

Extending Knowledge Tracing to allow Partial Credit: Using Continuous versus Binary Nodes

Yutao Wang and Neil Heffernan

Worcester Polytechnic Institute
yutaowang@wpi.edu and nth@wpi.edu

Abstract. Both knowledge tracing and performance factors analysis, are examples of student modeling frameworks commonly used in AIED systems (i.e., intelligent tutoring systems). Both use student correctness as a binary input, but student performance on a question might better be represented with a continuous value representing a type of partial credit. Intuitively, a student who has to make more attempts, and has to ask for more hints, deserves a score closer to zero, while students who asks for no hints and just needs to make a second attempt on a question should get a score close to one. In this work, we present a simple change to knowledge tracing and a simple (non-optimized) method for giving partial credit. We report on fit to real data comparing the original knowledge tracing (OKC) model with this new KT that uses partial credit (KTPC), which outperforms traditional knowledge tracing reliably. The practical implication of this work is that this new techniques can be widely used easily, as it's a small change from the traditional way of fitting KT models. But it also holds promise as a nice way to incorporate other features like response time, into a model. We discuss how future models could build out of this to make more effective student models.

Keywords: Knowledge Tracing, Intelligent Tutoring Systems, Student Responses, Partial Credit

1 Introduction

In many important student models, such as the Knowledge Tracing model and the Performance Factor Analysis (Pavlik, Cen and Koedinger 2009), student performance is presented as a binary value of correct or incorrect. The amount of assistance a student needed to eventually get a problem correct is ignored in these models. Feng and Heffernan (2010) showed that we can predict student performance better by accounting for amount of assistance they received, but Feng & Heffernan did not provide the field with a model that could be used as “run time” to predict individual responses. Arroyo, et al.(2010) showed how to use this information to predict learning gains. Their work suggests that using hints and attempts to model student behavior online could be effective.

There is good work in the psychometric literature on using types of partial credit, which goes back 30 years. Psychometricians have shown that different multiple

choice answer might be more worth of credit than others (for instance, choice A might be totally wrong but choice B is close, choice C is the correct answer). See Masters (1982) for the first such model and Tang (1996) for a review.

More recently a new type of partial credit is coming online, and this is one that replies of students having multiple attempts and feedback on the correctness of their answers. For instance, Attila and Powers (2010) at the Educational Testing Service showed they could better predict student GRE scores if they let student make multiple attempts. Their score on a question would go down by a third for each attempt (they could only make three attempts). Our work generalizes their work in two ways. First, we show how to incorporate it into a model with learning (i.e., knowledge tracing) as their model did not model learning. Second, we show how to incorporate penalties for each hint they request. The fact that ETS is investigating this is a good sign, as ETS has one of the contracts for the US PARCC test that will be used in 25 states in 2014-15 (parconline.org). Test that allows students to get feedback is a good step forward to reduce time wasted on testing.

In our previous work (Wang and Heffernan, 2010), we presented a naïve algorithm to assign partial credit, and showed it accounts for some variance in student knowledge. But in that work, we did not present a model that could do this task. In this paper we want to see if we can improve one of the dominant methods of student modeling (i.e., the Knowledge Tracing model) by relaxing the assumption of binary correctness: replacing the discrete performance node with continuous partial credit node.

In the next section, we describe our modification to the original knowledge tracing (OKT) model, and the method we use to make the correctness continuous. Section 3 contains the tutoring system and dataset used in our experiments and the experimental results. In Sections 5 and 6 we discuss our conclusions and future directions for our work.

2 Approach

2.1 Knowledge Tracing with Continuous Performance Node

The Knowledge Tracing model shown in Fig.2 has been widely used in ITS to model student knowledge and learning over time. It has become the dominant method of student modeling and many variants have been developed to improve its performance (Baker et al., 2010, Pardos and Heffernan 2010). Knowledge Tracing uses one latent and one observable dynamic Bayesian network to model student learning. As shown in Fig.2, four parameters are used for each skill, with two for student knowledge (initial knowledge and probability of learning the skill) and the other two for student performance (the probability of guessing correctly when the student doesn't know the skill and the probability of slipping when the student does know the skill).

