Effort-based Tutoring: An Empirical Approach to Intelligent Tutoring

Ivon Arroyo[1], Hasmik Mehranian[2], Beverly P. Woolf[1]

[1] Department of Computer Science, University of Massachusetts Amherst
[2] Dept. of Mechanical and Industrial Engineering, University of Massachusetts Amherst

Abstract. We describe pedagogical and student modeling based on past student interactions with a tutoring system. We model student effort with an integrated view of student behaviors (e.g. timing and help requests in addition to modeling success at solving problems). We argue that methods based on this integrated and empirical view of student effort at individual items accurately represent the real way that students use tutoring systems. This integrated view helps to discern factors that affect student behavior beyond cognition (e.g., help misuse due to meta-cognitive of affective flaws). We specify parameters to the pedagogical model in detail.

1 Introduction

The traditional structure of intelligent tutoring systems consists of a student model and a pedagogical model to guide activity selection, generally based on heuristics. The most traditional student models are based on tracing student cognitive knowledge [4]. Knowledge tracing (KT) procedures encode cognitive mastery of the different skills being tutored. KT consists of four parameters fit to each knowledge component (KC), and include: *initial learning, learning rate, guess* and *slip parameters.* One advantage of these models is that parameters are interpretable, and being just four, they may also be fit from prior student data for each knowledge component in a domain, using expectation maximization or other techniques [5][6]. While this model is simple and has been used in many tutors, the main disadvantage of this method is that it relies on a definition of student performance in terms of number of incorrect attempts –it does not address how students help requests (via glossaries or hint requests) or issues of timing affect performance. This fact has been addressed in later work [2][3][5][6][7] by creating separate models of engagement or more complex Bayesian models that address the impact of help on student knowledge. The problem that still remains is that issues of timing or hints are not addressed in an *integrated* way within these knowledge estimation models, leading to biased estimations of knowledge (e.g., the student answers very fast incorrectly may lead to an estimation of unknowing, however, it really reflects disengagement). One attempts was [7], who modeled student engagement and knowledge at the same time within one model. It helped to keep knowledge more stable instead of an apparently decreasing knowledge, as if students were unlearning while they were actually disengaged. This work encoded math knowledge as single latent variable, though, which is not practical to make decisions in the pedagogical model, so this work is still preliminary.

In summary, while progress has been made in student modeling and intelligent learning environments, there is a need for student models that *integrate* and discern between engagement, student knowledge and other factors such as affect and meta-cognition, or descriptions about how to juggle different models to make optimal pedagogical decisions. This paper provides one approach that discerns among the reasons for student effort (or not) at individual practice items, based on different

dimensions of student behavior. We then thoroughly document a pedagogical model, which is heavily based on empirical estimates of student effort and problem difficulty. We last provide results of a randomized controlled study that shows the adaptive nature of this tutor does improve learning, compared to an "unintelligent" version that makes less smart moves when selecting practice activities. We also provide a methodology to evaluate that the estimates of problem difficulty are accurate. The level of detail in the methodology allows for replication of the tutoring mechanism and estimates of effort on other learning environments, even for systems without a large amount of content available, or ill-defined content (ill-defined domains).

1.1 Wayang Outpost: A Mathematics Tutoring System

Wayang Outpost is a software tutor that helps students learn to solve standardized-test type of questions, in particular for a math test called the Scholastic Aptitude Test, and other state-based exams taken at the end of high school in the USA. This multimedia tutoring system teaches students how to solve geometry, statistics and algebra problems of the type that commonly appear on standardized tests. To answer problems in the Wayang interface, students choose a solution from a list of multiple choice options, providing immediate feedback on students' entries and offering hints that students can accept or reject. Students are encouraged to ask the tutor for hints that are displayed in a progression from general suggestions to bottom-out solution. In addition to this domain-based help, the tutor currently provides a variety of affective and meta-cognitive feedback, delivered by learning companions designed to act like peers who care about a student's progress and offer support and advice [1][8]. Both decisions about content sequencing and characters response are based on a model of student effort, used to assess the degree of cognitive effort a student invests to develop a problem solution, described in the next sections.

