International Journal of Learning Technologies and Learning Environments
International Institute of Applied Informatics
2026, Vol. 9, No. 1, IJLTLE896

Log Analysis Using Bayesian Networks in a Card Operation Based Programming Learning Support System

Toshikazu Kiyotaki [*], Natsumi Tanabe [†],
Shimpei Matsumoto [†], Shuichi Yamagishi [†]

Abstract

The Card Operation Programming Study Support System (COPS) was developed to help beginning students understand the structure of programs. COPS is a card-based programming method in which learners solve problems and create programs by rearranging cards according to problem statements. By sorting cards according to the problem statement, learners can visually understand the structure of the program needed to solve the problem. It is expected to have the same learning effect as the conventional coding format, and its educational effect has been suggested from the perspective of Learning Analytics (LA), which utilizes learning history data. In particular, a method for estimating learners ' thinking patterns by analyzing COPS usage logs using a Bayesian network (BN) has been proposed and its usefulness has been partially confirmed. However, several issues remain in the conventional LA approach, and factors such as the constraints of the probrems faced by the learner have not been adequately taken into account. Therefore, in this study, we propose a method to analyze COPS learning history data with BN to visualize how learners tackle problems in more detail. Specifically, we extract the correct and incorrect patterns of learners ' card operations from the logs and use BN to learn their structure in order to estimate how learners tackle and understand the problems. In our experiments, we analyzed the data obtained for basic problems in the C language. The results showed that certain card manipulation patterns were associated with incorrect answers, confirming that analysis using BN is effective in supporting learners.

Keywords: programming learning, programming education, Bayesian network, learning analytics

1 Introduction

In recent years, the importance of information education has increased due to the global spread and advancement of digital technology. A particular emphasis in programming learning is the acquisition of Computational Thinking (CT)[1], which refers to the ability to use computers to understand problems and devise procedures and methods to solve them. CT includes the ability

[*] Hiroshima University, Hiroshima, Japan
[†] Hiroshima Institute of Technology, Hiroshima, Japan

to decompose complex problems and to construct solutions in a logical manner, CT includes the ability to identify patterns and regularities and convert them into algorithms, as well as the ability to predict and evaluate results. Google has identified four main components of CT: Decomposition (problem decomposition), Pattern Recognition (pattern recognition), Abstraction (abstraction), and Algorithm Design (algorithm design) [1]. (Abstraction), and Algorithm Design (Algorithm Design) are the four main components of CT, and Google has established a course to provide these components [1].

Here, although CT does not simply refer to programming, programming is considered to support CT and to help learn important problem solving and design strategies that are carried over to domains other than programming [3]. For example, grammar, typing, etc. are not related to CT, but algorithms, etc. are considered to be related to CT. We assume that the card exercise method in this study allows students to concentrate on the very algorithms that are related to CT. Therefore, it is valuable not only for achieving the learning objectives of the programming class, but also for acquiring a part of CT.

However, since programming is multifaceted, it is considered to be a heavy learning burden for beginning students [4][5]. Furthermore, it has been pointed out that beginning students are prone to many grammatical errors due to typos, etc. [6]. Conversely, reducing the learning burden may reduce learners ' anxiety about programming and foster an enjoyable learning experience. Furthermore, the potential for improved programming learning outcomes through increased motivation and the ability to explore more content is significant.

To reduce the learning burden and effectively support programming learning, an information structure-oriented approach that breaks down the various technical elements required for programming and structures the entire learning process into open-ended tasks so that cognitive resources can be easily allocated to essential learning and is considered to be the most effective approach [7]. Under the information structure oriented approach, a card manipulation-based learning support system (hereafter referred to as COPS) has been developed to reduce task extrinsic load[8] in programming learning that considers the relationship among segmented meaningful parts[9]. in COPS, the learner divides the program code of a task into into meaningful segments, present them as multiple cards, and arrange them in the correct order. This approach allows the learner to focus on understanding the structure of the program, thereby focusing cognitive resources on essential learning.

The card manipulation method is not intended to completely replace traditional coding exercises, but rather to be used in conjunction with traditional teaching methods and the card manipulation method. Ishii (2016) conducted an experiment on cognitive load using a learning support system based on the card manipulation method in an actual programming class, and found that the “card manipulation method” reduced the cognitive load of grammar, typing, debugging, etc., and allowed students to focus their cognitive resources on algorithms, etc. The results suggest that the “card manipulation method” reduces the cognitive load of grammar, typing, debugging, etc., and focuses cognitive resources on algorithms, etc.[8]. In the model of cognitive load by Pass et al. performance is measured by the number of correct answers, time spent on the task, and so on. Among these, mental effort is considered to be the actual cognitive load.

There are several methods for measuring cognitive load (mental effort), including rating forms and physiological methods [11]. The rating form is based on a report that humans can properly rate their own mental effort [12], and asks learners to rate how much effort they have put in on a scale of several levels. This method has been used in many studies because it is very useful and easy to use.

