On Students' Experiences with Algorithm Tracing using Pair Programming

Oana Andrei
oana.andrei@glasgow.ac.uk
University of Glasgow
Glasgow, UK

Syed Waqar Nabi
syed.nabi@glasgow.ac.uk
University of Glasgow
Glasgow, UK

ABSTRACT

We have seen students struggling with fully understanding the algorithms when required to trace pseudocode for an algorithmics course, despite the fact that examples of tracing each algorithm are provided in lectures and tutorials exercises. This course is being taught in year 2 of the Graduate Apprenticeship in Software Engineering programme (GA) [1], and covers fundamental string and graph algorithms and a brief introduction to automata.

Code tracing helps students develop valid mental models of the program [5], while sketching [3, 7] – a pedagogical tool for tracing code via pen-and-paper drawing of visualisation of program states or other computing processes – may help students manage cognitive load while understanding the notional machine [6] behind an algorithm. Hence sketching is fitting for tracing algorithm pseudocode.

Pair programming (PP) [2] has been successfully adopted in CS education as a collaborative learning activity [4] where both learners work in pairs to solve a task, with one being the driver and the other the navigator, and switching roles regularly. In a previous investigation on using PP for constructing finite state automata [1], we have found that GA students enjoyed working in pairs and one student noted that *"[working in pairs] was nice to work with others to help build each others understanding and this approach would help more with some of the tricky sections"*.

We wish to explore if we can use PP for algorithm tracing similar to how it is used in a programming context. Our initial research question is: *How do GA students experience tracing the pseudocode of fundamental string and graph algorithms using pair programming?*

Algorithms were covered in two weeks, delivered in block-mode. Students were encouraged to use PP for all tracing exercises. In March 2023 we ran a pilot study to answer the research question above to gather students' opinions on *what they liked* and *what they didn't like* about using PP for algorithm tracing via an anonymous, online survey consisting of two open-ended questions. Seven out of 25 students enrolled in the course responded. The inductive thematic analysis of the survey responses shows that most students enjoyed collaboration to build up their understanding of algorithms. Overall, themes around insufficiently supportive setup and time-consuming activity have been identified. One response recommended encouraging students to pair up with colleagues at other tables.

Some future actions to address the shortcomings identified are: increase tutorial sessions structure with timing guidelines for each tracing exercise, provide blank paper and/or digital templates for tracing each algorithm, check progress in pairs regularly and remind students to ask for help when the pair is stuck. A few students in the class chose not to engage in PP, preferring to work on their own. We will investigate ways to engage these students such as identifying and use existing computer-based tools to support visualising algorithm execution and providing extra tutor support.

We will continue using PP for tracing algorithms in the next teaching session and further investigate the students' experiences and their perceived and assessed performance when using PP.

ACM Reference Format:
Oana Andrei and Syed Waqar Nabi. 2023. On Students' Experiences with Algorithm Tracing using Pair Programming. In *Proceedings of ICER 2023 (ICER2023).* ACM, New York, NY, USA, 1 page. https://doi.org/10.1145/nnnnnnn.nnnnnnn

REFERENCES

[1] Oana Andrei. 2022. A Practice Enquiry Design to Investigate How Pair Programming Can Help with Constructing Automata. In *Proceedings of UKICER 2022*, Keith Quille, Joseph Maguire, and Brett A. Becker (Eds.). ACM, 20:1.
[2] K. Beck and C. Andres. 2004. *Extreme Programming Explained: Embrace Change* (2nd ed.). Addison-Wesley Professional.
[3] Kathryn Cunningham, Sarah Blanchard, Barbara Ericson, and Mark Guzdial. 2017. Using Tracing and Sketching to Solve Programming Problems: Replicating and Extending an Analysis of What Students Draw. In *Proceedings of the 2017 ACM Conference on International Computing Education Research, ICER 2017*, Josh Tenenberg, Donald Chinn, Judy Sheard, and Lauri Malmi (Eds.). ACM, 164–172.
[4] B. Hanks, S. Fitzgerald, R. McCauley, L. Murphy, and C. Zander. 2011. Pair programming in education: a literature review. *Computer Science Education* 21, 2 (2011), 135–173.
[5] Matthew Hertz and Maria Jump. 2013. Trace-based teaching in early programming courses. In *The 44th ACM Technical Symposium on Computer Science Education, SIGCSE 2013*, Tracy Camp, Paul T. Tymann, J. D. Dougherty, and Kris Nagel (Eds.). ACM, 561–566.
[6] Juha Sorva. 2013. Notional machines and introductory programming education. *ACM Trans. Comput. Educ.* 13, 2 (2013), 8:1–8:31.
[7] Benjamin Xie, Greg L. Nelson, and Amy J. Ko. 2018. An Explicit Strategy to Scaffold Novice Program Tracing. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education, SIGCSE 2018*, Tiffany Barnes, Daniel D. Garcia, Elizabeth K. Hawthorne, and Manuel A. Pérez-Quiñones (Eds.). ACM, 344–349.

[1] https://www.gla.ac.uk/research/az/ccse/workbasedlearning/

ICER2023, Chicago,

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn