Identifying Students' Inquiry Planning Using Machine Learning

Orlando Montalvo[2], Ryan S.J.d. Baker[2,1], Michael A. Sao Pedro [1], Adam Nakama[2], and Janice D. Gobert[2,1]
{amontalvo, rsbaker, mikesp, nakama, jgobert}@wpi.edu
[1]Computer Science Department, Worcester Polytechnic Institute
[2]Social Science and Policy Studies Department, Worcester Polytechnic Institute

Abstract. This research investigates the detection of student meta-cognitive planning processes in real-time using log tracing techniques. We use fine and coarse-grained data distillation, in combination with coarse-grained text replay coding, in order to develop detectors for students' planning of experiments in Science Assistments, an assessment and tutoring system for scientific inquiry. The goal is to recognize student inquiry planning behavior in real-time as the student conducts inquiry in a micro-world; the eventual goal is to provide real-time scaffolding of scientific inquiry.

1 Introduction

Self-regulation is recognized as a highly important aspect of learning [3, 12, 25]. Self-regulation includes planning, meta-cognitive monitoring, reflection, and checking outcomes. While several studies on self-regulation within computer-based learning environments have been conducted [15, 18, 23, 27], there is no consensus about how to automatically measure self-regulation [3, 23]. Furthermore, very few studies have addressed planning within the context of scientific inquiry. Some research has shown that deliberate scaffolding of self-regulation leads to better learning in science [22], but it is difficult to figure out what to measure [15]. Our study seeks to demonstrate a method for detecting one aspect of self-regulation, students' planning in the context of scientific inquiry. Planning is one of the inquiry skills outlined by the National Science Education Standards [21]. Since inquiry problems require several meta-cognitive processes, one of which is planning [11], detecting students' inquiry strategies and skills, including planning, is a critical first step in order to provide students with support in the form of computer-based adaptive scaffolding during real time inquiry [13, 14]. This study brings together research on self-regulation and planning during scientific inquiry.

In this paper, we present a machine-learned model that detects student planning by tracing time spent looking at data tables and hypothesis lists within our inquiry-based learning environment, Science Assistments (http://users.wpi.edu/~sci_assistments; [13, 14]) and microworld for Phase Change. Planning is required especially when applying the control for variables strategy (CVS), a key cognitive strategy within the domain [10], but also in deciding what experiments are needed. We leverage from the success of [6], in using text replays [4] to provide training instances for machine-learned detectors of gaming the system within intelligent tutors. Specifically, by manually inspecting and coding a proportion of the student inquiry sequences using text replay tagging of log files, we extended this approach in order to develop detectors that determine whether a student is planning by viewing data from their previous trials and/or their hypotheses. Our text replay coding approach differs from previous text replays in two ways. First, whereas text replays allow for the classification of a replay clip as a single category (out

of a set of categories), text replay tagging allows multiple tags to be associated with one clip. For example, a clip may be coded as using the data table for planning, using the hypothesis table for planning, both, or neither. Second, the behaviors we are studying are temporally more coarse-grained than in [6], displaying the entire sequence of experimental trials for part of a hypothesis rather than specific trials. In addition to this coding, we summarized each clip by creating a feature set from the action data. In accordance with our coding, we consider problem-level features of the student data rather than step or transaction-level data, unlike in many prior EDM models of student behavior (e.g. [2, 8, 9, 24, 26]). Using the coding and the feature set, we created detectors for planning.

2 Learning Environment

Our phase change environment (Figure 1), hosted by the Science Assistments [13, 14], enabled students to engage in authentic inquiry using a microworld and inquiry support tools. Each problem in our learning environment required students to conduct experiments to determine if a particular independent variable, e.g., container size, affected various outcomes like the melting point or boiling point of a substance.