In addition, Morinaga 's study (2017) revealed that it is an efficient learning method that can reduce learning time more than conventional methods while having the same learning effect as conventional coding-based learning [13].

However, programming learning using the card manipulation method is not a complete process on its own, but rather a complement to conventional lectures and coding exercises. Since class time is limited in educational institutions such as universities, it is a great burden for learners to perform all programming activities within that time. Therefore, a system such as the Card Operation Learning Support System is designed to divide classes into grammar-centered lectures and algorithm-centered exercises, with coding exercises conducted during self-study time.

Figure 1 shows the actual problem exercise screen of COPS, displaying question sentences, answer columns, and choice cards. The learner follows the instructions in the question text, inserts the cards with the program codes in the answer column, and rearranges them in the correct order. COPS is a learning support system intended to reduce the impact of non-intrinsic cognitive load as much as possible in programming learning, which focuses on thinking about relationships among parts. COPS is a learning support system intended to reduce the impact of non-essential cognitive load as much as possible in programming learning focused on thinking about relationships between parts. Feedback on correctness is provided upon submission, and administrators have access to a variety of log data on the learner.

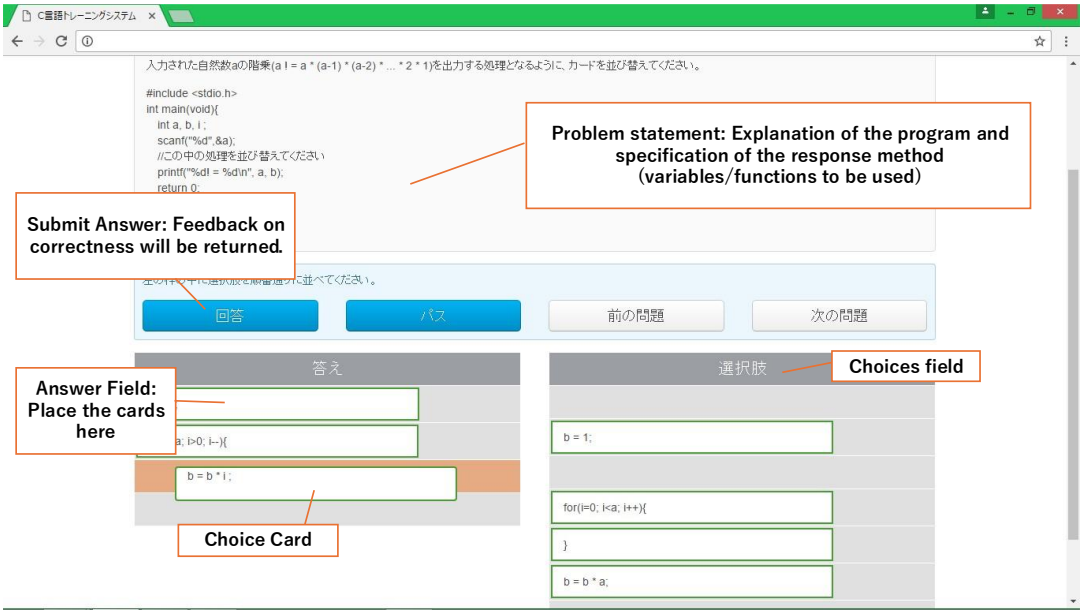

Figure 1: exercise screen of COPS

Recently, an initiative called LearningAnalytics (hereafter, LA) has been attracting a lot of attention. An example of LA targeting educational institutions is a system that detects learners who are in danger of dropping out and enables them to receive consideration so that they do not fail in a particular course [14]. As shown above, many educational improvements have been attempted based on the aggregation and statistical presentation of learning history data to date [15]. With this point in mind, previous studies have shown the possibility of educational improvement by the LA of COPS. For example, it has been suggested that there is a statistically significant relationship between the number of card operations and learners ' comprehension [16]. In addition, research has also been conducted to analyze COPS learning logs using a Bayesian Network (hereinafter referred to as BN). A method for estimating learners ' thinking patterns has been proposed, and its usefulness has been partially confirmed[17]. However, in the LA of COPS using BN, analysis using a model of BN that takes into account the constraints of the learning task and the characteristics of the learning task, such as dummy cards, have not been sufficiently addressed for problems in situations. Therefore, in this study, we practice LA of COPS by BN and analyze the model of BN that takes into account the constraints of the learning task and the characteristics of the learning task, two points that were left as future issues in the previous study [17]. The purpose of this study is to clarify the usefulness of the BN for the LA of COPS by conducting experiments to verify the validity of the estimated thinking patterns of learners.