2 Modeling and Acting Upon Student Effort

We start by estimating the expected behavior that a student should have on a problem based on three indicators of effort: 1) number of attempts to solve a problem; 2) number of hints requested for a problem; 3) time required to solve a problem. These are three orthogonal axes that help understand student effort. The only pre-processing done for this data set was to use data corresponding to "valid" student users (instead of test users), and discarding outliers just for the "time" variables.

Figure 1 shows examples of problem solving behavior for nearly 600 students in one problem. This one problem may seem evidently too easy at first glance, as the majority of students made zero or few incorrect attempts, saw no hints, and solved the problem in less that 5 seconds. However, this is not the case. It is common to find problem-student interaction instances where students spend little time and effort. It is also common that students under-use the help in the system. We find it essential to take into account that this is the real way that students use the tutoring system, and we need to take into account what are likely student behaviors when considering how to adjust instruction and the presentation of the material to students. Note that the distributions are not normal, but more similar to Chi-Square distributions.

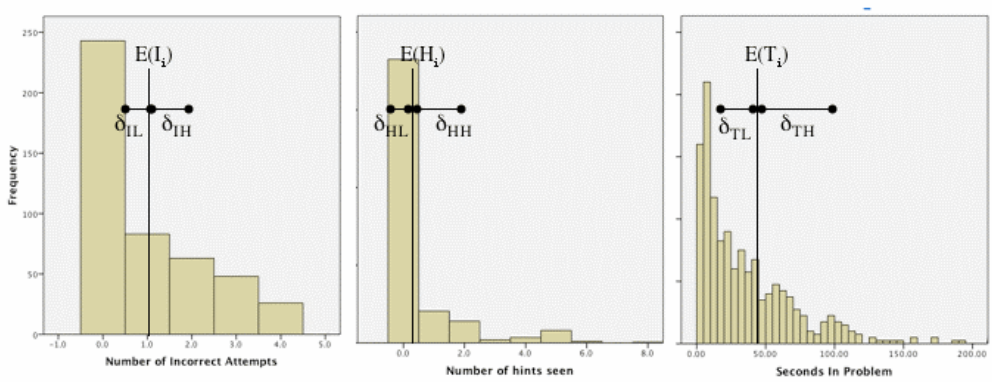

Figure 1. Distribution of attempts, hints and seconds in one problem. Expected and delta values.

The combination of mistakes, hints and time as shown in Figure 1 will allow to estimate higher-level scenarios of mastery or disengagement, see Table 1. For each of the hundreds of problems or practice items in an intelligent tutor, we compute the median (or the sample mean after discarding the top 10 percentile, which was a good approximation in our data and much easier to compute using SQL) and standard deviation for the whole population of students. This median or mean is considered the expected value, i.e. the expected number of incorrect attempts for a problem p_i ($E(I_i)$) where $i=1...N$, and N=total practice items in the tutoring system. Expected hints seen is $E(H_i)$ and time required to solve the problem is $E(T_i)$. We also define two delta values for each $E(I_i)$, $E(H_i)$ and $E(T_i)$, a total of six delta values (see Figure 1) for each problem p_i, which represent a fraction of the standard deviation, regulated by two parameters, θ_{LOW} and θ_{HIGH} in the interval [0,1]. For example, if $\theta_{LOW}=1/4$ and $\theta_{HIGH}=1/2$, then $\delta_{IL}=\theta_{LOW}SD(I_i)= SD(I_i)/4$ (a fourth of the standard deviation of I_i) and $\delta_{IH}= SD(I_i)\theta_{HIGH}= SD(I_i)/2$, half of the standard deviation of I_i. θ_{LOW} and θ_{HIGH} are the same for all problems in the system. These values help define what is "expected behavior" for a practice item within the tutoring system. Note that the notation for δ values has been simplified (e.g. δ_{IL} should really be δ_{I_iL}, as it refers to an individual practice item).

