

Learning by Teaching SimStudent: Technical Accomplishments and an Initial Use with Students

Noboru Matsuda¹, Victoria Keiser¹, Rohan Raizada¹, Arthur Tu¹,
Gabriel Stylianides², William W. Cohen¹, Kenneth R. Koedinger¹

¹ School of Computer Science, Carnegie Mellon University.
5000 Forbes Ave. Pittsburgh PA 15213 USA

² School of Education, University of Pittsburgh.
5517 Posvar Hall, Pittsburgh PA 15260 USA

[noboru.matsuda, keiser, atu, wcohen, koedinger]@cs.cmu.edu
rohan@gmail.com, gstylian@pitt.edu

Abstract. The purpose of the current study was to test whether we could create a system where students can learn by teaching a live machine-learning agent. SimStudent is a computer agent that interactively learns cognitive skills through its own tutored-problem solving experience. We have developed a game-like learning environment where students learn algebra equations by tutoring SimStudent. While Simulated Students, Teachable Agents and Learning Companion systems have been created, our study is unique that it genuinely learns skills from student input. This paper describes the overview of the learning environment and some results from an evaluation study. The study showed that after tutoring SimStudent, the students improved their performance on equation solving. The number of correct answers on the error detection items was also significantly improved. On average students spent 70.0 minutes on tutoring SimStudent and used an average of 15 problems for tutoring.

Keywords: SimStudent, Learning by teaching, tutor-learning effect, algebra equation solving, machine learning

1 Introduction

There is ample evidence that students learn when they teach their peers [1]. Such an *effect of tutor learning* has been observed across different subjects, age groups, format of tutoring, and so forth. Yet, little is known about when tutors' learning would be facilitated and why. A scientific contribution of the current study is at our exploratory effort to study cognitive and social factors for tutor learning. Even when tutor learning is effective, there are practical difficulties to exercise peer tutoring in an actual classroom – not only would it be time consuming (the students must take turns) but, also, the tutees might not learn as much as tutors do. Thus, on the engineering side of our contribution, building an effective and efficient learning environment that facilitates tutor learning is one of our primary research goals.

We have developed a game-like learning environment where students learn by teaching a computer agent, called *SimStudent*. SimStudent is a machine-learning agent that learns cognitive skills from examples and through its own tutored problem-solving experiences [2]. In this learning environment, students interactively tutor SimStudent the same way as they tutor their peers. Our long-term research goal is to investigate the effect of tutor learning with SimStudent as a teachable agent.

The aim of this paper is to provide an overview of our learning by teaching system and discuss results from an evaluation study. The primary research question in the current paper addressed whether or not the students learn by teaching SimStudent at all, and if so, how effective the system is.

In the following sections, we first discuss the issues in learning by teaching. We then introduce our learning by teaching system and the SimStudent architecture. To evaluate the effectiveness of the proposed system, we have conducted an evaluation study that is described in Section 4, followed by general discussions.

2 Learning by Teaching

2.1 Type and Domain of the Proposed Learning by Teaching Environment

The *effect of tutor learning* has been studied in many different domains, across ages, and in various tutoring settings [3]. Various forms of tutoring have been observed including reciprocal teaching [4] and collaborative passage learning [5]. The effect of tutor learning has been observed for all age groups including college [6], high school [7], middle school [8], and elementary school students [9]. The tutor learning effect has been shown to be relatively more effective in math than reading [3, 10].

In the current study, we focus on one-on-one tutoring where a single student acts as a tutor and a computer agent plays the tutee's role. Although the SimStudent technology and the overall framework of the proposed learning environment are domain independent, the current learning system is built for algebraic linear equations – one of the more challenging subjects in mathematics.

2.2 Related Studies

There have been a number of simulated students (also called *teachable agents*) developed so far [11-14]. VanLehn et al. [11] developed one of the earliest simulated students and demonstrated its benefit for *teacher training* in physics. Betty's Brain [15] and its variations are the most recent examples of a teachable agent used to study the tutor learning effect. Betty's Brain learns causal relations from a conceptual map created by student by entering nodes (each representing a concept) and links (each representing a causal relation among the concepts). Students can also quiz Betty's Brain with a problem asking a causal relation (e.g., "If dead organisms increase, what happens to the animals?").

