Making Educational Recommendations Transparent through a Fine-Grained Open Learner Model

Jordan Barria-Pineda
University of Pittsburgh
Pittsburgh, PA, USA
jab464@pitt.edu

Peter Brusilovsky
University of Pittsburgh
Pittsburgh, PA, USA
peterb@pitt.edu

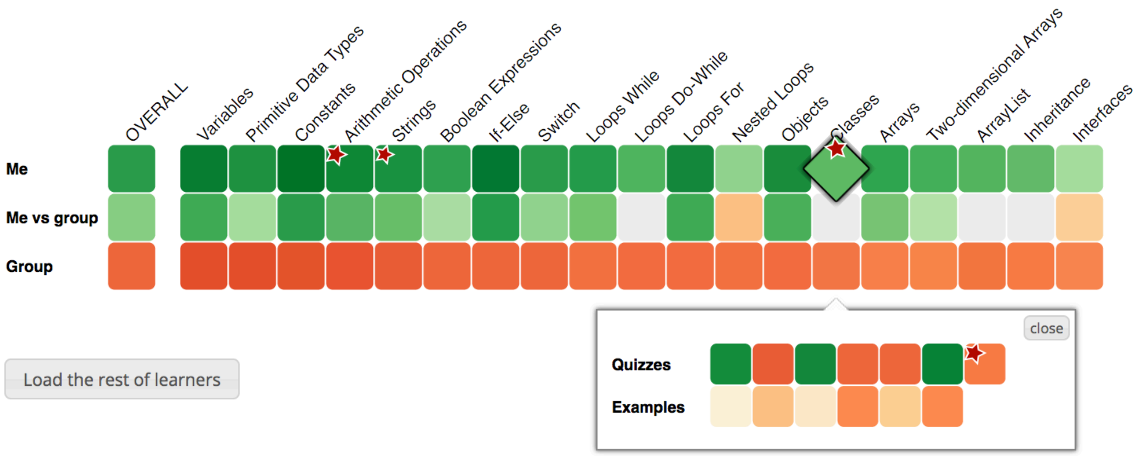

Figure 1: Social navigation support and recommendations in the context of Mastery Grids' OLM interface, a cell with a star symbol represents a recommended item

ABSTRACT

Recommendations for online educational systems generally differ from recommendations generated in other contexts (e.g. movies, e-commerce), given that students' level of knowledge rather then their interests is key for suggesting the most appropriate content. Thus, the challenge of making recommendations more transparent is closely tied to how student skills are estimated and conveyed. In this paper, we present an approach based on Open Learner Model visualization as a first step for making the learning content recommendation process more transparent. A preliminary analysis of students who used the visualization for navigating the content of an introductory programming course showed that considerable time was spent exploring the explanatory interface, which could be linked to the significant likelihood of opening/attempting the recommended activities.

CCS CONCEPTS

• **Applied computing** → **Interactive learning environments**; • **Human-centered computing** → *Information visualization*.

KEYWORDS

Educational Recommender Systems; Transparency; Open Learner Models

IUI Workshops'19, March 20, 2019, Los Angeles, USA

ACM Reference Format:
Jordan Barria-Pineda and Peter Brusilovsky. 2019. Making Educational Recommendations Transparent through a Fine-Grained Open Learner Model. In *Joint Proceedings of the ACM IUI 2019 Workshops, Los Angeles, USA, March 20, 2019* , 5 pages.

1 INTRODUCTION

Over the past few years, the research on *Explanations for Recommender Systems* attracted attention of many researchers along with the broader trend of *Explainable AI/Machine Learning*. These efforts aim on helping recommender system users understand why a specific item or a certain decision is being recommended. Explanations have been studied in many contexts, like e-commerce, people, and location recommender systems [9]. However, little work has been done in the context of online educational systems, i.e., exploring how explanations can benefit or hinder the adoption of recommendations in learning scenarios. In fact, [7] argues that explainability is one of the challenges for educational recommender systems and points out to information visualizations as a possible way to address this issue.

