Improving Programming Instruction with Subgoal Labeled Instructional Text

Lauren Margulieux (l.marg@gatech.edu)
Georgia Institute of Technology, School of Psychology
Atlanta, GA 30332-0170 USA

Richard Catrambone (rc7@prism.gatech.edu)
Georgia Institute of Technology, School of Psychology
Atlanta, GA 30332-0170 USA

Abstract

In science, technology, engineering, and mathematics (STEM) education, problem solving tends to be highly procedural, and these procedures are typically taught with general instructional text and specific worked examples. Subgoal labels have been used in worked examples to help learners understand the procedure being demonstrated and improve problem solving performance. The effect of subgoal labels in *instructional text*, however, has not been explored. The present study examined the efficacy of subgoal labeled instructional text and worked examples for programming education. The results show that learners who received subgoal labels in both the text and example are able to solve novel problems better than those who do not. Subgoal labels in the text appear to have a different effect, rather than an additive effect, on learners than subgoal labels in the example. Specifically, subgoal labels in text appear to help the learner articulate the procedure, and subgoal labels in the example appear to help the learner apply the procedure. Furthermore, having subgoal labels in both types of instruction might help learners integrate the information from those sources better.

Keywords: STEM education; subgoal learning; worked examples; procedural text.

Introduction

Knowledge of computing is increasingly necessary in our society. As computing advances, individuals need to understand more about it to understand technical information and make well-informed decisions. Moreover, individuals with advanced computing knowledge are needed to fill increasingly technical jobs and promote innovation. To reflect these societal goals, a major learning goal for computing is that students understand core concepts and principles with the underlying expectation that they can transfer their knowledge to solve problems or critically evaluate information.

In computing like in other STEM subjects, both instructional text and worked examples are used to provide instruction that is abstract enough to apply to novel problems and concrete enough to grasp (Trafton & Reiser, 1993). Instructional text describes a procedure abstractly (LeFevre & Dixon, 1986) and provides information about reasoning within a domain (Reder & Anderson, 1980); worked examples demonstrate how to apply procedures to specific problems. Worked examples are typically used by students as the primary method to learn procedures (LeFevre & Dixon, 1986) because they take less effort to understand than instructional text (Eiriksdottir & Catrambone, 2011). Using worked examples in this way, however, can inhibit transfer to novel problems because they are specific to a particular context, and learners are commonly not able to glean abstract information from these concrete examples. To improve this type of transfer, examples that emphasize subgoals have been used (e.g., Catrambone, 1998; see Figure 1).

Subgoal Labeled Worked Example

Create Component

1. From the basic palette drag out a label.
2. Place the label underneath the image.

Set Properties

3. Set the text to *Click button to see your fortune*.
4. Rename it to *fortuneLabel.*

Unlabeled Worked Example

1. From the basic palette drag out a label.
2. Place the label underneath the image.
3. Set the text to *Click button to see your fortune*.
4. Rename it to *fortuneLabel.*

Figure 1: Worked examples with and without subgoal labels.

To understand what a subgoal is, consider a complex problem solution. Achieving the solution would be the overall goal, and the problem solver takes many individual steps towards that goal. Subgoals are in-between; they are functional pieces of the solution achieved by completing one or more individual steps. The same subgoals tend to appear across problems within a topic area; therefore, teaching learners to identify and achieve subgoals increases their success at solving novel problems (Catrambone & Holyoak, 1990).

Research on subgoal labeled worked examples suggests that improved outcomes caused by subgoal labels stems from three sources: highlighting the structure of the worked example for the learner (Atkinson & Derry, 2000; Catrambone, 1995a), helping the learner mentally organize information (Catrambone, 1995b), and inducing the learner to self-explain the examples (Catrambone, 1998; Renkl & Atkinson, 2002). Though subgoal labels improve learning from worked examples, the effect of subgoal labels in instructional text has not been explored.

Subgoal labels in instructional text (see Figure 2) might provide extra guidance that would help learners use and understand the information in the text better. Subgoal labels in both types of instructional material also might help text and examples complement each other better by connecting related information with the same subgoal labels. This type of presentation might help learners integrate information presented in each type of instruction.