While Simulated Students, Teachable Agents and Learning Companion systems have been created, some were never used with real students and others do not ge-

naively learn from student input. While the VanLehn's system [11] incorporated machine learning and could be used for theory generation and to analyze instructional materials, it was not designed for use with students. On the other hand, while Betty's Brain has been used extensively by students and subject to numerous evaluations, it does not have a machine-learning component. Students teach Betty's Brain by editing a concept map, but the system does not learn from the concept map any more than making straightforward inferences from following the links in the map. This paper provides perhaps the first demonstration of a machine learning system being used as a teachable agent by real students and with significant pre-to-post learning outcomes.¹

3 SimStudent as a Teachable Peer Learner

3.1 Overview of SimStudent

The underlying technology for SimStudent's learning is inductive logic programming in the form of programming by demonstration [16, 17].

There are two learning strategies implemented for SimStudent so far – *learning from examples* and *learning by tutored problem solving* [18]. The learning strategy used for the current research context is learning by tutored problem solving, which requires a *tutor agent* that interactively tutors SimStudent. The tutor agent first poses a problem for SimStudent to solve. To solve the problem, SimStudent attempts to apply production rules that are already learned. When a rule is applied (i.e., a step is performed), the tutor agent provides flagged (binary) feedback that merely shows a correctness of the rule application. When the feedback is negative (*regardless of the accuracy of this tutor feedback*), SimStudent attempts to apply another rule.

When SimStudent cannot perform a step “correctly,” then SimStudent asks the tutor agent for a hint about what to do next.² The tutor agent then *demonstrates* the step for SimStudent as a hint, which is equivalent to a so-called bottom-out hint.

SimStudent is an inductive learner – it learns skills by generalizing the examples demonstrated. There are two types of examples: a *positive example* is generated either when SimStudent applies a rule “correctly” (i.e., the tutor agent provides an affirmative feedback), or when the tutor agent demonstrates a step as a hint. A *negative example* is generated when SimStudent receives negative feedback from the tutor agent on a step that SimStudent performed.

As a result of generalization, SimStudent generates a *production rule* that covers all positive examples (i.e., an application of the rule yields the same step mentioned in a positive example) but does not cover any of the negative examples. In other words,

¹ [Comment for the reviewers, not intended to be in the final manuscript] We know of no other such systems, but we welcome reviewers to point out something we may be missing.

² The “correctness” of the step, by definition, is determined by the feedback from the tutor agent. Thus, SimStudent could fail to perform a step correctly when the tutor agent disconfirms the step regardless of the true correctness of the rule application.

SimStudent generates a set of *production rules* sufficient to solve problems that share the underlying domain principles which have been demonstrated.

The above-mentioned examples are accumulated for each of the skills demonstrated. In other words, when a step is demonstrated, it must be annotated with a *skill-name*. Such annotation drastically reduces the complexity of the search space when learning rules. There is another piece of information that improves the search – the *focus of attention* (FoA, for short), which is knowledge about where to pay attention when performing a step. When tutoring SimStudent, the FoA specifies the elements on the tutoring interface. As an example, see the tutoring interface shown in **Fig. 1**. In this particular interface, FoA is a set of cells representing either the left- or right-hand side of an equation or a transformation (see the next section for the details of these interface elements). Technically speaking, the FoA composes the left-hand side of a production rule encoding a pattern matching for a rule application. Without FoA,