2.1 Pedagogical Decisions based on Student Effort

The large benefit of an effort model based on different orthogonal axes of behavior (hints, time and correctness) is that it can help researchers discern between behaviors related to student engagement (affective) and behaviors related to help misuse (meta-cognitive or affective) in addition to behaviors related to cognitive mastery. Table 1 shows the estimations of most likely scenarios made by the pedagogical model in Wayang Outpost, and the pedagogical decisions made in terms of content difficulty, plus other pedagogical moves related to affective and meta-cognitive feedback. Note that disengagement (e.g. lines 3 and 5) produces a reduction in problem difficulty, based on the assumption that if a student is not working hard enough on the current problem, they probably won't work hard on a similar or harder problem. However, the key intervention is that Learning Companions deemphasize the importance of immediate success.

Table 1. Empirical-based estimates of effort at the recently completed problem lead to adjusted problem difficulty and other affective and meta-cognitive feedback

	Student Model Estimate most likely scenario for student on problem i				Pedagogical Model Moves Cognitive or Affective or Metacognitive	
	Mistakes	Hints	Time	Most Likely	Decision	Other Actions
1	< E(Ii) – δIL	< E(Hi) – δHL	< E(Ti) – δTL	Mastery without effort	Increase Problem Difficulty	Show learning progress
2	< E(Ii) – δIL	< E(Hi) – δHL	> E(Ti) + δTH	Mastery with high effort	Maintain Problem Difficulty	Affective feedback: Praise Effort
3	< E(Ii) – δIL	> E(Hi) +δHH	< E(Ti) – δTL	Hint abuse, low effort	Reduce Problem Difficulty	Deemphasize importance of immediate success
4	< E(Ii) – δIL	> E(Hi) +δHH	> E(Ti) + δTH	Towards mastery, effort	Maintain Problem Difficulty	Praise effort
5	> E(Ii) + δIH	< E(Hi) – δHL	< E(Ti) – δTL	Quick guessing, low effort	Reduce Problem Difficulty	Deemphasize importance of immediate success
6	> E(Ii) + δIH	< E(Hi) – δHL	> E(Ti) + δTH	Hint avoidance and high effort	Reduce Problem Difficulty	Offer hints upon incorrect answer in the next problem
7	> E(Ii) + δIH	> E(Hi) +δHH	< E(Ti) – δTL	Quick guess and hint abuse	Reduce Problem Difficulty	Deemphasize importance of immediate success
8	> E(Ii) + δIH	> E(Hi) +δHH	> E(Ti) + δTH	Low mastery and High Effort	Reduce Problem Difficulty	Emphasize importance of effort and perseverance
9	Otherwise			Expected Behavior	Maintain Problem Difficulty	

The retrieval of an increased difficulty item is based on a function *Harder*(H[*1..n*], γ) that returns a problem of higher difficulty; H is a sorted list of *n* practice items the student has not yet seen, all harder in difficulty than the one the student has just worked on; H[1] is the item of lowest difficulty, and H[*n*] is the item of highest difficulty, and γ is a natural number greater than zero. The problem returned by *Harder* is specified in Eq. 1. For example, *Harder* with γ=3 will return the problem at the 33rd percentile of items in list H[1..m].

$$Harder(H[1..m],\gamma) = H\left[ceiling\left(\frac{m}{\gamma}\right)\right] \qquad \textbf{(1)}$$

Similarly, a problem of lesser difficulty is selected with function *Easier*(E[*1..m*], where E is a sorted list of problem items, all items are easier than the problem just seen by the student; E[1] is the item of lowest estimated difficulty, and E[n] is the item of highest difficulty. Eq. 2 shows *Easier* as a function of m and γ. Easier with γ=3 will return the item that at the 66th percentile of items in list E[*1..m*].

$$Easier(E[1..m],\gamma) = E\left[ceiling\left(m - \frac{m}{\gamma}\right)\right] \qquad \textbf{(2)}$$

Both *Easier* and *Harder* work upon the assumption that there are easier or harder items to choose from. The next section addresses what happens when $m=0$ or $n=0$.

2.2 Progression through Knowledge Units

In Wayang Outpost, the curriculum is organized in a linear set of topics or knowledge units (KU), which is a classification of problems in sets of items that involve similar skills (e.g. polygon perimeter measurement problems). Pedagogical decisions about content sequencing are made at two levels: within a topic and between topics, skills or knowledge units. This section addresses between topic decisions.