Subgoal Labeled Instructional Text

Create Component

Components are the pieces that provide your app functionality, such as a *button* that users can press or a *label* to display…

Set Properties

You'll be able to change the properties of each component in the App Inventor Designer as well. For example, you can change…

Unlabeled Instructional Text

Components are the pieces that provide your app functionality, such as a *button* that users can press or a *label* to display…

You'll be able to change the properties of each component in the App Inventor Designer as well. For example, you can change…

Figure 2: Instructional text with and without subgoal labels.

In summary, subgoal labels in the instructional text and worked examples are expected to

- help learners understand problem solving procedures in a way that enables transfer to novel contexts,
- guide learners' mental organization of knowledge,
- help learners integrate information from various sources,
- and help learners understand information by encouraging learning strategies like self-explanation.

Overview of Experiments

The present study explored the effectiveness of subgoal labeled instructional materials compared to unlabeled instructional materials to teach computer programming. Participants learned to create applications (apps) for Android devices using Android App Inventor. This computer programming language was chosen because it is a drag-and-drop language. Drag-and-drop programming languages are effective for teaching novices because, instead of writing code to create programs, users drag components from a menu and place them together like puzzle pieces. This type of code creation is more easily understood by novices (Hundhausen, Farley, & Brown, 2009). Instructions from the ICE Distance Education Portal (Ericson, 2012) were used to develop instructional materials. Materials in all conditions were identical except for the subgoal labels. Subgoals were determined using the Task Analysis by Problem Solving (TAPS; Catrambone, 2012) technique with subject-matter experts.

For instruction, participants received text detailing how to create apps (i.e., instructional text; excerpt in Figure 2) and a video demonstration and textual step-by-step guide detailing how to create a Fortune Teller app (i.e., worked example; excerpt in Figure 1). A video demonstration (of an expert making the app and explaining the procedure) was used because videos can quickly and naturally show learners to use direct-manipulation interfaces (Palmiter, Elkerton, & Baggett, 1991) like App Inventor. Participants were also asked to make the app themselves using the step-by-step guide because studying an example and applying the procedure can lead to better learning than studying alone (Trafton & Reiser, 1993).

Experiment 1

Experiment 1 explores the efficacy of subgoal labels in instructional text. The assessment tasks in this experiment were designed to measure participants' skill in problem solving and mental representations of information learned.

Method

Participants Participants were 120 students from a mid-sized university who received class credit for participation. Participants must not have had experience with App Inventor or taken more than one course in computer science or programming. These restrictions were necessary because instructions were designed for novices.

Procedure Sessions were between 70 and 90 minutes depending on how quickly participants completed the protocol. During the sessions, experimenters provided technical support and answered questions about the study (e.g., "Can I watch the video again?") but did not answer questions about the instructions or App Inventor (e.g., "How do I make a button?"). First, participants filled out a demographic questionnaire to provide information about possible predictors of programming performance (Rountree, Rountree, Robins, & Hannah, 2004; i.e., age, gender, field of study, SAT scores, high school and college GPA, year in school, number of completed credits, primary language, computer science experience, comfort with computers, and expected difficulty of learning App Inventor).

Next was the instructional period. During this time, participants received the instructional materials. Examples of subgoal labeled and unlabeled materials are in Figures 1 and 2. For participants who received subgoal labeled worked examples, the video presented subgoal labels in pop-up text boxes that did not cover the part of the interface that was being used. Participants had up to 30 minutes to create the app using the instructions and the App Inventor website. Next was the assessment period. During the assessments, participants could not access the instructional materials, but they could access the App Inventor website and the app that they had created (to serve as a memory cue to aid problem solving).

The assessment tasks included 1) a problem solving task, 2) an explanation task, and 3) a generalization task. The problem solving task asked participants to list the steps that they would take to make parts of an app (e.g., "Write the steps you would take to italicize the fortune presented," or "Write the steps you would take to create a list of colors and make the ball change to a random color whenever it collided with something"). This assessment was meant to measure how well participants could solve novel problems. In the explanation task, participants were given an expert's solutions for the previous problem solving tasks and asked to group steps of the solutions however they thought apt. Then, participants described what each group achieved. This assessment was meant to measure how well participants could explain solutions. The generalization assessment asked participants to describe the general procedure that they would take to create an app with given specifications. This assessment was meant to measure how well participants could abstractly describe the problem solving procedure that they learned in the session.

