A Framework for Tutoring Computational Thinking: Learning Environment and Task Analysis

Kazuhisa Miwa

Graduate School of Information Science
Nagoya University
Nagoya, Japan 464-8601
e-mail: miwa@is.nagoya-ua.jp

Kazuaki Kojima

Learning Technology Lab.
Teikyo University
Utsunomiya, Japan 320-8551
e-mail: kojima@lt-lab.teikyo-u.ac.jp

***Abstract*—Computational thinking refers to thinking like a computer scientist. In this paper, we posit an approach of class design to train general university students in computational thinking. In our class practice, we let participants build a rule-based model to solve the following problem: There is a robot in a room with a banana and a box. Build a model with the knowledge to be given to the robot in order to make the robot get the banana. Computational thinking has four functions: decomposition, pattern recognition, abstraction, and algorithm (procedure). We discuss the participants' engagement to solve this problem and the four functions of computational thinking.**

Keywords - Computational Thinking; Cognitive Modeling; Tutoring

I. INTRODUCTION

Computational thinking refers to thinking like a computer scientist. In fact, computational thinking has great influence not only in the natural sciences of physics, chemistry, biology, but also in psychology, economics, literature, and psychiatry. The stage of the activity of computer scientists is now spreading in broad area.

Computational thinking is not reserved only for experts in these areas, but rather is applicable for anyone who engages in problem solving. In this respect, various efforts to implement computational thinking training in education should be emphasized. Above all, many attempts to develop computational thinking in the context of problem-solving education have been made in K-12 education [1][2].

In this paper, we posit an approach to including computational thinking training into the curriculum for general university students. In fact, in her paper on computational thinking, Jeannette M. Wing insists that professors of computer science should teach university freshmen subjects such as "a way to think like a computer scientist" in the department of computer science but also to other areas of study [3].

We have developed a framework called "Learning by Building Cognitive Models" in which general university students build rule-based cognitive models [4]-[7]. We developed a production system architecture for education, DoCoPro (Production system for anytime and anywhere), as learning environment for that purpose [8][9].

The first learning effect obtained there is the promotion of theory-based thought, which tries to understand the data in relation to the theory. For many university students, it is easy to explain data descriptively, but difficult to interpret data on a theoretical basis. As such, there is a gap between the data and the theory. A model is built by refining the theory, while predicting the data. That is, a model has the function of bridging theory and data. Based on this perspective, building a model to explain the data will promote activities to theoretically interpret the data [10].

The second learning effect is the refinement of the mental model and the improvement of mental simulation [11]. When behavior is observed, it is difficult to infer the mental model behind it. In our approach, when the result of calculation including error was observed, it was requested to identify the mental model behind it, i.e., the bug model. In doing so, participants were required to create a cognitive model that simulates the behavior (the result of the calculation including the error). By creating a cognitive model, participants can more clearly understand bugs that cause errors in cognitive procedures. In addition, using the mental model, simulating error generation becomes possible [12].

These efforts support that Learning by Building Cognitive Models can be used as a learning framework for fostering computational thinking. Based on this insight, this paper examines the following:

- In DoCoPro as learning environment, we propose an available learning task for fostering computational thinking.
- We examine that the proposed learning task is useful for fostering computational thinking based on the task analysis.

In Section 2, we indicate four core functions of computational thinking. In Section 3, we introduce a task and learning environment to test our approach. In Section 4, we discuss how computational thinking is specified in the task referring to the definition of computational thinking in this paper. Section 5 is our conclusions.

II. FOUR ELLEMENTS OF COMPUTATIONAL THINKING

Jeannette M. Wing posits that thinking like a computer scientist means more than just being able to program a computer, which requires multiple levels of abstract thought [3]. Computational thinking has several definitions; however,

 ISBN: 978-1-61208-705-4

computational thinking is consistently said to have the following four functions [13].

1) Decomposition
   Disassemble complex problems so they can be solved.
2) Pattern Recognition
   To see periodicity and law.
3) Abstraction
   Cut out branches and leaves and extract only important elements.
4) Algorithm (Procedure)
   Step by step, to clarify the problem-solving procedure.

We test if participants engage in cognitive activities with the four functions in our class practice.

III. TASK AND LEARNING ENVIRONMENT

We assessed whether participants engaged in cognitive activities using the four functions in our class practice.

