Integrating Personalized Parsons Problems with Multi-Level Textual Explanations to Scaffold Code Writing

Xinying Hou
University of Michigan
Ann Arbor, Michigan, USA
xyhou@umich.edu

Barbara J. Ericson
University of Michigan
Ann Arbor, Michigan, USA
barbarer@umich.edu

Xu Wang
University of Michigan
Ann Arbor, Michigan, USA
xwanghci@umich.edu

ABSTRACT

Novice programmers need to write basic code as part of the learning process, but they often face difficulties. To assist struggling students, we recently implemented personalized Parsons problems, which are code puzzles where students arrange blocks of code to solve them, as pop-up scaffolding. Students found them to be more engaging and preferred them for learning, instead of simply receiving the correct answer, such as the response they might get from generative AI tools like ChatGPT. However, a drawback of using Parsons problems as scaffolding is that students may be able to put the code blocks in the correct order without fully understanding the rationale of the correct solution. As a result, the learning benefits of scaffolding are compromised. Can we improve the understanding of personalized Parsons scaffolding by providing textual code explanations? In this poster, we propose a design that incorporates multiple levels of textual explanations for the Parsons problems. This design will be used for future technical evaluations and classroom experiments. These experiments will explore the effectiveness of adding textual explanations to Parsons problems to improve instructional benefits.

CCS CONCEPTS

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

KEYWORDS

Introductory Programming, Code Explanations, Parsons Problems, Code Writing, Scaffolding, Hint, Large Language Models

ACM Reference Format:
Xinying Hou, Barbara J. Ericson, and Xu Wang. 2024. Integrating Personalized Parsons Problems with Multi-Level Textual Explanations to Scaffold Code Writing. In *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 2 (SIGCSE 2024), March 20–23, 2024, Portland, OR, USA.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3626253.3635606

1 INTRODUCTION

One main goal of introductory programming courses is to develop basic coding skills. Although AI generation tools are now widely used in professional code development, it is still crucial for beginners to participate in code-writing activities to fully acquire fundamental programming concepts. In recent work, we explored providing Parsons problems as scaffolding for students who are struggling while writing code independently in Python [2–4]. By leveraging the power of LLMs, we designed and implemented a system called *CodeTailor* to provide multi-stage personalization through a Parsons problem to assist students with code writing (Figure 1). An initial study with 18 novice programming students showed that CodeTailor is engaging and benefits learning. However, some students reported challenges as they could not fully understand the meaning of some code blocks. Textual explanations could help users understand the program's components, objectives, and structure [5]. They can assist beginners in learning more from the Parsons problem, providing an additional opportunity to read code blocks, understand the problem, and comprehend the solution. Hence, we propose a design to incorporate multiple levels of natural language explanations into CodeTailor. While explanations have been given with various programming learning materials or activities, there is limited research on integrating them with Parsons problems. Given the increasing interest and demonstrated effects of using Parsons problems to scaffold novice programmers' learning, more research is required to explore the integration of explanations into Parsons problems.

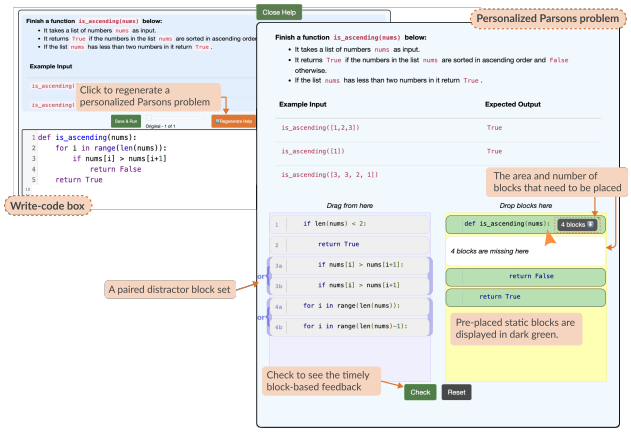

Figure 1: A write-code box (left) with a pop-up personalized Parsons problem as scaffolding (right).

2 SYSTEM DESIGN

CodeTailor is a large language model (LLM)-powered system that provides real-time, on-demand, and multi-staged personalized Parsons problems to support students while writing code. It differs from existing hint systems for programming in that CodeTailor provides an active learning opportunity through the help. Specifically, students need to engage in problem-solving with the support

SIGCSE 2024, March 20–23, 2024, Portland, OR, USA

ACM ISBN 979-8-4007-0424-6/24/03.
https://doi.org/10.1145/3626253.3635606

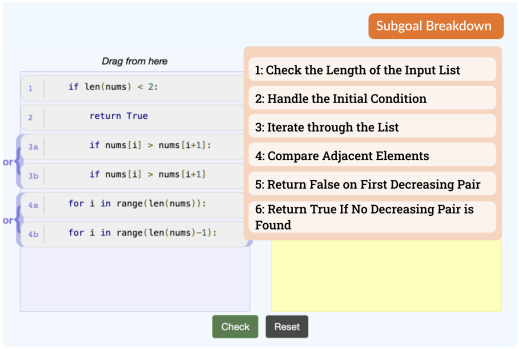

Figure 2: A subgoal list about the Parsons problem solution.

