Title: How can we design effective instructions to promote transfer?

Author(s): Hee Seung Lee, Shawn Betts, and John R. Anderson

Hee Seung Lee  
Carnegie Mellon University  
heeseung@andrew.cmu.edu

Shawn Betts  
Carnegie Mellon University  
sabetts@andrew.cmu.edu

John R. Anderson  
Carnegie Mellon University  
ja0s@andrew.cmu.edu
Abstract Body

Background / Context:
A major hurdle to learning occurs when a student has to master a skill that requires appreciating hidden structure in a problem. For instance, in understanding that $5x - x = 4x$, the student has to appreciate that $5x - x = 5x - 1x$ and that this can be factored to be $(5 - 1)x$. Without this understanding, the student may learn superficial rules to deal with specific cases, but not be able to generalize to new cases. For instance, in this example a student may infer the superficial rule of decrementing the integer and faced with $-5x - x$ produce $-4x$. To enable understanding of the hidden structure, typical instruction combines an effort to represent the hidden structure along with verbal explanation.

A large number of studies (e.g., Carroll, 1994; Tuovinen & Sweller, 1999) have shown that learning is facilitated by providing worked examples that try to lay out this hidden structure. Although there is strong evidence that learning is facilitated by provision of worked examples, if solution steps are not explained or relevant features are not appropriately highlighted, students are left to discover underlying rationale for each of solution steps and have to generate their own explanations. This often leads to illusion of understanding and thus prevents comprehension of learning materials (Renkl, 2002). However, provision of instructional explanation also does not always guarantee a positive learning outcome (Wittwer & Renkl, 2008). When explanation is provided, students often do not actively make sense of learning materials. Solution steps can be learned and practiced even without conceptual understanding (Hiebert & Lefevre, 1986; Skemp, 1976), but learning without understanding can lead to poor transfer performance. For example, Perry (1991) showed that although children who received principle-based instruction and those who received procedure-based instruction showed comparable learning performance, principle-based instruction led better transfer performance in learning of mathematical equivalence concept. Kieras and Bovair (1984) also reported that when people learned a device model in learning to operate a device, they executed procedures in a faster and more efficient way than those who leaned the procedures by rote without a relevant device model.

Purpose / Objective / Research Question / Focus of Study:
We investigated the role of verbal explanation and problem representation on learning and transfer in mathematics. We used a computer-based tutoring system (Brunstein, Betts, & Anderson, 2009) that was designed to study the learning of an algebra isomorph. The advantage of this system is that can be used to study algebra learning anew in a college population. Learners have to solve algebra problems that are represented as data-flow diagrams as shown in Figure 1(a). Here, an unknown number flows from the top box into the boxes below and the final result is 29 at the bottom. The task is to find what the unknown number is. Figure 1(a) shows the isomorph of algebraic expression, $(5 - x) + (5 * x) = 29$ and Figure 1(d) is equivalent to the expression $5 + 4x = 29$. In Figure 1(d), the unknown top value can be easily determined by propagating the number up from the bottom and doing the arithmetic operations. First, we place 24 into the empty tile above 29, put 6 above it, and then finally put 6 in the top box. However, in Figure 1(a), this simple strategy does not work and at this point learners have to transform the diagram in Figure 1(a) into the form in Figure 1(d). This step is called linearization.

The linearization involved two subtasks: selection and execution. In the selection task, learners have to determine what boxes have to be selected for linearization. To decide what boxes to select, participants have to be able to identify a loop in the diagram. Figure 1(b), a loop is highlighted by the dotted line. The loop starts from the top unknown box where two arrows
diverge and then the loop continues all the way down to the bottom box where the left and right branches converge. In the execution task, students have to determine what values to fill in on the linearized diagram like Figure 1(c). The execution rule is to find the coefficient that multiplies the unknown values and the constant. In this example, the selected boxes are simplified into the expression, \(4x + 5 \((5 - x) + (5 * x) = 4x + 5\). The coefficient 4 goes to the box next to the multiplication operator and the constant 5 goes to the box next to the addition operator.

In this task, this linearization step creates major difficulty that is eased by understanding the relation between the data-flow diagram and algebraic expressions. The present study used this critical relation to manipulate transparency of problem structure and investigated how provision of verbal instruction interacts with this manipulation.

**Research Design:**

A 2 x 2 between-subjects study was designed by crossing instruction type (direct instruction vs. discovery) and representation type (computation vs. expression). First, two types of instruction were provided, direct instruction versus discovery depending on presence of verbal explanation. In the direct instruction condition, participants studied a worked-example where each of solution steps was verbally explained in text message. Each solution step appeared when clicking a forward or backward button and the corresponding text explanations appeared at the bottom of the page. The explanation characterized an action and provided explanation of why a certain action was necessary and how an intermediate answer was computed. In the discovery condition, participants were provided with the identical worked example but without verbal explanations. The text provided for each solution step was about general interface issue such as “click a box” and “enter a number.” This manipulation was applied during the entire learning session.

