Development of Production System for Anywhere and Class Practice

Kazuhsia MIWA a,1, Ryuichi NAKAIKE b, Jyunya MORITA c, and Hitoshi TERAI d

a Graduate School of Information Science, Nagoya University
b Graduate School of Education, Kyoto University
c School of Knowledge Science, Japan Advanced Institute of Science and Technology
d School of Information Environment, Tokyo Denki University

Abstract. The authors have developed a web-based production system that users can use whenever and anywhere by the Internet. The authors held two cognitive science introductory classes with the system. In our class activities, participants were required to construct running cognitive models on the production system architecture that can solve pulley problems. The participants not only constructed cognitive models but also produced original problems, which were distributed to the other class members. The participants found the defects in their models while trying to solve the problems from other members and trying to improve their models. A posttest indicated that the participants who successfully constructed high performance models revealed deeper understanding of pulley systems.

Keywords. Cognitive Science Class, Production System, Cognitive Modeling, Group Learning.

Introduction

In studies of cognitive science, the roles of running computer models are crucial [1]. The most standard architecture is a production system. Popular architectures such as ACT-R and SOAR have been utilized in various types of cognitive science studies. The authors held two cognitive science classes where the participants constructed production system models that actually ran on a computer. In this paper, we report our web-based production system developed for the classes, the designs of the class activities, and the results of the class practice. Many cognitive science classes worldwide are dealing with the issues of cognitive modeling; however few classes exist where the participants actually construct running computer models and experience such adventure and effectiveness (some preceding reports can be seen in [2][3]).

It costs too much for instructors to construct such a class. One big practical obstacle is to construct an educational environment. Usually, the behavior of each type of cognitive architecture depends on a lower computer platform on which the architecture is installed. Instructors must experience many setbacks in the process of preparing cognitive architecture using available computer facilities. It is not exciting to install cognitive architecture into many computers for class practice.

1 Corresponding Author. Nagoya 464-8601, Japan: E-mail: miwa@is.nagoya-u.ac.jp
To overcome such difficulties, we developed a web-based production system architecture called DoCoPro, which is a nickname of a Japanese phrase that means a production system for anywhere. Learners can use DoCoPro whenever and from anywhere in an environment with access to the Internet. Instructors are completely released from the frustration of preparing educational environments. Our architecture also provides novice learners with multiple learning supports that reduce the difficulties that emerge in the process of programming.

Another big issue is class design. Beginners are usually required to construct a basic cognitive model that solves a simple example problem for exercise in the initial stage of learning to understand the grammatical structures and the semantics of the programming language. They are required to engage in dead-alive trainings individually for a relatively long time. Many students may lose interest in cognitive modeling at this stage before they experience the joy of cognitive modeling. We address these difficulties by proposing an innovative class design. In our class activities, a learning context is created where interaction among class members emerges naturally in the learning process, and the participants are guided to learn cognitive modeling in such a social context.

1. **DoCoPro: Production System for Anywhere**

1.1. **Concepts**

Figure 1 shows the system concepts of DoCoPro. Learners access the DoCoPro server from terminals connected to the Internet, access the system through the URL of the server opened to the public by a Web browser, and login by a password. The installation and the settings of specific software are not needed. DoCoPro is utilized by various types of web browsers such as IE, Firefox, and Safari that learners use daily. The users’ individual data are saved and managed based on the database and the login mechanism; therefore if learners move to a different terminal, they can continue to program their cognitive models uninterruptedly.

DoCoPro is established Ruby on Rails (RoR) with MySQL for a database and Ajax for an interface. DoCoPro is basically a web application; however it functions as an individual application installed onto the user's personal computer.

![Figure 1. System concepts of DoCoPro](image)

1.2. **Interface**

Figure 2 shows an example screenshot of the interface of DoCoPro.
Knowledge Building: Learners can add and revise every clause of the production rules from the right side window of the interface indicated in Figure 2.

Process Management: The status of the working memory is displayed in the left upper window. The process of the simulations is managed by manipulating the allow buttons below the WM window.

Learning Support: One of the big difficulties that learners suffer when constructing production system models lies in understanding the matching mechanisms of the lists of WM and the if clauses of production rules and variable instantiations. Our system indicates both the matching status of WM and the rules and variable instantiations in the left lower window for learning support. The "CHECK ALL" button displays a list of rules that can fire at every point of the simulations. Clicking one of the rules that can fire from the list displays the variable instantiations of the if clauses of the rule. Clicking a rule that cannot fire informs the learners of the status at which the matching process was broken.

2. Curriculum Design

2.1. Task and Participants

The task in our class activities was pulley systems. The subjects and the cognitive models expected to be constructed are based on [4].

Two classes participated in our practice.

Class A: 18 liberal arts undergraduates without advanced programming skills. Pairs of participants were allowed to construct a single cognitive model collaboratively.