The structure of the Knowledge Tracing model with a continuous performance node is the same as the original Knowledge Tracing model, with the only difference:

how we set up the “Student Performance” node. The idea is straight forward, yet there has never been positive result reported in this field. Some other Intelligent Tutoring System groups, such as LISTEN (<http://www.cs.cmu.edu/~listen/>) tried this approach before but failed for unknown reasons.

The trickier part in the implementation is that, instead of assign the “*Guess*” and “*Slip*” parameters in a CPT table, we assume instead we have two Gaussian distribution for “*Guess*” and “*Slip*” with given standard deviations. We use four parameters: *guess_mu*, *guess_sigma*, *slip_mu*, *slip_sigma*, to describe the two Gaussian distributions corresponding to the “*Guess*” and “*Slip*” situations.

Similarly, when we predict the performance node, we also get a Gaussian distribution with a mean and a standard deviation parameter, in which the mean value will be the prediction and the standard deviation contains the information of how good the prediction is. In this work, we are not utilizing the standard deviation of the prediction, but it has potential to be useful in the future to determine how confident we are toward our prediction. This can be seen as another benefit that caused by relaxing the assumption of binary correctness in the knowledge tracing model.

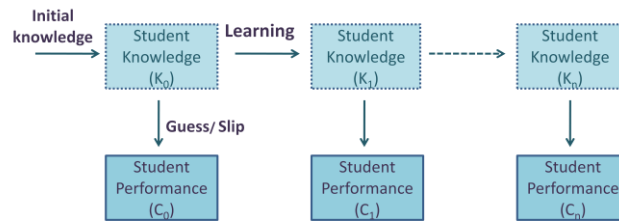


Fig. 1. Knowledge Tracing model

(Figure comes from Gong, Beck et al. 2010)

In our experiment, we used the Bayes Net Toolbox for Matlab developed by Murphy (2001) to implement Knowledge Tracing and the Expectation Maximization (EM) algorithm to fit the model to the dataset. The EM algorithm finds a set of parameters that maximizes the likelihood of the data. Since EM can be sensitive to initial conditions (Pardos & Heffernan 2010b) we report the initial settings. We used *initial knowledge* = 0.5, *learning* = 0.1, *guess_mu* = 0.1, *guess_sigma* = 0.02, *slip_mu* = 0.1, *slip_sigma* = 0.02 for the KTPC model, and *knowledge* = 0.5, *learning* = 0.1, *guess* = 0.1, *slip* = 0.1 for the OKT model.

2.2 Make the Correctness Continuous: Partial Credit

There are different ways of assign partial credit, some of them are nature. For example, most humans give partial credit as degrees of correctness for open ended tasks. In our experiment, we are using the algorithm that was mentioned in our previous poster [11] to make the correctness to be continuous. Since we never introduce the algorithm completely, it is described in this section in detail.

In a binary performance model, a student would get a ‘1’ if they got the problem correct on their attempt without asking for a hint. For the purpose of this paper we “made up” a scoring method that would give students a score between ‘0’ and ‘1’ according to how many attempts and hints were used. We are well aware that this could be optimized in lots of ways (should each hint cost the same, or should the first hints cost less?). As you will see this simple method is effective and we leave to others different ways to optimize this.

Intuitively, the more hints that are asked for, the less likely it is that the student understands the skill, so we penalize a student for each hint asked for by what we call the hint penalty, which is 1 divided by the total number of hints available. For example, if there are 4 hints possible and a student asks for three of them and then gets the problem correct they would get a .25 score. In a somewhat similar manner, more attempts indicate a lower possibility of understanding the required skill, and we penalize each attempt. The size of the penalty depends upon whether the questions are multiple choice or “fill in the blank” type. We have about 80% of our questions to be “Fill In The Blank” as they are mathematics questions. We picked a penalty for each attempt to be 0.1 (Attali & Powers used a penalty of 1/3 which might be better). For multiple choice questions with x choices, the penalty was given by one over the number of multiple choice options minus one. So a true false question will have a penalty of one if they guess wrong. If there were 4 choices, each time the student guess wrong the penalty would be 1/3.