Second, two types of representation were employed to illustrate hidden structure behind the execution subtask. In the **Computation** condition we illustrated the steps that proficient problem solvers in this system report using. They report separately calculating the coefficient to go in top empty tile and the constant to go in the bottom tile (e.g., separately calculating the 4 and the 5 in going from Figure 1(c) to Figure 1(d). Therefore, the tutor first shows the left part of Figure 2(a) to illustrate the calculation of the coefficient and then the right part to illustrate the calculation of the constant. By focusing on just one component working memory load is reduced. However, a potential problem of directly teaching how to calculate the components is that the equivalent algebraic transformation is not apparent. In the **Expression** condition, illustrated in Figure 2(b), we forgo illustrating how to compute the components and focus on illustrating how the diagram is equivalent to an algebraic expression. This shows how the components of the diagram come together to be the equivalent of an algebraic expression on which they can perform collection of like terms. If students were to actually follow these steps mentally in solving a diagram they would face larger working memory demands but they would have a basis for understanding of the values that they are entering for coefficient and constant.

**Intervention / Program / Practice:**

The study consisted of two sessions, learning session and transfer session. Each session lasted 2 hours and there was 1 or 2 day interval between the two sessions. The transfer session was identical across all experimental conditions and experimental manipulations occurred only in the learning session. The learning session consisted of two problem sections, propagate and linearize problems. Participants solved 20 propagate problems and this required learning to propagate numbers up or down using simple arithmetic calculations. These problems did not involve
linearization. In the linearize problems, participants were given one worked-out example and one guided puzzle first. In these two examples, hints were automatically provided for each solution step based on experimental conditions. After this brief introduction, participants solved 40 linearize problems under one of the four conditions. During the 40 problems, whenever participants made errors and requested help, hints popped up based on experimental conditions.

The transfer session occurred 1 or 2 days after the training session and all participants were tested under identical test conditions. On the transfer session, instead of solving a complete form of data-flow diagrams, students were asked to solve only selection portion and execution portion of the problem separately. In the selection task, participants were shown data-flow diagrams and asked to select a set of boxes which they think should be linearized. In the execution task, the diagram was already linearized by the system and participants had to determine what values to enter on the linearized diagrams. To familiarize participants with this new format of the task, there were 10 warm-up problems for each of selection and execution task. These warm-up problems were randomly selected from the ones participants already solved in the training session. Transfer test items were constructed by changing the structure of the diagram to create new conditions that required understanding the basis of the operations learned in the first session. Examples of selection and execution transfer problems are shown in Figure 3. In the selection task, there were three problem types and eight problems per problem type, resulting in 24 problems in total. In the execution task, there were eight different problem types, and eight problems per problem type, resulting in 64 problems in total. Each problem had a 1-min time limit. After solving a problem, an immediate feedback was given on the correctness of the answer. Participants observed their own answer and correct answer presented on the same page for 2 seconds for the case of correct and 10 seconds for the case of incorrect answer.

Participants / Data analysis / Results:

Forty-four undergraduate and graduate students (20 male and 24 female, \(M = 20.74\) years, \(SD = 1.73\) years) at the Carnegie Mellon University participated in the study. Participants received $10/hour plus performance based bonus. Two participants did not show up on the transfer session and these two were excluded from the data analysis (1 instruction-expression, 1 discovery-expression). There were individual differences in terms of number of problems solved during the learning session and three participants who solved less than 10 problems were removed from the analysis as well (1 discovery-computation, 2 discovery-expression). This resulted in a total of 39 participants (10 instruction-computation, 10 instruction-expression, 10 discovery-computation, and 9 discovery-expression). Due to the individual differences in terms of the number of solved problems, only the first 16 problems were chosen for analysis of learning performance. Two types of errors were analyzed, selection error and execution error. The numbers greater than 5 were re-coded as 5.

A 2 x 2 between-subjects analysis of variance (ANOVA) was performed. The mean number of selection and execution errors participants made during the learning session is shown in Figure 4. Although students from the discovery-computation condition seemed to show more selection errors than those in other conditions, the interaction did not reach significance, \(F(1, 35) = 2.43, p = .13\). In contrast, there was a significant effect of instruction type on execution errors in that students who were given direct instruction \((M = 0.71, SD = 0.47)\) made significantly fewer execution errors than the students who learned under discovery condition \((M = 2.31, SD = 1.42)\), regardless of computation type, \(F(1, 35) = 22.71, p < .001\).
After the learning session, some participants felt lost and wanted to quit the study. Also, participants who showed more than the mean of 2.5 execution errors during the learning session were removed from the transfer data analysis. As a result, only 7 people and 4 people were left for the discovery-expression and discovery-computation condition. This could have left only better learners in these two conditions whereas all learners were chosen in the other two conditions. To avoid possible selection bias, only top half of the participants were chosen in instruction conditions as well. This finally resulted in 5 instruction-computation, 5 instruction-expression, 4 discovery-computation, and 7 discovery-expression.

