Long Term Retention of Programming Concepts Learned Using Tracing Versus Debugging Tutors

Amruth N Kumar [0000-0002-1951-3995]

Ramapo College of New Jersey, Mahwah NJ 07430, USA
amruth@ramapo.edu

Abstract. We studied long-term retention of the concepts that introductory programming students learned using two software tutors on tracing the behavior of functions and debugging functions. Whereas the concepts covered by the tutor on the behavior of functions were interdependent, the concepts covered by debugging tutor were independent. We analyzed the data of the students who had used the tutors more than once, hours to weeks apart. Our objective was to find whether students retained what they had learned during the first session till the second session. We found that the more the problems students solved during the first session, the greater the retention. Knowledge and retention varied between debugging and behavior tutors, even though they both dealt with functions, possibly because debugging tutor covered independent concepts whereas behavior tutor covered interdependent concepts.

Keywords: Retention of Learning, Programming, Code-tracing, Debugging.

Introduction. Researchers have studied interventions for improving long term retention of learning, such as data-driven examples (e.g., [1]), game-based environments (e.g., [2]), task interleaving (e.g., [9]), spacing (e.g., [3]) and active construction of digital artifacts (e.g., [4]). They have also attempted to incorporate retention into student models (e.g., [10]) in order to be able to predict the performance of a student on the next problem on a concept, when the problem is attempted a few hours, days or weeks later.

In order to find out if students retained the concepts learned using a tutor over the long term, in this observational study, we analyzed the data collected by two tutors when students used them more than once, a few days or weeks apart, of their own volition and on their own time. One tutor was on function behavior wherein students were asked to identify the output of a program and the other was on debugging functions wherein students were asked to identify bugs in a program. Both had reified interface [11], making it hard to guess the correct answer. The tutor on function behavior covered ten concepts: four on function call, two on function definition and four on parameter passing. The concepts are **interdependent**, i.e., a student who learns one parameter passing concept is likely to be able to solve problems on other parameter-passing concepts correctly. The tutor on debugging functions covered nine concepts: three on function call, four on function definition and two on parameter passing. The bugs are **independent**, i.e., knowledge of one bug is unlikely to help a stu-

dent solve problems on another bug correctly. The tutors presented isomorphic problems generated as randomized instances of parameterized templates which are still challenging for novices [5, 6]. So, students saw different problems each time they used the tutors.

The tutors administered pre-test-adaptive practice-post-test protocol every time they were used [7]. The pre-test was used to prime the student model. Practice was provided on only the concepts on which the student solved a pretest problem incorrectly. Practice was provided on a concept until the student had mastered the concept by solving a minimum number and percentage of problems correctly. Post-test was presented on only the concepts mastered during practice. Pretest, practice and post-test were administered by the tutors back-to-back, all online and without any interruptions. The entire protocol was limited to 30 minutes. Each concept covered by the tutors can be classified as known, tested, practiced or learned for each student, as summarized in Table 1.

Table 1. Types of Learning Experience with the Tutors.

Pretest	Practice	Posttest	Type of Learning
Correct			Known
Incorrect	None		Tested
Incorrect	Some		Practiced
Incorrect	Mastered	Incorrect	Practiced
Incorrect	Mastered	Correct	Learned

If a student who returns to use the tutor a second time at a later date or time solves the pretest problem on a concept correctly, the student has **retained** the concept from the previous session. If the student solves the pretest problem incorrectly, the student has **forgotten** the concept from the previous session. Based on the student's learning experience during the first tutoring session and pretest performance in the second tutoring session, the eight possible retention behaviors of a student on a concept are: known-retained, known-forgotten, tested-retained, tested-forgotten, practiced-retained, practiced-forgotten, learned-retained and learned-forgotten. Neither known-retained nor known-forgotten concepts are affected by the use of the tutor. These served as the comparison group in the study. On the other hand, tested-retained, practiced-retained and learned-retained all provide evidence in support of long-term retention of what was learned using the tutor, the hypothesis of this study, whereas tested-forgotten, practiced-forgotten and learned-forgotten all provide evidence disproving retention. These served as experimental data points in the study.