After computing hint penalty (*phint*) for each hint and attempt penalty (*pattempt*) for each attempt, we add them together to compute the total hint penalty (*total_phint*) and the total attempt penalty (*total_pattempt*) for this problem. If the number is less than zero we make it zero. The last column of Table 2 shows two examples of formula doing this calculation.

Table 1 details the case for scaffolding questions. Our dataset has a special type of feedback called scaffolding that we also had to deal with. Since its only a small amount of our data this detail might not be that important. But for completeness, we wanted to describe this. (Please note that all of our code and our data set are available at <http://users.wpi.edu/~yutaowang/> so that others can attempt to replicate and improve upon our work.) For those problem sets with scaffolding questions, if a student gets the original question wrong, the system will give the student a series of questions we call “scaffolding” that walk the students through the steps. Each of those scaffolding question has hints and so can be scored with this partial credit function just like any question. The only question left is how to score the “original question.” If a student gets a question wrong and is given say three scaffolding questions, the value of the whole problem is a average partial credit score of the three scaffolding penalized by 10% (If a student got the original question wrong but then got all the scaffolding questions correct they should get something close to 1, which in this case would be exactly 0.9). Again these parameters like 0.9 are not optimized and could be learned from data.

Table 1. The whole algorithm of computing partial credit.

```
function pc = partial_credit(problem){  
    if first attempt correct then  
        return pc = 1  
    else if problem has no scaffold then  
        pc = 1 - #hint * phint - total_pattempt  
        if pc < 0 then return pc = 0  
        return pc  
    else  
        for each scaffold question i in the problem do  
            pc_scaffold(i) = partial_credit(scaffold(i))  
        end for  
        pc = 0.9 * average(pc_scaffold(i))  
        return pc  
    }  
}
```


The algorithm is used only for testing the effect of relaxing the assumption of binary correctness in a Knowledge Tracing model.

3 Evaluation

3.1 The Tutoring System and Dataset

Our dataset consisted of student responses from ASSISTments, a web based tutoring system for 7th-12th grade students that provides preparation for the state standardized test by using released math items from previous years' tests as questions. The tutorial helps the student learn the required knowledge by breaking the problem into sub questions called scaffolding or giving the student hints on how to solve the question. Fig.1. shows an example of a hint. A second type of assistance is presented if they click on (or type in) an incorrect answer, at which point the student is given feedback that they answered incorrectly (sometimes, but by no means always, students will get a context-sensitive message we call a "buggy message"). Examples can be seen at "tinyurl.com/buaesc2".

Triangles ABC and DEF shown below are congruent.



What is the perimeter of triangle ABC ? [Comment on this question](#)

Perimeter is defined as the sum of all sides of a figure. [Comment on this hint](#)

Show me hint 2 of 3

Select one:

$2x + 8$

$\frac{1}{2} * 8x$

$2x + x + 8$

$\frac{1}{2} * x(2x)$

Submit Answer

No. You might be thinking that the area is $\frac{1}{2}$ base times height, but you are looking for the perimeter.