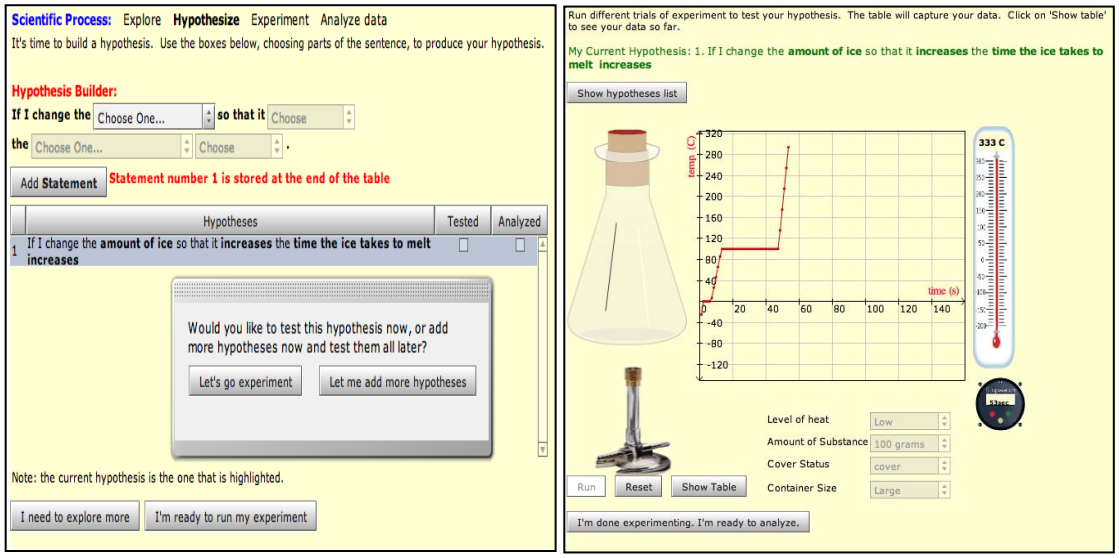

Figure 1. Hypothesizing widget (left) and data collection panel (right) for the phase change microworld.

We scaffold students' inquiry processes by organizing these tasks into different inquiry stages, namely, "observe", "hypothesize", "experiment", and "analyze data". Students start in the hypothesizing stage and move between stages in a suggested order but can navigate back and forth between some of the inquiry phases. For example, from the "analysis" stage students can collect more data by returning to the "experiment" stage, they can create new hypotheses by returning to the "hypothesize" stage (starting a new inquiry loop), or can submit their final experimentation procedures and analyses and begin the next problem. While in the hypothesizing stage, they can either explore the microworld or begin collecting data in the experiment phase. Finally, within the experiment phase, students can only move to the analysis phase.

This learning environment has a moderate degree of learner control, less than in purely exploratory learning environments [2], but more than in model-tracing tutors [17] or constraint-based tutors [20]. Though our scaffolding restricts when students can switch inquiry phases, there is enough freedom such that students can approach these inquiry tasks in many ways, e.g., while experimenting, students could set up and run as many different experiments as they desired.

In the Hypothesis stage, the student is prompted to build a hypothesis using drop down boxes (Hypothesis Builder). The fields are: independent variable, change to the independent variable, dependent variable, and change to dependent variable. So, for example, the student can change the first [Choose One…] box to "amount of ice", which enables the next box. Proceeding, the student can create the hypothesis "If I change the [amount of ice] so that it [increases], the [melting point][doesn't change].

When students reach the experimentation stage, they can then change independent variables, such as Level of Heat, and see the results within the microworld by running a trial (by clicking on Run). They can also view representations of their full set of hypotheses (by clicking on Show hypotheses list) and they can view the trial run data (clicking on Show Table). Both the data table and the hypothesis list provide external memory aid, allowing the student use information about previous decisions to reflect and plan new experimental trials.