The criteria of "chunking" problems in knowledge units is based on the idea that similar problems should be seen close to each other, to maximize the transfer of what a student has learned, as the concepts are still in working memory to be applied to the next cognitive transfer task. Cognitive effort is then reduced, and the likelihood of applying a recently learned skill to the next task is enhanced.

Each knowledge unit may be defined at a variety of levels, and is composed of a variety of problems involving a set of related skills. For instance, within the "Statistics" topic, a student may be presented with problems about finding the median of a set of numbers, or deciding whether the mean or median were larger, from a picture of a stem and leaf plot. While overlap of skills exists, not all problems within a topic involve the same skills, and their difficulties may vary to a large degree.

Topics are arranged according to pre-requisites (problems presented in KU2 will not include skills introduced in KU3). When a topic begins students are presented an explanation of the kinds of problems that will follow, generally introduced aloud by pedagogical animated creatures. Sometimes this involves an example problem, accompanied by a worked-out solution via multimedia features.

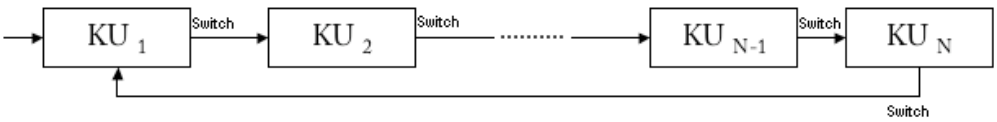

Figure 2. Spiral curriculum in which Knowledge Units are ordered according to pre-requisites

Table 2. Conditions for topic switching in Wayang Outpost

Topic Switch Criterion	Reason	Parameter
2.1 Topic Mastery was reached (e.g. enough "hard" problems answered correctly)	Cognitive	M_{KU}
2.2 Persistent failure to find a problem of desired difficulty	Content limitation	F_{KU}
2.3 Maximum time in Topic condition, or Maximum Number of Problems allowed	Classroom Implementation	T_{KU} N_{KU}

A student progresses through these knowledge units depending on a variety of criteria, specified in Table 2 beyond cognitive mastery. For instance, condition 2.2 shows how a topic switch may be forced based on limitations of content --the system failed F_{KU} times to find a problem of the difficulty it believes the student should get for the topic. If the pedagogical model suggests the student should increase problem difficulty, but there are no harder problems remaining, then a counter for the number of failures for the current topic is increased. Because failures < F_{KU} an easier problem is provided instead. Another possibility is condition 2.3, where the teacher has allocated a specific amount of time for the student to study or review a certain topic.

2.3 Problem Difficulty Estimates

The pedagogical model must be able to estimate problem difficulty in order to assign problems for students in specific scenarios. We identify two faces of problem difficulty in intelligent tutors. From the perspective of a knowledge engineer, problems have *objective difficulty* (e.g., based on number of skills and steps involved in each problem). However, students may perceive each problem differently according to a *student perceived difficulty* (SPD). While objective problem difficulty should be similar to SPD, they are not necessarily the same. Proper estimation of problem difficulty is essential for this pedagogical model, and not possible to do with simple Item Response Theory because tutoring involves more dimensions (help, engagement) than testing (accuracy). We capture SPD from the three independent sources of evidence of students' effort to solve a problem: 1) correctness in term of number of required attempts to solve a problem (random variable C_i); 2) amount of time spent in a problem (random variable T_i); 3) amount of help required or requested to solve the problem correctly (random variable H_i).