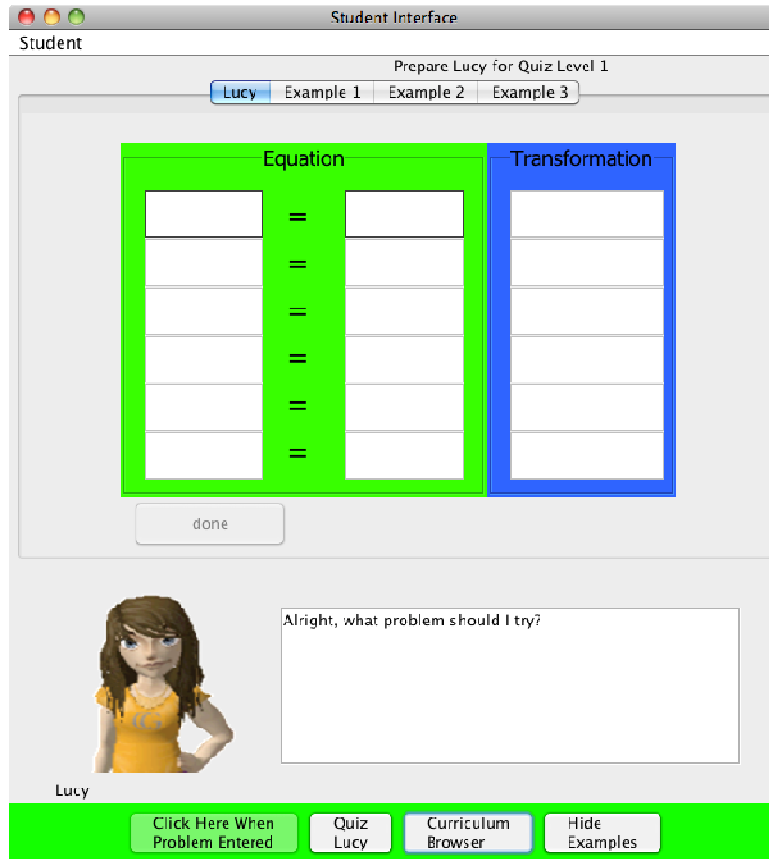


Fig. 1. Learning by Teaching environment. Worked-out examples appear in the interface by clicking the [Example X] tabs at the top of the interface. SimStudent is called Lucy in this example.

SimStudent must search for such a pattern matching, which significantly increases the search complexity. See [18] for details of the SimStudent learning algorithm.

An earlier pilot study showed that asking students to annotate a skill name each time they demonstrate a step is unnatural and confusing. Likewise, asking the students to specify FoA is not practical. Thus, in the proposed learning by teaching environment, both skill names and FoA are implicitly provided to SimStudent using domain dependent ad-hoc heuristics. The next section introduces our Learning-by-Teaching environment and explains how the SimStudent’s learning is naturally integrated into the environment.

3.2 Learning-by-Teaching Environment

Fig. 1 shows a screenshot of our Learning-by-Teaching learning environment (the *LBT environment*, for short). As shown in the figure, the LBT environment is composed of (1) a tutoring interface, (2) a Curriculum Browser, (3) example problems, (4) a communication window, (5) a SimStudent avatar, called Lucy in this example, and (6) the buttons to control the flow of tutoring.

The tutoring interface is used by both SimStudent and the student to collaboratively solve problems. In the current tutoring interface, a single *equation-solving step* is implemented with three *tutoring steps* that are modeled with two *knowledge components* (or skills, if you will). For example, suppose the equation $3x+2=8$ is transformed into $3x=6$ by subtracting 2 from both sides, which, by definition, is a single equation-solving step. In the tutoring interface shown in **Fig. 1**, this equation-solving step consists of (a) specifying a *transformation skill*, which in this case is “subtract,” and (b) *typing in* a left- and a right-hand side of the transformed equation. The tutoring interface thus has three columns (two under “Equation” and one under “Transformation”), corresponding to these three steps.

In the LBT environment, a student performs the role of the tutor agent mentioned in section 3.1. In addition to the tutoring activities mentioned above, the student also gives SimStudent a quiz to gauge its competency. Indeed, the goal of the LBT environment is to tutor SimStudent well enough to pass the quiz, which provides a bit of the flavor of an on-line game to the LBT environment.

To pose a problem for SimStudent to solve, the student simply enters an equation into the tutoring interface, and clicks the [Solve the Equation] button. The *communication window*, the box next to the SimStudent avatar, is used to display a message from SimStudent (e.g., a request for feedback on a step performed or a request for a hint). The student provides feedback by clicking on a [Yes]/[No] button that also appears in the communication window. To provide a “hint”, the student simply performs a step in the tutoring interface.

One obvious question is “What if the student cannot provide a hint?” We have two supporting materials for the student to prepare for tutoring. The *Curriculum Browser* shows an overview of the curriculum to learn (“equations with variables on both sides,” in the current study). It has descriptions on the goal of the subject, skills to be learned, and an example that explains how to solve a typical problem. The Curriculum Browser appears when the student clicks on the [Curriculum Browser] button. A set of examples is also available when the [Example] button is clicked. Unlike the exam-

ple in the Curriculum Browser, these examples are shown inside the *tutoring interface* with the exact tutoring steps. Thus, the student can learn how to use the tutoring interface as well.