We used the data collected by the tutors over 14 semesters: Fall 2012 – Spring 2019. The tutors were used by introductory programming students in high schools and colleges as after-class assignments. The students could use the tutors as often as they pleased. We used data only from the students who had used the tutors at least twice and gave us permission to use their data for research purposes.

Function Behavior Tutor Results. 513 students used the tutor more than once. They solved problems at least twice on 3918 concepts, representing an average of 7.64 concepts per repeat user. Table 2 lists the number of student concepts N in each type

of retention behavior, the percentage of the total student concepts represented by that retention behavior O%, the percentage of retained and forgotten concepts within the learning category L%, the mean pretest score on the first and second pretests, and the mean time between the two sessions in hours. The score on each problem was normalized to the range 0 → 1.0.

Table 2. Functions Behavior Tutor - Types of Retention Behavior

Retention Behavior Type	N	O%	L%	Pretest1	Pretest2	Time (hours)
Known-Retained	2226	56.81	94.56	1.0	1.0	402.15 ± 107.0
Known-Forgotten	128	3.27	5.44	1.0	0.08	872.22 ± 446.4
Tested-Retained	760	19.40	67.86	0.12	1.0	27.07 ± 183.2
Tested-Forgotten	360	9.19	32.14	0.10	0.13	87.16 ± 266.2
Practiced-Retained	167	4.26	70.17	0.14	1.0	688.39 ± 390.8
Practiced-Forgotten	71	1.81	29.83	0.15	0.14	549.51 ± 599.4
Learned-Retained	164	4.19	79.61	0.17	1.0	588.52± 394.4
Learned-Forgotten	42	1.07	20.39	0.24	0.20	1622.82 ± 779.3

Known-forgotten concepts represent transience, the deterioration of learning over time. The student concepts in this category were 5.44% of all known student concepts. Based on the column titled L%, *students retained over 67% of the concepts covered by the tutor on function behavior*. Conversely, tested-forgotten, practiced-forgotten and learned-forgotten figures were all greater than known-forgotten percentage (5.44%) attributable to transience of learning. So, although students retained over 67% of the concepts, *there is room for improvement of the tutor to promote retention of learning*. We note two additional patterns in the descriptive statistics: in the column L%, learned-retained was greater than both practiced-retained and tested-retained. Since students solved more problems on learned concepts than practiced concepts and on practiced concepts than on tested concepts, this supports the observation that *the more the practice problems solved during the first session, the more likely students retained the concept till the second session.* From Table 2, we also note that the mean time between sessions is 2-3 times greater for forgotten concepts in each learning category compared to retained concepts, except in practiced category. It is possible that this observational study captured retained and forgotten student concepts in each category at different points in time, and eventually, more retained student concepts will convert to forgotten concepts without additional reinforcement of learning.

Debugging Functions Tutor Results. 642 students used the tutor more than once. They solved problems at least twice on 5489 concepts, representing an average of 8.55 concepts per repeat user. Table 3 lists the retention behavior figures for debugging tutor. Since students either correctly identified a bug or did not, the score on a problem was either 0 or 1.

Table 3. Debugging Functions Tutor - Types of Retention Behavior

Retention Behavior Type	N	O%	L%	Pretest1	Pretest2	Time (hours)
Known-Retained	2224	40.52	88.68	1.0	1.0	63.16 ± 21.52
Known-Forgotten	284	5.17	11.32	1.0	0.0	210.43 ± 60.23
Tested-Retained	1616	29.44	72.53	0.0	1.0	12.46 ± 25.25
Tested-Forgotten	612	11.15	27.47	0.0	0.0	93.43 ± 41.03
Practiced-Retained	402	7.32	76.14	0.0	1.0	52.55 ± 50.62
Practiced-Forgotten	126	2.30	23.86	0.0	0.0	125.41 ± 90.42
Learned-Retained	205	3.73	91.11	0.0	1.0	56.72 ± 70.89
Learned-Forgotten	20	0.36	8.89	0.0	0.0	137.89 ± 226.96

Known-retained is far smaller than 56.81% for function behavior tutor. Known-forgotten as a percentage of known concepts, which accounts for transience of learning, is larger (11.32%) than that for function behavior tutor. *So, knowledge and retention of learning varied between debugging and tracing skills, even though they both pertained to functions.* This confirms the results from our earlier study conducted using selection tutor [8]. Tested-retained and practiced-retained percentages (L%) on the other hand were greater for debugging than behavior of functions. One explanation is that each bug is unique and **independent**, and the short explanation provided for it clarifies the genesis of the bug. On the other hand, students must synthesize a lot of **interdependent** concepts to understand and predict the behavior of functions, making the behavior of functions harder to learn and retain.