A hint message

A buggy message

Fig. 2. Assistance in ASSISTment

The Data set was drawn from ASSISTments. These data were from 72 twelve-through fourteen-year old 8th grade students in an school district of the Northeast United States. There were 106 skills (e.g., area of polygon, Venn diagram, division, etc.) students were working on. The data we analyzed consisted of 52,529 log records during the period Jan 2009-Feb 2009 where each row is similar to Table 2, showing one row for each problem done by one student. Table 2 is meant to show that we use a commonly used data format. We use the same data format as the KDD in EDM (<https://pslcdatashop.web.cmu.edu/KDDCup/FAQ/#data-format>). In Table 2, we can see an example of the type of data we used. The first nine columns in the table are straight from (https://pslcdatashop.web.cmu.edu/KDDCup/rules_data_format.jsp), but we added three extra columns, which are used to calculate the partial credit. In particular, we add a column to help us determine if the problem is multi-choice or not, and how many choices there were. In a manner that is somewhat analogous, to compute the partial penalty per hint, it is important to know how many hints there are. The last hint always gives away the answer so if a student asked for all the hints their score should be zero. The second to last column in Table 1 shows we added the total number of hints that were available for that question, enabling us to give a bigger penalty for hints if the number of hints is small. The last column is to show how we compute the partial credit score that the student would get between 0 and 1. Note that original KT will only use the 7th column (named “Error rate” which is deceiving as its just 1 if they get it correct else zero) while the KTPC model will use only the 12th column.

Row	Student	Problem	Step	Incorrects	Hints	Error Rate	Knowledge component	Opportunity Count	If Multiple Choices the number of Choices	Number Of Hints Available	Partial Credit Score
1	1	WATERING_VEGGIES	(WATERED-AREA Q1)	0	0	0	Circle-Area	1	4 Choice Multiple Choice	2	1
2	1	WATERING_VEGGIES	(TOTAL-GARDEN Q1)	2	1	1	Rectangle-Area	1	Fill In The Blank	3	$1-2*0.1-1*1/3=0.46$

Table 2. A example of a few rows of data, showing how we calculate partial credit.

3.2 Results

To evaluate how well the new model fits the data, we used the Root Mean Squared Error (RMSE) to examine the predictive performance on an unseen test set. The lower the RMSE is, the better the prediction. We did not use cross fold validation but instead are predicting complete performance for unseen data. We stratified our data set by student. There were randomly 2,313 student data in the test set and randomly 3,297 students in the training set.

Table 3 shows the results of the comparison for the two different models, the original Knowledge Tracing model and the Knowledge Tracing with partial credit model.

We compared the RMSE for predicting the partial credit performance and for predicting the traditional binary performance respectively. The Knowledge Tracing with partial credit model has lower RMSE value in both situations. The real comparison is in the right column where you can see that OKT has higher RMSE than the KTPC in predicting binary performances. The lower left cell shows that of course, KTPC does a great job of predicting partial credit scores. The top left cell shows that OKT, can do some reasonable job of predicting partial credit scores.

Table 3. original KT (OKT) vs KT with partial credit (KTPC)

Model	Predicting Performance	
	Partial Credit	Binary Performance
OKT	0.4128	0.4637
KTPC	0.2824	0.4572

We determined whether the difference between these two models is statistically significant by computing the RMSE value for each student to account for the non-independence of their actions, and then compared these two models using a two tailed

paired t-test. To be precise, to compute student level RMSEs we calculated the average RMSE across all of a students' data points.

The p value of the RMSE between using the original Knowledge Tracing and the Knowledge Tracing with partial credit model to predict the partial credit is 0. The t-test p value between using the original Knowledge Tracing and the Knowledge Tracing with partial credit model to predict the binary performance is $p < .001$. The degree of freedom is 2,313 (since we are doing a student level t-test, the degree of freedom is the same as the number of students in the test set). Thus, the Knowledge Tracing with partial credit model is statistical reliably better at predicting student performance over the original Knowledge Tracing model.

4 CONCLUSIONS AND FUTURE WORK

In this paper, we extended Bayesian Network student modeling to include continuous performance node. The effectiveness is demonstrated by incorporating a partial credit algorithm that assigns continuous performance given detailed student responses. Experiment results show that relaxing the assumption of binary correctness can help improve predictions of student performance in student modeling. This also proves that our intuition based heuristic for partial credit might be broadly applicable.