4 Evaluation Study

To evaluate the LBT environment, we have conducted a lab study. This section describes the details of the study and the results.

The goal of the study is to measure the degree of tutor learning (the effectiveness) and the efficiency and usability of the system. The effectiveness of the study was measured as learning gain using pre- and post-tests. For the efficiency and usability, we conducted a protocol analysis by video-recording the full learning sessions.

4.1 Method

The study involved 12 students ranging from 6th to 8th grade. The participants were recruited from local middle schools for monetary compensation. The study sessions described here were run individually.

All 12 participants followed the same procedure. Before the study session began, the experimenter explained to the participants that for the whole study sessions they were to think aloud. During the study sessions, the experimenter gave an occasional prompt to the participants when they became quiet.

The participants first took an on-line pre-test (the details of the test are described in the next section). The pre-test took 46 minutes on average. They then watched a video (10 minutes) to learn how to use the LBT environment. After watching the video, the participants were told the goal of the study session – to have Lucy (the name of SimStudent in this particular study as shown in **Fig. 1**) pass the quiz. The participants then tutored Lucy. After an hour, the experimenter told the participants that they were welcome to quit the session if they wanted even when Lucy had not passed the quiz.

Finally, the participants took a post-test. The pre- and post-tests were isomorphic, and the difference in the tests was counter-balanced across the participants. After the post-test, all of the participants completed a post-study questionnaire (due to space limitations, we will not discuss the results of the questionnaire in this paper).

All of the study sessions, including the pre- and post-tests as well as the tutoring sessions, were video recorded. The participants' activities during the tutoring session were logged into an open data repository, called DataShop, maintained by the Pittsburgh Science of Learning Center [19].

4.2 Study materials

The pre- and post-tests were implemented as on-line tests authored using the Cognitive Tutor Authoring Tools. When taking a test, the participants were given a piece of paper to write down their work.

The on-line test consists of question items for (1) equation solving, (2) term identification, (3) what to do next, (4) equivalent expressions, and (5) error identification. The *equation solving* items are to solve equations (e.g., $-3y+6 = 8+5y$) on paper and then only enter the final answer (e.g., $y = -1/4$) into the on-line test form. The *term identification* questions are to identify variable and constant terms in a given expression (e.g., $3 = 4 - 5b$). There are six term-identification questions, which are implemented as multiple-choice questions with six or seven choices. The *what-to-do-next* questions are to identify an appropriate next step for a given equation. There are three what-to-do-next questions, are implemented as multiple-choice questions with four choices. The *equivalent expression* questions are to find an expression that is equivalent to a given expression (e.g., $4x+3$). There are two equivalent-expression questions, which are implemented as multiple-choice questions with five choices. The *error identification* questions are to identify the incorrect step in a given worked-out example of equation solving. The participants are also asked to provide a reason why the step is incorrect in a free text response. An error-identification question is scored as correct when the incorrect step is correctly selected and the corresponding reason is correctly provided. There are five error-identification questions with four to six steps.

4.3 Results

This section shows the participants' learning outcomes, the analysis of the tutoring activities, and a qualitative analysis of the participants' think aloud protocols, which were recorded during the tutoring sessions.

4.3.1 Overall learning gain

We started comparing the overall test scores, which were computed as a percent of correct answers to the total number of question items on the test. There was no main effect on the test (pre- vs. post-) for the overall test scores – average 0.58 on pre-test and 0.61 on post-test (paired- $t=2.36$, $p=0.29$). There was, however, a main effect found on the test for the equation solving (1.5 out of 6 correct on pre-test and 2.5 correct on post-test; paired- $t=2.36$, $p=0.03$) and error detection test items (1.38 out of 5 correct on pre-test and 2.63 correct on post-test; paired- $t=2.36$, $p=0.01$).

After tutoring SimStudent, the participants had improved their skills with solving equations and identifying errors in given solutions, although their conceptual understanding related to equation solving did not improve significantly.

4.3.2 Students activities

In addition to the test scores, we have analyzed the activity logs recorded during the sessions. At the beginning of the experiment, there was a technical issue in logging and we excluded the first five participants from the activity analysis.

On average, the participants spent 70.0 minutes tutoring SimStudent. On average, the participants posed 15 problems to SimStudent.