2 Method

Regarding the BN used in this study, the BN is a model that approximates the simultaneous distribution of discrete probability distributions by an acyclic directed graph network structure with random variables as nodes and a set of conditional probability parameters. Directed links indicate dependencies among random variables, and the dependencies are quantitatively represented by a conditional probability table (CPT; Conditional Probability Table) (Ueno, 2013; Motomura & Iwasaki, 2006). The relationships between events are easy to understand visually, and various conditional probabilities can be easily obtained. By setting a definite value (evidence) for a node, the values of other nodes in the network are estimated based on Bayes ' theorem. This operation is called probabilistic inference, whereby in a BN that has already learned the structure and CPT, it is possible to estimate the probability values of the target variables by setting the obtained data as evidence. efficient inference. Figure 2 is a concrete example of a BN.

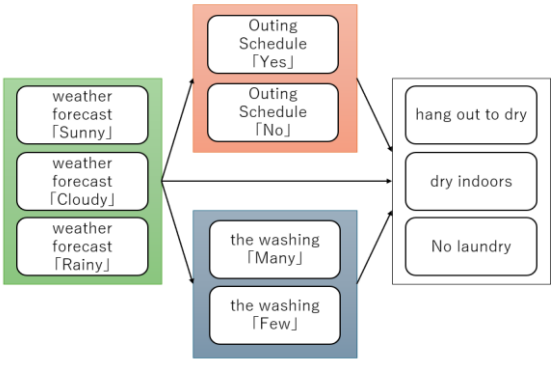

Figure 2: Visualization using multidimensional scaling method (conventional method)

Furthermore, interpreting a person 's thinking during a COPS exercise, one can interpret a person 's thinking as an operation of information structures, and that the operation is specified based on the information structure of the learning task. Therefore, in this study, we assume that human thoughts are represented as operations of information structures specified by the data and control dependencies that define the information structure of the program, and that the knowledge of the inference behind the operations can be inferred a posteriori by considering probabilities.

The study content used in this analysis consisted of two problems: 1) arrays and 2) iterations (for loops), each of which covered basic programming concepts. The experimental subjects were 20 undergraduate and graduate students who had already mastered the basics of the C programming language.

These tasks were designed to train beginning programmers to deeply understand data and control structures, and are suitable for analyzing learners ' behavior and thought processes through operation logs. Based on the data of card permutations recorded for each assignment, we aim to construct a BN and identify the thinking patterns of the learner in each problem.

Based on the learner 's operation logs, the correct and incorrect answer patterns of the card permutations are defined and these are set as input data to the BN. This makes it possible to identify the process leading to correct answers and the tendency leading to wrong answers, and to infer the thinking process of each learner. In addition, by visualizing the relationship between the frequency of correct and wrong answer patterns, it is expected to efficiently identify learners who are at high risk of dropping out and learners who are prone to certain wrong answer patterns.

Regarding the method of analysis, first, the state of the card permutation set in the answer column is obtained when a correct or incorrect diagnostic event occurs for each learning task. This card state is called the “card pattern. The card pattern is represented by a three-digit number, for example, if the question is to find the correct three-card permutation. Suppose there is a problem where the correct answers are cards numbered 1-3 and the dummy card is numbered -1. In this case, the correct answer state is represented as “([1][2][3])”. If there are cards with toggles, each toggle is considered a separate card. For example, if card number 3 has three toggles, it is considered that three patterns “([3_1][3_2][3_3])” existed, and these are reflected in the input data. Examples of cards that do not contain a toggle and cards that do contain a toggle are shown in Figure 3. An example of the logs obtained is shown in Figure 4.

