







Involving Teachers in the Data-driven Improvement of Intelligent Tutors: A Prototyping Study

Meng Xia¹ , Xinyi Zhao² , Dong Sun³ , Yun Huang¹ , Jonathan Sewall¹ , and Vincent Alevan¹ 

¹ Carnegie Mellon University, Pittsburgh, USA
mengxia@andrew.cmu.edu, yunhuanghci@cmu.edu, sewall, va0e@andrew.cmu.edu

² Fudan University, Shanghai, China
xinyizhao19@fudan.edu.cn

³ Nio Inc., Shanghai, China
sundongcandy@gmail.com

Abstract. Several studies show that log data analysis can lead to effective redesign of intelligent tutoring systems (ITSs). However, teachers are seldom included in the data-driven redesign of ITS, despite their pedagogical content knowledge. Examining teachers’ possible contributions is valuable. To investigate what contributions teachers might make and whether (and how) data would be useful, we first built an interactive prototype tool for visualizing student log data, SolutionVis, based on needs identified in interviews with tutor authors. SolutionVis presents students’ problem-solving processes with an intelligent tutor, including meta-cognitive aspects (e.g., hint requests). We then conducted a within-subjects user study with eight teachers to compare teachers’ redesign suggestions obtained in three conditions: a baseline “no data” condition (where teachers examined just the tutor itself) and two “with data” conditions in which teachers worked with SolutionVis and with a list representation of student solutions, respectively. The results showed that teachers generated useful redesign ideas in all three conditions, that they viewed the availability of data (in both formats) as helpful and enabled them to generate a wider range of redesign suggestions, specifically with respect to hint design and feedback on gaming-the-system behaviors and struggle. The current work suggests potential benefits and ways of involving teachers in the data-driven improvement of ITSs.

Keywords: Intelligent tutoring system · Data-driven redesign · Students’ log visualization.

1 Introduction

Data-driven optimization of intelligent tutoring systems (ITSs) using design insights derived from student learning data and log data can substantially improve student learning outcomes [14]. Existing work has largely consisted of researchers’ analyzing and refining the system’s knowledge component model and

then redesigning the system to optimize the learning of the revised knowledge components [9].

In this paper, we present a first exploration of whether and how teachers might be involved in the data-driven redesign of tutoring systems. Teachers have often been involved in the initial design and development of tutors [11,17] as well as analytics dashboards for use with ITSs [8,7]. However, we are not aware of any projects in which teachers were involved in data-driven redesign of tutors, although in one instance they successfully used data-driven methods to adapt the text of course materials for teaching English to non-native speakers [10]. In another project, teachers redesigned hints for an ITS, but without using data [25]. Beyond teachers’ proven role on tutor design teams, there is reason to think that teachers could make valuable contributions to the data-driven refinement of tutoring systems. Teachers’ rich pedagogical content knowledge and practical experience in giving effective feedback might put them in a great position to suggest improvements to an intelligent tutor. In particular, data from student problem solving with the given ITS might jog teachers’ memory of what happened in the classroom, and reveal trends or events that they were not aware of. It is worth investigating what contributions teachers might make to the data-driven redesign process, given their unique sources of knowledge, and what tools would be useful in this regard, given their unique sources of knowledge.

Visual analytics have shown potential in summarizing and presenting problem-solving processes to enhance understanding of domain competencies or demands. The CTAT Behavior Graph visualizes pre-defined solutions within a given problem [19]. DataShop’s widely used “learning curves” are line charts depicting students’ mastery of targeted knowledge components over time. [13]. Later studies [16,20,23,24] visualized the problem-solving behaviors of groups of students to find common and distinct strategies. For example, Xia et al. [23] proposed QLens, a glyph-based Sankey diagram, to show how a group of students solve drag-and-drop problems step by step, allowing question designers to easily identify common difficult situations encountered. However, existing work misses detail in the visualization of student actions (e.g., hint requests) and has not investigated how to help teachers generate ideas for improving a tutoring system.