Five out of six participants reviewed examples an average of 7.8 times (counted as the number of times they switched the examples in the tutoring interface).

SimStudent requested hints 31.8 times on average during a single tutoring session. SimStudent applied rules and asked for their correctness 79.5 times on average. Finally, the participants quizzed SimStudent 4.3 times on average.

4.3.3 Protocol analysis

To see differences in tutoring activities among the students who improved their performance in equation solving and those who did not, we performed a protocol analysis with the videos taken during the tutoring sessions. We divided the students into three categories: those who showed significant improvement on the equation solving test-items (Category High, or CH in **Table 1**), those who showed moderate improvement, and those who showed minimal improvement (Category Low, or CL). The scores on the equation solving section for CH and CL students are shown in **Table 1**. Recall that there were six equation-solving items on each of the tests.

It turned out that the CH students used equations in the examples and quiz items for tutoring more often than the CL students. The CH students often copied those equations to tutor SimStudent, hence more likely tutored SimStudent correctly. On the other hand, the CL students tended to make equations by themselves and failed to solve them correctly. Indeed, one particularly interesting finding is that *being able to pose equations for tutoring is one task, but being able to solve them is another*. When making a new problem, one of the CL students always started with a number in his mind and some arithmetic to make another number and then performed those operations to construct an equation. For example, he said, “*I’ll start with 4, that is an x . If I multiply it with 3, I get 12. I’ll add 5, which is 17. So, $3x+5 = 17$. This is the equation.*” However, he could not solve equations by reverse engineering them when he did not know the “starting” place.

Another interesting comparison is that the CH students went back and forth between examples and the Curriculum Browser contents when they got stuck more often than the CL students did. On the other hand, the CL students did not seem to learn much from the examples and the contents of the Curriculum Browser. The CH students also tended to pose the *exact same problems* used in the Quiz and examples to tutor SimStudent, so that students could at least show SimStudent what to do next without getting stuck.

Table 1. Number of correctly solved equation-solving test items. There were six equation-solving questions.

	Pre-test	Post-test	Category
Student A	0	3	CH
Student B	1	3	CH
Student C	0	0	CL
Student D	1	1	CL

5 Discussion

5.1 Learning by Teaching SimStudent

Overall, the students in the evaluation study did improve their skills in equation solving after tutoring SimStudent for about 70 min in average. A purpose of the evalua-

tion study was to see if the students learn by teaching SimStudent at all, and so we did not compare learning by teaching with other existing interventions. We plan to conduct a further study to compare learning by teaching with learning by being tutored, in which an existing Algebra I Cognitive Tutor will be used for the control condition.

5.2 Low Proficiency Students do not learn from Examples

The protocol analysis often showed that the participants with low proficiency on equation solving often got stuck when providing a hint in response to SimStudent's request for what to do next. It is particularly interesting to see that participants could pose a problem for tutoring, but could not solve it (as mentioned in 4.3.3).

The current LBT environment is designed to encourage students to ask a classroom teacher or consult a textbook when they get stuck. The learning environment also has the Curriculum Browser and example problems. However, it turned out that these supplemental materials are not enough for some of the low proficiency participants. In particular, these participants apparently needed more direct instruction (as opposed to learning from worked-out examples) on how to solve equations.

5.3 How would SimStudent's Prior Knowledge affect Tutor Learning?

The current evaluation study used a version of SimStudent that starts with a "tabula rasa." Namely, it does not know anything about equation solving. Therefore, the students must start with teaching very basic equations (i.e., one-step equations), which forces more time for tutoring. It would be worth investigating how the difference in SimStudent's prior knowledge affects the tutor's learning outcome. A related issue is has to do with the *quality* of SimStudent's prior knowledge. A prior study shows that differences in the quality of the prior knowledge affects SimStudent's learning in both speed and the types of errors made on the test [20], which would also affect the tutor learning.

6 Conclusion

Students actually learned equation solving by tutoring SimStudent. The participants who did not show proficiency on solving equations on the pre-test did improve their skills at solving equations after tutoring SimStudent.

The higher-level goal of the current project is to explore cognitive and social factors that facilitate the effect of tutor learning. We plan to conduct a series of studies in actual classroom settings. The studies include a self-explanation study (SimStudent will ask the student to provide justifications for the student's tutoring activities), a game-show study (several SimStudents tutored by different students will compete against each other in a virtual game-show environment in solving challenging problems), and a meta-tutor study (there will be an embedded third agent in the learning environment that monitors the student's activities and provides feedback and hints).