A 2 x 2 between-subjects ANOVA was performed on the transfer results. In both selection and execution transfer test, transfer performance differed depending on representation type regardless of instruction type. Figure 5 shows the mean number of incorrectly solved problems from selection (left) and execution (right) transfer test. In the selection task, students who were provided with illustrations of algebraic expressions showed fewer errors ($M = 6.32$, $SD = 4.14$) than those who were given illustrations of component computations ($M = 11.07$, $SD = 2.11$), $F(1, 17) = 8.66$, $p = .009$. Likewise, in the execution task, students from the expression condition showed fewer errors ($M = 19.57$, $SD = 13.34$) than those from the computation condition ($M = 33.35$, $SD = 4.94$), $F(1, 17) = 8.71$, $p = .009$.

Conclusions:

To summarize, provision of verbal instructions and choice of problem representation seemed to have effects on different aspects of learning. First, the instruction type mostly affected on initial period of learning in that students given direct instruction showed less floundering as shown in fewer execution errors than students who were left to learn by discovery. The instruction, however, had a significant effect on only execution task, not on the selection task. This suggests that when a relevant feature is highlighted by a scaffolding method such as a loop highlighter shown in Figure 1(b) besides verbal instruction, instruction becomes irrelevant. Second, choice of problem representation affected the transfer performance in that students provided with representation of algebraic expressions transferred better to novel problems than those provided with representation of component computation. Although this manipulation did not seem to affect during initial learning, selection of problem representation appeared to develop different levels of understanding of the problems. When students learned with an algebraic representation, the structure of the diagram appeared to become transparent and they showed greater conceptual understanding of the problems. In contrast, students who learned with a representation of component computation seemed to focus mostly on how to compute values without understanding of how the diagram works and in turn they could not transfer their algorithmic learning to novel problems when problems had different structures on the transfer test.

The most striking finding of the current study was the better transfer performance from the expression students than the computation students in the selection task. The representation manipulation was supposed to help students understand execution rules and all students were given the same kind of feature in terms of selection rules. In both representation conditions, the identical loop highlighter was used to help students learn selection rules as shown in Figure 1(b) except different types of verbal instruction based on instructional conditions. This suggests that it is critical to develop problem representation not only for directly related aspects of problem solution but also for indirectly related aspects of problem solution. It seems that the better representation of the execution subtask helped learners acquire better problem schema and this overall increased transfer performance regardless of task type.
Appendices

Appendix A. References


Appendix B. Tables and Figures

Selection task: Equivalent of \((5 - x) + (5 * x) = 29\)

(a) Loop highlighter is OFF

\[\begin{array}{c}
5 - \text{[Box]} \\
\downarrow \\
+ \\
\downarrow \\
29
\end{array}\]

(b) Loop highlighter is ON

\[\begin{array}{c}
5 - \text{[Box]} \\
\downarrow \\
+ \\
\downarrow \\
29
\end{array}\]

Instruction: “29 cannot be propagated up because the box above it has two little tiles resulting in two paths leading back to the unknown. You can eliminate this loop. You start by clicking the box at the bottom of the loop.”

Discovery: “Click a set of boxes highlighted by the green line.”

Execution task

(c) Intermediate state of linearization

\[\begin{array}{c}
5 \text{[Box]} \\
\downarrow \\
* \\
\downarrow \\
29
\end{array}\]

(d) Result of linearization:

\[
\begin{align*}
5 + 4x &= 29 \\
5 + &\quad \text{[Box]} \\
\downarrow &\quad \\
29
\end{align*}
\]

Figure 1. Stages of an example problem. Depending on the instructional conditions, different hint texts were provided for the selection task and examples are shown below the selection task example for direct instruction and discovery condition.
(a) Component Computation

Coefficient

Constant

(b) Algebraic Expression

Instruction: “Enter 4 because the sum of the X’s is 4 (-1X + 5X = 4X)”
“Enter 5 because the sum of the numbers is 5 (5 + 0 = 5)”
Discovery: “This illustrates how to fill in blue tiles.”

Figure 2. Example illustration for computation and expression condition. Depending on the instructional conditions, different hint texts were provided and examples are shown below the figures for direct instruction and discovery condition. Hint texts were identical for computation and expression condition.
Figure 3. An example of (a) selection and (b) execution transfer problems. (a) Selection problem: Correct boxes are indicated by arrows. Participants have to find a minimal loop among the multiple loops present in the diagram. (b) Execution problem: Correct values are indicated by arrows. Here students have to understand \( x + \{(-2 + x) + -6\} = -22 \) \( = 2x - 8 = 2*(-4 + x) \).
Figure 4. Mean number of selection errors (left) and execution errors (right) observed in the learning session. The Error bars represent 1 standard error of the mean.

Figure 5. Mean number of incorrectly solved problems from the selection (left) and execution (right) task in the transfer session. The Error bars represent 1 standard error of the mean.