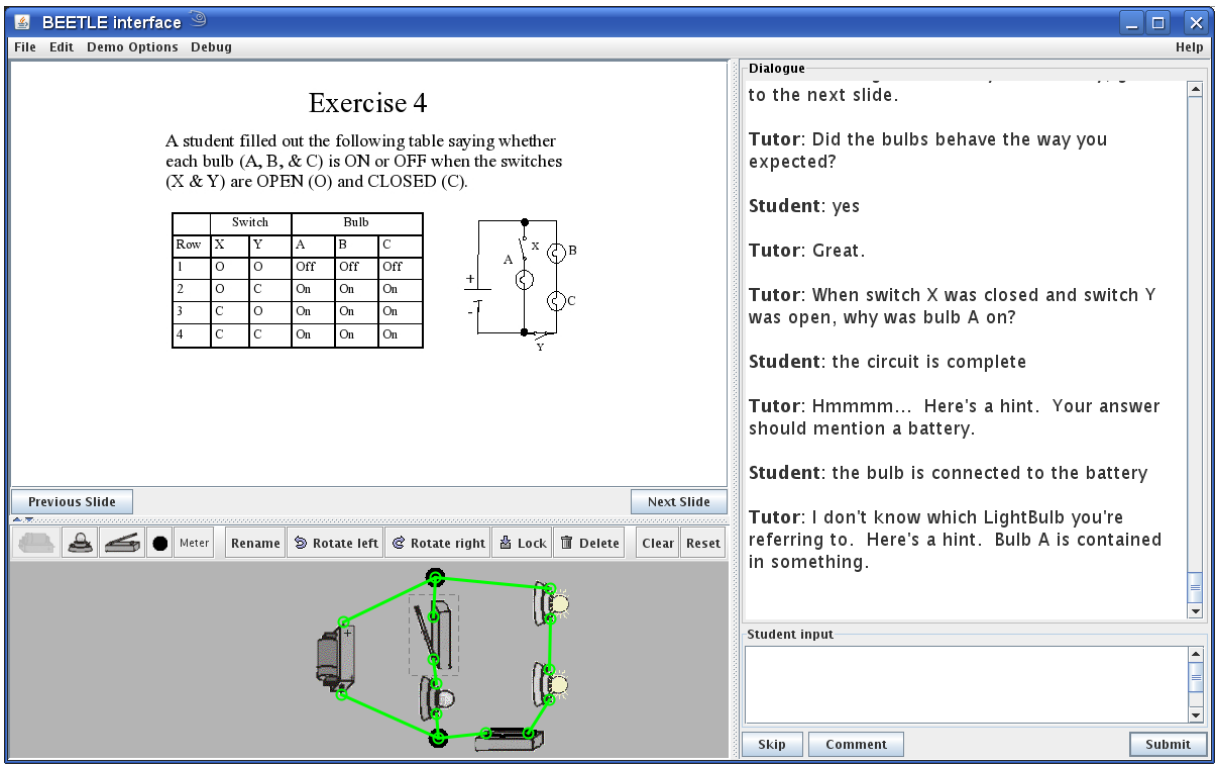

Figure 1: Screenshot of the BEETLE II system

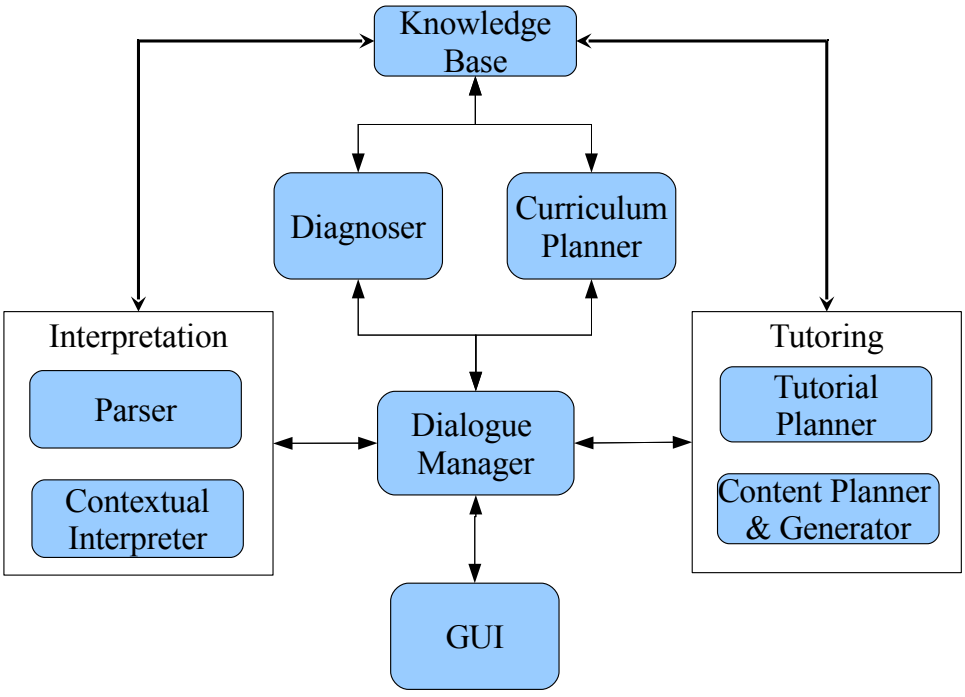

Figure 2: System architecture diagram

an as-yet unmentioned object and hints at it, for example, "Here's a hint: Your answer should mention a battery." For high-specificity, it attempts to hint at a two-place relation, for example, "Here's a hint: the battery is connected to something."

The tutorial policy makes a high-level decision as to which strategy to use (for example, "acknowledge the correct part and give a high specificity hint") based on the answer analysis and dialogue context. At present, the system takes into consideration the number of incorrect answers received in response to the current question and the number of uninterpretable answers.[1]

In addition to a remediation policy, the tutorial planner implements an error recovery policy (Dzikovska et al., 2009). Since the system accepts unrestricted input, interpretation errors are unavoidable. Our recovery policy is modeled on the TargetedHelp (Hockey et al., 2003) policy used in task-oriented dialogue. If the system cannot find an interpretation for an utterance, it attempts to produce a message that describes the problem but without giving away the answer, for example, "I'm sorry, I'm having a problem understanding. I don't know the word *power*." The help message is accompanied with a hint at the appropriate level, also depending on the number of previous incorrect and non-interpretable answers.

2.4 Generation

The strategy decision made by the tutorial planner, together with relevant semantic content from the student's answer (e.g., part of the answer to confirm), is passed to content planning and generation. The system uses a domain-specific content planner to produce input to the surface realizer based on the strategy decision, and a FUF/SURGE (Elhadad and Robin, 1992) generation system to produce the appropriate text. Templates are used to generate some stock phrases such as "When you are ready, go on to the next slide."

2.5 Dialogue Management

Interaction between components is coordinated by the dialogue manager which uses the information-state approach (Larsson and Traum, 2000). The dialogue state is represented by a cumulative answer analysis which tracks, over multiple turns, the correct, incorrect, and not-yet-mentioned parts of the answer. Once the complete answer has been accumulated, the system accepts it and moves on. Tutor hints can contribute parts of the answer to the cumulative state as well, allowing the system to jointly construct the solution with the student.

