Supporting Mathematical Problem Posing with a System for Learning Generation Processes through Examples

Kazuaki Kojima, Learning Technology Laboratory, Teikyo University, Tochigi, Japan
Kazuhisa Miwa, Graduate School of Information Science, Nagoya University, Nagoya, Japan
Tatsunori Matsui, Faculty of Human Sciences, Waseda University, Saitama, Japan

Abstract. Problem posing, by which learners create new problems by themselves, is an important activity in mathematics education. However, novice learners have difficulty in posing problems, particularly when formulating appropriate solution structures of problems. Although they are provided with example problems that can serve as hints for composing novel problems, they do not necessarily understand the key ideas used to generate the examples. To improve problem posing for novices, this study discusses an approach that supports learning from examples as a production task. We propose a method of learning from examples through imitation, where a learner reproduces problems identical to given examples. We implement a system that presents examples of problem posing and supports learners in understanding the examples by having the learners reproduce them. We conducted an experimental evaluation in which learners learned from an example that embeds useful ideas to alter solution structures in the system. The results demonstrated that the learners successfully adapted the example when posing their own problems if they learned the example by the reproduction method. Thus, learning from examples through reproduction appears to be effective in the domain of problem posing as a production task.

Keywords. Problem posing, production task, learning from examples, learning through reproduction

INTRODUCTION

Generally, students learn mathematics by solving problems provided by a teacher or from textbooks. However, in addition to problem solving, problem posing, by which learners create problems by themselves, has also been identified as an important activity in mathematics education. In fact, some mathematicians and mathematics educators have pointed out that problem posing lies at the heart of mathematical activity (e.g., English, 1997b; Polya, 1945; Silver, 1994).

Problem solving and problem posing are not entirely different cognitive activities but are closely related. On the other hand, they differ, of course, in the features and formats of their tasks. Problem solving is a convergent task, by which a learner extracts a mathematical structure from given information and reaches a correct answer. In contrast, problem posing is a synthetic activity and a divergent task that fundamentally has multiple answers. Here we call the former task a comprehension task and the latter a production task. Problem posing requires productive thought. In problem posing, a learner must generate novel ideas because generally, new problems cannot be composed using only
Learning from examples is necessarily adopted in initial skill acquisition. Regardless of the problem domain and task format, examples are indispensable for learning (e.g., example problems in mathematics or physics, papers on related research in academic research, existing products in art or mechanics design, or performances by experts in sports). However, activities that facilitate learning from such examples and supporting methods for these activities must be designed according to the problem domain and task format. In learning by problem solving, a learner comprehends the solution methods in problems provided as examples. However, when learning problem posing, the learning activities used in problem solving may not be sufficiently effective because the cognitive processes used in comprehension and production tasks are different. It is difficult for novice learners to pose new problems, even if they can easily solve similar problems in general. They have difficulty particularly in generating ideas for the formulation of mathematical structures in the solutions of new problems (Kojima, Miwa & Matsui, 2010a). Even if they are provided examples that can serve as hints for composing the structures, they cannot necessarily learn useful ideas by only seeing the examples (Kojima & Miwa, 2008). To benefit from learning by problem posing, learners have to engage in composing appropriate and effective problems. For this purpose, they have to improve their problem-posing skills in terms of the composition of mathematical structures. Thus, activities that facilitate learning from examples appropriate to problem posing as a production task must be designed.

To improve problem posing by novices, this study proposes an approach that supports learning from examples in the domain of problem posing as a production task. In particular, we focus on the design of an activity that facilitates the learning of ideas for composing solution structures in problem posing. To facilitate appropriate problem posing in terms of the composition of solution structures, we introduce the learning activity imitation, where a learner learns from an example by reproducing it. We then implement a support system for this type of learning and experimentally verify its effect on problem posing.

**SUPPORT FOR LEARNING FROM EXAMPLES IN PROBLEM POSING**

**Significance and Difficulties of Problem Posing**

As mentioned above, problem posing is closely related to problem solving. It has been experimentally confirmed that problem-solving ability and problem-posing performance are correlated and that problem posing positively influences problem solving (Bernardo, 2001; Ellerton, 1986; Nikata and Shimada, 2005; Siver & Cai, 1996). For example, Ellerton (1986) investigated the differences in problems posed by mathematically more able and less able students in the sixth to eighth grades. The two groups of students were selected according to their performance on mathematical questions designed to measure their range of reasoning skills. The students were asked to pose difficult problems for their friends without any other constraints and observed that the more able students posed computationally complex problems and could work out the answers to their problems. On the other hand, the less able students posed simpler problems and had difficulty in solving them. Silver and Cai (1996) gave problem-posing and -solving tasks to middle school students. In the problem-posing task, the students were asked to pose mathematical questions from a story that included quantitative information. The performance on the problem posing task was scored in terms of solvability and the
complexity of the linguistic and mathematical structures. The results revealed that students exhibiting high performance in the problem-solving tasks showed higher performance in the problem-posing task than those who exhibited low performance in the problem-solving tasks. Bernardo (2001) experimentally studied analogical transfer from source problems to target problems. In this experiment, junior or senior high school students solved word problems after simply reading worked-out examples (a control group) or posing analogous problems from the examples (an experimental group). The word problems could be solved by transferring what was learned from the examples. The results showed that the performance was higher in the experimental group than in the control group, suggesting that posing analogous problems promotes analogical transfer. In the same way, Nikata and Shimada (2005) proved that the experience of posing analogous problems facilitated problem solving in the domains of Duncker’s radiation problem and analogous problems and word problems involving linear equations. Problem posing offers many benefits; for example, it enhances problem-solving ability and the grasp of mathematical concepts, generates diverse and flexible thinking, alerts both teachers and learners to misunderstandings, and improves learners’ attitudes and confidence in mathematics (English, 1998; Silver, 1994). Although problem posing is rarely adopted in general education owing to certain constraints in practical classrooms, it is as critical a skill as problem solving.

In the research field of mathematics education, recent studies have increasingly addressed teacher education that entails problem posing by prospective teachers. In addition to novice students, those who subsequently become mathematics teachers also have limited experience in problem posing (Crespo & Sinclair, 2008). Toluk-Ucar (2009) experimentally tested prospective teachers’ understanding of mathematical concepts in the domain of fractions before and after they were engaged in learning activities including problem posing. The tests evaluated their understanding of two aspects of working with fractions: correct execution of operations and appropriate representation and explanation of mathematical statements. The results showed that problem posing positively affected both aspects. Similarly, Akay and Boz (2010) examined the effects of problem posing on self-efficacy beliefs in mathematics and attitudes toward mathematics, each of which is positively related to mathematical performance. They experimentally confirmed that problem posing improved the self-efficacy and attitudes of prospective teachers. Furthermore, Crespo and Sinclair (2008) empirically improved problem posing by prospective teachers by two interventions, in which the teachers explored given materials before posing problems from them and developed criteria to describe the quality of problems. These studies demonstrated that prospective teachers benefitted from problem posing in various ways. However, to provide such benefits to novice students who are not particularly skilled in mathematics, some supportive interventions that provide a scaffold for problem posing are required.

It is frequently argued that there is a relationship between problem posing and creativity (Leung, 1997; Silver, 1994), because problem posing has an aspect of creative generation that requires productive thinking. In fact, it has been used in some creative thinking tests to measure the fluency, flexibility, and originality of individuals’ thinking (Sternberg & O’Hara, 1999). Problem posing is needed when using mathematics in daily life or when engaging in mathematical and scientific work (Ishida & Inoue, 1983; Mestre, 2002; Silver, 1994). Mathematics and science advance by continuous problem posing followed by the solution to the problems. In such problem posing, it is critical to distinguish between problem contexts and their structures and to appropriately associate them. Here a context denotes information in problem descriptions indicating something that exists or happens, and a structure denotes the relationships of key elements in the problem.