Design The experiment was a two-by-two, between-subjects, factorial design: the format of instructional text (subgoal labeled or unlabeled) crossed with the format of the worked example (subgoal labeled or unlabeled). The dependent variables were performance on tasks.

Results and Discussion

Of the demographic information collected as possible predictors, none correlated with performance on the tasks and will not be discussed further.

Problem Solving Performance For this task, participants earned one point for each correct step they took towards the correct problem solution. This scoring scheme afforded more sensitivity than judging an entire solution as correct or incorrect. The maximum score that participants could earn was 22. Participant responses were scored by two raters, and interrater reliability was measured with intraclass correlation coefficient of absolute agreement (ICC(A)). ICC(A) for this assessment was .94.

There was a main effect of example design consistent with previous literature (e.g., Margulieux, Guzdial, & Catrambone, 2012). Participants who received subgoal labels in the example (*M* = 13.1, *SD* = 6.0) performed better than those who did not (*M* = 5.5, *SD* = 4.8), *F* (1, 116) = 70.19, *MSE* = 24.47, *p* < .001, est. ω^2 = .32, *f* = .76. A main effect of text design was also found. Participants who received subgoal labels in the text (*M* = 11.0, *SD* = 7.1) performed better than those who did not (*M* = 7.6, *SD* = 5.7), *F* (1, 116) = 13.90, *MSE* = 24.47, *p* < .001, est. ω^2 = .06, *f* = .34. In addition, there was an ordinal interaction between text design and example design, *F* (1, 116) = 12.82, *MSE* = 24.47, *p* = .001, est. ω^2 = .05, *f* = .57. This interaction shows that participants who received subgoal labels in the text performed better than those who did not only when they also received subgoal labels in the example. This pattern suggests that the interaction caused a main effect of text, but closer evaluation showed that there was no simple main effect of text design (see Table 1). This interaction between text type and example type might have occurred because learners in procedural domains typically rely on worked examples to demonstrate how to apply domain knowledge to problem solving (LeFevre & Dixon, 1986). Therefore, in order for subgoal labeled text to aid problem solving performance, it might need to be accompanied by subgoal labeled examples to guide application.

Table 1: Post-hoc analyses of problem solving task. Note: SL = subgoal labeled, UL = unlabeled, and ex. = example.

Condition	***n***	***M***	***SD***	***t***	**Std. error**	***p***
SL text, SL ex.	30	16.4	4.3			
				5.08	1.30	<.01
UL text, SL ex.	30	9.8	5.6			
				3.18	1.36	<.01
SL text, UL ex.	30	5.6	4.8			
				.106	1.33	.92
UL text, UL ex.	30	5.5	4.9			

Attempted Problem Solutions To better understand participants' performance, the problem solving tasks were also scored for how much of the solution participants attempted. This score is meant to measure how many functional components of the solutions the participants attempted, regardless of whether their answers were correct. A high score would suggest that a participant recognized the components needed in the solution, even if they could not correctly achieve each component.

To calculate this score, the correct solutions for the problem solving tasks were deconstructed into the subgoals, or functional components, that were necessary to complete the solution. Participants earned a point for each subgoal that was attempted. Attempting a subgoal was operationally defined as listing at least one step required to achieve the subgoal, listing a step that would achieve a similar function (e.g., listing a step to change a property regardless of whether it was the correct property), or describing the subgoal. The maximum score that participants could earn was 10. ICC(A) for this assessment was .95.

There was a main effect of example design. Participants who received the subgoal labeled example (*M* = 6.9, *SD* = 2.7) attempted more subgoals than those who did not (*M* = 4.1, *SD* = 2.8), *F* (1, 116) = 30.43, *MSE* = 7.73, *p* < .001, est. ω^2 = .20, *f* = .50. No other statistically significant differences were observed (see Table 2). These results, in conjunction with problem solving performance, suggest that the subgoal labeled text did not prompt participants to attempt more components but, when paired with the subgoal labeled example, helped them correctly achieve more of their attempted components.