In our class practice, we requested that participants build a rule-based model to solve the following problem. There is a robot in a room with a banana and a box (Figure 1). Build a model with the knowledge to be given to the robot to retrieve the banana (Figure 2). In order to retrieve the banana, move the robot to the same place as the banana. The robot can move to a high place by standing on the box. The robot can also move the box.

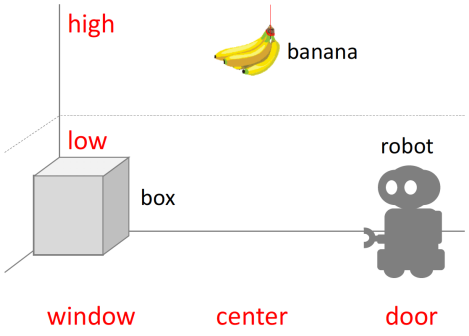

Figure 1. Initial state (stage (a))

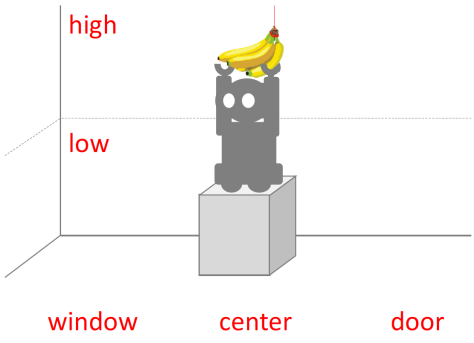

Figure 2. Goal state (stage (b))

DoCoPro was used as learning environment for building models. Below is a screenshot of DoCoPro (Figure 3). Representations of the states observed during the problem-solving processes are shown in the working memory in the left frame. The students created their models by editing rules in the editor in the right frames and simulating and evaluating problem-solving processes by executing the models with the controller in the upper frame.

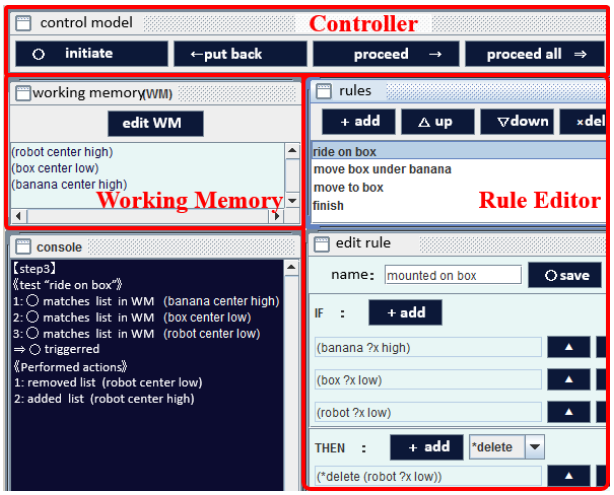

Figure 3. An example screenshot of DoCoPro as learning environment

The participants confirm that the robot can grasp the banana properly from this initial condition according to the model that they create.

Subsequently, the participants address the following problems in the next learning stage. The participants are presented with the scenario (as four new initial stages (c) to (f)) presented in Figure 4. If the conditions and operations are successfully set, the model can reach the goal state and stop even from any of the four newly presented initial states, or the model will be improved for reaching the goal.

IV. TASK ANALYSIS

Next, we discuss the participants' engagement in solving this problem and the four functions of computational thinking.

A. *Decomposition*

In order for the robot to acquire the banana, the participants are required to break down and assess the problem. To reach the target state (Figure 2) from the initial state (Figure 1), this problem is typically decomposed into the following four sub-problems.

- The robot moves toward the box.
- The robot moves the box under the banana.
- The robot stands on the box.
- The robot retrieves the banana.

B. *Pattern recognition*

See the initial stages (a) and (d). The initial knowledge for each situation is as follows.

Rule for (a):