To this end, we first conducted a needs-finding study through semi-structured interviews with five intelligent tutor researchers and two school math teachers. From these interviews, we derived initial design requirements for how to present students’ multi-step problem-solving processes to teachers in a way that might spur ideas for redesign. From these requirements, we then developed an interactive visualization interface prototype, SolutionVis. The tool presents students’ problem-solving processes, including meta-cognitive aspects (e.g., hint requests). Finally, we conducted a within-subjects user study with another eight math teachers to explore the following research questions: RQ1. What kinds of redesign suggestions do teachers generate? RQ2. Does giving them a representation of students’ interaction data help them generate redesign ideas? RQ3. To help teachers generate redesign suggestions, how does an interactive visualization of the data compare to presenting it in a standard list format?

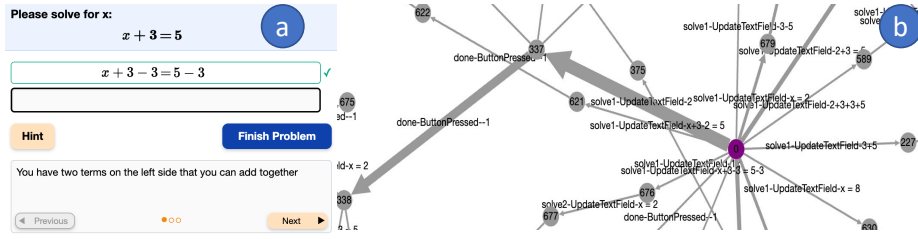


Fig. 1. (a) An example problem $x+3=5$ of the intelligent tutor. (b) The node-link diagram of students’ log data from the intelligent tutor with the problem: $x+3=5$.

2 Needs-finding Study

We conducted semi-structured interviews with five ITS researchers and two school math teachers, with more than 10 years of research or teaching experience on average. The purpose was to derive initial design requirements for how to visualize the data of a tutoring system in a way that would help teachers understand students’ interactions with that tutoring system and generate ideas for improvements of that tutoring system. Having a mix of experienced authors and teachers gives us a wide variety of perspectives.

Data Used and Procedure. We selected Lynnette, a linear-equation-solving tutor [15] for our investigation because it had key features that are characteristic of an ITS [4,21] and a reasonable level of complexity. Specifically, it supports problems with multiple steps, multiple solution paths within each problem, and step-level guidance in the form of hints and feedback. Its design was based on instructional design research [12], but this unit had not been data-tuned. As shown in Figure 1 (a), we used the problem $x + 3 = 5$ as an example for the interview. We used 27 students’ log data (records of individual student interactions with the tutoring software) from DataShop [13]. Each record includes the timestamp, the action (e.g., button pressed, text filled), and the evaluation results (e.g., correct or incorrect) from the intelligent tutor.

The interview lasted one hour for each participant. We presented a visualization of students’ log data in the form of a lo-fi node-link diagram, derived from existing work [16], to learn how we could improve from that. See Figure 1 (b) above. The graph shows the solution paths (including errors and hint requests) of all 27 students. Each node in the graph represents a state in the problem-solving process; and each link represents a student’s action. The number in the node shows the index of the transactions in the log data. Wider links indicate more students taking that action. The link label “solve1-updateText-2” means that the student answered “2” at the first step (indicated by 1 in solve1). We asked questions such as “*What information can you find from the visualization that can be used to improve the tutor design?*”

Initial Design requirements. After the interviews, three authors analyzed the interview transcripts. The author who conducted the interviews extracted all suggestions for improving the visualization to gain insights for the tutor

redesign from the transcripts, and all three authors worked together to derive the following initial design requirements using an affinity diagram [6].

- *R1: Make long paths easier to follow.* Participants mentioned that the initial radial layout makes it hard to identify long sequences and align different paths. They preferred a horizontal layout like a train station map.
- *R2: Make it easier to identify paths that capture similar problem-solving strategies by merging steps of paths.* Participants mentioned different ways to merge, such as, merging the same steps in different paths and collapsing steps from paths into stages (e.g., performing distribution of “()”).
- *R3: Make it easier to identify common problematic steps using visual cues.* Participants did not find it easy to distinguish correct and incorrect steps. They mentioned it might be important to pay attention to common incorrect steps for diagnosing problems in the tutor and suggested using different colors to encode the edges and highlight the problematic steps.
- *R4: Make it easier to understand students’ metacognition by showing hint request behaviors.* Participants emphasized that students’ hint requests could reflect hint quality and students’ metacognition.
- *R5: Make it easier to understand the problem-solving context by showing the tutor and a list of steps in a solution path.* Participants wanted to know what problems the students are solving and also a detailed list of steps, perhaps vertically, could contribute to a better understanding.