In learning to pose problems, it is not useful for learners to repeatedly generate similar problems (Hirashima, Yokoyama, Okamoto & Takeuchi, 2007). As described above, problem posers have to
appropriately combine contexts and structures in the solutions of problems. In analogy studies in cognitive psychology, two attributes of problems are recognized as crucial: one is surface features such as contextual settings in problem texts (e.g., purchase of goods or transfer by vehicles), and the other is structural features such as mathematical structures of solutions (Gentner, 1983; Forbus, Gentner & Law, 1995; Holyoak & Thagard, 1995). We refer to these two attributes as situations and solutions. Although it is important for learners to generate diverse problems by extracting several solutions from one situation or by recalling multiple situations to which one solution can be adapted, such diverse problem posing is quite difficult for them. It has been confirmed that problems generated by novice learners lack diversity because of narrow associations between situations and solutions (English, 1998; Mestre, 2002).

We investigated the variety of problems posed by novices to understand the difficulties they encounter in problem posing (Kojima et al., 2010a). Undergraduates were asked to generate new problems from initial problems presented as bases. The bases were simple word problems easily solved by equations. The undergraduates were then encouraged to generate problems as varied and unique as possible. The variety of problems they posed was evaluated according to the four categories shown in Figure 1, indicating similarities in the situations and solutions between their problems and the bases. Category I/I indicates problems that are almost the same as the bases; D/I indicates problems generated by altering the situations of the bases; I/D indicates problems generated by altering the solutions; and D/D indicates problems generated by combining alterations in both situations and solutions. Figure 2 presents examples of posed problems in each category solved by simultaneous equations. The results of the investigation confirmed that the undergraduates posed many problems in categories I/I and D/I, and few problems in I/D. They also revealed that problems in D/I that had situations different from the bases were appropriately composed. On the other hand, problems in I/D and D/D, where the solutions different from the bases were involved, were relatively simple and inappropriate. Although the bases were elementary problems, many of the posed problems were simpler than the bases. These results indicate that the novices could generate novel situations, but failed to create new solutions in problem posing; thus, it is difficult for undergraduates to pose new problems, even if they can easily solve them. We asked undergraduates to evaluate the difficulty of the problem-posing tasks using questionnaires and confirmed that they found the tasks difficult (Kojima & Miwa, 2008). Because of the difficulty, the undergraduates in the investigation appeared to have insufficient knowledge to adequately compose new solutions to the posed problems. Thus, to enhance problem posing by novice learners, we need to design methods to support learners’ idea generation for composing solutions.

![Figure 1. Categories for evaluating posed problems](image-url)
I bought some 60-yen oranges and 120-yen apples for 1020 yen. The total number of oranges and apples was 12. How many oranges and apples did I buy?

Solution:
Let $x$ denote the number of oranges and $y$ denote the number of apples.

\[ x + y = 12 \]
\[ 60x + 120y = 1020 \]

According to the equations above, $x = 7$ and $y = 5$.

Support Systems for Learning to Pose Problems

The research field of Intelligent Tutoring Systems/Artificial Intelligence in Education has long addressed learning from examples. Interactive scaffolding that enhances learning from worked examples has been implemented and its effects have been discussed (e.g., Conati & VanLehn, 2000; Koedinger & Aleven, 2007; Schonke, Renkl, Krieg, Wittwer, Aleven & Salden, 2009; McLaren & Isotani, 2011). However, the central issue in such research is basically limited to problem solving and does not include problem posing.

In several studies, problem posing was adopted as a learning task to facilitate knowledge sharing or interactions among learners in e-learning systems (Barak & Rafaeli, 2004; Hirai, Hazeyama & Inoue, 2009; Takagi & Teshigawara, 2006; Yu, Liu & Chan, 2005). These systems aid problem posing by learners and support peer assessment of problems among learners. It has also been experimentally indicated that learning through such activities in these systems improves learning performance as well as the quality of learner problems. Generally, these e-learning systems adopt multiple-choice format
problems as learning materials without incorporating any function for understanding the structural information of problems; thus, they cannot evaluate the variety of learner problems.

Hirashima et al. implemented several systems that can understand the structures of problems posed by learners. They developed their systems on the basis of a computational representation method (Hirashima, Niitu, Hirose, Kashihara & Toyoda, 1994; Hirashima, Umeda & Takeuchi, 2001), in which solution structures of problems are represented as semantic networks, and surface information is represented as node values in the networks. This enables the implementation of computer-supported learning exercises in order to generate problems solved by specified solutions (Hirashima et al., 2007; Nakano, Hirashima & Takeuchi, 2002) and to alter the instance problems into new ones (Hirashima, Yoshida, Nakano & Takeuchi, 2004; Waki, Ura, Horiguchi & Hirashima, 2009). The major purpose of their studies was to improve learners’ understanding of solution methods or the relationships among problems through problem posing. They empirically confirmed that learning to pose problems through their systems enhanced problem-solving abilities, although they have not addressed the variety of ideas in the problems posed by learners.

Previously, we addressed the implementation of a support system to facilitate learners’ posing of diverse problems (Kojima & Miwa, 2008). In the system, learners are engaged in a task that involves posing mathematical word problems from an initial problem given as a base. They pose new problems and input the texts and equations of their solutions into the system. Our system can automatically understand the situations and solutions in the problems and can evaluate their variety. It estimates the situations in the problem texts on the basis of situation-estimating models, each constructed from independent words in the texts of problems stored in the system’s problem database. It also extracts the structures of solutions by parsing the equations and constructs schemata that represent the structures; the schemata are based on the computational representation of problems by Hirashima et al. (2001). It can also present learners with problems as examples of task output to provide hints for idea generation. The variety of learners’ problems is evaluated, and the presentation of examples is controlled on the basis of the four categories shown in Figure 1. Experimental evaluations of the system confirmed that it can facilitate learners’ posing of diverse problems to some extent. The number of problems posed in the I/I category decreased and those in D/I and D/D increased after the learners had posed problems with the system and had been shown various examples belonging to D/I and I/D by the system. However, the presentation of examples did not increase the number of problems in I/D. The lack of problem posing in I/D was consistent with the results in the investigation mentioned above (Kojima et al., 2010a). Thus, novices found it difficult or unfamiliar to alter the solutions while controlling the situations in problem posing.

Although the system presents examples to learners and prompts them to compare the base with their posed problems, it does not give any instructions on how to learn from the examples. The examples are merely shown to the learners. We have not examined how the learners learned from the presented examples. The learners may have simply read the presented examples, or they may have solved them. In other words, the learners may have understood the examples through performing comprehension tasks. The comprehension of examples may have helped in generating various situations; however, it may not have necessarily facilitated the understanding of the structural features of solutions. For learners to adequately study the structural features from examples, further support must be introduced. Because problem posing is a production task, it must be effective and allow a learner to examine each example through a productive activity. The learner may then gain better understanding about its structural features. According to this insight, we designed a method that allows learning from examples through the productive activity of imitation.
Learning from Examples through a Production Task

In comprehension tasks, several aspects of learning from examples and methods to enhance learning effects have been studied. One such research area is self-explanation (Chi, Bassok, Lewis, Reimann & Glaser, 1989), an activity by which learners generate explanations of learning content. Self-explanation has been demonstrated to improve knowledge acquisition (Chi, de Leeuw, Chiu & LaVancher, 1994). The effects of eliciting and computationally supporting self-explanation in learning to solve problems from examples have also been studied (e.g., Conati & VanLehn, 2000; Renkl, Stark, Gruber & Mandl, 1998; Renkl, 2002). However, to what extent explanation activities can facilitate idea generation in production tasks has not been sufficiently investigated (Kanzaki & Miwa, 2010).