[1] Other factors such as student confidence could be considered as well (Callaway et al., 2007).

3 Evaluation

The first experimental evaluation involving 81 participants (undergraduates recruited from a Southeastern University in the USA) was completed in 2009. Participants had little or no prior knowledge of the domain. Each participant took a pre-test, worked through a lesson with the system, took a post-test, and completed a user satisfaction survey. Each session lasted approximately 4 hours.

We implemented two different tutoring policies in the system for this evaluation. The baseline policy used an "accept and bottom out" strategy for all student answers, regardless of their content. The students were always given the correct answer, but the system made no attempt at remediation, and never indicated whether the student was understood (or correct). In comparison, the full adaptive policy attempted to select an appropriate strategy based on student answer analysis and dialogue context, as described above. Students were randomly assigned to a condition. On average, each student session contained 232 student language turns in the fully adaptive condition ($SD = 25.6$) and 156 turns in the baseline condition ($SD = 2.02$). Additional time was taken by reading and interacting with the simulation environment.

Out of 81 students, 76 successfully completed the evaluation (data from 5 participants had to be discarded due to system crashes). There was no significant difference in learning gain between the two conditions (baseline: $M = 0.63, SD = 0.26$, full: $M = 0.57, SD = 0.23$, t-test $p = 0.32$), and students gave higher satisfaction scores to the baseline condition (baseline: $M = 3.32, SD = 0.65$, full: $M = 2.56, SD = 0.65$, t-test, $p < 0.01$). We are currently analyzing the data to investigate the reasons for this outcome. However, the evaluation demonstrated that the system can be successfully deployed in experiments with naive users and therefore can be used to study the impact of different natural language processing algorithms on learning gain and user satisfaction.