3 SolutionVis

To address all the requirements, we developed a visualization interface prototype, SolutionVis, to present students’ log data for teachers to explore. We used Python to process the log data and save each problem’s data as a JSON file. We then used Cytoscape.js [2] to visualize the node-link digram of the selected problem. SolutionVis consists of three views: Tutor View, Student Path View, and Sequence View.

The Tutor View, shown in Figure 2(A), displays the “live” tutor so that teachers can interact with it and understand the problem-solving context (see requirement R5, above).

The Student Path View, shown in Figure 2(B), is designed for teachers to understand how all the students (as a group) proceeded step by step. The steps in the paths are positioned from left to right for easy reading-of long paths especially (R1). This view supports zooming and panning. By default, it shows a fully-zoomed-out overview as shown in Figure 3 below. The user zooms in to show details, as in Figure 2(B). The nodes represent states on students’ paths to a solution; the links represent students’ actions. **Links:** The color of a link is determined by the evaluation of the student’s action—green for correct input, red for incorrect input, and yellow for hint requests. We merged duplicate actions and used the thickness of the link to represent how many times students repeated the step (R2, R3). **Nodes:** The nodes in the graph represent the intermediate states on students’ solution paths. The number in the node represents how many

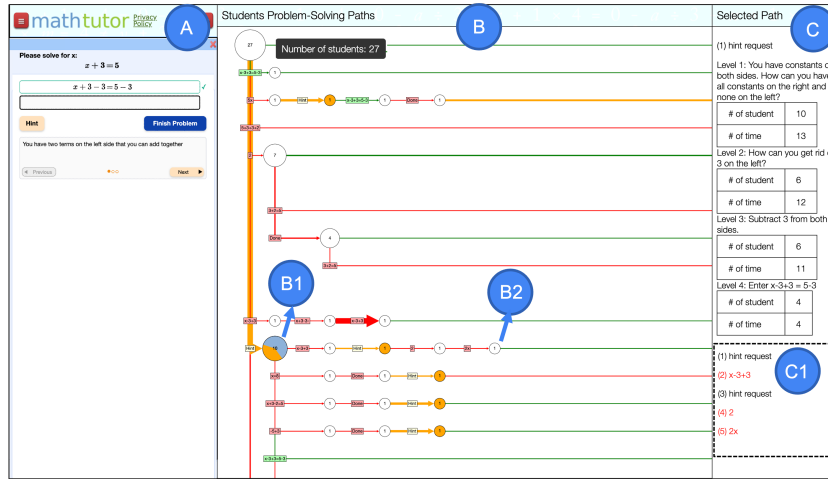


Fig. 2. SolutionVis: The Tutor View (A) shows the original intelligent tutor for teachers to interact with; The Student Path View (B) shows students’ aggregated problem-solving paths, and the Sequence View (C) lists, at the user’s request, a solution path or hints. Specifically, when the user clicks a step node is clicked (e.g., B2), the detailed steps will be listed (C1). When the user clicks a hint node (e.g., B1), apart from the steps, the details of the hint node will be shown (C).

students performed the same previous actions while the first node shows the total number of students. We use a pie chart to represent the ratio of students who asked for the bottom-out hint at that stage (R4). For example, as shown in Figure 2(B1), the pie chart shows that 4 out of 10 students asked for the bottom-out hint at the first step. By clicking the pie chart, the user can see the text shown at each hint level and the number of students asking for that level. **Layout:** The x-axis position of the node represents which stage the student is in, in an effort to address R2. Solving basic linear equations includes four stages: removing parentheses, adding/subtracting terms, combining like terms, and dividing both sides of the equation by the coefficient of the variable term. As shown in Figure 3, SolutionVis classified each node (state) along a solution path into different stages according to students’ attempts. (The example omits the first stage, removing parentheses, since there are no parentheses in this problem.) Moreover, when students do not follow the given order of the four stages, SolutionVis shifts the node a fixed distance down along the y-axis (see the inset in Figure 3).