Class B: 14 information science graduate students. Some had experience with advanced computer programming; however none had learned list-processing-type computer programming such as LISP and Prolog.

2.2. Overview

Our practice was performed in a series of cognitive science introductory courses. The following is an overview of our class activities.
Introduction: A lecture on the basics of production system modeling was given to the participants. They received a simple pulley problem and were instructed to construct a production system model that solves the problem.

Constructing Initial Models: All participants were required to generate an original pulley problem and construct a cognitive model to solve the problem generated by themselves. The models constructed in this earlier stage are called initial models.

Constructing Problem Set: The generated problems were withdrawn, and a set of problems was created. The problem set consisted of 14 problems in Class A and 15 problems in Class B.

Initial Model Challenges: The problem set was distributed the participants, and their initial models solved the problems. The solved and unsolved problems were identified in each of the initial models.

Constructing Revised Models and Challenges: The participants were required to improve their initial models by having the models solve as many problems as possible. The improved models are called revised models. The revised models solved the problems of the problem set, and the solved and unsolved problems were identified.

The most important point of our class activities is that the participants act not only as problem solvers but also as problem generators. This context naturally produces interaction among the participants and increases their motivations to construct sophisticated models.

3. Results

We mainly report the results of the practice in Class B and supplementarily add the results in Class A.

3.1. Collected Problems

The fifteen problems gathered in Class B were grouped into three categories (Figure 3).

(a) Simple problem  
(b) Specific problem  
(c) Difficult problem

Figure 3. Example problems in problem set

Simple Problems: These problems were solved by a set of basic knowledge, such as two forces acting on two ropes hanging with a pulley are identical; a force pulling up a pulley is identical to the sum of the forces of two ropes dragging the pulley down.

Specific Problems: These problems contain specific objects in the system. In the example problem in Figure 3(b), a rod connecting the two pulleys is contained. These problems are solved by adding specific knowledge applicable to a specific situation.
**Difficult Problems:** These problems are difficult to be solved because each of two forces acting on the two ropes supporting a weight is not calculated independently from the other; therefore the ratio of the two forces is inferred for problem solving. In the specific problems, a single rule corresponding to a specific situation makes the solution possible; however in the difficult problems, multiple relatively complex rules are needed for gradual inferences to calculate the ratio.

In Class B, six simple, five specific, and four difficult problems were gathered. In Class A, twelve simple and only two difficult problems were gathered; one invalid problem was eliminated.

### 3.2. Model Performance

Figure 4 shows the means of the ratios of the problems successfully solved by the initial and revised models to all problems. The figure indicates that the models' performances largely improved in both Classes A and B.

![Figure 4. Model performance](image)

Compared to Class A, the improvement of performance was drastic in Class B. As mentioned earlier, various types of problems were generated in Class B. Therefore, the initial models could not solve unexpected problems; the ratio of successful problem solving was considerably low in the early stage. This explains the drastic improvement in Class B.

Table 1 shows the combinations of the revised models and the problems that each model could solve. The "1" means successful solution whereas the "0" means unsolved.

<table>
<thead>
<tr>
<th>Problems each revised model could solve</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple</td>
</tr>
<tr>
<td>--------</td>
</tr>
<tr>
<td>Problem 1</td>
</tr>
<tr>
<td>Problem 2</td>
</tr>
<tr>
<td>Problem 3</td>
</tr>
<tr>
<td>Problem 4</td>
</tr>
<tr>
<td>Problem 5</td>
</tr>
<tr>
<td>Problem 6</td>
</tr>
<tr>
<td>Problem 7</td>
</tr>
<tr>
<td>Problem 8</td>
</tr>
<tr>
<td>Problem 9</td>
</tr>
<tr>
<td>Problem 10</td>
</tr>
<tr>
<td>Problem 11</td>
</tr>
<tr>
<td>Problem 12</td>
</tr>
<tr>
<td>Problem 13</td>
</tr>
<tr>
<td>Problem 14</td>
</tr>
<tr>
<td>Problem 15</td>
</tr>
<tr>
<td>Problem 16</td>
</tr>
<tr>
<td>Problem 17</td>
</tr>
<tr>
<td>Problem 18</td>
</tr>
<tr>
<td>Problem 19</td>
</tr>
<tr>
<td>Problem 20</td>
</tr>
<tr>
<td>Problem 21</td>
</tr>
<tr>
<td>Problem 22</td>
</tr>
<tr>
<td>Problem 23</td>
</tr>
<tr>
<td>Problem 24</td>
</tr>
<tr>
<td>Problem 25</td>
</tr>
<tr>
<td>Problem 26</td>
</tr>
<tr>
<td>Problem 27</td>
</tr>
<tr>
<td>Problem 28</td>
</tr>
<tr>
<td>Problem 29</td>
</tr>
<tr>
<td>Problem 30</td>
</tr>
</tbody>
</table>

Table 1. Problems each revised model could solve
The models (vertical axis) are sorted based on their problem solving performance, i.e., the number of problems that the model could solve. The problems (horizontal axis) are categorized as simple, specific, and difficult, as mentioned in 3.1. The revised models are categorized based on the solution patterns of the three types of problems.