We define problem difficulty d_i for a practice activity component i in Eq. 3, as the mean of these three factors: attempts to solve, time and help needed.

$$d_i = mean(dc_i, dt_i, dh_i) \quad \textbf{(3)}$$

Where dc_i is the difficulty factor in terms of correctness, dt_i is the difficulty factor in terms of time, and dh_i is the difficulty factor in terms of help needed. Alternatively, the three factors might be given a weight, to emphasize them differently.

d_i, dc_i, dt_i and dh_i are normalized values in the interval [0,1] and express SPD. Eq. 4, 5 and 6 show how each of the three difficulty factors are computed.

$$dc_i = \frac{E(I_i)}{Max_{j=1}^{N}(E(I_j))} \quad \textbf{(4)}$$

$$dt_i = \frac{E(T_i)}{Max_{j=1}^{N}(E(T_j))} \quad \textbf{(5)}$$

$$dh_i = \frac{E(H_i)}{Max_{j=1}^{N}(E(H_j))} \quad \textbf{(6)}$$

dc_i (Eq. 4) is the expected value of I_i (number of incorrect attempts while trying to solve a problem p_i) across all students who have seen that problem, divided by the

maximum $E(I_j)$ registered for any problem p_j in the system (N=the total number of problems or practice activities in the system).

Similarly, dt_i (Eq. 5) is the expected value of T_i (time spent on problem p_i) and is also normalized. This expected time is the mean value after removing outliers, or median.

dh_i (Eq. 6) is the expected value of H_i (number of attempts for problem p_i) divided by the maximum $E(H_j)$ registered.

2.4 Accuracy of Item Difficulty Estimations

We computed SPD estimates using a data set of 591 high school students who used Wayang Outpost tutoring software over past years, from 2003 until 2005. The tutors employed a variety of problem selectors during those years, with some percentage of students using a random problem selector.

Validating that student perceived difficulty estimates were reasonable seemed essential. The first reason is that the difficulties play a crucial role in the adaptive behavior of the tutor, and inappropriate difficulties would make the system behave in undesired ways (e.g. providing a harder problem when the student clearly needs an easier one). The second reason is that it is just too likely that the student perceived difficulty estimates are biased, because student behavior is contingent to the problem selector in place at the moment the data on problem performance was collected. Unless the raw data comes from a random selection of problems, student behavior and thus the data collected will be biased in some direction. This will make problems look easier or harder than they truly are.

We devised a variety of methods to assess the correctness of our estimation of perceived student difficulty, and implemented three of them. All of these are based on the following axiom: "*Pairs of Similar Problems Should have Similar Problem Difficulty Estimates"*. In other words, if two problems are very similar, the perceived differences in their difficulty should approach zero. We subsequently drew a subset of 60 mathematics problems (p_1 to p_{60}) from our tutoring system. These sixty problems are special because may be divided into 30 pairs of problems, where each p_i , with i=1…30, is extremely similar to p_{30+i}. In this domain of geometry problems, similar problems involved similar showing graphics with slightly different angles, or measurements. For example, same problems with a rotated figure (and different operands). Similar problems involve the application of the same skills the same amount of times. We call these *highly similar pairs* and now describe four criteria used to verify that these pairs are similar in their difficulty estimates.

2.4.1 Criteria 1: Correlations. We tested that such pairs had similar difficulty estimates with a simple Pearson correlation, which is the most familiar measure of dependence between two quantities. It is obtained by dividing the covariance of the two variables (d_{p_i} and $d_{p_{30+i}}$) by the product of their standard deviations. A Pearson correlation determined that pairs of problems were significantly correlated (N=30, p<0.000, R=.823), thus this test is passed.

2.4.2 Criteria 2: Mean Squared Error. Another criteria used was that the difference in objective difficulty between highly similar problems should be smaller than the difference in difficulty between either of these problems and any other problem in the

system that is not as similar –other problems will involve different skills, or different total amount of applications of the same skills. While it may be coincidental that a problem foreign to the pair might have a very similar difficulty to either problem in the pair, this should not be the general case.

The distance between the difficulty of a problem p_i and its highly similar problem pair p_{30+i} should be smaller than the mean distance between one of the problems in the pair and the remaining problems in the set. A more common jargon when talking about differences due to error is the mean squared error. Eq. 7 rephrases the above in terms of squared differences, where *N=total number of pairs=30*.

$$\left(d_{p_i} - d_{p_{30+i}}\right)^2 < \frac{\sum_{j=1, j\neq i}^{N} (d_{p_i} - d_{p_j})^2}{N} \qquad (7)$$

If we can show that this inequality holds in general for problems, we have some evidence that our system is doing a reasonable job at estimating difficulties. We computed the 30 square differences $\left(d_{p_i} - d_{p_{30+i}}\right)^2$, and their corresponding mean squared differences as specified in Eq. 7. The result was that the inequality holds for 29 of the thirty cases, which is a 97% success rate. A paired-samples t-test for the two inequality terms in Eq. 6 revealed that these two sides of Eq. 7 are significantly different t(29)=7.35, p<.000. The second test is then passed.

