VizPI: A Real-Time Visualization Tool for Enhancing Peer Instruction in Large-Scale Programming Lectures

Xiaohang Tang
Xi Chen
Virginia Tech
USA
{xiaohangtang,xic}@vt.edu

Sam Wong
University of California, San Diego
USA
c6wong@ucsd.edu

Yan Chen
Virginia Tech
USA
ych@vt.edu

ABSTRACT

Peer instruction (PI) has shown significant potential in facilitating student engagement and collaborative learning. However, the implementation of PI for large-scale programming lectures has proven challenging due to difficulties in monitoring student engagement, discussion topics, and code changes. This paper introduces VizPI, an interactive web tool that enables instructors to conduct, monitor, and assess PI for programming exercises in real-time. With features that visualize the progress of student discussions and code submissions, VizPI allows for more effective oversight of PI activities and the provision of personalized feedback at scale. Our work aims to transform the pedagogical approach to PI in programming education, making it more engaging and adaptable to student needs.

ACM Reference Format:
Xiaohang Tang, Xi Chen, Sam Wong, and Yan Chen. 2023. VizPI: A Real-Time Visualization Tool for Enhancing Peer Instruction in Large-Scale Programming Lectures. In *The 36th Annual ACM Symposium on User Interface Software and Technology (UIST '23 Adjunct), October 29-November 1, 2023, San Francisco, CA, USA.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3586182.3616632

1 INTRODUCTION

Peer Instruction (PI) is an instructional strategy developed by Mazur et al. that emphasizes students' active construction of conceptual understanding [1]. Initially employed in physics lectures for multiple-choice questions, students first individually respond to a question (the individual vote using wireless keypads or iclickers), then engage in peer discussion, and finally cast a re-vote influenced by the discussion (the group vote). Many recent studies have reported the benefits of PI in computing education [2] with students' survey responses overwhelmingly endorsing the efficacy of PI for learning and recommending its continued usage in further courses. Instructors, too, benefit from a refined focus on student difficulties, an improved ability to adapt lectures in real-time, and enhanced student involvement in teamwork and collaboration [4].

Recently, with the advent of supporting tools [3, 6], programming instructors have begun using in-class coding exercises — small scale programming exercises for students to practice during lectures or lab — as the basis for PI activities [5]. Instead of limiting discussions to options with immediate neighbors in a classroom, the use of online tools has enabled broader student interaction, facilitating code sharing and collaborative problem-solving. Despite these advancements, our preliminary interviews and previous research indicate that instructors struggle to assess student engagement during peer discussions, understand the topics being discussed, or ensure the relevance of the discussions to the exercise at hand. This becomes especially challenging in introductory programming classes, often comprising hundreds of students. Therefore, our research question arose: how can we aid instructors in effectively monitoring PI, specifically peer discussion activities, for coding exercises in large-scale programming lectures?

This poster presents our ongoing work aimed at enhancing the efficacy of PI in programming lectures. We introduce a prototype, VizPI[1], an interactive web tool that enables instructors to monitor numerous group discussions in real-time. With VizPI, instructors can create a coding exercise for students, while the tool monitors the submission rate and displays it to the instructor. Students are then grouped in pairs or trios, with a balanced expertise within each group. VizPI provides real-time visualization of each group's discussion progress and success rate. This allows instructors not only to gauge the overall progress of group discussions but also to discern common discussion points, prevalent mistakes, and other salient features of student discourse. As a result, instructors can easily highlight important codes or issues that VizPI has identified, creating a lecture that is considerably more adaptive to ongoing discussions. This innovative tool aids instructors in providing personalized, scalable feedback within the lecture setting.

2 INFORMAL NEEDFINDING INTERVIEWS

We conducted informal interviews with three instructors from our institution who teach introductory programming classes, gathering insights about their needs concerning peer instruction. These insights have significantly informed our system design rationale: (1) Instructors expressed a desire to comprehend and discuss overarching group performance metrics, particularly the primary mental, coding, and logical challenges that students discuss. (2) They also emphasized the utility of being able to present exemplary student responses on a large screen for class-wide discussion. This approach inverts the traditional pedagogical model, wherein instructors primarily share examples. (3) In addition, instructors highlighted the potential for innovation when our tool is integrated with a well-designed pedagogical strategy for classroom use. Such a blend of

[1]Short for visualization for peer instructions.

