Subgoal Learning Integration in a CS1 Course

Olivier Goletti[1] [0000−0002−1610−4985]

ICTEAM/INGI, UCLouvain, Louvain-la-Neuve, Belgium
olivier.goletti@uclouvain.be

Abstract. Introductory programming courses are sometimes too ambitious for novice programming students. Since a lot of university level programming courses make use of undergraduate teaching assistants, we decided to train them with evidence-based instructional strategies that will benefit students' learning. Over the course of five years, we iterated on the integration of instructional strategies through the training of the undergraduate teaching assistants of a CS1 course. Based on an analysis of the course, subgoal learning was selected as an evidence-based instructional strategy to promote learning transfer and reduce cognitive load for students. This paper details the methodology used to integrate subgoal learning in our third iteration. We discuss how insights from previous iterations instructed our design choices, challenges to change an established CS1 course and how we selected the resources and where changes were made in the course to integrate subgoal learning.

Keywords: Subgoal Learning · CS1 · Educational Change.

Context

I am in the fifth year of my doctoral studies at UCLouvain where I am a teaching and research assistant. This means that I dedicate nearly half of my time to teaching and the other half to research. My supervisor at UCLouvain is Prof. Kim Mens and Prof. Felienne Hermans is my co-supervisor at Vrije Universiteit Amsterdam.

I am in the process of completing the third and last iteration step of my research on the integration of evidence-based instructional strategies to undergraduate teaching assistants (UTAs). I hope that the Didapro 10 Doctoral Consortium will allow me to take a step back on this last step in order to help me plan the organisation of my dissertation text.

Motivation

A lot of computer science pedagogic approaches seem to be inspired by constructionism and learner-centered methodologies [6]. The goal of my research is to explore how I can foster learning transfer in this context through the training of UTAs to use explicit evidence-based programming strategies for difficult CS concepts.

Background

Learning CS and programming in particular has been described as hard [12] while others insist that the ambitions of CS1 courses are to blame instead [13]. In the course of multiple iterations guided by the principles of *Design-Based Research* [1][4], I selected and integrated instructional strategies to UTAs. Multiple evidence-based strategies were selected based on *learning transfer* and *cognitive load theory*. I then explore(d) how undergraduate teaching assistants could grasp and use these instructional strategies to teach their students. The third iteration of my research focuses on *Subgoal Learning*; other studied strategies were discussed in prior work [9, 10]. We will briefly define all these key notions below.

Design-Based Research is a research approach focusing on the iterative development of a pedagogical innovation while observing and researching it during the intervention. Research is done in an ecological setup such as classrooms, guided by design principles that will be refined, adjusted or dropped in subsequent iterations [1]. Each iteration cycle is starts by a preparation and design phase, then the intervention itself takes place and finally, a redesign is done based on the analysis of the previous iteration.

Learning Transfer is the ability to reuse previously acquired knowledge in a new target task. Learning transfer has been studied and modeled as multiple interacting processes, including knowledge encoding in memory and knowledge retrieval conditioned by its organisation and accessibility in long-term memory [18, 3].

Cognitive Load Theory (CLT) [19] is an instructional theory based on human cognitive architecture. CLT assumes a limited working memory. The impact of CLT on instructional design is that one should try to reduce load on the working memory when teaching new material. A lot of effects and instructional impact have been proposed by CLT proponents [17].

Subgoal Learning Among the effects predicted by cognitive load theory, the worked example effect shows that exposing students to detailed solutions to problems improves their learning [20] in programming courses among other disciplines [16, 5]. Labelling the steps used by experts to solve these problems draws the learner's attention to the generic aspect of these solution steps [14, 16]. Subgoal learning is the idea of explicitly teaching the steps of recurring patterns in the resolution of similar problems to favour learning transfer. Margulieux et al. [15] combined the idea of worked examples with labeling the steps behind common code constructs, proposing subgoal labels worked examples (SLWEs). In the end, the subgoals are taught to the students by interleaving SLWEs and practice exercises [15] progressively diminishing the guidance as prescribed by CLT [11] . The SLWEs studied by Margulieux et al. suggest that *"subgoal materials helped learners to solve problems using the procedure more effectively during the early stages of learning even though no performance difference between groups was found on the exams"*.