 ISBN: 978-1-61208-705-4

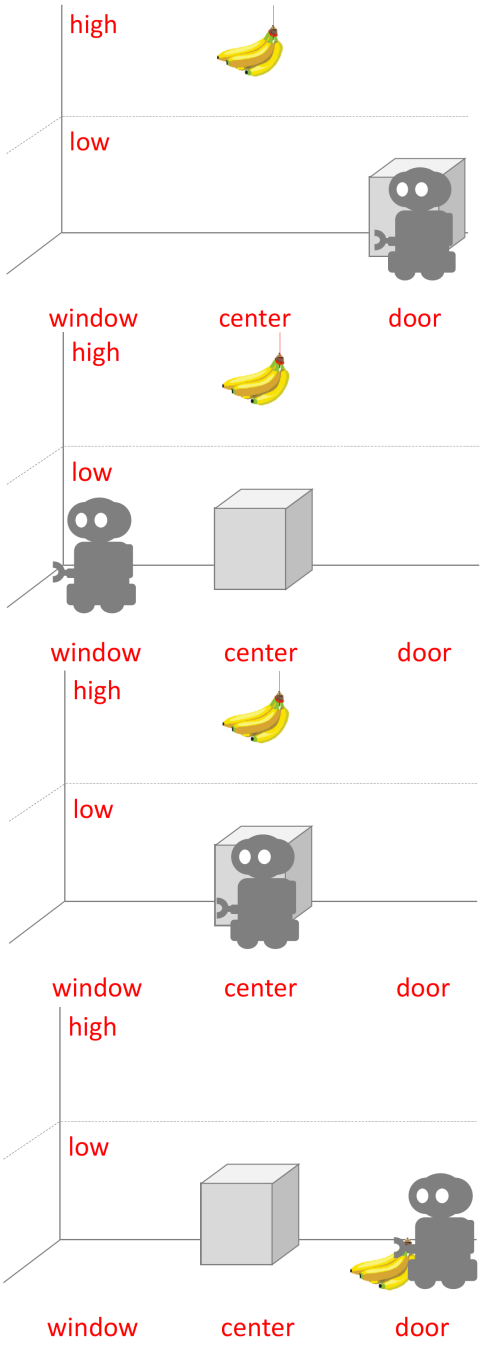

Figure 4. Four initial stages for model expansion: (c), (d), (e), and (f)

```
IF
Position of robot is RIGHT
Position of box is CENTER
THEN
Move robot to CENTER
```

Rule for (d):

```
IF
Position of robot is LEFT
Position of box is CENTER
THEN
Move robot to CENTER
```

If the participants can find the commonality of this knowledge, the following rule is drawn:

Rule for (a) and (d):

```
IF
Positions of robot and box are
different
THEN
Move robot to the position of box
```

C. Abstraction

The level of abstraction is represented in the if-clause of each rule. For example, the above rule for (a) and (d) is too abstract because it fires at the condition of the initial state (f), even though it should not because the robot does not need to go up on the box because the banana is on the floor. On the other hand, the above rules for (a) and (d) are too specific.

The adequate rule is as follows:

```
IF
Positions of robot and box are
different
Banana hangs from the ceiling
THEN
Move robot to the position of box
```

D. Automation

The crucial nature of our practice is that models are built as computer programs in DoCoPro. The participants can test if the model behaves as expected. The participants can improve the model while observing the behaviors of the model.

V. Conclusion

In this paper, we examined our learning framework to foster computational thinking for university students. We have established an educational production system architecture, DoCoPro, as learning environment. Our task analysis based on the four functions of computational thinking implies that our framework expect to work well for the educational setting.

The primary objective of this paper is to establish the foundation of our approach by formalizing the functions of computational thinking and analyzing the training task used in our approach. Additionally, we have begun to make an initial challenge for examining the utility of our framework through class practices.

We performed a preliminary class practice for evaluating our learning framework. The results indicated that the rules described in the pretests omitted many conditions in the pretests, whereas the presence of the conditions improved in the posttest. More detailed results were found in Kojima and Miwa, 2018 [14].

Acknowledgement

This work was supported by JSPS KAKENHI Grant Number 18H05320 and 15H02717.

 ISBN: 978-1-61208-705-4

VI. APPENDIX

Our production system, DoCoPro, is a specific notation of stages and rules. The following is the specific description of the two stages (a) and (b) seen in Figures 1 and 2.