The Sequence View shows detailed information about a selected path that one or more students took. When the user clicks on a node in the graph, the sequence view will display the steps from the start up to the selected node. For example, if the user clicks Figure 2(B2), the path will be shown as Figure 2(C1). In addition, if the node clicked is a hint request (e.g., Figure 2(B1)), the levels of hints requested will be listed in the Sequence View (e.g., Figure 2(C)).

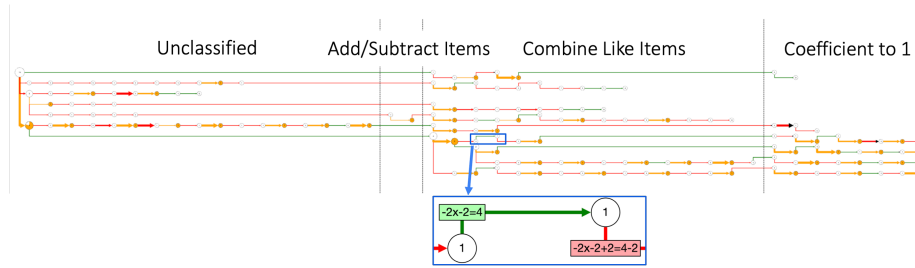


Fig. 3. The layout of the Student Path View in SolutionVis with the stages (at the top) for solving “ $-4x+1-3+2x=4$ ”. The enlarged part shows that the student input “ $-2x-2=4$ ” belongs to the stage “Combine Like Terms”, followed by “ $-2x-2+2=4-2$ ”, which does not belong to “Combine Like Terms” and is therefore shifted down.

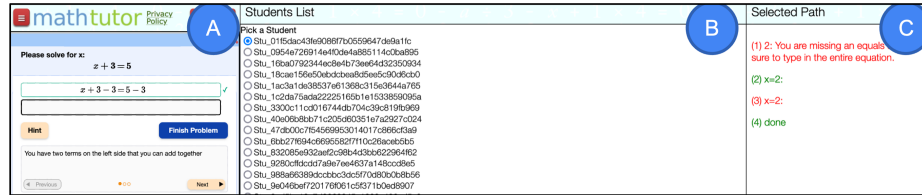


Fig. 4. List Interface: Tutor View (A), Students List (B), and Selected Path (C).

4 User Study

We conducted a user study to investigate the three research questions stated at the end of the Introduction.

Participants. Eight middle school math teachers (three males and five females, aged 30-59) participated in the user study, with an average of 12.9 years of experience in grades 5-12. They were from the US, Croatia, Taiwan, and Germany. While six had no ITS experience, one had experience with MATHia, and another used Khan Academy. Five had experience with visualization dashboards, four in creating data visualizations, two in reading interactive infographics, and one in reading self-tracking data charts.

Conditions and Datasets. To address our research questions, we compared three conditions: a baseline “no data” condition (i.e., working with just the tutor itself) and two “with data” conditions in which teachers worked with SolutionVis and with a list representation (List Interface, Figure 4), respectively. The List Interface shows a list representation of students’ IDs (Figure 4(B)) and, at the user’s request, a single student’s log data (Figure 4(C)). This kind of list representation is a common feature of commercial learning software such as Graspable Math [1]. Key differences between SolutionVis and the List Interface are that (a) SolutionVis has a graph of students’ aggregated paths and (b) the List Interface lists a single student’s solution path. We used a within-subjects design in which all participants experienced all conditions. All teachers

first worked with the Original Tutor and then with both the List Interface and SolutionVis. This way, we could ascertain whether data helps over and above working with just the tutor. To avoid a learning effect, we counterbalanced, across participating teachers, the order of the List Interface and SolutionVis. Participants were shown two equation-solving problems from the given intelligent tutor (Lynnette) with different difficulty levels: $x+3=5$ and $-4x+1-3+2x=4$. We presented log data from past use of these problems by 27 and 10 sixth-grade students, respectively; no data from the teachers' own students was included.