2.4.3 Criteria 3: Human Expertise. While pairs of highly similar problems should have similar student perceived difficulty levels, they don't necessarily need to have exactly the same difficulty (i.e. the difference in their difficulty levels will not be exactly zero. In other words, $\left(d_{p_i} - d_{p_{30+i}}\right)^2 = \varepsilon_i$, where ε_i is a small number. While it would be hard to determine the true value of epsilon for each problem pair, an expert human eye (e.g. a teacher or tutor) could probably make good predictions about whether $d_{p_i} > d_{p_{30+i}}$ or whether $d_{p_i} < d_{p_{30+i}}$. This kind of expert knowledge can help us establish that the latter problem should be harder for a student to solve than the former one. Other restrictions may have to do with operand size, involvement of decimals or negative numbers, or a small extra step. We managed to establish such restrictions for 21 of the 30 pairs of problems we considered, the other 9 were just too similar to each other. Such restrictions (true positives or true negatives) were correctly guessed in 14 of the 21 cases (67%), and a Chi-Square test revealed this is significantly better than chance (Pearson Chi-Square=5.25, p=.022). Thus, the third test is passed.

2.4.4 Criteria 4: Convergence. Ideally, the difference between highly similar pairs of problems would converge to ε_i as more data arrives to the logs, even if different problem selectors are in place at different moments. This test is still ongoing.

2.5 Evaluation of Effectiveness of Effort-Based Pedagogical Model

While we may be satisfied that difficulty of items are reasonably estimated, we need also to show that the adaptive mechanism underlying the pedagogical model makes a difference to student learning. A study was carried out in the 2003-2004 academic year with 60 students to evaluate the effectiveness of the adaptive sequencing of problems, compared to a random selection of problems within a topic (no learning companions or affective feedback).

Table 3. Pretest and Posttest Scores in Math Test

Adaptive/random		Pretest score SAT questions (correct/attempts)	Postest score SAT (correct/attempts)	Pretest correct SAT	Posttest correct SAT
Random problem selection	Mean	.3868	.4220	2.7576	3.3030
	N	33	33	33	33
	Std. Deviation	.25160	.19730	1.87133	1.51007
Adaptive problem selection	Mean	.4216	.5664	3.5217	4.9565
	N	23	23	23	23
	Std. Deviation	.23227	.19108	2.06419	1.77042
Total	Mean	.4011	.4813	3.0714	3.9821
	N	56	56	56	56
	Std. Deviation	.24230	.20590	1.97122	1.80395

Both the experimental and the control conditions implemented topic switching based on one parameter only, N_{KU}, so that the "topic switch" criterion was set to a fixed maximum number of problems per topic. This was established so that all students were exposed to the same number of problems in each topic. M_{KU}, F_{KU} and T_{KU} were then ignored. The main difference between conditions was the problem selection mechanism *within* the topic. For the experimental condition, it adjusted problem difficulty as described in previous sections, with the following parameters: $\gamma=2$; $\theta_{LOW}=0$; $\theta_{HIGH}=0$; this made the changes in problem difficulty quite marked. Control condition students received random problems within each topic.
Students were randomly assigned to either the Effort-based Adaptive Problem

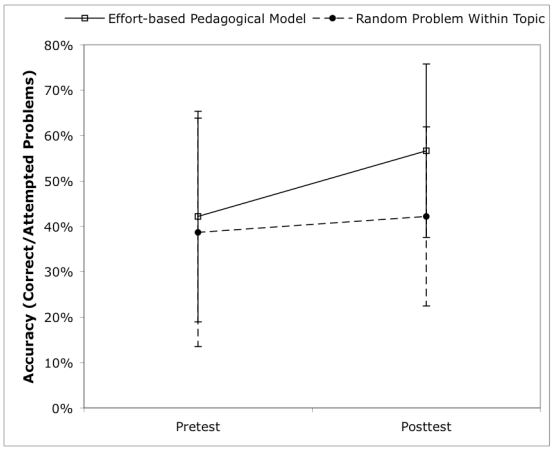

Figure 3. Pre to Posttest Improvement with Effort-based Pedagogical Model compared to a Random Problem Selector Within the Topic.