2 EXPLANATIONS AND KNOWLEDGE VISUALIZATION IN ONLINE EDUCATIONAL SCENARIOS

There is a small body of research on how explanations in recommender systems for learning can improve factors related to student engagement with recommendations, such as persuasiveness, learning efficiency, satisfaction, etc. [8].

On the other hand, there is a solid body of work on Open Learner Models (*OLMs*) focused on visualizing student knowledge [3]. In particular, in our earlier work [1] we explored a fine-grained visualization of student knowledge, which reflected the distribution of knowledge gained on every programming concept associated with every learning activity in the platform. This visualization helped students to understand their knowledge on a deeper level [2]. The work presented below attempts to fill that gap between OLM and educational recommendations. We argue that OLM interfaces could be used to explain learning content recommendations when they are generated based on student level of knowledge of the domain.

3 NAVIGATION SUPPORT AND CONTENT RECOMMENDATION IN MASTERY GRIDS

Mastery Grids is an intelligent interface which offers access to different kinds of practice content for introductory programming courses. To help students in accessing most relevant content, it offers provides several kinds of navigation support as well as direct recommendation. Figure 1 shows a a version of Mastery Grids for a Java programming course reviewed in [6]. The system organizes course contents into topics, displayed as columns of the grid. The first row shows topic-by-topic knowledge progress of the current student by using green colors of different density, the darker the higher the progress. This is, technically, a topic-level OLM of student Java knowledge. The third row shows the aggregated progress of the rest of the students of the class in shades of orange. The second row presents a differential color comparing the students progress and the class progress. For example, in Figure 1 the student has a higher progress than the class in most of the topics where the cells in the second row are green, but the class is more advanced in two of the topics (13th and 20th column) where the cells in the second row are orange. The student has same progress as the class in four topics with light gray color (11th, 15th, 18th, and 19th column). By clicking in cells, the student can access learning content for each topic. For example, in Figure 1, the student has clicked the topic *Classes* and the system displays cells to access questions and examples related with this topic. Note that the social and the comparison rows could be hidden to help students focusing on their own knowledge.

By presenting student's own knowledge, group knowledge, and their comparison, the system offers several kinds of navigation support, which could help students find most appropriate content for different kinds of learning goals. For example, personal part of OLM could help focusing on least learned topics, group model could help in locating "safe" topics that already mastered by a good part of the class, while the comparison could help to focus on the knowledge gaps. To augment this kind of navigation support, we also explored several personalized recommendation approaches. The older version of our recommendation interface shown in Figure 1 selects top three recommended content items at each given moment and displays their presence in the topic using red stars that appear on both, recommended items and their containing topics. The size of the stars shows the position of the recommended items in the top − 3 list. This presentation of recommended items is consistent with the navigation support nature of the interface: it does not force students to go to the recommended content, but informs the students and helps them to make their next navigational step. The resulting interface combines the social guidance of social OLM with the personal guidance provided by recommendation algorithms. Yet, directly recommended content differs from the navigation support provided by the OLM by the total lack of communicated reasons behind recommendation. While the low or high level of individual of social topic knowledge could be easily traced down to extensive or low work with topic content (clearly visualized in the content browser when the topic is opened), the system offered no hints on why a specific content item is recommended. In this paper we present an interface that attempts to address this problem by connecting recommended content with a finer-graned picture of student knowledge offered by a concept-level OLM.

4 ENABLING LEARNING CONTENT RECOMMENDATION TRANSPARENCY THROUGH A FINE-GRAINED OLM

The design of visual explanation of content recommendation is based on our earlier work on concept-level OLM [2]. This work explored the role of finer-grained OLM on student motivation and navigation support. Our visualization allowed students to see the overall level of their knowledge concept-by concept as well as to see concepts associated with each learning activity by mousing over this activity cell (see Figure 2B).

Given the recent interest in using visual interfaces to making recommendation processes more transparent to users, it was natural to explore the use of OLMs as an interface that could add transparency to educational recommendation. In this paper we show our first attempt to use concept-level knowledge visualization to explain the choices made by the learning content recommendation engine in order to make the reason behind these recommendations more clear to the users.