Procedure. We conducted and recorded one 90-minute study session for each participant remotely through Zoom. The user study consisted of the following activities: (1) We asked participants to sign in the consent form to get their permission for Zoom recording. (2) We provided background information about the project. (3) We first provided links to these two tutor problems. We then provided links to the List Interface (Figure 4) and the SolutionVis (Figures 2, 3) to participants in a counterbalanced order. (Thus, each participant experienced all three conditions.) For each of the three conditions, we spent three minutes introducing the interface. We then asked participants to do a task: explore different interfaces and give suggestions on how to improve the tutor for each of the two tutor problems. Participants were given 15 to 20 minutes for each interface. (4) Lastly, for each of the three interfaces, we asked them to fill in a 5-point Likert scale questionnaire with answers ranging from 1 (Strongly disagree) to 5 (Strongly Agree) (Figure 6) to rate the usability (Q1-Q5) and usefulness of the interface (Q6-Q10). We also asked which interface they preferred and why. The questionnaire was adapted from a standard usability test [5] and from questionnaires meant to evaluate the usefulness of teacher dashboards [22]. We also asked the participants to rate how easy it was to understand SolutionVis's interface. Each participant received an Amazon gift card (\$40) for participating.

5 Results

We address RQ1 with an analysis of teachers' redesign suggestions. For RQs 2 and 3, we report both teachers' redesign suggestions and of their survey data, including comments regarding the design of SolutionVis and the List Interface.

Analysis of Teachers' Redesign Ideas As shown in Figure 5, we extracted all the suggestions made by teachers in step (3) for how to improve the tutor. In total, we got 52 suggestions and 22 unique ones. On average, each participant generated three redesign suggestions when working with the Original Tutor, one additional suggestion when subsequently working with the List Interface, and two additional suggestions with SolutionVis. Two authors, using an affinity diagram, then grouped these suggestions into three main categories, namely, (1) interface/logic, (2) hints, and (3) feedback on gaming-the-system behaviors and persistent struggle. Category 2 (hints), was subdivided into four subcategories: adaptivity, clarity/correctness, visibility, and composition.

The main findings are: first, teachers generated many suggestions for how to improve the design of the tutor interface, the tutor's hints, and its feedback

	Original Tutor			List Interface	SolutionVis
Interface /logic design	Need to provide instruction that using "Enter" instead of "Finish" to go to next step. (5)	The correct step with a different order of the items in the equation is not accepted by the tutor. (5)	Cursor should go to the next line automatically.(1) Need to say explicitly to input the final answer or the intermediate step.(2)		
Hint Adaptivity	Address what the students did in the hint. (3)			If the answer is super close to the final answer (e.g., missing a negative sign), provide more concrete feedback (e.g., check your sign) (1)	Provide hints based on how many times the student asks for it. If the student asks a second time, showing a different hint. (1)
Hint Clarity/ Correctness	The hint "You can get the variable by itself by dividing both sides by the coefficient." is not correct for "-2x-4=2". Should "add 4 on both sides first".(3)	Rewrite the hint "Your input in not valid algebra." as "You need to have 'x ='"(2)	The hint "You have two terms on the left side that you can add together" is not well designed. Explain and give examples about "term".(4)	Provide the number line in the hint for students to understand positive and negative numbers for the steps where they need to move items from one side to the other side. (1)	Check the first hint of each step and make sure it is clear and easy to understand. (1)
Hint Visibility	Don't show the bottom hint.(1) Let the hints pop up automatically.(1)	Show part of the hints to let students think more at each step; teaching them but not correcting them.(1)		Don't encourage students to ask for hints at beginning, but ask "what would be your first step?" (1)	
Hint Composition	—	—	—	—	Ask a question about the knowledge in the hint to let them think. (1) Show some examples in the hint. (1)
Feedback on Gaming the system behavior/ protracted struggle	—	—	When seeing students gaming the system or inputting random things, provide feedback like "show me your efforts", "show your work"(2)	When seeing student submitting the same thing multiple times, let the tutor give the answer and move on.(1)	When seeing students gaming the system or inputting random things, provide feedback like "stop clicking this, please try again" (2) When seeing student submitting the same thing multiple times, provide feedback like "show me your efforts" (4) Show cartoons or funny animals to encourage them.(1)

Fig. 5. Suggestions made by teachers using different interfaces. For the List Interface and SolutionVis, the table lists only the suggestions that were not mentioned when working with the Original Tutor. “-” means no suggestion in the cell and empty means there are repetitive suggestions. Suggestions about hints are highlighted in yellow. The numbers in the parentheses indicate the number of participants.

(RQ1). Second, with the data provided in the List Interface and by SolutionVis, teachers generated additional suggestions, compared to the ones they generated with the Original Tutor (RQ2). With the Original Tutor, they generated a mixture of ideas regarding the redesign of the interface/logic, content of hints, and pedagogical aspects of hints. With the List Interface and SolutionVis (the “with data” conditions), they showed a greater focus on hint design (highlighted in yellow) and on feedback on gaming and struggling behaviors. For example, for the category hint adaptivity, P7 noticed, using the List Interface, that one student repeatedly submitted a solution that just missed a negative sign and suggested: “[I]f the answer is super close to the final answer (e.g., missing a negative sign), then provide more concrete feedback (e.g., check your sign).” Also, when teachers saw consecutive attempts like “x=1, x=2, ...,” they suggested feedback like “show me your efforts.” When they saw behaviors indicative of

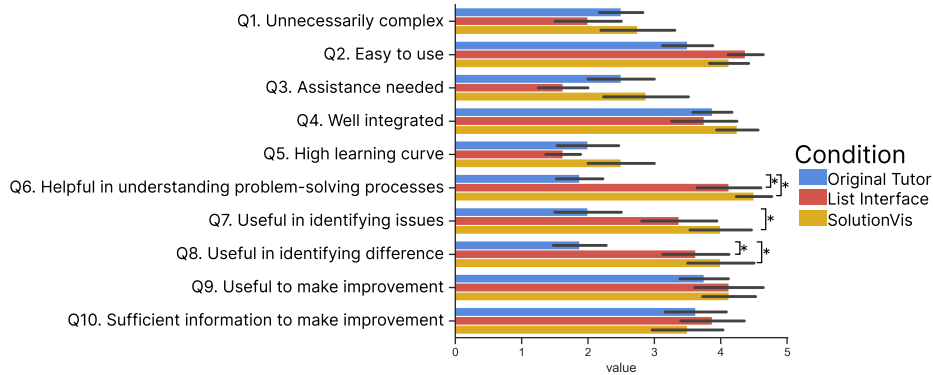


Fig. 6. Participants’ rating results for usability and usefulness on a 5-point Likert scale under three conditions: Original Tutor, List Interface, and EqLens. (* : $p < .05$)

struggle (e.g., long sequences), they suggested letting the tutor give the answer or cartoons to encourage students. Second, when using SolutionVis, participants focused on the steps with thick lines, large nodes, and paths containing both; they gave more suggestions addressing students’ common issues, e.g., gaming-the-system/struggling behaviors (RQ3).

Teachers’ Ratings of the Interfaces Figure 6 shows the ratings for the three interfaces in terms of usability (Q1-Q5) and usefulness (Q6-Q10). We ran a one-way ANOVA on each questionnaire item, with the three interfaces as the independent factor, followed by the Bonferroni posthoc test on measures with statistically significant differences [18]. Regarding usability, with SoutionVis the teachers felt they need slightly more assistance (Q3) and had a higher learning curve (Q5) than the Original tutor and the List interface, although the difference was not statistically significant. Participants could understand the visual representation in SolutionVis after the introduction. In terms of usefulness (see Q6-Q8), teachers found SolutionVis and the List Interface more helpful than the Original Tutor (without students’ data) for understanding students’ problem-solving behaviors, identifying common issues, and different problem-solving strategies (RQ2). There was no significant difference between the SolutionVis and the List Interface in this regard (RQ3).

Qualitative Feedback. We summarized participants’ comments and suggestions regarding SolutionVis and the List Interface (RQ3) and found that four participants preferred the List Interface and four preferred SolutionVis. All would like to have an integrated version of both interfaces.

SolutionVis. Teachers’ positive feedback about SolutionVis is that it helps them quickly understand students’ data by providing aggregation and highlighting common errors. “*SolutionVis is nice because it does some aggregation*” (P1). “*You can see, just by, like, the thickness of those arrows, you could see that the same hints were being asked many times or the same/similar answer was being entered over and over and over and to me, like, glaring red or those bold things*

would indicate that there’s a problem there” (P5). Teachers’ suggestions for improving SolutionVis are as follows. First, they hope SolutionVis could let them trace back to the specific students who made particular mistakes so they could help these students directly. Second, it would be better to provide a compact overview of the paths that fits on a single page. Third, the tool could highlight gaming-the-system behaviors. Fourth, a replay function could be added to show students’ actual interactions with the tutor interface [3].