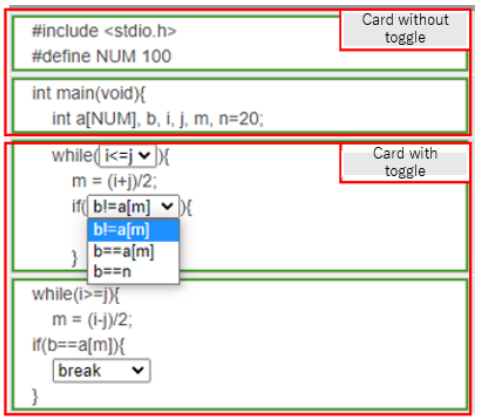

Figure 3: Example of cards with and without toggle

date	time	user_id	prot	stp	ope1	slot	card	check	exec	jdg	pattern
2023/7/25	15:30:22	12	7	0	PROB_BE	0	0	0	0	-1	0
2023/7/25	15:30:40	12	7	1	set	1	6	0	0	-1	[6]
2023/7/25	15:30:41	12	7	2	set	2	4	0	0	-1	[6][4]
2023/7/25	15:30:42	12	7	3	set	3	5	0	0	-1	[6][4][5]
2023/7/25	15:30:43	12	7	4	set	4	7	0	0	-1	[6][4][5][7]
2023/7/25	15:30:45	12	7	5	set	5	1	0	0	-1	[6][4][5][7][1]
2023/7/25	15:30:47	12	7	6	set	7	3	0	0	-1	[6][4][5][7][1][3]
2023/7/25	15:30:48	12	7	7	set	6	9	0	0	-1	[6][4][5][7][1][9][3]
2023/7/25	15:30:50	12	7	8	set	8	8	0	0	-1	[6][4][5][7][1][9][3][8]
2023/7/25	15:30:52	12	7	9	set	9	10	0	0	-1	[6][4][5][7][1][9][3][8][10]
2023/7/25	15:30:54	12	7	10	set	10	2	0	0	-1	[6][4][5][7][1][9][3][8][10][2]
2023/7/25	15:30:56	12	7	11	set	1	1	0	0	-1	[1][4][5][7][6][9][3][8][10][2]
2023/7/25	15:31:02	12	7	12	set	2	3	0	0	-1	[1][3][5][7][6][9][4][8][10][2]
2023/7/25	15:31:06	12	7	13	set	10	8	0	0	-1	[1][3][5][7][6][9][4][2][10][8]
2023/7/25	15:31:16	12	7	14	set	3	2	0	0	-1	[1][3][2][7][6][9][4][5][10][8]
2023/7/25	15:31:36	12	7	15	set	6	7	0	0	-1	[1][3][2][9][6][7][4][5][10][8]
2023/7/25	15:31:39	12	7	16	set	8	9	0	0	-1	[1][3][2][5][6][7][4][9][10][8]
2023/7/25	15:31:53	12	7	17	set	4	4	0	0	-1	[1][3][2][4][6][7][5][9][10][8]
2023/7/25	15:32:02	12	7	18	set	7	7	0	0	-1	[1][3][2][4][6][5][7][9][10][8]
2023/7/25	15:32:05	12	7	19	set	6	6	0	0	-1	[1][3][2][4][5][6][7][9][10][8]
2023/7/25	15:32:38	12	7	19	check	0	0	1	0	0	[1][3][2][4][5][6][7][9][10][8]
2023/7/25	15:33:39	12	7	19	check	0	0	1	0	0	[1][3][2][4][5][6][7][9][10][8]
2023/7/25	15:34:20	12	7	19	exec	0	0	0	1	0	[1][3][2][4][5][6][7][9][10][8]
2023/7/25	15:34:20	12	7	19	PROB_EN	0	0	0	1	0	[1][3][2][4][5][6][7][9][10][8]
2023/7/25	15:30:22	12	11	0	PROB_BE	0	0	0	0	-1	0
2023/7/25	15:34:26	12	11	1	set	1	10	0	0	-1	[10]
2023/7/25	15:34:29	12	11	2	set	3	10	0	0	-1	[10]

Figure 4: Example of logs to be obtained

Next, based on the log data obtained, a matrix is created to represent what kind of card patterns appeared by what kind of learners. The rows of the matrix represent learners, the columns represent card patterns, and each cell is set with a flag indicating “appearance”. The flag is set to 1 if learner X has made a particular card pattern Y appear, and 0 if not. In addition, a column named [result] is added to the final column of the matrix, and a flag is set for correct or incorrect answers based on the number of responses while checking the COPS operation log. Specifically, a flag of 1 is set if the number of responses is correct within 2 times in order to omit responses that are exhaustive and correct, and a threshold value of 0 is set for incorrect responses or responses that are correct 3 or more times. Such a matrix is created for each learning task to be analyzed and used as input data for the BN. An example of the matrix created is shown in Figure 5.

student_ID	0	-1	(1, 7)	(1, 2)	(1, 2, 3)	(1, 2, 3, 5)	(1, 2, 3, 4,	(1, 6, 3, 4,	(1,
12	1	1	1	1	1	1	1	1	
13	1	1	0	0	0	0	0	0	
14	1	1	0	1	0	0	0	0	
15	1	1	0	0	0	0	0	0	
16	1	1	0	1	0	0	0	0	
17	1	1	0	1	1	0	0	0	
18	1	1	0	1	0	0	0	0	
19	1	1	0	1	0	0	0	0	
20	1	1	0	0	0	0	0	0	
21	1	1	0	0	0	0	0	0	
22	1	1	1	0	0	0	0	0	
23	1	1	0	0	0	0	0	0	
25	1	1	0	1	1	0	0	0	
27	1	0	0	0	1	0	0	0	
28	1	1	0	0	0	0	0	0	
29	1	1	0	1	0	0	0	0	
30	1	1	0	1	0	1	0	0	

Figure 5: Example of matrix to be created for analysis

Next, structural learning is performed using the input data to build the BN model. During structural learning, card permutation nodes other than the correct card permutation "([1][2][3])" are set as prohibited nodes so that they do not have the correct permutation node as their parent. All other nodes are treated as candidate nodes. Greedy Search (greedy method) is used for the algorithm, and the Akaike Information Criterion (AIC) is used as the model evaluation criterion. As a condition for termination of structural learning, the search is set to stop when the average value of the cross tabulation reaches a threshold value of 0.01 or less.