Although the performance of comprehension and production tasks is generally correlated, there are gaps between the two tasks. An example of tasks showing such gaps is sentence comprehension and production by learners of a second language. It is often observed that such learners can understand a language but produce only limited output. That is because the comprehension of a language to understand messages is possible using only lexical information; it does not necessarily require syntactic processing (Krashen, 1982; Swain, 1985). On the other hand, the production of the language requires syntactic processing, and learner production is considerably limited by the required use of syntax. Takakuwa (2001) experimentally demonstrated the gap in sentence comprehension and production performance by Japanese learners of English as a foreign language (EFL). He observed that some learners applied different processing mechanisms in sentence comprehension versus sentence production. Thus, the skills of comprehension and production may not be simultaneously developed and improved through a common activity because structural processing is dispensable. In fact, it is reported that the learning of different tasks such as comprehension and production has no influence on the other (Singley & Anderson, 1989). It has been experimentally confirmed that there was marginal transfer from training on evaluating LISP code to generating LISP code, and vice versa. Such a gap can also arise between mathematical problem solving and posing. Mathematics problems are often solved by simply following routine solution procedures learned from examples, without a deep understanding of the solution structures. On the other hand, problems cannot be appropriately posed without structural understanding, particularly when composing new solutions. Therefore, to improve production performance by learning from examples, it is necessary to adequately learn the solution structures of the examples, and it is effective to do this through productive activities.

We design a method of learning from examples through imitation, a learning activity adopted in productive task domains. Imitation, by which a learner reproduces an existing example work, has long been adopted as a major learning activity in the domains of creative generation, such as art and music. The relationship between imitation and creation has been consistently noted in such creative domains. Thus, we adapt imitation as a tool for learning from examples in problem posing. The effects of imitation have been documented. For example, Ishibashi and Okada (2006) argue that imitating examples can prompt imitators’ understanding of examples and their conceptual background; imitation facilitates a creative performance by imitators. In their experiment, subjects were engaged in an artistic drawing task before and after they created copies of a presented example. It was observed that the subjects deeply understood the example through its imitation. The understanding of the example then elicited understanding of the subjects’ own expressions.

1 Japanese and English are grammatically quite different. For example, the sentence structure in Japanese is the Subject-Object-Verb (SOV) type, and subjects can often be omitted in Japanese sentences. This makes it difficult for Japanese EFL learners to process the syntactic structures of English sentences.
In the learning activity proposed in this study, imitation is used to reproduce a problem identical to a presented example. Learning by reproduction is intended to allow learners to understand the ideas used in formulating the example from the viewpoint of the poser. However, if a learner is asked to simply reproduce a shown example, she/he will merely duplicate the characters and symbols composing the text and solution of the example. The learner learns nothing from such duplication. Thus, we design and implement computational support for learning by reproduction. While learning, a learner is given information on the processes used to generate an example problem. The learner then reproduces the problem by following generation process information.

This study is an expansion of the study mentioned previously (Kojima Miwa, 2008), which implemented a support system that can evaluate problems posed by learners and provide examples of problem posing by adopting AI components such as a production system, case-based reasoning, and natural language processing. These components are used to provide the generation process information of examples because they actually generate concrete instances of problems. The new system in this study is implemented on the basis of the system in the previous study.

IMPLEMENTATION OF A SUPPORT SYSTEM

We implement a new system for learning by reproducing examples in problem posing as an enhancement of the system described in the previous section (Kojima & Miwa, 2008). Here we refer to the earlier system as the Presentation-Only system (P-Only system), because it merely presents examples and offers no further support, and the new system is referred to as the Presentation+Reproduction system (P+R system). The P+R system initially provides a learner with a base problem and a problem-posing task. The problem domain consists of word problems solved by simultaneous equations. The P+R system can present problems as examples generated by altering the base and information on the processes that show how the examples are generated from the base. A learner reproduces problems identical to the presented examples by following generation process information. A built-in system support provides the learner with various ideas regarding problem posing through learning by reproducing examples. Such learning by reproduction is expected to facilitate the understanding of the solution structures for the examples, which in turn is expected to expand the variety of problems the learner can appropriately pose.

The P+R system incorporates the functions of an automatic generation system for mathematical word problems (Kojima, & Miwa, 2005; 2006) in addition to the P-Only system. The generation system semi-automatically generates new problems through interactions with a user acting as a teacher and constructs a database containing problems that vary in their situations and solutions.

Because our system is designed according to the models of analogy which assume that human problem solving distinguishes surface and structural features of problems (Gentner, 1983; Forbus, Gentner & Law, 1995; Holyoak & Thagard, 1995), it can be adapted to any problem domain in which texts express the surface contextual settings and structures in solutions. However, a specific interface has to be designed and implemented for each domain.

Basic Framework of Learning through Reproduction
Figure 3 indicates the basic framework of learning by reproducing examples. In learning with our system, a learner is required to pose new problems from an initially given base. The learner is also presented with problems as examples, each generated by altering the base.

When a learner studies an example that was generated from the base, the system hides the example itself and shows its generation process information to indicate how it was generated from the base (the black bold arrows in Figure 3). Generation process information also includes sufficient information to reproduce the example. The learner generates a problem identical to the example by reproducing how to alter the base as indicated in generation process information (Figure 3 (a)). From the viewpoint of a poser, this learning activity must facilitate the understanding of the essential ideas used to generate the example, particularly those for composing a solution structure. The learner then transfers what is learned through the reproduction into the posing of new problems (Figure 3 (b)). For example, the learner may pose a novel problem by adapting ideas used in the example that add operations to a solution.

**Generation of Problems and their Generation Process Information**

As described above, our system presents examples and process information about how examples are generated. The generation system provides those data. It can generate new mathematical word problems. To generate such word problems, issues in common knowledge and common-sense reasoning must be overcome. Therefore, the generation system adopts case-based reasoning (CBR) and interaction with a user acting as a teacher. CBR is a problem-solving method that uses cases, concrete knowledge comprising past problems and their answers. The generation system reuses common knowledge embedded in these cases (e.g., relationships between fruits and a basket, or pens and a pen case) or learns new knowledge through interacting with the teacher.
The generation system uses problems initially stored in its problem database. Figure 4 indicates the basic framework of problem generation. The generation system creates and uses episodes, which is cases of CBR in the system. If a new problem is created by altering another problem, an episode can be formed as data comprising the two problems. In the episode, the new problem is referred to as a new instance and the original as a base. When another instance of a problem is given, the generation system produces a new problem by mapping the relationship in an episode. The problem first given is referred to as input and the one produced by the system is the output. In Figure 4, new problems B-2 and B-3 are generated from A-1, A-2, A-3, and B-1, problems initially stored in the system.

Episodes are created by a production engine, a production-system component in the generation system. The production engine uses one set of rules to investigate the differences between two problems, and another set to provide each episode with actions based on the differences. The actions, referred to as altering actions, can compose a new instance by altering a base in an episode. If a new episode is created from the I/I and I/D problems in Figure 2, for example, the production engine investigates indices of their data, each a set of an attribute and its value as indicated in each line in Table 1, and lists the differences such as the replacement of objects (entities appearing in the problem texts), the addition of an object, and the addition of operations (mathematics operations used to evaluate values that are included in the equations but not in the texts). In the system, the data for the two problems are represented with indices as shown in Table 1. On the basis of the differences listed, the production engine then provides altering actions that can be used to generate new problems. Actions that merely copy are provided for indices with the same values. Table 2 shows the altering actions provided to an episode. New problems are generated by composing problem data using these actions.
Table 1. Representation of I/I (left) and I/D (right) problems in Figure 2

| instance layer | object.a | pen | object.b | pencil | object.W | (total) | property.1 | (how many) | property.2 | = | property.3 | = | property.4 | = | property.5 | = yen | parameter.a_1 | = x | parameter.b_1 | = y | parameter.W_1 | = 8 | parameter.a_4 | = 50 | parameter.b_4 | = 120 | parameter.W_5 | = 520 |
|---------------|---------|-----|---------|-------|----------|--------|-----------|----------|-----------|----|-----------|---|-----------|---|-----------|-----|-------------|----|-------------|----|-------------|---|-------------|----|-------------|----|
| class layer   | verb    | = buy |
| domain-property layer | target | = stationery | situation | = purchase |
| equation.1    | = x + y = W_1 |
| equation.2    | = a_4 * x + b_4 * y = W_5 |