List Interface. Teachers liked the List Interface because they thought they could use it to review individual students’ data from their own class (the data used in the study was not from the teachers’ own students), better interpret the data, and provide help to the students directly “*I want to connect the data to students’ behaviors in class to understand them better*” (P2). They also mentioned several limitations of the List Interface. “*It is helpful for debugging but not helpful for the classroom*” (P5). “*There’s no way to have 50 or 60 or 70 students every day and then 13, 14, 15 questions*” (P2).

6 Discussion

In general, for RQ1, teachers generated useful redesign ideas in both without data and with data (graph and list format) conditions. For RQ2, data about student learning (whether in graph or list format) was rated helpful (over and above working with just the tutor) and enabled teachers to generate a wider range of redesign suggestions. For RQ3 (comparison of graph v. list format), some teachers preferred SolutionVis because showing the aggregated data in a graph helps them find the tutor’s problems efficiently, and others preferred the List Interface for tracing each individual student’s solution. From these results we may conclude that SolutionVis enables teachers to be more active participants in the data-driven improvement of tutoring systems. Below, we further discuss design considerations, generality, and limitations of the current study.

Design Considerations (1) *Combine access to aggregated and student-specific process information.* All participants would like to have an integrated version of both interfaces. and we suggest an integrated version with aggregated graphical representations of students’ problem-solving processes and easy access to each individual student’s problem-solving steps. (2) *Provide more context information to better support attribution analysis.* In the user study, we found that teachers wanted to utilize their understanding of a particular student’s background (e.g., hard-working or not) to analyze the data. For example, they suggested that if a good student is experiencing struggle with the tutoring system, then maybe the tutoring system is to blame. We suggest providing a student’s past data (e.g., previous error rate) to aid teachers in attributing results to features of the tutors. *Integrate the tool into a teacher’s workflow.* It is still open questions on how a tool like this might fit into a teacher’s workflow. For example, education institutions could invite teachers to the data-driven intelligent tutors’ redesign process periodically, or the tool could be connected with tutor authoring tools (e.g., CTAT) so teachers can customize the tutors themselves.

Generality. The design requirements and the visualization tool generated are applicable for data-driven optimization of step-level problem-solving (e.g., ITSs). ITSs have been built for many domains, where the method should apply by defining different problem-solving stages, although we have yet to deal fully with problems in which these stages do not have a fixed order.

Limitations Our study has some limitations, including a limited number of participants, testing with data from a single tutor and a small problem set, and no evaluation of the impact on student learning. Despite this, we engaged more participants than previous studies [23,16] and gathered valuable quantitative data. Future work will enhance the prototype, address design considerations, and expand evaluations to include more teachers, tutors, and problems.

7 Conclusion

We investigated whether involving teachers in the data-driven of an ITS might be valuable, and what tools would be useful to harness teachers' unique sources of knowledge. The current study may well be the first in the AIED literature to address these questions. First, we derived five initial design requirements through a needs-finding study. We then built a visualization tool prototype, SolutionVis, that embodies the five requirements to present students' log data. Results from a within-subjects user study showed that teachers generated useful redesign ideas especially when supported by data, presented either as aggregated data from a group of students using a graph visualization (SolutionVis) or individual students' data in a list (List Interface). Teachers rated both interfaces as more helpful than working with just the tutoring system itself. The study shows two formats have complementary strengths and could be integrated in the next iteration of SolutionVis. These findings open up data-driven redesign for an audience that was not previously involved but may make unique contributions.

References

1. Grastable math. <https://activities.graspablemath.com/> (2022), accessed: 2022-09-10
2. Cytoscape.js. <https://js.cytoscape.org/> (2023), accessed: 2023-04-29
3. Alevin, V., Blankestijn, J., Lawrence, L., Nagashima, T., Taatgen, N.: A dashboard to support teachers during students' self-paced ai-supported problem-solving practice. In: ECTEL. pp. 16–30. Springer (2022)
4. Alevin, V., Sewall, J.: The frequency of tutor behaviors: a case study. In: International Conference on Intelligent Tutoring Systems. pp. 396–401. Springer (2016)
5. Bangor, A., Kortum, P.T., Miller, J.T.: An empirical evaluation of the system usability scale. *IJHCI* **24**(6), 574–594 (2008)
6. Hartson, R., Pyla, P.S.: *The UX Book: Process and guidelines for ensuring a quality user experience*. Elsevier (2012)
7. Heffernan, N.T., Heffernan, C.L.: The assistments ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. *IJAIED* **24**(4), 470–497 (2014)