In the constructed BN model, the correct card permutation node is used as the objective variable and the other card pattern nodes as explanatory variables to predict correctness. In addition, probabilistic inference is conducted by setting evidence at the correct answer node to investigate the probability change of the objective variable. In this process, we identify card patterns that have high probabilities at each correct and incorrect answer and examine their effects. Based on the inference results, we will also examine the relationships among card patterns and the role of specific patterns in the learner 's thought process.

Finally, to test the validity of the BN model, we conducted an accuracy assessment by comparing the model 's predicted results with actual correct and incorrect answer data. By doing so, we confirmed the extent to which the model accurately explained learners ' behavior patterns, and furthermore, we quantitatively evaluated the impact of specific card patterns on correct and incorrect answers. Through this analysis, we aimed to identify learners at high risk of dropping out and to gain insights that could lead to improvements in the design of learning tasks.

3 Results

Let 1) arrays and 2) iterations (for loops) be Problem 1 and Problem 2, respectively.

Problems 1 and 2 are shown in Figures 6 and 7.

Figures 8 and 9 show the visualization results of analyzing the operation logs in COPS for Problem 1 and Problem 2 with BN using the proposed method. The result represents the correct or incorrect result, and the card patterns that are inferred to affect the result are linked together.

These visualization results mean that the dependencies of card patterns related to questions 1 and 2 are properly visualized using BNs. Specifically, based on the dependencies shown by the BN, the results correctly reflect which card patterns influence the results, and the results are as predicted based on the analysis. The visualization results in Figures 8 and 9 indicate that the analysis method using the BN is working as intended, and that proper visualization was achieved for all problems.

Problem 1: Array

【Contents】Rearrange the cards so that the total in the array is 150.

【code】	【options】

#include <stdio.h>
int main(void){
int sum=0, i;
 	
//Please reorder the processes in this
 	
printf("sum = %d¥n", sum);
return 0;
}
	int a[5] ={10, 20, 30, 40, 50};
	for(i = 0; i < 5; i++){
	sum = sum + a[i];
	}
	---------- 	
	for(i = 0; i <= 5; i++){
	for(i = 0; i >= 5; i++){
	sum = a[i] + a[i+1];
	int a[5]={20, 30, 40, 50, 60};

Figure 6: Detail of Problem 1

Problem 2: Repeat

【Contents】Rearrange the cards so that the process outputs “12345”.

【code】	【options】

#include <stdio.h>
int main(void){
int i;
 	
// Please reorder the processes in this
 	
return 0;
}
	i = 0;
	while(i < 5){
	printf("%d",i+1);
	i = i + 1;
	}
	---------- 	
	i = 1;
	while(i <= 5){
	while(i > 5){

Figure 7: Detail of Problem 2

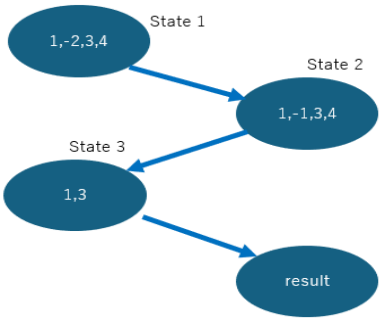

Figure 8: Visualization results using BN for problem 1

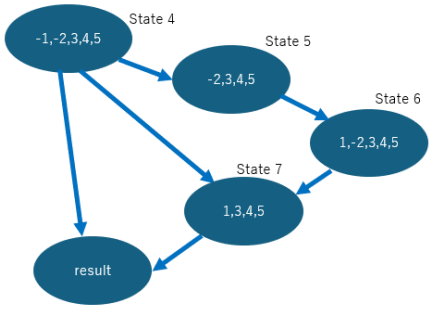

Figure 9: Visualization results using BN for problem 2

First, for Question 1, we set the evidence and check the probability of occurrence of each card pattern for states 1-3 when "result = correct" or "result = incorrect". The probability of occurrence of each card pattern in the case of "result = correct" is shown in Table 1.From Table 1, it is confirmed that the probability of occurrence of all card patterns is low when the correct answer is given.

Next, Table 2 shows the probability of occurrence of each card pattern when "result = incorrect" evidence is set. From Table 2, it can be inferred that the probability of occurrence of card patterns in state 3 is 0.7366 higher, indicating that state 3 has an impact on incorrect answers.

Furthermore, Table 3 shows the probability of occurrence of each card pattern when the evidence of "state 3 = 1 (occurrence)" is set.

Table 3 shows that there was no change in the percentage of correct answers when state 3 appeared.

Next, Table 4 shows the probability of occurrence of each card pattern when the evidence of "state 3 = 0 (non-occurrence)" is set. Table 4 confirms that when state 3 does not appear, the probability of correct answers is 0.9474.

From this result, it can be inferred that the probability of correct answers increases when state 3 does not appear.