An example dialogue with the system, taken from the evaluation corpus, is shown in Figure 3.

Tutor:	Why was bulb A on when switch Y was open and switch Z was closed?
Student:	because it had a closed path
Tutor:	Right. There is a closed path. Keep going.
Student:	there was a closed path not effected by the switch
Tutor:	I'm sorry, I'm having a problem understanding. I don't understand the word 'effect' here. Here's a hint. Your answer should mention a battery.
Student:	it has a closed path with the battery
Tutor:	You're on the right track. Here's the answer. Bulb A is still contained in a closed path with the battery.

Figure 3: Example interaction with the system from our corpus

It shows three key system properties: after the student's first turn, the system rephrases its understanding of the correct part of the student answer and prompts the student to supply the missing information. In the second turn, the student utterance could not be interpreted and the system responds with a targeted help message and a hint about the object that needs to be mentioned. Finally, in the last turn the system combines the information from the tutor's hint and the student's answers and restates the complete answer since the current answer was completed over multiple turns.

4 Conclusions and Future Work

The BEETLE II system we present was built to serve as a platform for research in computational linguistics and tutoring, and can be used for task-based evaluation of algorithms developed for other domains. We are currently developing an annotation scheme for the data we collected to identify student paraphrases of correct answers. The annotated data will be used to evaluate the accuracy of existing paraphrasing and textual entailment approaches and to investigate how to combine such algorithms with the current deep linguistic analysis to improve system robustness. We also plan to annotate the data we collected for evidence of misunderstandings, i.e., situations where the system arrived at an incorrect interpretation of a student utterance and took action on it. Such annotation can provide useful input for statistical learning algorithms to detect and recover from misunderstandings.

In dialogue management and generation, the key issue we are planning to investigate is that of linguistic alignment. The analysis of the data we have collected indicates that student satisfaction may be affected if the system rephrases student answers using different words (for example, using better terminology) but doesn't explicitly explain the reason why different terminology is needed (Dzikovska et al., 2010). Results from other systems show that measures of semantic coherence between a student and a system were positively associated with higher learning gain (Ward and Litman, 2006). Using a deep generator to automatically generate system feedback gives us a level of control over the output and will allow us to devise experiments to study those issues in more detail.

From the point of view of tutoring research, we are planning to use the system to answer questions about the effectiveness of different approaches to tutoring, and the differences between human-human and human-computer tutoring. Previous comparisons of human-human and human-computer dialogue were limited to systems that asked short-answer questions (Litman et al., 2006; Rosé and Torrey, 2005). Having a system that allows more unrestricted language input will provide a more balanced comparison. We are also planning experiments that will allow us to evaluate the effectiveness of individual strategies implemented in the system by comparing system versions using different tutoring policies.

Acknowledgments

This work has been supported in part by US Office of Naval Research grants N000140810043 and N0001410WX20278. We thank Katherine Harrison and Leanne Taylor for their help running the evaluation.

References

V. Aleven, O. Popescu, and K. R. Koedinger. 2001. Towards tutorial dialog to support self-explanation: Adding natural language understanding to a cognitive tutor. In *Proceedings of the 10th International Conference on Artificial Intelligence in Education (AIED '01)"*.

James Allen, Myroslava Dzikovska, Mehdi Manshadi, and Mary Swift. 2007. Deep linguistic processing for spoken dialogue systems. In *Proceedings of the ACL-07 Workshop on Deep Linguistic Processing*.

Mark Buckley and Magdalena Wolska. 2007. Towards modelling and using common ground in tutorial dialogue. In *Proceedings of DECALOG, the 2007 Workshop on the Semantics and Pragmatics of Dialogue*, pages 41–48.

Donna K. Byron. 2002. *Resolving Pronominal Reference to Abstract Entities*. Ph.D. thesis, University of Rochester.

Charles B. Callaway, Myroslava Dzikovska, Elaine Farrow, Manuel Marques-Pita, Colin Matheson, and Johanna D. Moore. 2007. The Beetle and BeeDiff tutoring systems. In *Proceedings of SLaTE'07 (Speech and Language Technology in Education)*.

Michelene T. H. Chi, Nicholas de Leeuw, Mei-Hung Chiu, and Christian LaVancher. 1994. Eliciting self-explanations improves understanding. *Cognitive Science*, 18(3):439–477.

Peter Clark and Bruce Porter, 1999. *KM (1.4): Users Manual*. http://www.cs.utexas.edu/users/mfkb/km.