Selection condition, or the Random Problem Selection Condition. Students used the Wayang Tutoring System for 4 class periods, completing a 10-item math test before starting and a similar posttest the last day. The test consisted of items drawn from the SAT (Scholastic Aptitude Test) and released by the College Board. The two tests were counterbalanced –half of students received pretest A, and half pretest B, and the tests were reversed for students at posttest time.

We measured the total number of correct items achieved in the test, and the accuracy at items (correct/test items attempted) as a measure of performance, see Table 3. We obtained full pre and posttest data for 56 students, 23 in the experimental adaptive condition, and 33 in the control condition. Table 3 shows the mean and standard deviation of pretest and posttest scores for the pretest and the posttest. Mean achievement in the posttest increased and standard deviations reduced for both groups. However, mean improvement was higher for the experimental adaptive

problem selection group (Figure 3). This difference is significant (ANCOVA for posttest score with pretest score as a covariate, group effect F(55,1)=8.4, p=.006). The group receiving adaptive effort-based pedagogical decisions about problem difficulty improved more than did the group receiving random problem selection control condition. We conclude that adaptive problem selection is better than random.

3 Summary

This paper presented a novel approach to the development of smart learning environments, based on empirical measures of student effort at individual items. It described a pedagogical model that uses empirical estimates of problem difficulty, specifying parameters that regulate behavior within knowledge units (γ, θ_{LOW} and θ_{HIGH}) and between knowledge units ($M_{KU}, F_{KU}, T_{KU}, N_{KU}$). Knowledge Units may be defined at different levels of abstraction, thus addressing restrictions of content. This allows for replication in other ILEs, even in ill-defined domains or in small ILEs that are trying to encode smart decisions about practice items or activity selection.

We have described criteria for evaluating that estimates of problem difficulty are not too biased to the problem selector in place at the time of data collection. Last, we have shown that this effort-based pedagogical model leads to improved learning compared to uninformed random decisions within a topic or knowledge unit.

References

[1] Arroyo, I., Woolf, B.P., Royer, J.M., Tai, M.: Affective Gendered Learning Companions. *Proceedings of AIED conference*, 2009. p. 41-48. IOS Press.

[2] Baker, R.S.; Corbett, A.T.; Koedinger, K. Detecting Student Misuse of Intelligent Tutoring Systems. *Proceedings of ITS Conference.* LNCS 3220, 2004, p. 54-76

[3] Beck, J.E. Engagement tracing: using response times to model student disengagement. *Proceedings of AIED conference*, 2005. p. 88-95. IOS Press

[4] Corbett, A.T., Anderson, J.R., Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, 1995, 4, p.253-278.

[5] Johns, J. and Woolf, B.P. A Dynamic Mixture Model to Detect Student Motivation and Proficiency. *Proceedings of AAAI Conference,* 2006, 1, p. 163-168.

[6] Ferguson, K.; Arroyo, I., Mahadevan, S., Woolf, B.P., Barto, A. Improving Intelligent Tutoring Systems: Using Expectation Maximization To Learn Student Skill Levels. *Proceedings of ITS conference.* LNCS 4053, 2006. p. 453-462.

[7] Mayo, M., Mitrovic, A. Optimising ITS Behaviour with Bayesian Networks and Decision Theory. *International Journal of Artificial Intelligence in Education,* 2001, 12, p. 124-153.

[8] Woolf, B.P. , Arroyo, I., Woolf, B., Muldner, K., Burleson, W., Cooper, D., Dolan, B., L., The Effect of Motivational Learning Companions on Low-Achieving and Learning Disability Students. *Proceedings of ITS conference*, 2010. Springer.