Table 1: Probability of appearance of each pattern when result is set to appear

State 1	State 2	State 3
0.1048	0.1007	0.1344

Table 2: Probability of appearance of each pattern when result is set to not appear

State 1	State 2	State 3
0.2706	0.3501	0.7366

Table 3: Probability of appearance of each pattern when state 3 is set to appear

State 1	State 2	result
0.3431	0.4592	0.5

Table 4: Probability of appearance of each pattern when state 3 is set to not appear

State 1	State 2	result
0.0678	0.0451	0.9474

The next step is to validate this Bayesian network model.The objective variable is set to "result" and the explanatory variable to "state 3," and the results of the validation are shown in Table 5.The correct response rate was 0.9048 and the mean log-likelihood was 0.1758, confirming that there is no problem with the accuracy of this BN model.

The card patterns for states 1-3 and result are also shown in Figures 10 through 13.Comparing the card patterns of state 3 and result, which are considered "these incorrect patterns that appeared in student responses" in this problem, state 3 was a sequence that omitted the for statement from result.This suggests that if the for statement is considered when answering the question, it is more likely to lead to a correct answer, and if the for statement is not considered when answering the question, it is more likely to lead to an incorrect answer.

Table 5: Validation results of the BN model for Problem 1.

item	value
Number of data	21
Number of correct data	19
percentage of correct answers	0.9048
mean log likelihood	0.1758

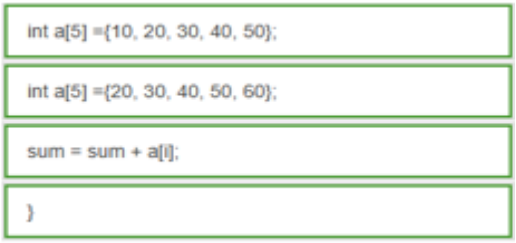

Figure 10: State 1 card pattern

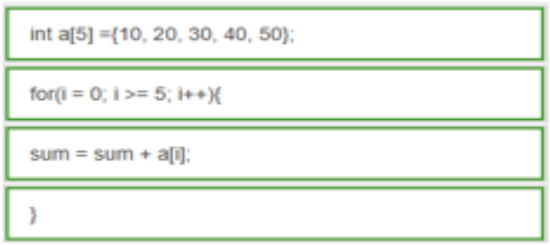

Figure 11: State 2 card pattern

Figure 12: State 3 card pattern

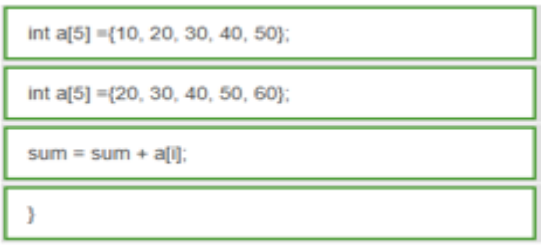

Figure 13: Result card pattern

Next, for Question 2, we set evidence in the result and checked the probability of occurrence of each card pattern from state 4 to 7 when the answer is correct or incorrect. Table 6 confirms that the probability of occurrence of any card pattern became lower when the correct answer was given. Next, Table 7 shows the probability of occurrence of each card pattern when the evidence for no occurrence is set. Table 7 shows that the probability of occurrence of the card patterns in state 4 and state 6 were 0.7367 and 0.7346, respectively, indicating that state 4 and state 6 had an effect on the incorrect answers. Table 8 shows the probability of occurrence of each card pattern when evidence of occurrence is set to state 4. Table 8 shows that the probability of correct answers did not change when state 4 appeared. Next, Table 9 shows the probability of occurrence of each card pattern when evidence is set for when state 4 does not appear. Table 9 shows that when state 4 did not appear, the probability of correct answers increased to 0.7971. From this result, it can be inferred that the probability of correct answers increases when state 4 does not appear. Table 10 shows the probability of occurrence of each card pattern when evidence of the appearance of state 6 is set. Table 10 shows that the probability of correct answers did not change when state 6 appeared. Next, Table 11 shows the probability of the occurrence of each card pattern when evidence is set for when state 6 does not appear. Table 11 shows that when state 6 does not appear, the probability of correct answers increases to 0.7111.

From this result, it can be inferred that the probability of correct answers increases when state 6 does not appear.

Table 6: Probability of appearance of each pattern when result is set to appear

State 4	State 5	State 6	State 7
0.0766	0.077	0.4815	0.4385

Table 7: Probability of appearance of each pattern when result is set to not appear

State 4	State 5	State 6	State 7
0.7367	0.5376	0.7346	0.4516

Table 8: Probability of appearance of each pattern when state 4 is set to appear

State 5	State 6	State 7	result
0.7241	0.8431	0.4384	0.1012

Table 9: Probability of appearance of each pattern when state 4 is set to not appear

State 5	State 6	State 7	result
0.0227	0.4443	0.4198	0.7941

Table 10: Probability of appearance of each pattern when state 6 is set to appear

State 4	State 5	State 7	result
0.5557	0.5	0.6865	0.4262

Table 11: Probability of appearance of each pattern when state 6 is set to not appear