Based on the column L%, *students retained over 72% of the concepts covered by the tutor on debugging functions*. Learned-forgotten (8.89%) was less than transience of learning. So, the mastery criterion used by debugging tutor during practice stage is robust. Here again, we found that *the more the practice problems solved during the first session, the more likely students retained the concept till the second session*: in column L%, learned-retained was greater than practiced-retained and tested-retained. Just as in the case of behavior tutor, we note that the mean time between sessions is at least twice as much for forgotten concepts in each category compared to retained concepts suggesting that we captured retained and forgotten student concepts in each category at different points in time.

In this study, we did not consider guesses and slips: the reified user interface makes it hard to guess the correct answer and error-flagging feedback provided by the tutors offers the opportunity for students to recover from slips. On the other hand, students who use a tutor repeatedly of their own volition are typically self-motivated. They are also likely to have had extraneous opportunities to practice the tutored concepts between the two tutoring sessions, which could have affected retention. These are confounding factors in terms of being able to generalize the results of this study.

Acknowledgments. Partial support for this work was provided by the National Science Foundation under grant DUE-1432190.

References

1. Mostafavi, B., Zhou, G., Lynch, C., Chi, M., Barnes, T.: Data-driven worked examples improve retention and completion in a logic tutor. In: Conati, C., Heffernan, N., Mitrovic, A., Verdejo, M.F. (eds.) AIED 2015. LNCS, vol. 9112, pp. 726–729. Springer, Heidelberg (2015)
2. Wouters, P., et al.: A meta-analysis of the cognitive and motivational effects of serious games. J. Educ. Psychol. 105, 249–265 (2013)
3. Cepeda, N.J., Vul, E., Rohrer, D., Wixted, J.T., Pashler, H.: Spacing effects in learning, a temporal ridgeline of optimal retention. Psychol. Sci. 19(11), 1095–1102 (2008)
4. Federici, S., Medas, C. and Gola, E. (2018) Who Learns Better - Achieving Long-term Knowledge Retention by Programming-based Learning. In Proceedings of the 10th International Conference on Computer Supported Education (CSEDU 2018) - Volume 2, pages 124-133
5. Bassok, M., and Novick, L.R. (2012). Problem solving. In K.J. Holyoak and R.G. Morrison (Eds.), Oxford handbook of thinking and reasoning (pp. 413-432). New York: Oxford University Press.
6. Martin, S.A., and Bassok, M. (2005). Effects of semantic cues on mathematical modeling: Evidence from word-problem solving and equation construction tasks. Memory and Cognition, 33(3), 471-478.
7. Amruth N. Kumar. A Model for Deploying Software Tutors. IEEE 6th International Conference on Technology for Education (T4E), Amritapuri, India, 12/18-21/2014, 3-9.
8. Amruth N. Kumar. Long Term Retention of Programming Concepts Learned Using Software Tutors. Proceedings of Intelligent Tutoring Systems (ITS 2020). Athens, Greece, 6/8-12/2020, LNCS 12149, 382-387.
9. LeBlanc, K., & Simon, D. (2008). Mixed practice enhances retention and JOL accuracy for mathematical skills. 49th Annual Meeting of the Psychonomic Society, Chicago, IL.
10. Yutao Wang and Joseph E. Beck. (2012) Using Student Modeling to Estimate Student Knowledge Retention. Proceedings of 5th International Conference on Educational Data Mining. 200-203.
11. Amruth N. Kumar. (2003) A Reified Interface for a Tutor on Program Debugging. Proceedings of Third IEEE International Conference on Advanced Learning Technologies (ICALT 2003), Athens, Greece, 7/9-11/2003, 190-194.