8. Holstein, K., McLaren, B.M., Alevan, V.: Co-designing a real-time classroom orchestration tool to support teacher–ai complementarity. *Journal of Learning Analytics* **6**(2) (2019)
9. Huang, Y., Lobczowski, N.G., Richey, J.E., McLaughlin, E.A., Asher, M.W., Harackiewicz, J.M., Alevan, V., Koedinger, K.R.: A general multi-method approach to data-driven redesign of tutoring systems. In: LAK21. pp. 161–172 (2021)
10. Jin, T., Lu, X.: A data-driven approach to text adaptation in teaching material preparation: Design, implementation, and teacher professional development. *Tesol Quarterly* **52**(2), 457–467 (2018)
11. Koedinger, K.R., Alevan, V.: An interview reflection on “intelligent tutoring goes to school in the big city”. *IJAIED* **26**(1), 13–24 (2016)
12. Koedinger, K.R., Anderson, J.R.: Illustrating principled design: The early evolution of a cognitive tutor for algebra symbolization. *Interactive Learning Environments* **5**(1), 161–179 (1998)
13. Koedinger, K.R., Stamper, J.C., Leber, B., Skogsholm, A.: Learnlab’s datashop: A data repository and analytics tool set for cognitive science. *Top. Cogn. Sci.* **5**(3), 668–669 (2013)
14. Koedinger, K.R., Stamper, J.C., McLaughlin, E.A., Nixon, T.: Using data-driven discovery of better student models to improve student learning. In: International conference on artificial intelligence in education. pp. 421–430. Springer (2013)
15. Long, Y., Alevan, V.: Enhancing learning outcomes through self-regulated learning support with an open learner model. *User Modeling and User-Adapted Interaction* **27**, 55–88 (2017)
16. McBroom, J., Yacef, K., Koprinska, I., Curran, J.R.: A data-driven method for helping teachers improve feedback in computer programming automated tutors. In: AIED. pp. 324–337. Springer (2018)
17. Murray, T., Woolf, B.P.: Tools for teacher participation in its design. In: International Conference on Intelligent Tutoring Systems. pp. 593–600. Springer (1992)
18. Norman, G.: Likert scales, levels of measurement and the “laws” of statistics. *Advances in health sciences education* **15**(5), 625–632 (2010)
19. Simon, H.A., Newell, A.: Human problem solving: The state of the theory in 1970. *American psychologist* **26**(2), 145 (1971)
20. Tsung, S., Wei, H., Li, H., Wang, Y., Xia, M., Qu, H.: Blocklens: Visual analytics of student coding behaviors in block-based programming environments. In: Proceedings of the Ninth ACM Conference on Learning@ Scale. pp. 299–303 (2022)
21. VanLehn, K.: The behavior of tutoring systems. *AIED* **16**(3), 227–265 (2006)
22. Xia, M., Sun, M., Wei, H., Chen, Q., Wang, Y., Shi, L., Qu, H., Ma, X.: Peerlens: Peer-inspired interactive learning path planning in online question pool. In: Proceedings of the 2019 CHI. pp. 1–12 (2019)
23. Xia, M., Velumani, R.P., Wang, Y., Qu, H., Ma, X.: Qlens: Visual analytics of multi-step problem-solving behaviors for improving question design. *IEEE Transactions on Visualization and Computer Graphics* **27**(2), 870–880 (2020)
24. Xia, M., Wei, H., Xu, M., Lo, L.Y.H., Wang, Y., Zhang, R., Qu, H.: Visual analytics of student learning behaviors on k-12 mathematics e-learning platforms. arXiv preprint arXiv:1909.04749 (2019)
25. Yang, K.B., Nagashima, T., Yao, J., Williams, J.J., Holstein, K., Alevan, V.: Can crowds customize instructional materials with minimal expert guidance? exploring teacher-guided crowdsourcing for improving hints in an ai-based tutor. *Proceedings of the ACM on Human-Computer Interaction* **5**(CSCW1), 1–24 (2021)