It is possible that receiving more instantiations of each subgoal label, whether in text or in additional subgoal labeled examples, would allow learners to compare more instances, refine their procedural rules, and solve problems better. Though this possibility is not directly explored in the present study, the results from other tasks suggest that subgoal labels have a different effect on learners when presented in instructional text than when presented in worked examples.

Table 2: Post-hoc analyses of attempted problem solutions. Note: SL = subgoal labeled, UL = unlabeled, and ex. = example.

Condition	***n***	***M***	***SD***	***t***	**Std. error**	***p***
SL text, SL ex.	30	7.0	2.6			
				.53	.70	.60
UL text, SL ex.	30	6.7	2.8			
				3.42	.71	<.01
SL text, UL ex.	30	4.2	2.8			
				.50	.74	.62
UL text, UL ex.	30	3.9	3.0			

Explanation Task The participants completed an explanation assessment to measure how well they could explain problem solutions. Participants received two scores for this assessment: a grouping score for how well they organized steps and a description score for how well they explained groups. To score the grouping portion of this task, participants received one point for each group that contained only structurally similar steps. They could earn up to nine points. ICC(A) for this assessment was .97.

Participants who received subgoal labels in both the text and example made more correct groups than all others, and there were no other statistically significant differences (see Table 3). To perform well on this task, participants needed to integrate procedural knowledge (to identify structural groups) and application knowledge (to apply the groups to specific problems), and subgoal labels in both types of instructional material might have aided this integration.

To score the description portion of this task, the descriptions that participants gave for the groups were analyzed qualitatively to determine if participants correctly identified their functions. Over 50% of the responses given by participants who received subgoal labeled text correctly described the function of a group of steps. In contrast, less than 10% of the responses given by participants who received unlabeled text correctly described the function.

There was no meaningful difference for example design. Both subgoal labeled and unlabeled example groups produced 30% functional descriptions. Incorrect responses included superficial information such as how the blocks were put together or where in the interface the steps were completed. These results suggest that subgoal labeled text helped learners to better articulate the purpose of steps.

Table 3. Post-hoc analyses of grouping task. Note: SL = subgoal labeled, UL = unlabeled, and ex. = example.

Condition	***n***	***M***	***SD***	***t***	**Std. error**	***p***
SL text, SL ex.	30	4.8	2.5			
				2.51	.57	.02
UL text, SL ex.	30	3.3	1.9			
				.06	.55	.95
SL text, UL ex.	30	3.3	2.3			
				.12	.55	.90
UL text, UL ex.	30	3.2	1.9			

Generalization Task The generalization task was meant to measure how well participants could create a high level description of the procedure. To score this task, participants received a point for each structural feature that they described. Participants did not receive points for specific descriptions (e.g., information about how to achieve a step using the interface) or unnecessary features. The maximum score on this assessment was six. The ICC(A) was .89.

There was a main effect of text design: people who received subgoal labeled text (*M* = 4.4, *SD* = 1.1) performed better than those who did not (*M* = 3.5, *SD* = 1.3), *F* (1, 116) = 15.11, *MSE* = 1.49, *p* < .001, est. ω^2 = .10, *f* = .35. There was no main effect of example design, *F* (1, 116) = 2.70, *p* = .10, and there was no interaction, *F* (1, 116) = .20, *p* = .66. These results are consistent with the explanation task in that subgoal labels in text aided articulation.

Experiment 1 explored the efficacy of subgoal labeled instructional text to teach a programming task. The results suggest that subgoal labeled text helps learners to explain a procedure and to solve novel tasks when paired with subgoal labeled worked examples. Experiment 2 continues this exploration in a different learning scenario.

Experiment 2

Experiment 2 attempted to replicate performance results from Experiment 1 in a more ecologically valid learning scenario. In Experiment 1, participants were not allowed to use instructional materials when solving novel problems, which is not typical in most learning environments. Experiment 2 allowed participants to use instructions during problem solving.

Method

The method for Experiment 2 was the same as for Experiment 1 (i.e., in sample size, selection of participants, procedure, and design). The only differences were that participants could use instructional materials during problem solving and the assessment period included only the problem solving task. The other tasks used in Experiment 1 were meant to measure mental organization of information; therefore, they were not relevant for this experiment.