4.1 The Visual Explanation Interface

The main features of our visual explanation interface are:

(1) *Concepts mastery bar chart*: as it can be seen in Figure 2C, the estimation of the student mastery of domain concepts is shown through a simple bar chart. In order to emphasize when the student model is more or less confident about the student mastery on a concept, we use 50% as the zero of the y-axis, as with this percentage the model is not sure about student mastery or lack of it, and also the model probabilities get initialized with this values in the *cold-start* scenario (no evidence of students' activity). Accordingly, whenever the student shows evidence that s/he is learning a concept the mastery percentage increases above this base probability and hence the corresponding concept bar increases it length towards the positive y-axis. In contrast, if the learner starts failing i.e. giving evidence that s/he is having troubles in learning a specific concept, the estimated mastery probability decreases below the base value and we reflect this through an increase in the corresponding concept bar length towards the negative part of the y-axis. We encode the bars' color following the same rule: when the mastery probability is above 50%, we use green and it gets more intense when

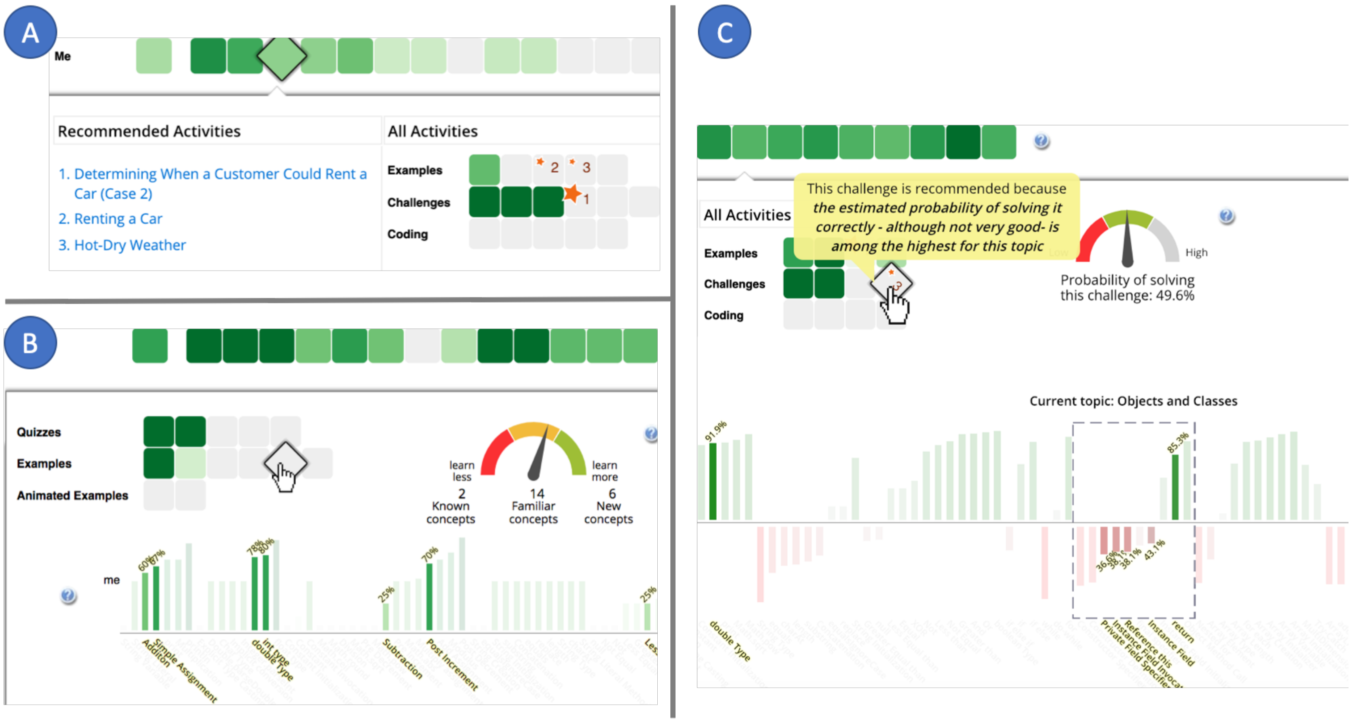

Figure 2: Different versions of Mastery Grids interface, going from less to more transparent. A: Mastery Grids interface showing recommendations as star icons at activity level | B. Mastery Grids interface with a concept-based knowledge visualization which shows a summary of the conceptual composition of an activity when mouseovered | C. Mastery Grids interface with recommendations plus a concept-based knowledge visualization and textual explanations for understanding why an activity was recommended.