One question we are interested in is to explore other partial credit schemes, for example, a method to refine the algorithm to better fit student data and more accurately infer student knowledge. Also, since we are observing some abnormal parameters in the performance parameters (guess/slip), we are interested in finding out why the parameters are so different compare to normal knowledge tracing, and what could be the plausible explanations.

5 CONTRIBUTIONS

In this paper, we explored the methodology that moving from binary performance to continuous which makes Intelligent Tutoring Systems more flexible. On one hand, we extended the Knowledge Tracing framework to include a continuous performance node to combine with all possible continuous performances such as essay scoring, speech recognition score. On the other hand, we presented an understandable and easy to refine algorithm to assign partial credit according to detailed student responses. This algorithm is one of many possible ways to convert more information than the binary performance into a continuous value.

In practical, we enhanced student model accuracy by improving upon the classic Knowledge Tracing model. The result we got shows statistical reliably improvement in predicting both students' partial credit performance and binary performance. Also, freely available code that actually gets positive results is shared online, which could be useful for researchers that are trying to do the same task since it is hard to get all the details correct.

6 Acknowledgements

This research was made possible by the U.S. Department of Education, Institute of Education Science (IES) grants #R305K03140 and #R305A070440, the Office of Naval Research grant # N00014-03-1-0221, NSF CAREER award to Neil Heffernan, and the Spencer Foundation. All the opinions, findings, and conclusions expressed in this article are those of the authors, and do not reflect the views of any of the funders.

7 REFERENCES

1. Arroyo, I., Cooper, D.G., Burleson, W., Woolf, B.P. Bayesian Networks and Linear Regression Models of Students' Goals, Moods, and Emotions. In: Handbook of educational data mining: pp. 323-338. Boca Raton, FL: CRC Press. (2010).
2. Attali, Y. & Powers, D. Immediate feedback and opportunity to revise answers to open-end questions. *Educational and Psychological Measures*, 70(1) 22-35. (2010)
3. Baker, R.S.J.d., Corbett, A.T., Gowda, S.M., Wagner, A.Z., MacLaren, B.M., Kauffman, L.R., Mitchell, A.P., Giguere, S. Contextual Slip and Prediction of Student Performance After Use of an Intelligent Tutor. In: Proceedings of the 18th Annual Conference on User Modeling, Adaptation, and Personalization. pp. 52-63.(2010).
4. Corbett, A., Anderson, J.: Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. *User Modeling and User-Adapted Interaction* 4:253-278. (1995)
5. Feng, M., Heffernan, N. T. Can We Get Better Assessment From A Tutoring System Compared to Traditional Paper Testing? Can We Have Our Cake (Better Assessment) And Eat It too (Student Learning During The Test)? The 10th International Conference on Intelligent Tutoring Systems, Pittsburgh, PA. (2010).
6. Masters, G. N. (1982). A rasch model for partial credit scoring. *Psychometrika*, 47, 149-174.
7. Pardos, Z.A., Heffernan, N.T.: Modeling Individualization in a Bayesian Networks Implementation of Knowledge Tracing. In: Proceedings of the 18th International Conference on User Modeling, Adaptation and Personalization. pp. 225-266. (2010a).
8. Pardos, Z.A., Heffernan, N.T. Navigating the parameter space of Bayesian Knowledge Tracing models: Visualization of the convergence of the Expectation Maximization algorithm. In Proceedings of the 3rd International Conference on EDM. (2010b)
9. Pavlik, P.I., Cen, H., Koedinger, K. Performance Factors Analysis – A New Alternative to Knowledge. In: Proceedings of the 14th International Conference on Artificial Intelligence in Education. pp. 531-538. (2009).
10. Tang, K. L. (1996) Polytomous item response theory (IRT) models and their applications in large-scale testing problems: Review of the literature. Educational Testing Service Technical Report www.ets.org/Media/Research/pdf/RM-96-08.pdf
11. Wang, Y., Heffernan, N.T. , Beck, J.E.. Representing Student Performance with Partial Credit. In: Proceedings of the 3rd International Conference on Educational Data Mining. Pittsburgh, PA. (2010).