Initial State (a) in Figure 1:

```
(Robot DOOR LOW)
(Box WINDOW LOW)
(Banana CENTER HIGH)
```

Goal State (b) in Figure 2:

```
(Robot CENTER HIGH)
(Box CENTER LOW)
(Banana CENTER HIGH)
```

The following is an example complete set of rules for solving the task. When the set of rules is applied to the initial stage (a): the robot moves to the box located in the window side by rule 1, the robot moves to the center of the room with the box by rule 2, the robot goes up to the box by rule 3, and the robot grasps the banana by rule 4. Through the above process, the goal state (b) is reached.

```
- name: Robot moves
  if:
  - (Robot ?x LOW)
  - (Box ?y LOW)
  - (Banana ?z HIGH)
  - (*test-not-equal ?x ?y)
  then:
  - (*delete (Robot ?x LOW))
  - (*deposit (Robot ?y LOW))
- name: Move box
  if:
  - (Robot ?x LOW)
  - (?z ?x LOW)
  - (Banana ?y HIGH)
  - (*test-not-equal ?x ?y)
  then:
  - (*delete (Robot ?x LOW))
  - (*deposit (Robot ?y LOW))
  - (*delete (?z ?x LOW))
  - (*deposit (?z ?y LOW))
- name: Ride on box
  if:
  - (Robot ?x LOW)
  - (Box ?x LOW)
  - (Banana ?x HIGH)
  then:
  - (*delete (Robot ?x LOW))
  - (*deposit (Robot ?x HIGH))
- name: Getting banana
  if:
  - (Robot ?y ?z)
  - (Banana ?y ?z)
  - (Hand EMPTY)
  then:
  - (*delete (Hand EMPTY))
  - (*deposit (Hand Banana))
```

REFERENCES

[1] A. Yadav, C. Mayfield, N. Zhou, S. Hambrusch, and J. T. Korb, "Computational thinking in elementary and secondary teacher education," Trans. Comput. Educ., vol. 14, no. 1, Mar. 2014, pp. 5:1–5:16. [Online]. Available: http://doi.acm.org/10.1145/2576872

[2] S. Grover and R. Pea, "Computational thinking in k-12: A review of the state of the field," Educational Researcher, vol. 42, no. 1, 2013, pp. 38–43.

[3] J. M. Wing, "Computational thinking," Communications of the ACM, vol. 49, no. 3, 2006, pp. 33–35.

[4] K. Miwa, "A cognitive simulator for learning the nature of human problem solving," Journal of Japanese Society for Artificial Intelligence, vol. 23, no. 6, 2008, pp. 374–383.

[5] K. Miwa, J. Morita, R. Nakaike, and H. Terai, "Learning through intermediate problems in creating cognitive models." Interactive Learning Environments, vol. 22, 2014, pp. 326–350.

[6] K. Miwa and K. Terai, H.and Shibayama, "Understanding procedural knowledge for solving arithmetic task by externalization," in Proceedings of ITS 2016, ser. LNCS, vol. 9684, 2016, pp. 3–12.

[7] K. Miwa and H. Terai, "Learning by building cognitive models that reflect cognitive information processing: A preliminary class exercise," in Proceedings of the ninth International Conference on Advanced Cognitive Technologies and Applications (Cognitive 2017), 2017, pp. 50–53.

[8] R. Nakaike, K. Miwa, J. Morita, and H. Terai, "Development and evaluation of a web-based production system for learning anywhere," in Proceedings of 17th international conference on computers in education, 2009, pp. 127–131.

[9] K. Miwa, R. Nakaike, J. Morita, and H. Terai, "Development of production system for anywhere and class practice," in Proceedings of the 14th International Conference of Artificial Intelligence in Education, 2009, pp. 91–99.

[10] H. Saito, K. Miwa, N. Kanzaki, H. Terai, K. Kojima, R. Nakaike, and J. Morita, "Educational practice for interpretation of experimental data based on a theory," in Proceedings of 21th international conference on computers in education, 2013, pp. 234–239.

[11] K. Miwa, J. Morita, H. Terai, N. Kanzaki, K. Kojima, R. Nakaike, and H. Saito, "Use of a cognitive simulator to enhance students' mental simulation activities," in Proceedings of ITS 2014, ser. LNCS, vol. 8474, 2014, pp. 398–403.

[12] K. Miwa, N. Kanzaki, H. Terai, K. Kojima, R. Nakaike, J. Morita, and H. Saito, "Learning mental models on human cognitive processing by creating cognitive models," in Proceedings of AIED 2015, ser. LNCS, vol. 9112, 2015, pp. 287–296.

[13] J. Krauss and K. Prottsman, Computational Thinking and Coding for Every Student The Teacher's Getting-Started Guide. CORWIN: A SAGE Company, 2017.

[14] K. Kojima and K. Miwa, "Preliminary study on fostering computational thinking by constructing a cognitive model," in Workshop Proceedings: 26th International Conference on Computers in Education, 2018, pp. 272–277.

 ISBN: 978-1-61208-705-4