- **High performance models**: solved all problems successfully.
- **Medium performance models**: solved some of the difficult problems, but did not some of the specific problems.
- **Low performance models**: solved none of the difficult problems, but almost all of the specific problems.

### 3.3. Posttest

A posttest was performed in the final stage of the practice.

#### 3.3.1. Material

The posttest consisted of ten pulley problems that were categorized into three types from the viewpoint of their problem structures.

- **Simple Problems w/One Rope**: The problems contain only weights supported by a single rope.
- **Simple Problems w/Two Ropes**: The problems contain weights supported by two ropes. However, a force acting on each rope is independently calculated; therefore the addition of a single relatively simple rule enables the models to solve such problems. An example is a problem indicated in Figure 3(a).
- **Difficult Problems**: These are the difficult problems defined in 3.1. The ratio of the forces acting on the two ropes supporting a weight must be calculated. An example problem is indicated in Figure 3(c).

The perceptual superficial characteristics of the problems are manipulated independently from these problem structures. For example, some of the simple problems contained three pulleys and others eight pulleys. In two simple problems, a rod, which is connecting the involved pulleys, was a dummy object; these problems were solved only by a set of basic knowledge.

#### 3.3.2. Tasks

Two tasks were given to the participants.

- **Problem solving task**: The participants were required to calculate a force acting on a target rope.
- **Categorization task**: The participants were required to categorize ten problems into three categories.

#### 3.3.3. Results

For the problem solving task, all participants except one accurately solved more than nine problems among ten, but for the categorization task, the results varied among the participants.

Analysis of the results of the categorization task was performed excluding one participant whose performance of the problem solving task was extremely low. The number of problems that were accidentally categorized that didn't follow the normative
categorization defined in the materials section was analyzed. Figure 5 shows the mean number of such erroneous categorization in each of the three types of participants who constructed high, medium, and low performance models. An ANOVA revealed that the main effect reached significance ($F(2, 9)=7.25, p<0.05$). The LSD analysis showed that the number in the low and medium models was larger than that in the high model ($p<0.05$ and $p<0.05$). The figure implies that the participants who constructed the high performance models followed the normative categorization, but other participants tended to perform erroneous categorization.

![Figure 5. Result of categorical test](image)

4. Discussion and Conclusion

4.1. Experts versus Novices

Note that the manners of the problem categorization are largely different in each of the participants even though the performance of the problem solving task in every participant was almost identical. The result shows that the participants who constructed high performance models categorized problems based on the problem structures, and were not confused by their perceptual appearances. Chi et al. reported that in physics, experts categorized problems based on the structures of the problems, and novices tended to do so based on appearances [5]. The result of our practice indicates that the participants constructing high performance models successfully acquired high quality knowledge consistent with experts. Note that it remains unknown whether such knowledge acquisition comes from the experiences of creating sophisticated models, or whether they could create high performance models because they initially understood such knowledge. Speculations about the benefits of creating cognitive models were reported[6]. The investigation of such a causal relation is future work.

4.2. Emergence of model's limitation

It has been confirmed that the feedback of negative information improves the performance of learning and discovery from various perspectives such as unexpected findings [7], surprising results [8], anomaly [9], and falsifications [10]. In learning cognitive modeling, learners are also expected to acquire various types of knowledge by facing the limitations of their models and overcoming them. This implies that
investigating the methods of providing learners with adequate negative information about the limitations of their models is important.

In the current practice, we designed class activities where interaction among the participants naturally emerges, and through the interaction each participant faced his/her model's limitations by receiving negative information from others. Figure 4 shows that the performance of the revised models was drastically improved from the initial models. This evidence suggests that the class design proposed in this practice functioned well.

4.3. Generality of knowledge

As mentioned earlier, the addition of specific rules corresponding to individual situations enables the basic models to solve the specific problems. On the other hand, the difficult problems needed more complex rules to be solved; however, actually the addition of simple rules also gave the models the ability to solve the difficult problems. For example, see the example problem in Figure 3(c). In terms of the situation in the figure, when the relationship among the two pulleys on the left side, the leftmost weight, and the ropes connecting those objects is described in an if clause, let the ratio of the forces acting on the two ropes be two versus one. If a basic model acquires this knowledge, it can solve this difficult problem even though this knowledge can be applied only to this specific situation. Actually some participants handled the difficult problems by describing such specific rules.

In the current practice, a difference of the performance in the posttest was not detected between the participants who handled the difficult problems by adding specific knowledge and those by producing more complex rules that can be generally applied to various situations. From the viewpoint of quality for learning, the relationship between learning performance and the manners of representing rules is crucial. Further investigation of this issue remains important future work.

References