closer to 100%, whereas when below 50% we use red and it gets more intense when it is closer to 0%.

Further, in order to give more context about the concepts that the student should set as her/his study goal, the "focus concepts" for the current topic are highlighted with a dashed frame (see C in Figure 2). It is important to mention that this visualization component can be used regardless the student modeling approach used for estimating student knowledge level, as it only uses the mastery estimates' values.

(2) *Recommendation gauge*: The score that represent the suitability of a certain learning content given its conceptual composition is shown through a gauge. When a learning activity is *mouseovered*, one of three gauge segments will be targeted by the needle (see C in Figure 2), according to its appropriateness to her/his level of knowledge. The three categories are the following: (1) *Too hard*: if the estimated probability of a successful attempt is too low (red segment), (2) *Learning opportunity*: activities in which some of the concepts are not mastered yet, but some important ones are mastered and can help on increasing student learning (green segment), and (3) *Too easy*: content that will not report any important learning increase, given that the underlying concepts are already mastered (gray segment).

(3) *Textual explanation*: a textual explanation of the recommendation rule that was triggered for the recommended item is shown when the activity cell is *mouseovered* (see C in Figure 2). We detail the rule-based recommendation algorithm used in the present study on the next section.

4.2 Recommendation approach

For this study, we used a rule-based recommendation algorithm based on the current level of knowledge of the student, which is updated every time an activity is attempted [4]. According to the correctness of each attempt, the nodes' values of the Bayesian network that represent the student model are recomputed (increased or decreased). These nodes reflect the probability of mastering each fine-grained concept, and also the probability of solving a problem correctly or understanding an example. These last probability values are considered as appropriateness scores for each activity; if the value is above 0.7 it is considered as a good candidate for being recommended.

Now, it is important to mention that examples and challenges (parsons-like activities) were created in groups that share the same learning goals [5]. Given this fact, the rule-based recommendation algorithm gives maximum priority to recommend a specific challenge whenever an related example was explored - regardless of its appropriateness score -. If the last is not the case, given the appropriateness score, the coding problems, non-related challenges and examples with higher scores (in that order of priority) are suggested up to complete a set of three recommended activities per topic. The whole rule set is described in more details in [4].

As stated in the previous section, the rule that triggered one of the top three recommended items is shown when the activity cell is mouseovered in the interface.

5 PRELIMINARY RESULTS

We released this version of Mastery Grids with visual/textual elements for making learning activities' recommendations more transparent in an intermediate Java programming course at the University of Pittsburgh (Fall term, 2018). In order to motivate students to use this non-mandatory practice system during the term, we offered extra-credit for completing a minimum activity threshold (7 coding problems, 5 parsons-like problems and 3 program examples' explorations). Half of students had access to textual explanations and half did not. Only 36 students out of 105 that had access to this interface version fulfilled the extra-credit requirement (13 with access to textual explanations and 23 without). This subset was used for the analysis of students' behavior on the system. We focused this general analysis on students navigation within the system and the likelihood of opening/attempting the learning activities' recommendations.

From the navigational side, we calculated the proportion of time that students spent using the Mastery Grids interface, i.e. not solving problems or reading program examples. In average, students explored the interface components in a 49.4% of the time (SD=13.2%). This shows that students used almost half of their time in the platform exploring the Open Learning Model components, which could be a sign that they took time for understanding why the recommended content was suggested to them at every moment.