| instance layer | object.a | orange | object.b | apple | object.W | (total) | object.c | = basket | property.1 | = (how many) | property.2 | = | property.3 | = | property.4 | = | property.5 | = yen | property.6 | = yen | property.7 | = times | property.8 | = yen | property.9 | = yen | parameter.a_1 | = x | parameter.b_1 | = y | parameter.W_1 | = 12 | parameter.a_4 | = a_4 | parameter.b_4 | = 120 | parameter.W_5 | = W_5 |
|---------------|---------|-------|---------|-------|----------|--------|-----------|----------|-----------|----|-----------|---|-----------|---|-----------|-----|-------------|----|-------------|----|-------------|---|-------------|----|-------------|----|
| class layer   | verb    | = buy |
| domain-property layer | target | = fruit | situation | = purchase |
| equation.1    | = x + y = W_1 |
| equation.2    | = a_4 * x + b_4 * y = W_5 |
| operation.a_4 | = a_6 * a_7 |
| operation.W_5 | = W_8 - c_9 |
Table 2. Example of altering actions in an episode

<table>
<thead>
<tr>
<th>Action Description</th>
<th>Action Details</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>copy (input, object.a, $AS$)</code></td>
<td></td>
</tr>
<tr>
<td><code>retrieveNoun ($AS$, 1, $SA_0$)</code></td>
<td></td>
</tr>
<tr>
<td><code>set (output, object.a, $SA_0$)</code></td>
<td></td>
</tr>
<tr>
<td><code>copy (input, object.b, $BS$)</code></td>
<td></td>
</tr>
<tr>
<td><code>retrieveNoun ($BS$, 1, $SB_0$)</code></td>
<td></td>
</tr>
<tr>
<td><code>set (output, object.b, $SB_0$)</code></td>
<td></td>
</tr>
<tr>
<td><code>create (output, object.c)</code></td>
<td></td>
</tr>
<tr>
<td><code>retrieve (object.c, $SC_c$)</code></td>
<td></td>
</tr>
<tr>
<td>Or <code>{copy (new instance, object.a, $SA_0$)},</code></td>
<td></td>
</tr>
<tr>
<td><code>copy (new instance, object.c, $SC_c$)</code></td>
<td></td>
</tr>
<tr>
<td><code>interact ($SA_0$, $SC_c$, $SA_0$, $SC_c$)</code></td>
<td></td>
</tr>
<tr>
<td><code>set (output, object.c, $SC_c$)</code></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td><code>set (output, equation.2, $SE_2$)</code></td>
<td></td>
</tr>
<tr>
<td><code>create (output, operation.a_1)</code></td>
<td></td>
</tr>
<tr>
<td><code>copy (new instance, operation.a_1, $SO_1$)</code></td>
<td></td>
</tr>
<tr>
<td><code>set (output, operation.a_1, $SO_1$)</code></td>
<td></td>
</tr>
<tr>
<td><code>create (output, operation.W_s)</code></td>
<td></td>
</tr>
<tr>
<td><code>copy (new instance, operation.W_s, $SO_2$)</code></td>
<td></td>
</tr>
<tr>
<td><code>set (output, operation.W_s, $SO_2$)</code></td>
<td></td>
</tr>
</tbody>
</table>

- Characters enclosed with $\$ indicate variables.
- The set of actions after “Or” is fired when the action before the Or fails.

To create the generation process information of a problem, the P+R system incorporates an enhanced generation system in this study. In addition to new problems, the system produces information on the processes used to generate the problems and records it in logs. When an output is generated from a problem as input to the episode illustrated above, the production engine first creates empty problem data and then composes the output by executing the altering actions in Table 2. The actions execute procedures such as getting the value of object.a from the input and binding variable A with it [copy (input, object.a, $AS$)], selecting a word $A_0$ from a conceptual class that belongs to the same superordinate concept as the parent concept of A [retrieveNoun ($AS$, 1, $A_0$)] in a dictionary database, and copying $A_0$ into the output as a value of object.a [set (output, object.a, $A_0$)]. The system records these actions executed as the alteration of an object from $A_0$ into A in a log. The procedure “interact ($A_0$, $SC_c$, $A_0$, $SC_c$)” is a special action that embeds an interaction command in the output, which requires the teacher to provide word knowledge by asking “Tell me a new word for $A_0$ corresponding to $SC_c$ for $A_0$.” Because common knowledge and common sense reasoning are necessary for composing word problems, the production engine can fail to fill in the values of the indices in the output. For example, the action “retrieve (object.c, $SC_c$)” examines a problem database to determine whether problem data that can offer the value of object.c to the output is available; however, this action often fails. After problem generation, the system always asks the teacher to check the output and revise it in terms of common knowledge if needed. It interactively learns word knowledge through these revisions and stores only output checked by the teacher in its problem.

---

2 The value of an object added to the output is obtained from problem data whose object.a and object.b are similar to (belong to the same conceptual class as) the output. This assumes that objects, such as oranges, apples, and basket, have a certain relationship.
database. After the interactions with the teacher\(^3\), the actions in the interactions are also recorded as the processes that generated the output. On the other hand, failed actions such as retrieve (object.c, $C0$) in this case are not recorded.

For example, if a problem is generated by adapting the episode above to the base in Figure 2, the following output is generated.

Output: Last year, I bought some 40-yen pencils and 110-yen pens. The total number was 13. This year, I bought 2 times as many pencils as last year, as many pens as last year, and a 300-yen pen case, all for 1430 yen. How many pencils and pens did I buy last year?

Solution:
Let \( x \) denote the number of pencils and \( y \) denote the number of pens.

\[
\begin{align*}
x + y &= 13 \\
40 \times 2x + 110y &= 1430 - 300
\end{align*}
\]

According to the equations above, \( x = 7 \) and \( y = 5 \).

Table 3. Example of a problem generation log

| copy (input, object.a, orange) |
| retrieveNoun (orange, 1, pencil) |
| set (output, object.a, pencil) |
| copy (input, object.b, apples) |
| retrieveNoun (apples, 1, pen) |
| set (output, object.b, pen) |
| create (output, object.c) |
| retrieve (object.c, $C0$) | failed |
| Or {copy (new instance, object.a, orange) |
| copy (new instance, object.c, basket) |
| interact (orange, basket, pencil, pen case) } |
| set (output, object.c, pen case) |
| set (output, equation.2, \(a_4x + b_3y = W_3\)) |
| create (output, operation.\(a_4\)) |
| copy (new instance, operation.\(a_4\), \(a_6\times a_7\)) |
| set (output, operation.\(a_4\), \(a_6\times a_7\)) |
| create (output, operation.\(W_3\)) |
| copy (new instance, operation.\(W_3\), \(W_8 - c_6\)) |
| set (output, operation.\(W_3\), \(W_8 - c_6\)) |

The system also produces a log as that shown in Table 3. The generation process information for the output indicates what was altered and how it was altered during output generation, and it is completed by transforming the log into instructional sentences. The sentences in the generation process information are automatically transcribed from the log using template rules. Each template rule describes a type of an instructional sentence based on procedures recorded in the log. In the system, the indices of problem data are categorized into three layers, as shown in Table 1: an instance layer denoting surface parameters in a problem text, a class layer denoting features of situations, and a

\(^3\) The teacher can freely change objects or parameters in the output without a request from the system when checking. The log is modified according to the changes made by the teacher.
domain-property layer denoting features of solutions. These layers were designed in our previous study (Kojima & Miwa, 2003) on the basis of models that represent processes for solving word problems in three steps (English, 1997a; Kintsh & Greeno, 1985). Sentences regarding surface parameters in the problem are described by rules regarding the indices of the instance layers, composed of an x-y-objects rule (describing a sentence indicating whether objects x and y are the same or different according to procedures regarding object.a and object.b), an object-addition rule, an object-removal rule, a parameter rule, a parameter-addition rule, and a parameter-removal rule. In the same way, a situation rule describes a sentence about the situation of the problem. Sentences about the solution are described by an equation rule, an operation-addition rule, and an operation-removal rule. In the problem generation described above, the sentences shown in Table 4 are described as the generation process information regarding the output. It includes (1) a situation and the answers (x and y), (2) surface parameters, (3) a solution structure of the output, and (4) an instruction to compose a text. According to the information, a learner composes a problem about purchasing by using the parameters and constructs the solution structure. No information about the text of the problem is provided. The last part of the generation process information simply instructs the learner to compose a text along with the situation and solution.