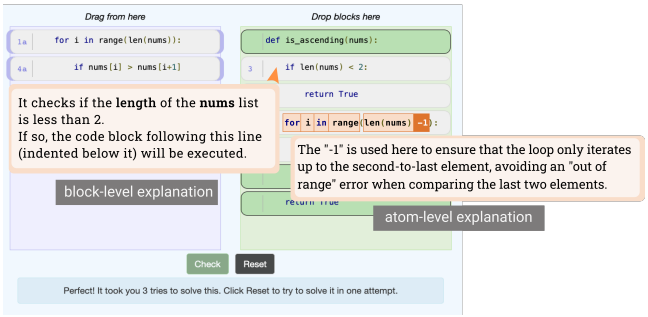

Figure 3: Block-level and atom-level explanations for the finished Parsons blocks.

provided rather than merely being consumers of a displayed help message [1]. An overview of CodeTailor's main interface is shown in Figure 1. After clicking the "Help" button, CodeTailor provides a personalized Parsons problem that is dynamically generated based on the student's latest code state. The personalized Parsons problem is partially complete, with the correct student-written code lines already in place and unmovable. The distractors, which are unnecessary blocks in the correct solution, are generated from the student's own incorrect code lines. Furthermore, CodeTailor adjusts the difficulty of the Parsons problem based on students' requests. If a student fails three times with a fully movable Parsons problem, CodeTailor allows them to combine two code blocks into one. Once the student successfully solves the Parsons problem, they can copy the solution to the write-code box.

We enhance CodeTailor by incorporating multiple levels of LLM-generated textual explanations. Four types of explanations will be generated and inserted in real-time, enabling CodeTailor to provide personalized explanations accompanying the Parsons problem. We are still exploring methods to assess the quality of these explanations automatically.

Subgoal guidance explanation: Students will have access to a subgoal summary of the solution provided in the unsolved Parsons problem. This is presented in numbered bullet points, breaking down the current programming task into 4-6 more manageable task subgoals (Figure 2). The subgoal guidance aims to help students gain a high-level abstract understanding and guide those who struggle to start solving a Parsons problem.

Block-level code explanation: After solving the Parsons problem, students can hover over each block to get an explanation of each block. These explanations clarify the behavior and purpose of the Parsons solution at the block level (Figure 3). For paired distractors, the explanation also provides reasoning for why the correct block is right and why the corresponding distractor block is incorrect.

Atom-level code explanation: For each block in the Parsons solution, students can also click on an atom (individual elements of the programming language, such as keywords and statements) within a code block to receive an explanation of the atom's text surface, execution (if any), and purpose (Figure 3).

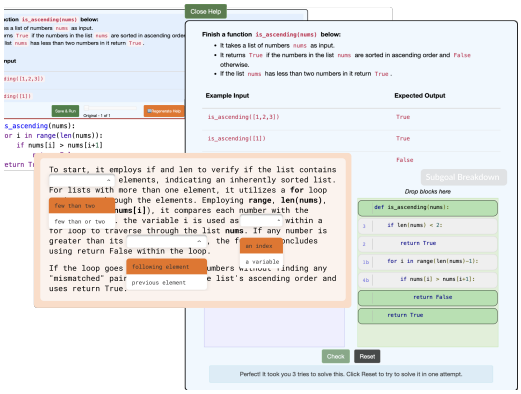

Figure 4: Students will receive a menu-based self-explanation question after solving the Parsons problem.

A self-explanation question to reflect on the Parsons solution: After getting the write-code problem correct, this Parsons solution will appear again without any textual explanations. This time, students need to explain the reasoning behind this solution in a menu-based self-explanation prompt. The prompt is designed to minimize guessing and enhance students' understanding. In this case, an LLM will generate the main structure of this explanation, leaving keywords to be filled in (Figure 4) by the student.

ACKNOWLEDGMENTS

The funding for this research came from the National Science Foundation award 2143028. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] Michelene TH Chi and Ruth Wylie. 2014. The ICAP framework: Linking cognitive engagement to active learning outcomes. *Educational psychologist* 49, 4 (2014), 219–243.

[2] Xinying Hou, Barbara Jane Ericson, and Xu Wang. 2022. Using adaptive parsons problems to scaffold write-code problems. In *Proceedings of the 2022 ACM Conference on International Computing Education Research-Volume 1*. 15–26.

[3] Xinying Hou, Barbara Jane Ericson, and Xu Wang. 2023. Parsons Problems to Scaffold Code Writing: Impact on Performance and Problem-Solving Efficiency. In *Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 2*. 665–665.

[4] Xinying Hou, Barbara J Ericson, and Xu Wang. 2023. Understanding the Effects of Using Parsons Problems to Scaffold Code Writing for Students with Varying CS Self-Efficacy Levels. *arXiv preprint arXiv:2311.18115* (2023).

[5] Cruz Izu, Carsten Schulte, Ashish Aggarwal, Quintin Cutts, Rodrigo Duran, Mirela Gutica, Birte Heinemann, Eileen Kraemer, Violetta Lonati, Claudio Mirolo, et al. 2019. Fostering program comprehension in novice programmers-learning activities and learning trajectories. In *Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education*. 27–52.