In order to study the influence of the recommendations showed in the system, we explore if there were differences between the attempts on recommended and not recommended activities for the whole group of students. The first metric we computed was the probability of opening a mouseovered activity ($p\_open\_mouseover$), calculated as the number of opened activities divided by the number of mouseovers on the Mastery Grids activity cells. A paired Wilcoxon Signed Rank test evidenced that $p\_open\_mouseover$ was significantly higher (V=552, p<.01) for recommended activities (Mdn = .171) than for the ones that were not recommended (Mdn=.127). For details, see Figure 3.

Furthermore, as students sometimes open an activity but they close it after feeling is not the right activity to be attempted, we decided to compute the probability of attempting an opened activity as a second metric for measuring recommendations' influence ($p\_attempt\_open$). This probability was computed as the proportion of activities that were attempted divided by the number of activities that were opened. A paired Wilcoxon Signed Rank test evidenced that $p\_attempt\_open$ was significantly higher (V=224 , p<.05) for recommended activities (Mdn=.941) than for non-recommended ones (Mdn = 0.839).

Finally, we explored deeper differences between students with and without textual explanations (i.e. more and less transparency).

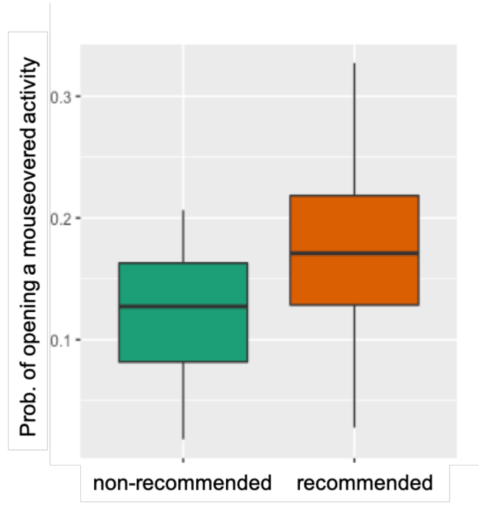

Figure 3: Difference between recommended and not recommended activities for the probability of clicking a mouseovered activity.

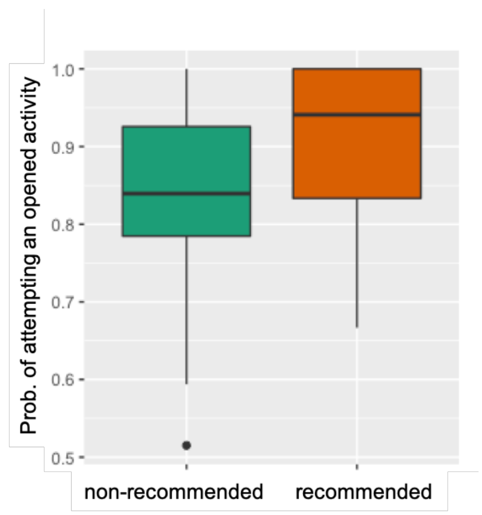

Figure 4: Difference between recommended and not recommended activities for the probability of attempt an opened activity.

	p_open_mouseover			*p_attempt_open*		
Textual exp	*rec*	*non_rec*	*p*	*rec*	*non_rec*	*p*
Yes	.188	.142	.094 .	.882	.789	.013 *
No	.167	.107	.006 **	.944	.881	.314

Table 1: Differences in recommendations' influence between students with and without access to textual explanations (. p<.1, * p<.05, ** p<.01). Values represent medians.

We focused this analysis on the likelihood of opening/attempting recommended activities. After performing four paired Wilcoxon

Signed Rank tests (see Table 1), we found that the general trend of having a higher probability of opening a recommended (*rec*) activity than a non-recommended (*non_rec*) one is still significant (*p_open_mouseover*), but marginal for students with textual explanations ($p<.1$). On the other hand, only the group with textual explanations exhibited significantly higher probability of attempting an opened activity when this is recommended rather than non-recommended (*p_attempt_open*). This result suggests that including textual explanations seems to be related to a higher students' confidence about the appropriateness of the activity, which could be triggering more attempts. It is important to mention that we need to be careful in interpreting this set of results given the low differences in medians (.05) and the unbalanced number of students on each subgroup.