As students solve these inquiry problems, they could engage in a number of behavior patterns. Particular to collecting data, systematic [24] students collect data that test their hypotheses by designing and running controlled experiments. Additionally, such students may use the table tool and hypothesis list to reflect upon their results and plan for additional experiments they may need to run. Students who are unsystematic in their experimental design and collection of data may exhibit haphazard behaviors such as: constructing experiments that do not test their hypotheses, not collecting enough data to support their hypotheses, not using CVS, or running the same experimental setup multiple times [17].

3 Data Set

Participants were 148 eighth grade students, ranging in age from 12-14 years, from a public middle school in Central Massachusetts. These students used the phase change microworld. Students engaged in authentic inquiry problems using the phase change and density microworlds within the Science Assistments learning environment. As part of the phase change activities, students attempted to complete four tasks using our interactive tools.

Each of these students completed at least one data collection activity in the phase change environment (two other students did not use the microworld, and were excluded from analysis). As students solved these tasks, we recorded fine-grained actions within the inquiry support tools and microworlds. The set of actions logged included creating hypotheses, setting up experiments, showing or hiding support tools, running experiments, creating interpretations of data, and transitioning between inquiry activities

(i.e., moving from hypothesizing to data collection). Each action's type, current and previous values (where applicable – for instance, a variable's value), and timestamp were recorded. In all, 27,257 student actions for phase change were logged. These served as the basis for generating text replay clips consisting of contiguous sequences of actions specific to experimenting.

4 Method

The data used for this study was collected by Science Assistments, which logs every widget action performed by the student including button clicks, checkbox choices, etc. Each action has a time stamp, student/problem identifiers, and widget information, and is tagged as to its step (step tag) in the inquiry process. The step tags are a level above a simple action (this is captured by the widget information), representing a step within the inquiry process, across microworlds. This allows us to analyze similar actions across microworlds. These step tags are used for two purposes: as markers to create clips for text replay coding and to categorize data for fine-grain feature extraction.

4.1 Text Replay Coding

Text replay hand coding presented our team with two significant challenges: specific codes and grain size. In designing our text replays, it was necessary to use a coarser grain-size than in prior versions of this method [4]. In particular, it is necessary to show significant periods of experimentation in order to put usage of the table and hypothesis list into context, while limiting clip size to reduce memory load. We decided to use clips that include both the hypothesis and the experiment stages, which is long enough to see context, but short enough to tractably code. Another important issue in grain-size selection is that trial run data from one hypothesis test can be used in another to make inferences about the hypothesis at hand (for instance, by comparing a current trial to one conducted earlier). To compensate for this, we code using both the actions in testing the current hypothesis, and cumulative measures which include actions performed when testing previous hypotheses.

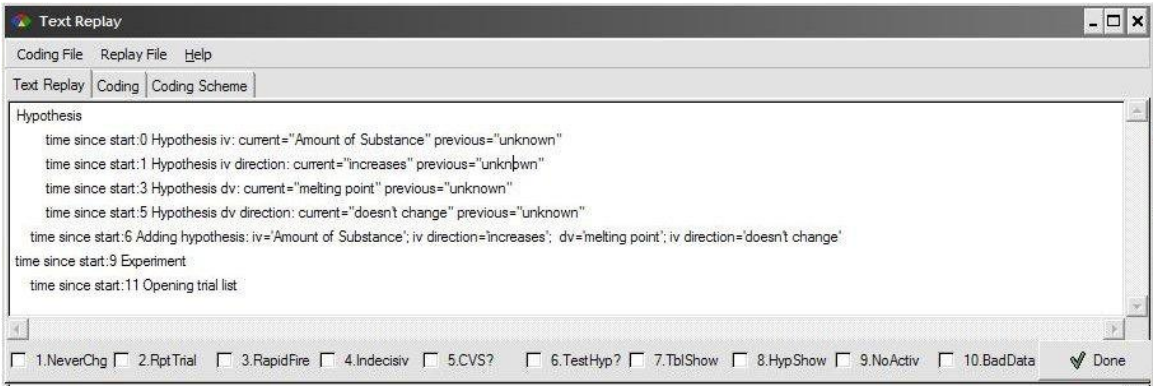

Figure 2 - Clip showing a single Hypothesis-Experiment run (clips may be significantly longer)