Research Goals and Progress

After our analysis of a CS1 course [7], we conducted a first iteration where we trained four undergraduate teaching assistants with four strategies that led us to introduce four criteria to select instructional strategies: the strategy should be easy to understand, straightforward to apply, useful on the long term and supported by literature [9].

Based on those criteria, our analysis of the first iteration led us to select only two strategies to not overload our UTAs and also to introduce both strategies earlier in the semester. We also created dedicated training material to help the UTAs integrate the different aspects of the strategies [10]. For the second iterations, we also measured the *Fidelity of Implementation* [2] of the two strategies by the UTAs by observing, recording and coding recordings of their teaching. While in the first iteration, we used subgoals from the literature, in the second iteration, we developed our own subgoals for the concepts and language of our own CS1 course [8].

Finally, in order to prepare our last iteration focusing on subgoal learning (SL), the main design change was to integrate the strategy globally in the course. The most mentioned issue by UTAs in the focus groups and follow up discussions for them to use SL in the course was that it took them too much time to properly present a specific worked example and the corresponding subgoals and labels. UTAs argued that they would benefit from an introduction of the labels done in the course material. Moreover, we observed that for the other strategy used in the second iteration, an important trigger for strategy use was to have prompts in students' exercises statements.

We identified the resources of the course that could be impacted by the integration of SL and designed a fading approach of the integration of subgoals and labels in the course, in accordance with CLT inspired instructional principles. We identify two different type of Subgoal Labels utilization. First, Step-By-Step SLWEs (SBS) are worked examples that use each label one by one to illustrate at each step how the generic corresponding subgoal for that concept is used in a concrete example. The integration of those SBSs is mainly done by the teachers during courses, but a UTA using the labels and solving procedure to provide students with an exercise solution could also be using SBSs. In order to integrate SBSs in the course, we had to modify teachers slide decks and this took some work and back and forth with teachers. Second, Subgoal Labeled Final Solutions (SLFS) that would be used after the labels have already been introduced once. The idea is to show final solution or solved examples without necessarily going through the whole solving process but by annotating what code pertains to what label. Typically, those could be shown in the teachers slides to remind students of the labels or the result of an UTA annotating code on the blackboard during a lab session.

The integration of SL in the course consisted in the modification of the slides with SBSs and SLFSs. This allowed for the introduction and repetition of the labels for the students in a passive way. Since we knew that prompts in students' exercises statements were also triggers for strategy use, we also added the labels

in the first exercises making use of the targeted concepts. The labels linked each time to a complete SLFS catalogue where the labels and annotated examples were gathered per concept. The last integration step was, next to training of the UTAs like in previous iterations, to add reminders in the correction guide for tutors so that they knew where and when new concepts were used and which labels to use. This information is also discussed each week during coordination meetings with the tutors.

Expected Contribution

We hope that this multi-level fading integration of subgoal learning in the course will push tutors to use the strategy more often than in our previous iteration. For this third iteration, since SL is completely integrated in the course as an instructional strategy, all UTAs and not just only volunteers like in previous iterations will need to apply the strategy. Our hypothesis is that observations will yield more strategy use and with better fidelity.