6 CONCLUSION

In this paper we proposed the use of a fine-grained Open Learner Model for supporting the understanding of how a learning content recommender engine works. In this way, the recommendation process became partially more transparent to the students, as it was made visible by showing estimations of students' concept-level knowledge (recommender's input) and part of the recommender rules (recommender's algorithm).

After releasing the system for testing it in a real introductory programming class, we found that transparent recommended activities by the system seemed to have an influence, as the probability of opening and attempting it and further, attempting it when opened, was significantly higher than non-recommended activities. Moreover, from this study can be inferred that adding transparency for explaining the outcome of the recommendation could lead to a higher confidence in attempting the activities that are recommended, however, a deeper data including students opinions should be collected to be sure about this claim.

7 FUTURE WORK

We plan to evaluate this interface in a controlled user study by using an eye-tracking setup, in order to study how students explore and make use of the different explanatory components for making their decisions on attempting activities - as otherwise it is very difficult to obtain this information -. Also, we plan to gather students thoughts about the value of adding transparency to an educational recommender system, with the aim of studying if the benefit of making the system more transparent surpasses the cost of increasing its understanding's complexity.

Additionally, we are working on analyzing previous students' activity data in order to define a more "data-driven" set of rules for the recommendation algorithm instead of the ad-hoc approach that we used for the setup of this study, which can open other visualization setups for making the recommendations transparent.

ACKNOWLEDGMENTS

This work was funded by CONICYT PFCHA/Doctorado Becas Chile/2018 - 72190680.

REFERENCES

[1] Jordan Barria-Pineda, Julio Guerra, Yun Huang, and Peter Brusilovsky. 2017. Concept-Level Knowledge Visualization For Supporting Self-Regulated Learning. In *Proceedings of the 22nd Int. Conference on Intelligent User Interfaces (IUI '17 Companion)*. 141–144.

[2] Jordan Barria-Pineda, Julio Guerra-Hollstein, and Peter Brusilovsky. 2018. A Fine-Grained Open Learner Model for an Introductory Programming Course. In *Proceedings of the 26th Conference on User Modeling, Adaptation and Personalization (UMAP '18)*. 53–61.

[3] Susan Bull and Judy Kay. 2007. Student Models That Invite the Learner In: The SMILI:() Open Learner Modelling Framework. *Int. Journal of Artificial Intelligence in Education* 17, 2 (2007), 89–120.

[4] Roya Hosseini. 2018. *Program Construction Examples in Computer Science Education: From Static to Adaptive Engaging Learning Technology.* Ph.D. Dissertation. University of Pittsburgh.

[5] Roya Hosseini, Kamil Akhuseyinoglu, Andrew Petersen, Christian D. Schunn, and Peter Brusilovsky. 2018. PCEX: Interactive Program Construction Examples for Learning Programming. In *Proceedings of the 18th Koli Calling International Conference on Computing Education Research (Koli Calling '18)*.

[6] Roya Hosseini, I-Han Hsiao, Julio Guerra, and Peter Brusilovsky. 2015. What Should I Do Next? Adaptive Sequencing in the Context of Open Social Student Modeling. In *Design for Teaching and Learning in a Networked World*, Gráinne Conole, Tomaž Klobučar, Christoph Rensing, Johannes Konert, and Elise Lavoué (Eds.). Springer International Publishing, Cham, 155–168.

[7] Nikos Manouselis, Hendrik Drachsler, Katrien Verbert, and Erik Duval. 2012. *Recommender Systems for Learning.* Springer Publishing Company, Incorporated.

[8] Nava Tintarev and Judith Masthoff. 2011. Designing and Evaluating Explanations for Recommender Systems. In *Recommender Systems Handbook*, Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor (Eds.). Springer US, Boston, MA, 479–510.

[9] Yongfeng Zhang and Xu Chen. 2018. Explainable Recommendation: A Survey and New Perspectives. *Foundations and Trends in Information Retrieval* (2018).