State 4	State 5	State 7	result
0.1569	0	0.0337	0.7111

Table 12: Validation results of the BN model for Problem 2.

item	value
Number of data	73
Number of correct data	61
percentage of correct answers	0.8356
mean log likelihood	0.4086

Next, we will test this BN model. Table 12 shows the results of the validation with [result] as the objective variable and [state 4] and [state 6] as explanatory variables. The correct response rate was 0.8356 and the mean log-likelihood was 0.4086, confirming that the accuracy of this BN model was not a problem. The card patterns for states 4 to 7 and the result are shown in Figures 14, 15, 16, 17, and 18. Comparing the card patterns of states 4, 6, and result, which should not appear in this problem, we found that states 4 and 6 either used a sequence with a wrong definition of a variable or a sequence with a wrong number of repetitions of a while statement, or both of them were different. This indicates that the variable and the state of the problem are not correct when solving the problem. From this, it was considered that if the variable and the number of iterations were kept in mind when solving the problem, the answer was more likely to lead to a correct answer, and if not, it was more likely to lead to an incorrect answer.

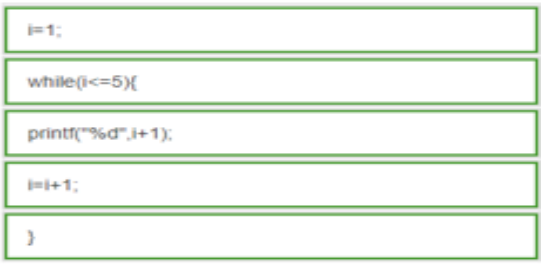

Figure 14: State 4 card pattern

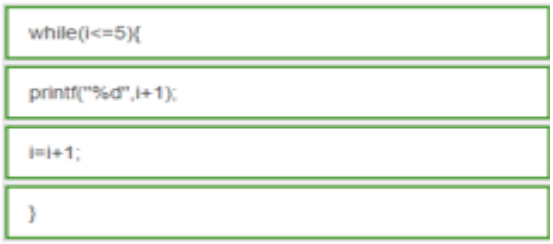

Figure 15: State 5 card pattern

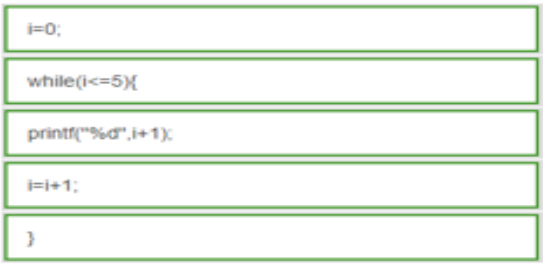

Figure 16: State 6 card pattern

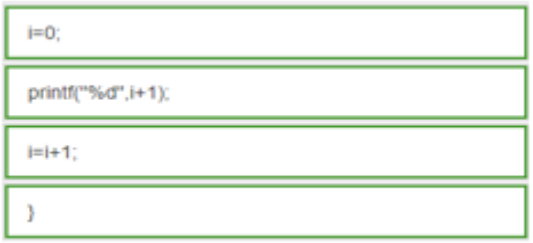

Figure 17: State 7 card pattern

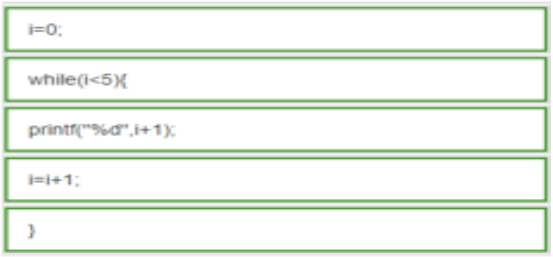

Figure 18: Result card pattern

4 Conclusion

This study examined the usefulness of introducing the concept of BN to evaluate and visualize the learning process in programming learning using COPS. by utilizing BN, it was possible to identify card patterns that influence the correct answer card patterns and to determine whether they appear or not. By setting the evidence for the occurrence of each card pattern, it was possible to determine the probability of occurrence of each card pattern. We also set evidence for the card patterns that had a higher probability of occurrence and investigated how they affected the probability of being able to answer correctly. Furthermore, we validated the model to prove that the results were appropriate and confirmed that the BN estimation results were consistent with the actual learner data. Then, by identifying and discussing the card patterns that are likely to lead to incorrect answers, we found that the proposed method is applicable to the analysis of COPS

learning logs and is useful for further helping learners.

The analysis results of this study confirm that certain card patterns are associated with a decrease in the percentage of correct answers. In particular, we found that codes omitting the for statement and patterns in which the order of processing was not appropriate frequently appeared as incorrect answers. This made it possible to clearly identify the points at which learners are prone to make mistakes, and provided important clues for formulating an appropriate instructional policy. Furthermore, in order to evaluate the validity of the estimation results obtained by the proposed method, we tested the model fit. The results showed that the BN model performed adequately in terms of mean log-likelihood and correct response rate, thus confirming the validity of the analysis results.