Results and Discussion

Of the demographic information collected as possible predictors, two were correlated with performance. High school GPA correlated positively, r = .30, p < .01, and number of college credits completed correlated positively, r = .25, p = .01, with score on the problem solving task. These predictors were not expected to confound the analyses of the performance metrics because the variance was evenly distributed among groups, and, therefore, no group had an advantage.

Problem Solving Performance This task and scoring was the same as in Experiment 1. The maximum score was 22. ICC(A) for this assessment was .94.

There was an ordinal interaction between text design and example design, F (1, 116) = 5.87, MSE = 24.26, p = .017, est. ω^2 = .07, f = .22. This interaction demonstrated that participants who received subgoal labels in the text and example outperformed all other groups. There were no other significant differences (see Table 4).

Table 4: Post-hoc analyses of problem solving task in Experiment 2. Note: SL = subgoal labeled, UL = unlabeled, and ex. = example.

Condition	***n***	***M***	***SD***	***t***	**Std. error**	***p***
SL text, SL ex.	30	10.5	6.0	2.20	1.51	.03
UL text, UL ex.	30	7.2	5.0	.37	1.28	.71
SL text, UL ex.	30	6.7	4.3	.30	1.16	.77
UL text, UL ex.	30	6.3	4.1			

Attempted Problem Solutions This score was calculated using the same method as in Experiment 1. The maximum score was 10. ICC(A) for this assessment was .95.

There was no main effect of example design, F (1, 116) = 2.70, p = .10, no main effect of text design, F (1, 116) = 2.21, p = .14, and no interaction, F (1, 116) = 1.40, p = .24. These findings were expected because participants were allowed to use the instructional materials during problem solving and the instructions were the same except for subgoal labels. All participants were equally likely to overlook components of the procedure. In conjunction with problem solving performance, these results suggest that receiving subgoal labels in both the text and example helped participants understand and/or reference the instructions better to solve novel problems.

Subgoal labels in instructional text, in addition to previously discussed benefits, could help learners find information in the text to help them resolve specific problem solving impasses. VanLehn, Jones, and Chi (1992) found that when participants had trouble with a problem, many participants spent a long time searching the text, but only a small proportion found relevant information. Subgoal labels in text might help students who are struggling with a problem to find relevant information more quickly.

Conclusion

The present research advances knowledge about strategies for improving novice problem solving in a STEM domain. The findings provide three important pieces of information about subgoal labeled instructional materials:

- Subgoal labeled text might improve performance only when paired with subgoal labeled examples.
- Subgoal labeled text seems to help learners explain procedures while subgoal labeled examples seem to help learners apply procedures.
- Subgoal labels can lead to better problem solving when the labels appear in both examples and text than when subgoal labels appear in examples alone.

Participants who received subgoal labels in both the text and example outperformed those in other conditions. This effect might have occurred for at least two reasons. First, when learners receive multiple representations of content (e.g., text and example), features that help them translate between those representations leads to better integration and understanding of the information (Ainsworth, 2006). Subgoal labels might have helped learners translate between the two types of instructional materials. Second, receiving the subgoal labeled text, similar to receiving principles in text (Eiriksdottir & Catrambone, 2011), might have helped participants organize information from the general procedure better. Better organization of the general procedure could have led to more effective processing of an example that used the same labels.

The results from the explanation and generalization tasks in Experiment 1 also suggest that subgoal labels in the text led to different benefits than subgoal labels in the example. If learners reviewed enough subgoal labeled examples, they might gather the same type of information offered by subgoal labeled text. This method of learning, however, would likely be less efficient, especially in a domain such as programming that contains complex tasks.

This subgoal intervention manipulates the instructional materials that students receive; therefore, distributing the intervention would be relatively easy. Furthermore, these interventions are not reliant on instructors; therefore, they can be used in a range of learning environments, such as online learning. This study did not explore the efficacy of this manipulation in a learning environment with an instructor, but it could still improve learning. Instructors, as experts, sometimes do not realize how to help learners form useful knowledge representations, partly because much of their procedural knowledge has become automated. Using subgoal labeled materials would ensure that students received the fundamental knowledge that they needed to understand procedures.