7 References

1. Roscoe, R.D. and M. Chi, *Tutor learning: the role of explaining and responding to questions*. Instructional Science, 2008. **36**(4): p. 321-350.
2. Matsuda, N., W.W. Cohen, and K.R. Koedinger, *Applying Programming by Demonstration in an Intelligent Authoring Tool for Cognitive Tutors*, in *AAAI Workshop on Human Comprehensible Machine Learning (Technical Report WS-05-04)*. 2005, AAAI association: Menlo Park, CA. p. 1-8.
3. Cohen, P.A., J.A. Kulik, and C.I.C. Kulik, *Education outcomes of tutoring: A meta-analysis of findings*. American Educational Research Journal, 1982. **19**(2): p. 237-248.
4. Palincsar, A.S. and A.L. Brown, *Reciprocal teaching of comprehension-fostering and comprehension-monitoring activities*. Cognition and Instruction, 1984. **1**(2): p. 117-175.
5. Bargh, J.A. and Y. Schul, *On the cognitive benefits of teaching*. Journal of Educational Psychology, 1980. **72**(5): p. 593-604.
6. Annis, L.F., *The processes and effects of peer tutoring*. Human Learning, 1983. **2**(1): p. 39-47.
7. Morgan, R.F. and T.B. Toy, *Learning by teaching: A student-to-student compensatory tutoring program in a rural school system and its relevance to the educational cooperative*. Psychological Record, 1970. **20**(2): p. 159-169.
8. Jacobson, J., et al., *Cross-Age Tutoring: A Literacy Improvement Approach for Struggling Adolescent Readers*. Journal of Adolescent & Adult Literacy, 2001. **44**(6): p. 528-536.
9. Fuchs, L.S., et al., *Enhancing Students' Helping Behavior during Peer-Mediated Instruction with Conceptual Mathematical Explanations*. Elementary School Journal, 1997. **97**(3): p. 223-49.
10. Cook, S.B., et al., *Handicapped Students as Tutors*. Journal of Special Education, 1986. **19**(4): p. 483-492.
11. VanLehn, K., S. Ohlsson, and R. Nason, *Applications of simulated students: An exploration*. Journal of Artificial Intelligence in Education, 1994. **5**(2): p. 135-175.
12. Chan, T.-W. and C.-Y. Chou, *Exploring the Design of Computer Supports for Reciprocal Tutoring*. International Journal of Artificial Intelligence in Education, 1997. **8**: p. 1-29.
13. Hietala, P. and T. Niemirepo, *The competence of learning companion agents*. International Journal of Artificial Intelligence in Education, 1998. **9**: p. 178-192.
14. Biswas, G., et al., *Learning by teaching: a new agent paradigm for educational software*. Journal Applied Artificial Intelligence, 2005. **19**(3&4): p. 363-392.
15. Biswas, G., et al., *Incorporating self regulated learning techniques into learning by teaching environments*, in *Proceedings of the 20th Annual Meeting of the Cognitive Science Society*. 2001: Chicago, IL. p. 120-125.
16. Muggleton, S. and L. de Raedt, *Inductive Logic Programming: Theory and methods*. Journal of Logic Programming, 1994. **19-20**(Supplement 1): p. 629-679.
17. Cypher, A., ed. *Watch what I do: Programming by Demonstration*. 1993, MIT Press: Cambridge, MA.
18. Matsuda, N., et al., *Why Tutored Problem Solving may be better than Example Study: Theoretical Implications from a Simulated-Student Study*, in *Proceedings of the International Conference on Intelligent Tutoring Systems*, B.P. Woolf, et al., Editors. 2008, Springer: Heidelberg, Berlin. p. 111-121.
19. Koedinger, K.R., et al., *An open repository and analysis tools for fine-grained, longitudinal learner data*, in *Proceedings of the international Conference on Educational Data Mining*. 2008.
20. Matsuda, N., et al., *A Computational Model of How Learner Errors Arise from Weak Prior Knowledge*, in *Proceedings of the Annual Conference of the Cognitive Science Society*, N. Taatgen and H. van Rijn, Editors. 2009, Cognitive Science Society: Austin, TX. p. 1288-1293.