To support coding in this fashion, a new tool for text replay tagging was developed in Ruby, shown in Figure 2. The start of the clip is triggered by a hypothesis variable change after the beginning of a new problem. The tool displays all student actions

(hypothesis and experiment) until the student transitions to the analysis stage. Subsequent clips include previous clips and any single new cycle which includes the Hypothesis and Experiment stage. A clip could be tagged with one of 10 tags: "Never Change Variables", "Repeat Trials", "Non-Interpretable Action Sequence", "Indecisiveness", "Used CVS", "Tested Hypothesis", "Used Table to Plan", "Used Hypothesis Viewer to Plan", "No Activity", and "Bad Data." Specific to our study, we tagged a clip as "Used Table to Plan" (TablePlan) if the clip contained actions indicative that the student viewed the trial run data table in a way consistent with planning for subsequent trials. "Used Hypothesis Viewer to Plan" (HypPlan) was chosen if the clip had actions indicating that the student viewed the hypotheses list in a way consistent with planning for subsequent trials.

4.2 Coding Agreement

Two coders (the third and fourth authors) tagged the data collection clips using at least one of the ten tags. To ensure that a representative range of student clips were coded, we stratified our sample of the clips on condition, student, problem, and within-problem clip order (e.g. first clip, second clip, etc.). The corpus of hand-coded clips contained exactly one randomly selected clip from each problem each student encountered, resulting in 581 clips. Each coder tagged the first 50 clips; the remaining clips were split between the coders. Of the 50 clips tagged, 7 were discarded because of a problem with an early version of the text replay tool where the problem number of the code did not match the problem number of the microworld.

For the 43 clips tagged by each coder, there was high overall tagging agreement, average $\kappa = 0.86$. Of particular relevance to this study, there was strikingly high agreement on the TablePlan, $\kappa = 1$ and of HypPlan, also $\kappa = 1$. Kappa at this level suggests particularly good agreement between coders, which was achieved in part through extensive discussion and joint labeling prior to the inter-rater reliability session. In particular, the coders found these two categories easy to code, as students either tended to spend significant amounts of time reflecting on these tools, or viewed them extremely briefly (or not at all). These categories were also relatively rare, potentially increasing κ by chance; only 8% of clips involved TablePlan and only 4% of clips involved HypPlan.

4.3 Feature Distillation

Features extracted can be grouped into 10 categories: all actions, total trial runs, incomplete trial runs, complete trial runs, pauses, data table display, hypothesis list display, field changes in Hypothesis Builder, hypothesis made, and microworld variable changes. For each of these categories we traced the number of times the action and the time taken for each action. Two other categories were included indirectly related to actions: the number trials where only one independent variable was different between the two trials and the number of times a trial was repeated. These last two had no times associated with them.

The microworld activity was divided into tasks in which the focus was a specific independent variable. Since there were four independent variables, there were four tasks. Within a task, the student is allowed to make and test several hypotheses. For each of the

12 categories above, we extracted data for each hypothesis the student worked on (non-cumulative data), and across all hypotheses in the task (cumulative data). The reason for this is that within each task, the data table accumulates the trial run data across hypotheses. This allows the students to compare trial runs testing previous hypotheses with the runs made in the current hypothesis.

Lastly, the time data was distilled to obtain the following values: minimum, maximum, standard deviation, mean and mode. It is these values plus the count which was used in the machine learning model. This data was arranged in a comma-delimited flat file suitable for input into RapidMiner. The data was divided into files, one for each coded feature. The coded feature being the first item on the line, followed by the distilled features described above.