Subgoal labeled worked examples have already been shown to significantly increase learners' problem solving performance (Catrambone, 1998). The present study demonstrated that subgoal labeled instructional *text* can increase this effect and improve other types of performance. This study suggests that subgoal labels should be used in both instructional text and worked examples designed to teach problem solving procedures.

Acknowledgments

This research was supported by an APA 2012 APF/COGDOP Graduate Research Scholarship. We thank Frank Durso and Mark Guzdial for their feedback. We also thank Gerin Williams for her help collecting and scoring data.

References

Ainsworth, S. (2006). DeFT: A conceptual framework for considering learning with multiple representations. *Learning and Instruction, 16*, 183-198. doi: 10.1016/j.learninstruc.2006.03.001

Atkinson, R. K., & Derry, S. (2000). Computer-based examples designed to encourage optimal example processing: A study examining the impact of sequentially presented, subgoal-oriented worked examples. *Proceedings of 2000 International Conference of the Learning Sciences*, 132-133.

Catrambone, R. (1994). Improving examples to improve transfer to novel problems. Memory and Cognition, 22, 605-615.

Catrambone, R. (1995a). Following instructions: Effects of principles and examples. Journal of Experimental Psychology: Applied, 1(3), 227-244.

Catrambone, R. (1995b). Aiding subgoal learning: Effects on transfer. Journal of Educational Psychology, 87(1), 5-17.

Catrambone, R. (1998). The subgoal learning model: Creating better examples so that students can solve novel problems. Journal of Experimental Psychology: General, 127, 355-376.

Catrambone, R. (2012). Task Analysis by Problem Solving (TAPS): Uncovering expert knowledge to develop high-quality instructional materials and training. http://cunningham.columbusstate.edu/technology symposium/docs/Catrambone%20white%20paper.pdf

Catrambone, R., & Holyoak, K. (1990). Learning subgoals and methods for solving probability problems. Memory & Cognition, 18(6), 593-603.

Eiriksdottir, E., & Catrambone, R. (2011). Procedural instructions, principles, and examples: How to structure instructions for procedural tasks to enhance performance, learning, and transfer. Human Factors, 53(6), 749-770.

Ericson, B. (2012, February 12). ICE Distance Education Portal. Retrieved from http://ice.cc.gatech.edu/dl/?q =node/641

Gecer, A. (2013). Lecturer-student communication in blended learning environments. *Educational Sciences: Theory and Practice*, *13*(1), 362-367.

Hundhausen, C. D., Farley, S. F., & Brown, J. L. (2009). Can direct manipulation lower the barriers to computer programming and promote transfer of training?: An experimental study. ACM Transactions in CHI, 16(3).

LeFevre, J., & Dixon, P. (1986). Do written instructions need examples? Cognition and Instruction, 3, l-30.

Margulieux, L. E., Guzdial, M., & Catrambone, R. (2012). Subgoal-labeled instructional material improves performance and transfer in learning to develop mobile applications. Proceedings of the Ninth Annual International Conference on International Computing Education Research (pp. 71-78). New York, NY: Association for Computing Machinery.

Palmiter, S., Elkerton, J., & Baggett, P. (1991). Animated demonstrations versus written instructions for learning procedural tasks: A preliminary investigation. International Journal of Man-Machine Studies, 34, 687-701.

Reder, L. M., & Anderson, J. R. (1980). A comparison of texts and their summaries: Memorial consequences. Journal of Verbal Learning and Verbal Behavior, 19, 121-134.

Renkl, A., & Atkinson, R. K. (2002). Learning from examples: Fostering self-explanations in computer-based learning environments. Interactive Learning Environments, 10(2), 105-199.

Rountree, N., Rountree, J., Robins, A., & Hannah, R. (2004). Interacting factors that predict success and failure in a CSI course. SIGCSE Bulletin, 33(4), pp. 101-104.

Trafton, J. G., & Reiser, B. J. (1993). The contributions of studying examples and solving problems to skill acquisition. In Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society (pp. 1017-1022). Boulder, CO.

VanLehn, K., Jones, R., & Chi, M. T. H. (1992). A model of the self-explanation effect. The Journal of the Learning Sciences, 2, 1-59.