International Society of the Learning Sciences

Scaffolding Debugging That Uses Tinkering

ChanMin Kim, Brian R. Belland
cmk604@psu.edu, bbelland@psu.edu
The Pennsylvania State University

Abstract: We designed scaffolding for debugging that uses tinkering. The scaffolding was implemented in a context for teacher learning of computing education. Data included responses to scaffolding prompts, pair debugging processes recorded through video screencast recording, and interviews. We studied how debugging processes differ where the quality of the rules written to justify debugging success (or unsuccess) differ. Findings are organized around the two paired groups who generated different quality of rules in response to the scaffolding.

Introduction

Central to creating culturally responsive computer science (CS) classrooms is a need to prepare teacher candidates to teach CS. Some recent efforts have attempted to identify and strengthen methods of preparing teacher candidates to program (e.g., Kim et al., 2018). But little is known how to facilitate their learning to teach CS in culturally responsive ways. The present study researched methods to do so, through scaffolding for abductive reasoning while tinkering.

Our research question was: How do participants engage in debugging when scaffolded through abductive reasoning?

Theoretical foundation

We used abductive reasoning as a theoretical lens. Abductive reasoning is a process of thinking while attempting to figure out a way of describing a perplexing phenomenon. The process of abduction begins with the best possible guess at the moment and moves to subsequent guesses when the presently tested guess does not explain the phenomenon of investigation (Magnani, 2009; Peirce, 1931-1935).

Method

Paired debugging sessions were recorded using video screencast software. In addition to debugging screencast recordings, pairs' responses to scaffolding prompts were recorded. Our scaffold design for abductive reasoning aimed to (a) leave tinkering as a natural process of debugging and (b) guide rule discovery and refinement. This design was grounded in our prior research in which teacher candidates (a) tinkered through abduction and (b) exhibited reflective debugging, but (c) did not notice a rule across cases (Kim et al., In press). Thus, our scaffolding provided prompts to help them discover a rule that may explain what they were seeing as a result of their tinkering. Artifact-based individual interviews were done after the debugging tasks were completed.

Our analytic strategy involved using conversation analysis (ten Have, 2007) and open and axial coding (Miles et al., 2013) done through the lens of ethnomethodology (Garfinkel, 1967).

Findings and discussion

We present findings from two groups who generated rules of varying quality in response to the scaffolding (e.g., Table 1). As described above, participants were asked to write down a rule that could explain what they just saw during tinkering. The design rationale was to help them (a) connect the change they made in the code to the result they were seeing, (b) reflect on what they guessed as a cause for the buggy code, and (c) notice relations between previously guessed causes and actual results. Such design aimed for reflective abstraction (Abrahamson, 2012) and "plausible abductive generalization" (Rivera & Becker, 2007, p. 140), which would in turn create a collection of potentially relevant rules to be used in abductive reasoning during future debugging.

The following themes were uncovered as to how the debugging processes of these two groups differed.

Reading before (re)writing the code

The initial reading of the given buggy code was rather quick for Group 1. Group 2 read the code line by line also while paying attention to the repeat while block. This finding is interesting because the total time Group 1 spent debugging (38.13 minutes) was longer than that of Group 2 (21.86 mins). Both groups questioned the meaning of "iteration" in the code but Group 1 searched for the term online and read out loud its definition and Group 2 figured out it was a label for a variable. When asked to guess what the code was making the robot do during the initial reading of the code, both guessed incorrectly yet: Group 1 guessed a star shape and Group 2 guessed a

pentagon shape. As such, both groups' initial reading of the buggy code was imperfect, but different processes of initial reading, quick versus thorough, were observed between the two groups.

Table 1: Example responses of Group 1 and Group 2 to prompts related to rule generation

	Group 1	Group 2
Final rule	When you put the wrong number in the distance, it will not complete the right shape.	If you program the robot to move forward 50 mm and then have it rotate 45 degrees, it will create one side of an octagon
Rule use plan	We plan on playing around with the numbers in order to make sure we are doing it right.	When considering how many sides we need for a shape, we need to remember to divide the number of sides by 360 to get the angle we should set it to.

Follow-through guesses

Still without information about the programming goal, participants were asked to run the buggy code and guess how the robot would travel. Their guesses were not correct yet: Group 1 guessed a pentagon and Group 2 guessed a hexagon and then a pentagon. But the two groups differed in that Group 2 tinkered through the numeric values in the turn angle in line with their guess but Group 1 did not revisit their guess later.

Parsing the code and robot movement

Group 2 broke down the code in relation to the specific behaviors of the robot whereas Group 1 did not. During such parsing, Group 2 noticed which part of the code worked to lead to a desired behavior of the robot, whereas Group 1 did not notice the misfunctioning of the angle value and ended up changing the correct distance.

Conclusion and implications

The unique contributions of this study are (a) the asset-based scaffolding in which tinkering through abduction among teacher candidates is respected and utilized, and (b) the potential lessons on how tinkering through abduction can be valued and supported among youth learning to program. This in turn has the potential to invite teachers and youth from diverse backgrounds to identify with and engage in programming.

Acknowledgments

This study is supported by grants 1927595 and 1906059 from the National Science Foundation. Any opinions, findings, or conclusions are those of the authors and do not necessarily represent NSF's official positions.

References

Abrahamson, D. (2012). Rethinking intensive quantities via guided mediated abduction. *Journal of the Learning Sciences*, *21*(4), 626–649. https://doi.org/10.1080/10508406.2011.633838

Berland, M., Martin, T., Benton, T., Smith, C. P., & Davis, D. (2013). Using learning analytics to understand the learning pathways of novice programmers. *Journal of the Learning Sciences*, *22*(4), 564–599. https://doi.org/10.1080/10508406.2013.836655

Garfinkel, H. (1967). *Studies in ethnomethodology*. Englewood Cliffs, N.J. http://hdl.handle.net/2027/uc1.-b3986101

Kim, C., Belland, B. R., Baabdullah, A., Lee, E., Dinç, E., & Zhang, A. (In press). An ethnomethodological study of abductive reasoning while tinkering. *AERA Open*.

Kim, C., Yuan, J., Vasconcelos, L., Shin, M., & Hill, R. B. (2018). Debugging during block-based programming. *Instructional Science*, *46*(5), 767–787. https://doi.org/10.1007/s11251-018-9453-5

Magnani, L. (2009). *Abductive cognition: The epistemological and eco-cognitive dimensions of hypothetical reasoning* (Vol. 3). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-03631-6

Miles, M. B., Huberman, A. M., & Saldaña, J. (2013). *Qualitative data analysis: A methods sourcebook*. SAGE.

Peirce, C. S. (1931-1935). *The collected papers of Charles Sanders Peirce* (C. Hartshorne & P. Weiss, Eds.; Vols. 1–6). Harvard University Press.

Rivera, F. D., & Becker, J. R. (2007). Abduction–induction (generalization) processes of elementary majors on figural patterns in algebra. *The Journal of Mathematical Behavior*, *26*(2), 140–155. https://doi.org/10.1016/j.jmathb.2007.05.001

ten Have, P. (2007). *Doing conversation analysis*. SAGE Publications, Ltd. http://srmo.sagepub.com/view/doing-conversation-analysis/SAGE.xml