Myroslava O. Dzikovska, Charles B. Callaway, and Elaine Farrow. 2006. Interpretation and generation in a knowledge-based tutorial system. In *Proceedings of EACL-06 workshop on knowledge and reasoning for language processing*, Trento, Italy, April.

Myroslava O. Dzikovska, James F. Allen, and Mary D. Swift. 2008a. Linking semantic and knowledge representations in a multi-domain dialogue system. *Journal of Logic and Computation*, 18(3):405–430.

Myroslava O. Dzikovska, Gwendolyn E. Campbell, Charles B. Callaway, Natalie B. Steinhauser, Elaine Farrow, Johanna D. Moore, Leslie A. Butler, and Colin Matheson. 2008b. Diagnosing natural language answers to support adaptive tutoring. In *Proceedings 21st International FLAIRS Conference*, Coconut Grove, Florida, May.

Myroslava O. Dzikovska, Charles B. Callaway, Elaine Farrow, Johanna D. Moore, Natalie B. Steinhauser, and Gwendolyn C. Campbell. 2009. Dealing with interpretation errors in tutorial dialogue. In *Proceedings of SIGDIAL-09*, London, UK, Sep.

Myroslava O. Dzikovska, Johanna D. Moore, Natalie Steinhauser, and Gwendolyn Campbell. 2010. The impact of interpretation problems on tutorial dialogue. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics(ACL-2010)*.

Michael Elhadad and Jacques Robin. 1992. Controlling content realization with functional unification grammars. In R. Dale, E. Hovy, D. Rösner, and O. Stock, editors, *Proceedings of the Sixth International Workshop on Natural Language Generation*, pages 89–104, Berlin, April. Springer-Verlag.

A. C. Graesser, P. Wiemer-Hastings, P. Wiemer-Hastings, and R. Kreuz. 1999. Autotutor: A simulation of a human tutor. *Cognitive Systems Research*, 1:35–51.

Beth Ann Hockey, Oliver Lemon, Ellen Campana, Laura Hiatt, Gregory Aist, James Hieronymus, Alexander Gruenstein, and John Dowding. 2003. Targeted help for spoken dialogue systems: intelligent feedback improves naive users' performance. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics*, pages 147–154, Morristown, NJ, USA.

Pamela Jordan, Maxim Makatchev, Umarani Pappuswamy, Kurt VanLehn, and Patricia Albacete. 2006. A natural language tutorial dialogue system for physics. In *Proceedings of the 19th International FLAIRS conference*.

Staffan Larsson and David Traum. 2000. Information state and dialogue management in the TRINDI Dialogue Move Engine Toolkit. *Natural Language Engineering*, 6(3-4):323–340.

Diane Litman, Carolyn P. Rosé, Kate Forbes-Riley, Kurt VanLehn, Dumisizwe Bhembe, and Scott Silliman. 2006. Spoken versus typed human and computer dialogue tutoring. *International Journal of Artificial Intelligence in Education*, 16:145–170.

Diane Litman, Johanna Moore, Myroslava Dzikovska, and Elaine Farrow. 2009. Generalizing tutorial dialogue results. In *Proceedings of 14th International Conference on Artificial Intelligence in Education (AIED)*, Brighton, UK, July.

Rodney D. Nielsen, Wayne Ward, and James H. Martin. 2008. Learning to assess low-level conceptual understanding. In *Proceedings 21st International FLAIRS Conference*, Coconut Grove, Florida, May.

Amruta Purandare and Diane Litman. 2008. Content-learning correlations in spoken tutoring dialogs at word, turn and discourse levels. In *Proceedings 21st International FLAIRS Conference*, Coconut Grove, Florida, May.

C.P. Rosé and C. Torrey. 2005. Interactivity versus expectation: Eliciting learning oriented behavior with tutorial dialogue systems. In *Proceedings of Interact'05*.

N. B. Steinhauser, L. A. Butler, and G. E. Campbell. 2007. Simulated tutors in immersive learning environments: Empirically-derived design principles. In *Proceedings of the 2007 Interservice/Industry Training, Simulation and Education Conference*, Orlando, FL.

Kurt VanLehn, Pamela Jordan, and Diane Litman. 2007. Developing pedagogically effective tutorial dialogue tactics: Experiments and a testbed. In *Proceedings of SLaTE Workshop on Speech and Language Technology in Education*, Farmington, PA, October.

Arthur Ward and Diane Litman. 2006. Cohesion and learning in a tutorial spoken dialog system. In *Proceedings of 19th International FLAIRS (Florida Artificial Intelligence Research Society) Conference*, Melbourne Beach, FL.