UIST '23 Adjunct, October 29-November 1, 2023, San Francisco, CA, USA

ACM ISBN 979-8-4007-0096-5/23/10.
https://doi.org/10.1145/3586182.3616632

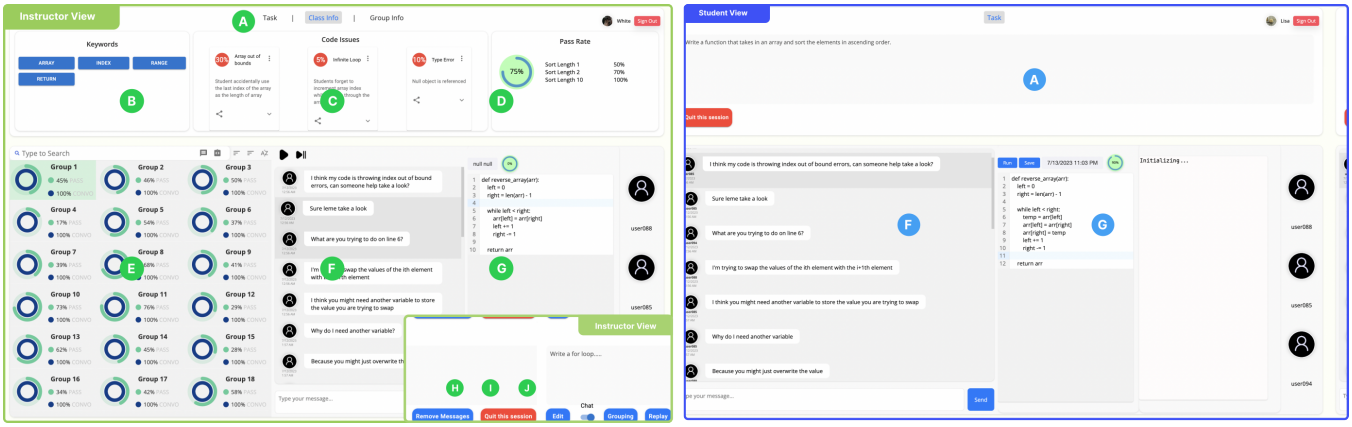

Figure 1: Instructor (in Green) and Student view (in Blue) of VizPI. A) Instructor can change between Task panel and Group/Class Performance, B) Keywords from group conversations, C) Common code issues found among groups, D) Class pass rate on instructor defined unit tests, E) Group list showing pass rate and conversation relevance, search bar filters relevant results, F) Group chat of assigned discussion group, G) Code editor activity of associated messages and code output window, H) Instructor control on turning off chat, I) Instructor control on grouping students, J) Instructor control on replaying group discussion.

technology and teaching methodology could significantly enhance the effectiveness of peer instruction in large-scale programming lectures.

3 SYSTEM OVERVIEW

VizPI is designed such that instructors can carry out peer instruction using in class programming exercises. Our interface has two views (Instructors view and Student view). Each view is separated into the Top Bar (Figure 1A,B,C,D), Group List (Figure 1E), Chat (Figure 1F) and Code Editor (Figure 1G). We will walk through how our interface aids each step of the peer instruction process.

3.1. Instructors propose a programming task for students: In VizPI, instructors will firstly input the programming task description and unit tests to test against the student solution in the Task tab (Figure 1A). In real time, students are able to view task description (Figure 1A) and start coding. During this phase, instructors can turn off chatting functions (Figure 1H) such that students are able to work on their code individually. As students write and run their code in the code editor, instructors are able to view the unit test pass rate of the whole class and the pass rate of individual unit tests (Figure 1D).

3.2. Instructors review students' performance and open up group discussion: Instructors are able to share initial unit test results and give preliminary feedback after students attempt the programming task. Afterwards, instructors can group students into groups (Figure 1I) and turn on chatting functions (Figure 1H). For the remaining discussion time, students are able to discuss with their peers and work on the programming assignments together. During this time period, instructors are allowed to observe student performance in 2 ways: (1) Class/Group Performance and (2) Individual Performance.

Class/Group Performance: This section is divided into 3 main components: *Keywords*, *Code Issues* and *Pass Rate*. *Keywords* summarizes the key themes of discussion based on the analysis of conversation messages across all groups (Figure 1B), and they can be used to filter groups that mentioned the corresponding keyword. *Code Issues* demonstrate common errors found in student submission. Each error is presented in a form of a card, where instructors can view the percentage of students making the mistake, as well as the summary of the error made by these students (Figure 1C). These components can also be viewed on a group level, which summarizes information based on each group instead of the whole class.

The group panel displays information for instructors to assess group performances in rings (Figure 1E). The outer light green ring denotes the pass rate of each group against all the unit tests, and the inner dark blue ring measures the conversation relevance of each group's messages. The closer the conversation is to the solution, the higher the conversation rate is. Instructors can use these metrics to evaluate how close each group is in reaching the correct answer.

Individual Performance: When a student sends a message in their group chat, the associated code written at that moment will be captured and stored. By selecting each chat message, instructors are able to view each students' code editor activity at the time each message is sent (Figure 1F,G). This gives instructors a better understanding of student's progress as they can trace through the student's coding progress to understand their approach to tackling the problem.

4 FUTURE WORK

We will continue our system implementation for features that we listed below and also conduct system evaluation to examine the effectiveness of this tool. **(1) Advanced Grouping:** We'll develop a refined algorithm that uses students' pass rates and personality traits for improved group formation. **(2) Enhanced NLP Models:** We plan to use better Natural Language Processing (NLP) models to highlight key discussion points, incorporating instructors' feedback. **(3) Customizable Feedback:** We aim to offer instructors the ability to provide feedback at varying levels during peer instruction.

In conclusion, our interactive web tool, VizPI, has the potential to transform the instructional approach by enabling real-time

monitoring of numerous group discussions, thereby empowering instructors to provide personalized and scalable feedback within the lecture setting.

REFERENCES

[1] Catherine H Crouch and Eric Mazur. 2001. Peer instruction: Ten years of experience and results. *American journal of physics* 69, 9 (2001), 970–977.

[2] Pranita Deshpande, Cynthia B Lee, and Irfan Ahmed. 2019. Evaluation of peer instruction for cybersecurity education. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*. 720–725.

[3] Philip J Guo. 2015. Codeopticon: Real-time, one-to-many human tutoring for computer programming. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*. 599–608.

[4] Leo Porter, Dennis Bouvier, Quintin Cutts, Scott Grissom, Cynthia Lee, Robert McCartney, Daniel Zingaro, and Beth Simon. 2016. A multi-institutional study of peer instruction in introductory computing. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*. 358–363.

[5] April Yi Wang, Yan Chen, John Joon Young Chung, Christopher Brooks, and Steve Oney. 2021. PuzzleMe: Leveraging Peer Assessment for In-Class Programming Exercises. *Proceedings of the ACM on Human-Computer Interaction* 5, CSCW2 (2021), 1–24.

[6] Ashley Ge Zhang, Yan Chen, and Steve Oney. 2023. VizProg: Identifying Misunderstandings By Visualizing Students' Coding Progress. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. 1–16.