<table>
<thead>
<tr>
<th>Table 4. Example of generation process information</th>
</tr>
</thead>
<tbody>
<tr>
<td>The situation “purchase of goods” is the same as the base</td>
</tr>
<tr>
<td>Objects are altered to “pencils” and “pens”</td>
</tr>
<tr>
<td>x object: pencils</td>
</tr>
<tr>
<td>y object: pens</td>
</tr>
<tr>
<td>values: x = 10 and y = 3 (how many)</td>
</tr>
<tr>
<td>Two more parameters are added</td>
</tr>
<tr>
<td>Parameters in the example: (total) 13, pencil 40 yen, pencil 2 times, pen 110 yen, (total) 1430 yen, pencil box 300 yen</td>
</tr>
<tr>
<td>A pencil box and 300 yen are added as the 3rd object</td>
</tr>
<tr>
<td>Solved in the solution different from the base</td>
</tr>
<tr>
<td>[x pencils] + [y pens] = [(total) 13]</td>
</tr>
<tr>
<td>[pencil #1][x pencils] + [pen 110yen][y pens] = [(total) #2]</td>
</tr>
<tr>
<td>#1 Operation [pencil 40 yen][pencils 2 times] is added</td>
</tr>
<tr>
<td>#2 Operation [(total) 1430 yen] - [pencil box 300 yen] is added</td>
</tr>
<tr>
<td>Compose a problem text of “purchase of goods” using the objects and parameters, which is solved with the solution</td>
</tr>
</tbody>
</table>

**System Architecture**

Figure 5 illustrates the architecture of the system implemented in this study. Our system consists of three interfaces: a problem input interface to receive problems from learners, a process display interface to show generation process information for the examples, and a feedback interface to show evaluations of problems posed by learners and examples presented to learners. Situation estimation models and an equation parser offered by the P-Only system are used to understand situations and solutions of problems input by learners. The enhanced generation system offers a dictionary database used to understand keywords in problems, a problem database to provide problems as examples, and a
log database. A log transformation tool creates generation process information using the template rules previously described. To implement the function that supports learning by reproduction, the log database, log transformation tool, and process display interface were added to the P+R system.

![Architecture of our system](image)

**Figure 5. Architecture of our system**

**Understanding of Learner Problems and Feedback**

Our system can automatically understand problems posed and inputted by learners and present examples. These functions are also offered by the previous P-Only system.

**Understanding of learner problems**

In our system, a learner poses a new problem in the domain of the base and inputs it into the problem input interface. In this phase, the problem input interface requires objects for variables \( x \) and \( y \) (such as oranges or pencils) and answers (values of \( x \) and \( y \)), numeric values appearing in the text of the problem, equations for solving the problem, and the text itself. It then represents the problem in data format, as shown in Table 1. Equations and operations in the data are obtained by the equation parser, and a situation’s label is inferred from the situation-estimating models.

The values of equations in the problem data are created by transforming the equations into the standard form (e.g., \( ax + by = c \)). If operations that evaluate coefficients or constant terms in the standard form are needed in the transformation, the equation parser records the processes of the operations and then creates indices of these operations. The problem data are constructed on the basis of the structure represented in the equations and operations.

To estimate the label denoting a problem’s situation, the problem-input interface uses situation-estimating models, each of which is automatically constructed from independent words in the texts of problems in the problem database comprising identical situations. The situation of a learner problem is inferred by evaluating the similarity scores between the problem text and situations in the models. The similarity scores are evaluated by a method for computing the weight values, which is developed by
transforming term frequency/inverse document frequency (tf/idf). Therefore, the situation-estimating models can never identify novel situations that are not included in problems in the problem database. Essentially, they can estimate only situations that are seen as typical and well-known problems; they suppose that other situations are novel in some way.

**Feedback**

After the system understands a problem posed by a learner, it then provides feedback to the learner. The feedback interface displays an evaluation of the posed problem, indicating similarities between the base and learner problem using the categories shown in Figure 1. It can identify the category of the learner problem, because the features of the problem’s situation and solution are represented in the data using the indices of the three layers. Simultaneously, it retrieves and presents some examples. The categories of the examples are also indicated.

The system can interactively present various examples according to evaluations of problems posed by learners. For example, it can present examples whose categories differ from that of a learner problem, or those whose solutions are identical to and situations are different from the learner problem. The presentation can be arbitrarily controlled by setting up conditions in the if-then format (e.g., IF an I/I problem is posed, THEN present a D/I problem). We assume that a teacher sets up the case presentation.

**Procedures of Learning with our System**

As mentioned, our system is an enhanced version of the P-Only system. Thus, we briefly state the learning procedures in the P-Only system before explaining those in the P+R system.

**Learning procedures in the P-Only system**

Before giving the P-Only system to learners, problem and log databases have to be prepared. A teacher first interacts with the enhanced generation system to provide the databases.

In learning with the system, a learner is first given an initial problem as a base and is prompted to generate a new problem from it. Suppose that a learner poses the following problem.

I bought some 98-yen persimmons and 128-yen peaches for 806 yen. The total number of persimmons and peaches was 7. How many persimmons and peaches did I buy?

**Solution.**

Let $x$ denote the number of persimmons and $y$ denote the number of peaches.

\[
\begin{align*}
x + y &= 7 \\
98x + 128y &= 806
\end{align*}
\]

According to the equations above, $x = 3$ and $y = 4$.

The learner inputs the problem into the problem input interface, as shown in Figure 6. The left-hand window in the figure indicates the base. The problem input interface requires the learner to input the problem in four steps. In the first step, it first requires information about $x$ and $y$. The learner then inputs “persimmon,” “3,” “peach,” and “4.” The problem input interface then creates the table at the top of the window. In the second step, the learner adds the parameters appearing in the problem texts.
(“7,” “98 yen,” “128 yen,” and “806 yen”) into the table. These are used as the values of indices in the instance layer. In the third step, the learner is required to input the two equations by using the parameters inputted in the previous step. As mentioned above, the features in the domain-property layer are obtained from the equations. The learner edits the problem text in the fourth step. As shown at the bottom right in Figure 6, keywords and numeric parameters necessarily included in the text are listed in the edit window. The situation of the problem is estimated from the words in the text.

Figure 6. Screenshot of the problem input interface

The evaluation of the learner problem is fed back in the feedback interface as shown in the middle window in Figure 7. Because this is a problem of purchase whose solution is the same as the base, the system indicates that it is an I/I problem in the dialog on the right. Simultaneously, it retrieves and presents some examples as shown at the bottom of Figure 7. The categories of the examples are also indicated. In this case, the system was set up in advance to present D/I and I/D
problems as examples if a learner poses an I/I problem, I/D and D/D examples if a learner poses a D/I, D/I and D/D examples if a learner poses an I/D, and I/D and D/I examples if a learner poses a D/D. Thus, the I/D problem in the dialog on the right can be presented as an example. The base is shown at the top of Figure 7. The learner poses further problems while comparing the base with the learner’s problem or the presented examples. In the P+R system, those procedures function when a learning-by-posing mode is set in the system configurations.