References

1. Bakker, A.: Design research in education: A practical guide for early career researchers. Routledge (2018)
2. Borrego, M., Cutler, S., Prince, M., Henderson, C., Froyd, J.E.: Fidelity of Implementation of Research-Based Instructional Strategies (RBIS) in Engineering Science Courses. Journal of Engineering Education **102**(3), 394–425 (2013). https://doi.org/10.1002/jee.20020, https://onlinelibrary.wiley.com/doi/abs/10.1002/jee.20020, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/jee.20020
3. Bracke, D.: Un Modèle Fonctionnel Du Tranfert Pour l'éducation, pp. 77–106. Presses Université Laval (2004)
4. Collective, D.B.R.: Design-based research: An emerging paradigm for educational inquiry. Educational researcher **32**(1), 5–8 (2003)
5. Ericson, B.J., Margulieux, L.E., Rick, J.: Solving Parsons Problems Versus Fixing and Writing Code. In: Proceedings of the 17th Koli Calling International Conference on Computing Education Research. pp. 20–29. Koli Calling '17, ACM, New York, NY, USA (2017). https://doi.org/10.1145/3141880.3141895
6. Falkner, K., Sheard, J.: 15 pedagogic approaches. The Cambridge handbook of computing education research p. 445 (2019)
7. Goletti, O.: En quoi le dispositif mis en œuvre dans le cours d'introduction à l'informatique en BAC1 ingénieur civil basé sur l'apprentissage par problèmes soutient les processus du transfert des apprentissages : l'encodage et l'accessibilité aux connaissances ? Tech. rep., UCLouvain (2019), http://hdl.handle.net/2078.1/245579
8. Goletti, O., De Pierpont, F., Mens, K.: Création d'exemples résolus avec objectifs étiquetés pour l'apprentissage de la programmation avec python. In: Didapro 9–DidaSTIC (2022)

9. Goletti, O., Mens, K., Hermans, F.: Tutors' Experiences in Using Explicit Strategies in a Problem-Based Learning Introductory Programming Course. In: ITiCSE '21. p. 7. ACM Press, Virtual Event, Germany (Jun 2021). https://doi.org/10.1145/3430665.3456348
10. Goletti, O., Mens, K., Hermans, F.: An analysis of tutors' adoption of explicit instructional strategies in an introductory programming course. In: Proceedings of the 22nd Koli Calling International Conference on Computing Education Research. pp. 1–12 (2022)
11. Kirschner, P.A., Sweller, J., Clark, R.E.: Why Minimal Guidance During Instruction Does Not Work: An Analysis of the Failure of Constructivist, Discovery, Problem-Based, Experiential, and Inquiry-Based Teaching. Educational Psychologist **41**(2), 75–86 (Jun 2006). https://doi.org/10.1207/s15326985ep4102$_1$
12. Lister, R., Adams, E.S., Fitzgerald, S., Fone, W., Hamer, J., Lindholm, M., McCartney, R., Moström, J.E., Sanders, K., Seppälä, O.: A multi-national study of reading and tracing skills in novice programmers. In: ACM SIGCSE Bulletin. vol. 36, pp. 119–150. ACM (2004)
13. Luxton-Reilly, A.: Learning to program is easy. In: Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education. p. 284–289. ITiCSE '16, Association for Computing Machinery, New York, NY, USA (2016). https://doi.org/10.1145/2899415.2899432, https://doi.org/10.1145/2899415.2899432
14. Margulieux, L.E., Guzdial, M., Catrambone, R.: Subgoal-labeled instructional material improves performance and transfer in learning to develop mobile applications. In: Proceedings of the ninth annual international conference on International computing education research. pp. 71–78 (2012)
15. Margulieux, L.E., Morrison, B.B., Decker, A.: Design and Pilot Testing of Subgoal Labeled Worked Examples for Five Core Concepts in CS1. In: Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education - ITiCSE '19. pp. 548–554. ACM Press, Aberdeen, Scotland Uk (2019). https://doi.org/10.1145/3304221.3319756
16. Morrison, B.B., Margulieux, L.E., Guzdial, M.: Subgoals, Context, and Worked Examples in Learning Computing Problem Solving. In: Proceedings of the Eleventh Annual International Conference on International Computing Education Research. pp. 21–29. ACM Press (2015). https://doi.org/10.1145/2787622.2787733
17. Sweller, J., van Merriënboer, J.J., Paas, F.: Cognitive architecture and instructional design: 20 years later. Educational Psychology Review pp. 1–32 (2019), publisher: Springer
18. Tardif, J.: Le Transfert Des Apprentissages. Editions logiques (1999)
19. van Merriënboer, J.J.G., Sweller, J.: Cognitive Load Theory and Complex Learning: Recent Developments and Future Directions. Educational Psychology Review **17**(2), 147–177 (Jun 2005). https://doi.org/10.1007/s10648-005-3951-0
20. Ward, M., Sweller, J.: Structuring effective worked examples. Cognition and instruction **7**(1), 1–39 (1990)