4.4 Machine Learning Algorithms

Machine-learned detectors of the two behavioral patterns were developed within RapidMiner 4.6 [19] using the default settings. Detectors were built using J48 decision trees, with automated pruning to control for over-fitting, the same technique used in [26] and [6]. Six-fold cross-validation was conducted at the student level (e.g. detectors are trained on five groups of students and tested on a sixth group of students). By cross-validating at this level, we increase confidence that detectors will be accurate for new groups of students. We assessed the classifiers using two metrics. First, we used A' [16]. A' is the probability that if the detector is comparing two clips, one involving the category of interest (TablePlan or HypPlan) and one not involving that category, it will correctly identify which clip is which. A' is equivalent to both the area under the ROC curve in signal detection theory, and to W, the Wilcoxon statistic [16]. A model with an A' of 0.5 performs at chance, and a model with an A' of 1.0 performs perfectly. In these analyses, A' was used at the level of clips, rather than students. Statistical tests for A' are not presented in this paper. The most appropriate statistical test for A' in data across students is to calculate A' and standard error for each student for each model, compare using Z tests, and then aggregate across students using Stouffer's method [5] – however, the standard error formula for A' [16] requires multiple examples from each category for each student, which is infeasible in the small samples obtained for each student in our text replay tagging. Another possible method, ignoring student-level differences to increase example counts, biases undesirably in favor of statistical significance.

Second, we used Kappa (κ), which assesses whether the detector identifies is better than chance at identifying the correct action sequences as involving the category of interest. A Kappa of 0 indicates that the detector performs at chance, and a Kappa of 1 indicates that the detector performs perfectly. As Kappa looks only at the final label, whereas A' looks at the classifier's degree of confidence, A' can be more sensitive to uncertainty in classification than Kappa.

5 Results

We constructed and tested detectors using our corpus of hand-coded clips. TablePlan and HypPlan detectors were constructed from a combination of the subset of the first 43 clips

that the two coders agreed on, the remaining clips, tagged separately by the two coders. In total, 570 tagged clips were used for each detector. Of these clips, 47 out of 570 were tagged with TablePlan (8%) and 20 out of 570 (4%) were tagged with HypPlan.

Table 1. Best results for detectors of each coding category

Category	A'	κ	Attribute type	% students in data
HypPlan	.93	.14	Non-cumulative	8%
TablePlan	.94	.46	Cumulative	4%

Detectors were generated for each behavior using J48 decision trees and two sets of attributes, cumulative and non-cumulative attributes. Thus, four different detectors were constructed two for TablePlan and two for HypPlan. The TablePlan detector using cumulative attributes (A' = .94, κ = .46) performed slightly better than the detector built with non-cumulative attributes (A' = .96, κ = .36). Both versions of the detector achieved excellent performance, comparable to detectors of gaming the system refined over several years (e.g., Baker & de Carvalho, 2008), and are very likely to be appropriate for use in interventions. The HypPlan detectors did not perform as well, achieving A' = 0.93, κ = 0.14 for the non-cumulative attributes and A'=.97, κ = 0.02 for the cumulative attributes. The substantial difference between A' and κ is unusual. It appears that what happened in this case is that the model, on cross-validation, classified many clips incorrectly with low confidence; in other words, A' by considering pair-wise comparisons catches the overall rank-ordered correctness of the detector across confidence values even though many clips were mis-categorized at the specific threshold chosen by the algorithm. One possibility is that the low number of HypPlan labels in the data set made the detectors more prone to over-fitting. This result suggests that the HypPlan detector is probably acceptable for fail-soft interventions, where students assessed with low confidence (in either direction) can receive interventions that are not costly if mis-applied.

6 Discussion and Conclusions

In this paper, we have presented models for detecting planning within science inquiry learning. Our efforts to detect planning from data table usage have met with greater initial success than our attempts to detect planning within the hypothesis list, although both detectors are, we feel, good enough to use for some forms of instructional intervention. The detector for showing data table use (TablePlan) in planning can detect a student using the data table effectively from one not using the data table effectively for planning 94% of the time. The κ=.46 is respectable, so this detector can be used robustly to scaffold table use for planning during inquiry. If we detect that a student is not using the table effectively, we can suggest that the user look at the table and provide hints on how to compare one table row with another, and how to use this to plan the next trial.