![Figure 7. Screenshot of the feedback interface](image-url)
Learning procedures in the P+R system

The P+R system functions when a learning-by-reproduction mode is set. A learner is prompted to reproduce each presented example in this mode. Figure 8 shows a screenshot of the mode when the learner reproduces the example selected in Figure 7, which is the same as the output described above. The left side of Figure 8 is the process display interface, which provides generation process information for the example, and the right side is the problem input interface. The example itself is hidden while the learner is working with the problem input interface in the learning-by-reproduction mode. The learner is instructed to input the example by following generation process information. All the information needed to formulate the example is provided in each step, and differences between the base and the example are highlighted.

Figure 8. Screenshot of the problem input interface in the learning-by-reproduction mode
Because the input procedures are the same as the learning-by-posing mode, the process display interface presents each piece of generation process information relevant to each of the four steps in the problem input interface. In the first step, the learner inputs \(x\) and \(y\) (“pencil,” “10,” “pen,” and “3”) into the problem input interface according to the information presented at the top of the process display interface. The information also notes that the objects of the example differ from those of the base. The situation of the example is indicated in the first step. In the second step, the learner inputs the parameters of the example (“13,” “40 yen,” “2 times,” “110 yen,” “1430 yen,” and “300 yen”). In this step, generation process information notes that parameters and an object are added, as shown in the middle of the process display interface. In the third step, the learner composes the equations in the example (“\(x + y = 13\)” and “\(40 \times 2x + 110y = 1430 - 300\)”). To clearly show the differences from the base, the standard forms and operations of the equations are indicated separately, as shown at the bottom. However, in the fourth step, generation process information does not provide any information about the text in the example. It only instructs the learner to compose a text according to the situation and solution. It also states that the text does not need to be literally identical to the example as long as it can be solved by the solution identical to the example. To prevent superficial duplication of the example by blindly following the information, the example’s text is not shown. Through this activity, the learner is expected to understand the ideas used to generate the presented example from the viewpoint of its poser.

**EXPERIMENTAL EVALUATION**

We experimentally evaluated the effect of learning with the P+R system. In the evaluation, we confirmed whether learners can understand the ideas used to alter solutions through learning by reproducing an example and can transfer the ideas into their problem posing. As mentioned above, merely viewing the examples did not increase the number of I/D problems composed by altering the solution structures in the P-Only system. Therefore, the effect of learning by reproduction was verified through a comparison with learning by viewing an example, the learning method in the P-Only system.

**Experimental Method**

Sixty undergraduate students simultaneously participated in the experimental evaluation in a lecture class on cognitive science and artificial intelligence. Each participant was seated at a desk where a PC was set up. They engaged in problem-posing tests after learning with our system. Each participant was asked to pose a new problem from an initial problem presented as a base in each task. The experimental task procedures were as follows.

1. Training in a system operation
   The participants practiced system operation by inputting a base problem given in the following learning phase.

2. Learning phase (15 min)
   Each participant posed a problem with our system in the domain of word problems solved by simultaneous equations. The base in Figure 2 was initially given in problem posing.
I bought some 60-yen oranges and 120-yen apples for 1020 yen. The total number of oranges and apples was 12. How many oranges and apples did I buy?
Solution.
Let $x$ denote the number of oranges and $y$ denote the number of apples.
\begin{align*}
x + y &= 12 \\
60x + 120y &= 1020
\end{align*}
According to the equations above, $x = 7$ and $y = 5$.

In this phase, participants were assigned to the condition groups described below.

3. Post-tests (30 min)
The participants were given two problem-posing tests. One of the tests used the following word problem of simultaneous equations as a base, which had a situation and solution identical to and surface parameters different from the one in the learning phase.

I bought some packages of colored paper containing 50 sheets and packages containing 80 sheets. The total number of packages was 6, and the total number of sheets of colored paper was 360. How many packages of 50 sheets and 80 sheets did I buy?
Solution.
Let $x$ denote the number of packages containing 50 sheets and $y$ denote the number of packages containing 80 sheets.
\begin{align*}
x + y &= 6 \\
50x + 80y &= 360
\end{align*}
According to the equations above, $x = 4$ and $y = 2$.

The other test used the following problem solved by a unitary equation as a base.

I want to buy some boxes of cookies. If I buy 110-yen boxes of cookies, then I have 50 yen left. If I buy 120-yen boxes of chocolate cookies, then I need 20 yen more. How many boxes do I want?
Solution.
Let $x$ denote the number of boxes.
\begin{align*}
110x + 50 &= 120x - 20
\end{align*}
According to the above equation, $x = 7$.

We refer to the former test as a near-transfer (NT) test, and the latter as a far-transfer (FT) test.
Before the start of each test, the participants were given the following instruction.
“Pose a new problem from the given base problem. In this problem posing, try to adapt what you learned from problem posing with the system.”
Before the start of each test, the participants were instructed to apply what they had learned in the learning phase to their problem posing.

To verify whether learning by reproduction can actually facilitate the alteration of solutions, we conducted the FT test, in which participants needed to adapt what they learned from the example, in addition to the NT test, in which new problems could be posed immediately by duplicating the example.
We did not conduct a pre-test because of a time constraint on the lecture class. Because our empirical data in the previous studies have consistently shown that without any intervention, undergraduates posed many I/I problems and few I/D problems, and most of their solution structures were simple, we verified the effect of how they learned the example by comparing the condition groups described below in the post-tests.

**Condition Groups**

In the learning phase, each participant was randomly assigned to one of three condition groups as follows.

*Imitation group (n = 21):* An example was presented in addition to the base in the training. Participants were given the following instruction.

“Now, you will experience problem posing. The problem presented below the base is an example of problems solved with simultaneous equations. Pose the same problem as the example. The system will instruct how to formulate the example. Information needed in formulating the example is provided in the next screen. Although you cannot view the example while inputting your problem, you need not exactly memorize the text of the example. The text of your problem does not have to be literally the same as long as it can be solved with the same solution as the example.”

They then reproduced the example in the system, as shown in Figure 8. They could see the base and generation process information for the example during the learning phase.

*Observation group (n = 20):* As for the imitation group, the base and example were presented. Participants were given the following instruction.

“Now, you will experience problem posing. Pose a new problem in the domain of the base and input it in the system. The problem should be solvable with simultaneous equations and middle school students should be able to solve it. The problem presented below the base is an example of problems solved with simultaneous equations. It’s one of good responses when posing a new problem in this domain.”

They then inputted their problems into the system, as shown in Figure 6. They could see the base and example during the learning phase.

*Control group (n = 19):* The base was presented. Participants were given the following instruction.

“Now, you will experience problem posing. Pose a new problem in the domain of the base and input it in the system. The problem should be solvable with simultaneous equations and middle school students should be able to solve it.”

Because they were given no example, they could see only the base.

In each group, participants posed one problem by using the system. Therefore, the imitation group did not pose their own problems. Because the imitation group used the system in the learning-by-reproduction mode, they learned to pose problems through learning by the reproduction method proposed in this study. On the other hand, the observation group learned with the system in the learning-by-posing mode. The imitation and observation groups were presented the following problem as an example, which is identical to the output described above.
Last year, I bought some 40-yen pencils and 110-yen pens. The total number was 13. This year, I bought 2 times as many pencils as last year, as many pens as last year, and a 300-yen pen case for 1430 yen. How many pencils and pens did I buy last year?

Solution.
Let \( x \) denote the number of pencils and \( y \) denote the number of pens.
\[
\begin{align*}
x + y &= 13 \\
40 \cdot 2x + 110y &= 1430 - 300
\end{align*}
\]
According to the equations above, \( x = 10 \) and \( y = 3 \).

This problem has the situation “purchase of goods,” which is identical to that of the base, and a solution formed by adding a third object other than the \( x \) and \( y \) objects and an operation for calculating a coefficient of \( x \) in the lower equation in the base. Because problems in the I/D category are generated by altering the solution in the base and are rarely posed by novices, we used that problem as the example in the learning phase. Although our system interactively provides examples according to the evaluations of learner problems, the example was controlled according to the focus of this evaluation.