As a future direction, based on the results of this study, we plan to implement a feedback function utilizing BN in COPS. Specifically, we plan to construct a system that enhances learning effectiveness by generating real-time feedback based on learners' answer patterns and clearly indicating the tendency of wrong answers. For example, when a learner exhibits a particular pattern of wrong answers, the system may incorporate a function to analyze the cause of the pattern and provide appropriate hints to facilitate understanding. In addition, in order to generalize this method, we will also consider its application to programming learning environments other than COPS. By linking the method to online learning systems and conducting real-time analysis on a larger number of learners, it could be used in a wide range of educational settings.

By utilizing the results of this research and developing it into a support tool to deepen learners' understanding, it is expected to contribute to improving the efficiency of programming learning and realizing individually optimized instruction. In the future, we intend to further verify the effectiveness of the proposed method by implementing the feedback function in COPS and conducting demonstration experiments, with the aim of creating a more effective learning support environment.

Acknowledgement

This work was partly supported by Grant-in-Aid for Scientific Research (C) No. 22K02815 and No.23K02697 from the Japan Society for the Promotion of Science (JSPS).

References

[1] Google： “Exploring Computational Thinking.” <https://research.google/blog/exploring-computational-thinking/> (see 3.18.2025).

[2] Mungunsukh Jambalsuren, Zixue Cheng: "An Interactive Programming Environment for Enhancing Learning Performance," The University of Aizu, pp. 201–212 (2002).

[3] Resnick M.， et al. : “Scratch: programming for all.” Communications of the ACM，52.11，pp.60-67（2009）.

[4] S.Matsumoto, Y.Hayashi, and T.Hirashima, Development of a program-ming learning system through card manipulation focusing on thinking about relationships between parts, Transactions of the Institute of Electrical Engineers of Japan C, Vol.138, No.8, pp.999-1010 (2018).

[5] M.Makino, Data Mining in e-Learning, Transactions of the Japan Society for Educational Technology, vol.31, no.3, pp.271-283 (2007).

[6] Okamoto, M., Kida, I.: "A Typology of Stumbling Stones of Beginning Learners in 'Copy-book Learning' of Programming and Its Consideration." Bulletin of the Practice Center, Vol. 22, pp. 49-53 (2014).

[7] S.Garner, A Tool to Support the Use of Part-Complete Solutions in the Learning of Pro-gramming, Proceeding deconference, pp.222-228, 2001.

[8] Motoki Ishii et al, Evaluation of a Programming Learning Support System Using a Card Method from the Perspective of Cognitive Load, Proceedings of the 79th Conference on Advanced Learning Science and Engineering, SIG-ALST/SIG-ALST,B5(03),pp.11-16, 2017.

[9] S. Matsumoto, Y. Hayashi, and T. Hirashima, Development of a Programming Learning System by Card Manipulation Focused on Considering Relationships among Parts, IEEJ Transactions C, Vol. 138, No. 8, pp. 999-1010, 2018.

[10] S. Iwamoto, S. Morinaga, S. Matsumoto, Y. Hayashi, and M. Hirashima, Learning Analytics in Programming Learning Support System in Card Operation Method, Proceedings of the 2019 Student Research and Presentation Meeting of the Association for Information Science and Systems in Education, Chugoku Region, O06, pp.187-188, 2020 .

[11] T. Hirashima, Learning task-centered learning research: redefining learning tasks as information structures and designing learning activities as structural operations. Artificial Intelligence, 30(3), 277-280.2015.

[12] Garner S. : "Learning to program using part-complete solutions." Computer Science, p.8-14, 2003.

[13] Ruka Murakami, L. Morinaga, S. Matsumoto, Y. Hayashi, and T. Hirasshima, Learning Effects of a Programming Learning System with Card Operation Method, Proceedings of the 2016 JSiSE Student Research Conference, pp. 203-204, 2018.

[14] K.E. Arnold and M.D. Pistilli. Course signals at purdue: using learn ing analytics to increase student success. Proceedings of the 2nd International Conference on Learning Analytics and Knowledge, pp. 267–270, 2012.

[15] Ueno M. Data Mining in e-Learning. Transactions of the Japan Society for Educational Technology, Vol. 31, No. 3, pp. 271-283, 2007.

[16] S. Iwamoto, S. Morinaga, S. Matsumoto, Y. Hayashi, and T. Hirashima. Learning Analytics in a Programming Learning Support System with Card Operation. In Proceedings of the 2019 Student Research Conference of the Association for Information Science and Systems in Education, pp. 187-188, 2020.

[17] T. Okudaira, T. Shigematsu, S. Matsumoto. A Study of Learning Analytics Using Probabilistic Models in a Programming Learning Support System with Card Manipulation.The 25th IEEE Hiroshima Section Student Symposium, pp.115-118, 2023.