On the other hand, the detector for using the hypotheses table for planning (HypPlan) did not perform as well. Although it had a very good A' (.93 and .97), the κ was low, meaning that if we used this detector for scaffolding, we will need to do it in a fail-soft manner. There is reason to believe this approach may be successful. For example, an early detector of gaming the system [7] with a similar κ and lower A' was found to be effective for improving gaming students' learning when used in a fail-soft manner. In addition, combining the HypPlan detector with another (for example, one that detects control for variables strategy or CVS) may compensate for its low κ. So for example, if a detector indicated that CVS was not being used, this detector also can be used to decide if scaffolding should include a hint regarding how the student should use the hypothesis table in order to reflect on their work. In this fashion, interventions based on this detector will only be given when there is additional reason to believe that intervention is needed.

Future work will include improving our κ for HypPlan and finding other meta-cognitive tasks that can be detected effectively. This would require an expansion of the tags we used and perhaps a way to track student progress from one problem to another, since lesson-wide attributes may be useful for measuring students' progress.

By detecting planning in real time, rich adaptive scaffolding becomes feasible [13, 14]. In addition, with helping students learn both content and inquiry skills, scaffolding for planning can help them become better learners, possibly by influencing their meta-cognitive skill development [1, 23]. This study makes an important contribution towards linking these two areas of research, namely, meta-cognitive skills and planning during scientific inquiry.

Acknowledgements

This research is funded by the National Science Foundation (NSF-DRL#0733286)**,** Janice Gobert, Principal Investigator**,** Neil Heffernan, Ryan Baker, and Carolina Ruiz, Co-Principal Investigators, and the U.S. Department of Education (R305A090170), Janice Gobert, Principal Investigator**,** Neil Heffernan, Ken Koedinger, and Joe Beck, Co-Principal Investigators. Any opinions expressed are those of the authors and do not necessarily reflect those of the funding agencies.

References

[1] Aleven, V., McLaren, B., Roll, I., Koedinger, K.: Toward Meta-cognitive Tutoring: A Model of Help Seeking with a Cognitive Tutor. International Journal of Artificial Integlligence in Education 16(2), p. 101-128, 2006

[2] Amershi, S., Conati, C. Combining Unsupervised and Supervised Machine Learning to Build User Models for Exploratory Learning Environments. *Journal of Educational Data Mining*, 2009, 1(1), p. 71-81.

[3] Azevedo, R.: Theoretical, conceptual, methodological, and instructional issues in research on metacogntion and self-regulated learning: A discussion. Metacognition Learning 4(1), p. 87-95, 2009

[4] Baker, R. S. J. d., Corbett, A. T., Wagner, A. Z. Human Classification of Low-Fidelity Replays of Student Actions. *Proceedings of the Educational Data Mining Workshop at the 8th International Conference on Intelligent Tutoring Systems*, 2006. p. 29-36.

[5] Baker, R. S. J. d., Corbett, A. T., Aleven, V. Improving Contextual Models of Guessing and Slipping with a Truncated Training Set. *Proceedings of the 1st International Conference on Educational Data Mining*, 2008. p. 67-76.

[6] Baker, R. S. J. d., de Carvalho, A. M. J. A. Labeling Student Behavior Faster and More Precisely with Text Replays. *Proceedings of the 1st International Conference on Educational Data Mining*. p. 38-47.

[7] Baker, R.S.J.d., Corbett, A.T., Koedinger, K.R., Evenson, E., Roll, I., Wagner, A.Z., Naim, M., Raspat, J., Baker, D.J., Beck, J. (2006) Adapting to When Students Game an Intelligent Tutoring System. *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*, 392-401.[8] Beck, J. Engagement tracing: using response times to model student disengagement. *Proceedings of the 12th International Conference on Artificial Intelligence*, 2005. p. 88-95.