We confirmed whether the imitation group adapted what they could learn from the example into their problem posing. We hypothesized that only the imitation group could adapt the example. The experimental evaluation in a previous study (Kojima & Miwa, 2008) predicted that the observation group could not adapt it.

Analysis

Two characteristics of the problems posed by the participants were analyzed: their variety and the strategies used to alter the solutions of the bases given in the problem-posing tests. The variety was measured using the four categories shown in Figure 1. The solution-altering strategies were labeled by the first author according to the differences between formal structures of solutions in the posed problems and those in the bases. Figure 9 indicates the classifications of the strategies used to alter solutions. First, all the posed problems were divided into two groups depending on whether their solutions were the same as or different from those of the bases. If a problem of a participant had a solution of \( x + y = W_1, a_4x + b_4y = W_3 \) in the NT test or \( a_3x + a_3 = b_2x - b_3 \) in the FT test, it was labeled not altered. The other problems were then divided into two groups depending on whether their solutions were partially altered or altered overall. Each of the partially-altered problems had a solution structure that can be composed by adding or removing operations in the structures of the bases (e.g., \( (a_3 + a_4)x + a_2 = b_1x - (b_3 - b_4) \) in the FT test). Finally, the partially altered problems were divided into two groups depending on whether their solutions were altered using the same operations as the example (with operations same as the base) or operations different from those in the example (with operations different from the base). The effects of learning from the example were verified by comparing the control group with the imitation or observation group. The adaptation of the example during problem posing by the participants was observed as an increase of the number of posed problems with the same operations, which corresponded to the transfer of ideas in the example.

---

4 Each of these problems includes at least an operation that adds an absolute term or one that adds a multiplication to evaluate a coefficient.
As described above, we predicted that understanding the example would increase the number of I/D problems in terms of the variety and increase the number of those posed using the same operations in terms of solution-altering strategies. Therefore, we examined the effects of the example by comparing the differences in the frequencies of each category and solution-altering strategy between the control group and the imitation or observation group. To confirm the differences in the frequencies, we used the chi-square test for a comparison. We also examined the effect of how participants were involved with the example by comparing the imitation and observation groups.

### Results

Two participants who did not reproduce the example in the imitation group were excluded from the data analysis. Those who did not complete the task in the learning phase (one in the imitation group, four in the observation group, and five in the control group) were also excluded, because time ran out. (Some examples of participants’ problems are shown in the Appendix.)

#### Categories of posed problems

Figure 10 indicates the proportions of posed problems in each category in the NT test. All three groups posed many I/I problems. The imitation group alone posed many I/D problems. We compared the control group with the imitation or observation group using the chi-square test; the results indicated a significant difference between the imitation and control groups ($\chi^2 (2) = 8.30, p < .05$). Furthermore, the results of residual analysis indicated that the number of I/D problems in the imitation group was significantly high and that in the control group was significantly low. No significant difference was observed between the observation and control groups ($\chi^2 (3) = 5.23, n.s.$). We also compared the imitation and observation groups, with the result indicating no significant difference ($\chi^2 (3) = 4.24, n.s.$).

Figure 11 indicates the proportions of posed problems in each category in the FT test. As in the NT test, many I/I problems were posed, and many I/D problems were posed by the imitation group. The chi-square test indicated a moderately significant difference between the imitation and control groups ($\chi^2 (3) = 6.40, p < .10$) and no significant difference between the observation and control groups ($\chi^2 (3)$
These results confirmed that the imitation group posed many I/D problems after learning the example. On the other hand, the example was not observed to have a significant effect on the observation group. We also compared the imitation and observation groups, with the result indicating no significant difference ($\chi^2 (3) = 3.30, \text{n.s.}$).

Solution-altering strategies in posed problems

Figure 12 indicates the proportions of posed problems composed with each solution-altering strategy in the NT test. In the control group, because participants did not pose any problems whose solutions differed from the base, all their problems were labeled “not altered.” It was also noteworthy that many participants in the imitation group used the same operations as the example in their problem posing. The chi-square test indicated a significant difference between the imitation and control groups ($\chi^2 (2) = 8.30, p < .05$). Residual analysis revealed that the number of the same operations in the imitation group was significantly high and that in the control group was significantly low; also, the
number of “not altered” problems in the control group was significantly high and that in the imitation group was significantly low. No significant difference was observed between the observation and control groups ($\chi^2 (3) = 3.76, n.s.$). We also compared the imitation and observation groups, with the result indicating no significant difference ($\chi^2 (3) = 2.54, n.s.$).

Figure 13 indicates the proportions of posed problems composed with each solution-altering strategy in the FT test. Those results were similar to those of the NT test. The chi-square test indicated a moderately significant difference between the imitation and control groups ($\chi^2 (3) = 6.36, p < .10$) and no significant difference between the observation and control groups ($\chi^2 (3) = 1.34, n.s.$). These results confirmed that participants in the imitation group could adapt solution-altering strategies used in the example to their problem posing. We also compared the imitation and observation groups, with the result indicating no significant difference ($\chi^2 (3) = 4.57, n.s.$).

![Figure 12](image1.png)

Figure 12. Proportions of posed problems with each solution-altering strategy (NT test)

![Figure 13](image2.png)

Figure 13. Proportions of posed problems with each solution-altering strategy (FT test)

**Discussion**
The results described above demonstrated the effect of learning the example on the imitation group. Learning by reproducing the example increased the number of I/D problems and facilitated the adaptation of ideas feasible for altering the solutions, which were used to generate the example, to the posing of new problems. On the other hand, viewing the example was not observed to have a significant effect on the observation group. Although the results did not confirm any difference in the effect of how participants were involved with the example, the effect of the example was actually effective only in the imitation group. Therefore, learning through reproduction was proven to be more effective in understanding of ideas embedded in the solution structures than learning through merely viewing.

We made a preliminary investigation of the effect of learning by solving on problem posing (Kojima, Miwa & Matsui, 2010b). In that investigation, undergraduates engaged in the same tasks as in the learning phase and FT test in this study, except that they learned the same example by solving it. The result proved that learning by solving the example moderately increased the number of I/D problems, even though it did not increase the number of problems whose solutions were more complex than the base. As in a previous study where no example was learned (Kojima et al., 2010a), most of their problems in the post-test were simple. This may indicate that learning by reproduction is more effective than learning by comprehension. This point has to be empirically confirmed in future work.

The P+R system we developed in this study is similar to the computational learning environments of Hirashima et al. (Hirashima et al., 2004; Waki et al., 2009), which support problem posing by changing the instance problems. In these environments, learners can compose new problems by altering the settings of given problems through operations in graphical user interfaces. Such problem posing corresponds to the generation of I/D problems, as shown in Figure 1. Although the primary purpose of the computational environments is not to stimulate idea generation for novel problem posing but to deepen the understanding of the relationships among problems, both these environments and our system enable learners to discover how to alter given problems into new ones. They reported that learning in their environments improved learners’ performance on a test of problem association by extracting similar and different elements in the problems. However, they observed that the environments did not increase the associations between problems based on the similarities in the solutions. Thus, because these environments scaffold problem posing using rich graphical interfaces, their learners who had experienced problem posing could not appropriately pose I/D problems without the environments. That also suggests the difficulty in learning solution structures, which was discussed previously.

One other study similar to this one is, of course, our previous study (Kojima & Miwa, 2008). In research on creative support, many systems that provide examples as hints regarding novel ideas have been proposed and implemented in a variety of tasks (e.g., Orihara, 1994; Restrepo & Christiaans, 2005; Young, 1987). The P-Only system in the previous study was designed on the basis of idea-generation support in problem posing. In ill-defined problem solving such as idea generation tasks, the reuse of past experiences or existing exemplars can increase the quality of the products (Leak, 1996). However, as indicated by both the experimental evaluations in the previous study and the results of the observation group in this study, merely presenting examples did not increase the number of I/D problems.

Why is it that viewing examples does not increase the posing of I/D problems? Furthermore, why does it not provide hints for problem posing? The reason may be hint abuse, a behavior in which novice learners deliberately do not read mediated hints but request correct answers instead (e.g.,
Aleven & Koedinger, 2000; Ringenberg & VanLehn, 2006). Even though the examples include helpful hints in problem posing, they never provided immediate answers for problem posing by the learners. To receive the hints from the examples, the learners must extract important features from them. However, it is difficult for them to do that, particularly for the structural features related to the solutions, merely by seeing the examples. Analogy studies have argued that people retrieve knowledge adapted to problem solving based on surface information such as situations, not structural information such as solutions (Gentner, 1983; Forbus et al., 1995). Human problem solving occasionally suffers because of similarities in the surface information of problems (e.g., Novik, 1988; Reed, Dempster & Ettinger, 1985; Ross, 1987). Because the examples of I/D problems have situations identical to a base given initially in problem-posing tasks, the surface similarity may have prevented learners from carefully examining the examples. Therefore, this study proved that reproduction can be an effective activity for engaging learners in such a careful examination of examples.

Fewer participants in the imitation group did not complete the task in the learning phase than in the observation and control groups. Participants did not complete the task, because they ran out of time. The learning task in the imitation group was a reproduction of the example by following generation process information. In other words, the task was scaffolded in the imitation. Thus, that must have aided the imitation group in completing the task in the given time.

This study has some limitations. First, although the participants in the experimental evaluation were undergraduates, the problem domains were middle school mathematics. As mentioned, undergraduates had difficulty in posing problems in the domains. Therefore, we must introduce further support to facilitate problem posing by middle school students, who are supposed to learn the domains by problem solving. Another limitation is that the participants only reproduced one example in the learning phase. To examine support for facilitating problem posing more effectively, further study must be conducted to confirm the effect of learning multiple and diverse examples.

**CONCLUSIONS**

We discussed an approach to support learning from examples in problem posing through the productive task of imitation, along with support that facilitate learners’ understanding of solution structures. We then implemented a support system for learning and experimentally evaluated its effect; the results indicated that learning through reproduction can facilitate the adaptation of ideas embedded in the solution structure of an example into learners’ problem posing.

It is important to further study the effects of learning from examples in problem posing. We confirmed that the reproduction of an example enables learners to reuse ideas that were used to compose the solution to an example. In the next step, we have to explore the extent to which experience in learning from the example can enhance problem posing by learners. As our previous investigation revealed (Kojima et al, 2010a), novice learners tended not to vary the solutions when they posed problems, although they generated various situations. This suggests that problem posing by learners is most likely constrained by a specific solution they have learned. Knowing unfamiliar solutions from examples may relax this constraint and broaden that range of solutions the learners try to formulate.

It is also important to empirically study the effects of problem posing on problem solving. As mentioned earlier, it is beneficial to pose appropriate and effective problems in many aspects. In the next step, we have to confirm the benefits of problem posing. One critical aspect is, of course, the
transfer of problem-posing skills to problem solving. Again, we need to pursue more support strategies for problem posing as a crucial cognitive activity in mathematics learning.

REFERENCES


**APPENDIX**

**Examples of problems posed in the Near Transfer test in the experimental evaluation**

I bought some packages containing 20 strawberries and packages containing 5 loquats. The total number of packages was 5, and the total number of pieces of fruit was 70. How many packages of strawberries and loquats did I buy?

Solution.

Let $x$ denote the number of packages of strawberries and $y$ denote the number of packages of loquats.

\[
\begin{align*}
x + y &= 5 \\
20x + 10y &= 70
\end{align*}
\]

According to the equations above, $x = 2$ and $y = 3$.

(I/I problem posed in the observation group)
It took 22 minutes to travel 1590 m to a station by walking for some minutes at a speed of 50 m/h and running for some minutes at a speed of 120 m/h. Find the time spent walking and running.

Solution.
Let \( x \) denote the minutes of walking and \( y \) denote the minutes of running.

\[
\begin{align*}
  x + y &= 22 \\
  50x + 120y &= 1590
\end{align*}
\]

According to the equations above, \( x = 15 \) and \( y = 7 \).

Last year, I bought some packages of colored paper containing 50 sheets and packages containing 80 sheets. The total number of packages was 6. This year, I bought twice as many packages containing 50 sheets as last year, the same number of packages containing 80 sheets as last year, and another package containing 40 sheets. The total number of sheets of colored paper was 560. How many packages of 50 sheets and 80 sheets did I buy last year?

Solution.
Let \( x \) denote the number of packages containing 50 sheets and \( y \) denote the number of packages containing 80 sheets.

\[
\begin{align*}
  x + y &= 6 \\
  50(2x) + 80y &= 560 - 40
\end{align*}
\]

According to the equations above, \( x = 2 \) and \( y = 4 \).

Last week, Maruo bought 6 packages of colored paper, some containing 50 sheets and some containing 80 sheets, but he has already used all the sheets. This week, he bought the same number of new packages containing 50 sheets and 80 sheets. He then gave 2 packages containing 50 sheets to a friend. The total number of sheets of colored paper left for Maruo was 260. How many packages of 50 sheets and 80 sheets did he buy last week?

Solution.
Let \( x \) denote the number of packages containing 50 sheets and \( y \) denote the number of packages containing 80 sheets.

\[
\begin{align*}
  x + y &= 6 \\
  50(x - 2) + 80y &= 260
\end{align*}
\]

According to the equations above, \( x = 4 \) and \( y = 2 \).

A game gives the winner some money as a reward, and also gives the loser some money. “A” won 5 times and lost 3 times in the matches and earned 560 yen. “B” won 4 times and lost 4 times in the matches and earned 480 yen. Find the amount of money given to the winner and the loser in a single match.

Solution.
Let \( x \) denote the money given to the winner and \( y \) denote the money given to the loser.

\[
\begin{align*}
  5x + 3y &= 560 \\
  4x + 4y &= 480
\end{align*}
\]

According to the equations above, \( x = 100 \) and \( y = 20 \).

Examples of problems posed in the Far Transfer test in the experimental evaluation

I want to buy some pens. If I buy 80-yen pens, then I have 30 yen left. If I buy 100-yen pens, then I need 30 yen more. How many pens do I want?

Solution.
Let $x$ denote the number of pens.
\[80x + 30 = 100x - 30\]
According to the equation above, $x = 3$
(I/I problem posed in the control group)

Seventh-grade students are going to a musical concert. The concert hall has some benches. If 3 students are seated on each bench, 10 students cannot be seated. If 4 students are seated on each bench, there is an excess of seating for 10 persons. How many benches does the concert hall have?
Solution.
Let $x$ denote the number of benches.
\[3x + 10 = 4x - 10\]
According to the equation above, $x = 20$.
(D/I problem posed in the observation group)

I want to buy 3 times as many sweets as I bought last week. If I buy 100-yen pies, then I have 70 yen left. If I buy 120-yen chocolate cakes, then I need 110 yen more. How many sweets did I buy last week?
Solution.
Let $x$ denote the number of sweets last week.
\[3 \times 100x + 70 = 3 \times 120x - 110\]
According to the equation above, $x = 3$.
(I/D problem with the same operations posed in the imitation group)

I want to buy some boxes of cookies. If I buy 110-yen boxes of cookies, then I have 40 yen left. If I buy 2 110-yen boxes of cookies and enough 120-yen boxes of chocolate cookies so that the number of boxes is the same, then I have no money left. How many boxes do I want?
Solution.
Let $x$ denote the number of boxes of cookies.
\[110x + 40 = 120(x - 2) + 110 \times 2\]
According to the equation above, $x = 3$.
(I/D problem with different operations posed in the imitation group)

A boy is 30 years younger than his father. His father will be 2 times as old as the boy in another 20 years. How old is the boy?
Solution.
Let $x$ denote the age of the boy.
\[2(x + 20) = x + 30 + 20\]
According to the equation above, $x = 10$.
(D/D problem with an overall alteration posed in the imitation group)