[9] Cetinas, S., Si, L., Xin, Y. P., Hord, C. Automatic Detection of Off-Task Behaviors in Intelligent Tutoring Systems with Machine Learning Techniques. *IEEE Transactions on Learning Technologies*, in press.

[10] Chen, Z., Klahr, D.: All Other Things Being Equal: Acquisition and Transfer of the Control of Variables Strategy. Child Development, 70(5), 1098-1120 (1999)

[11] de Jong, T. Computer simulations - Technological advances in inquiry learning. *Science, 312*, 2006, , p. 532-533.

[12] Dignath, C., Buttner, G. Components of fostering self-regulated learning among students: A meta-analysis on intervention studies at primary and secondary school level. , 2008, 3, p. 231-264.

[13] Gobert, J., Heffernan, N., Baker, R., Ruiz, C.: AMI: ASSISTments Meets Inquiry (NSF-DRL #0733286)., Awarded September 2007 from National Science Foundation

[14] Gobert, J., Heffernan, N., Koedinger, K., Beck, J.: ASSISTments Meets Science Learing (AMSL; R305A090170)., Awarded February 2009 from the U.S. Dept. of Education

[15] Hadwin, A. F., Nesbit, J. C., Jamieson-Noel, D., Code, J., Winne, P. H.: The use of computer-based environments for understanding and improving self-regulation. Metacognition Learning 2, p. 107-124, 2007

[16] Hanley, J. A., McNeil, B. J. The Meaning and Use of the Area under a Receiver Operating Characteristic (ROC) Curve. *Radiology*, vol. 143, 1982. p. 29-36.

[17] Koedinger, K. R., Corbett, A. T.: Cognitive Tutors: Technology bringing learning sciences to the classroom. In (Ed.), R. K. (Eds.) *The Cambridge handbook of the learning sciences*, 2006.: Cambridge University Press. p. 61-77.

[18] Manlove, S., Lazonder, A. W., Dejong, T.: Software scoffolds to promote regulation during scientific inquiry learning. Metacognition Learning 2, p. 141-155, 2007

[19] Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., Euler, T. YALE: Rapid Prototyping for Complex Data Mining Tasks. *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2006)*, 2006. p. 935-940.

[20] Mitrovic, A., Mayo, M., Suraweera, P., Martin, B. Constraint-based tutors: a success story. Springer-Verlag (Eds.) *Proceedings of the Industrial & Engineering Application of Artificial Intelligence & Expert Systems Conference IEA/AIE-2001*, 2001. p. 931-940.

[21] NSES: National Committee on Science Education Standards and Assessment. (1996)., National Science Education Standards, Washington, D.C., National Academy Press, 1996

[22] Roll, I., Aleven, V., McLaren, B., Koedinger, K.: Designing for metacognition--applying cognitive tutor principles to the tutoring of help seeking. Metacognition Learning 2, p. 125-140, 2007

[23] Schraw, G.: A conceptual analysis of five measures of metacognitive monitoring. Metacognition Learning 4, p. 33-45, 2009[24] Shih, B., Koedinger, K., Scheines, R. A Response Time Model for Bottom-Out Hints as Worked Examples. *Proceedings of the 1st International Conference on Educational Data Mining*, 2008. p. 117-126.

[25] Veenman, M. V. J., Van Hout-Worters, B. H. A. M., Afflerback, P.: Metacognition and learning: conception and methodological considerations. Metacognition Learning 1, p. 3-14, 2006

[26] Walonoski, J. A., Heffernan, N. T. Detection and Analysis of Off-Task Gaming Behavior in Intelligent Tutoring Systems. *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*, 2006. p. 382-391.

[27] Winne, P. H., Nesbit, J. C., Kumar, V., Hadwin, A. F., Lajoie, S. P., Azevedo, R., Perry, N. E. Supporting Self-Regulated Learning with gStudy Software: The Learning Kit Project. *Tech., Inst., Cogntive Learning*, 2006. p. 105-113.