

# A Statistical Analysis on the Order of Understanding for the Java Programming Language

**Yasuhiro Takemura\***

\*Osaka University of Arts  
Junior College  
2-14-19 Yata, Higashi-sumiyoshi-ku,  
Osaka 546-0023 Japan  
+81-6-66941-7341  
yasuhi-t@is.aist-nara.ac.jp

**Ken-ichi Matsumoto<sup>†</sup>**

<sup>†</sup>Graduate School of Information Science  
Nara Institute of Science and Technology  
8916-5 Takayama, Ikoma  
Nara 630-0101, Japan  
+81-743-72-5312  
[matumot,torii]@is.aist-nara.ac.jp

**Katsuro Inoue<sup>‡</sup>**

<sup>‡</sup>Graduate School of Engineering Science  
Osaka University  
1-3 Machikaneyama, Toyonaka  
Osaka 560-8531, Japan  
+81-6-6850-6571  
inoue@ics.es.osaka-u.ac.jp

**Koji Torii<sup>†</sup>**

## ABSTRACT

By extracting the order of understanding for the Java programming language from results of examinations, we investigate a learning process for efficient acquisition of knowledge on programming languages, and positions of knowledge items in textbooks. The order of understanding is extracted by statistically analyzing the ordering relation of knowledge (the relation that one piece of knowledge is required for understanding another piece of knowledge) and by constructing the whole structure of those relations. In the statistical analysis, “correct random guesses” and “careless mistakes” are statistically revised to extract ordering relations.

## Keywords

Java programming language, order of understanding, learning process

## 1 INTRODUCTION

For subjects who have a long history of accumulated teaching practice and knowledge, the ordering relation of knowledge (the relation where one piece of knowledge is required for understanding another piece of knowledge) has been extracted, and a systematic teaching order derived from the order relation of knowledge has been realized. Since the teaching of programming languages does not have a long history of teaching practice compared with other subjects, there exists no systematic teaching method which has obtained a consensus among experts on education [1], [2]. For example, some textbooks can have different orders of knowledge items to each other. In the learning phase for a new domain, some teachers can use learning materials in a different order to each other. Therefore, in order to realize an efficient and systematic teaching of programming, the order of understanding of the programming language should be clarified [3], [4], [5], [6], [7], [11], [12].

In this paper, we extract the order of understanding for the Java programming language from results of examinations, and investigate a learning process for efficient acquisition of knowledge on programming languages and positions of knowledge items in textbooks. The order of understanding

is extracted by statistically analyzing the ordering relation of knowledge (the relation that one piece of knowledge is required for understanding another piece of knowledge), and by constructing the whole structure of those relations. In the statistical analysis, “correct random guesses” and “careless mistakes” are statistically revised to extract ordering relations. We also perform factor analysis on knowledge items and construct a relation structure.

## 2 ANALYSIS OF ORDERING RELATIONS IN THE JAVA PROGRAMMING LANGUAGE


Knowledge of the Java language can be modeled by the Ordering Theory because, as in mathematics or natural science, there are two types of knowledge of the language: one is fundamental and independent of other knowledge, and the other is dependent on other knowledge. In this paper, we adopt the model of Ordering Theory proposed by P. W. Airasian [16] to extract the ordering relation of knowledge.


In the Ordering Theory, test item X and test item Y are regarded as having an ordering relation if the ratio of the size of set M01 (the set consisting of the learners who answered questions incorrectly to an easy test item X, and correctly to a difficult test item Y) to N01 (the number of all the learners) is small. However, in Multiple-choice questions, there can be so-called “correct random guesses” and “careless mistakes”. In the Ordering Theory, the noise is not taken into consideration in the derivation of ordering relations.


In this paper, we adopt a statistical analysis method [15], which can revise the inaccurate estimates caused by the so-called “correct random guesses” and “careless mistakes” and extract ordering relations. In the statistical analysis, we suppose that answers contradicting the Ordering Theory are caused by “correct random guesses” and “careless mistakes” and we use Binomial distribution to revise the inaccurate estimates and extract ordering relations. In addition, we simplify the relational structure by applying Factor Analysis to each knowledge item. The change in numbers of errors and correct answers caused by “correct random guesses” and “careless mistakes” in test item X and test item Y is shown in Table 1.


**Table1** Changes in numbers of errors and correct answers caused by “correct random guesses” and “careless mistakes”

		X	
		error	correct
Y	error	$N_{00}$ $S_0$	$N_{10}$
	correct	$k_0 + e_0 + k_1 + e_1$	$N_{11}$ $S_1$

 set  $S_0$  of including “random correct guesses”

 set  $S_{01}$  of including “careless mistakes”

 Answer transition caused by “correct random guesses” of test item Y

 Answer transition caused by “careless mistakes” caused by “careless mistakes” of test item Y

$S_0$  • Set of  $k$  learners who gave “random correct guesses”

$e_0$  • Number of learners in both  $e$  and  $S_0$  •  $\mathbf{0} \cdot .e_0 \cdot .k$

$k_0$  • Number of learners who gave a “correct random guess” for test item Y,  $k_0 = k - e_0$

$e$  • Number of learners who gave neither “correct random guesses” nor “careless mistakes” •  $\mathbf{e} = e_0 + e_1 \cdot \mathbf{0} \cdot .e \cdot .N_{01}$

$e_1$  • Number of learners in both  $e$  and  $S_1$  •  $\mathbf{0} \cdot .e_1 \cdot .N_{01} - k$

$k_1$  • Number of learners who gave a “careless mistake” to test item Y

In this paper, we name the knowledge contained in a test item a “knowledge item”, a group of test items a “cluster”, the learners transiting to a contradictive set M01, “contradictive learners”, the average probability of the occurrence of a contradictive learner in M01 a “inconsistency rate”, the function for computing a inconsistency rate “contradiction function”, and the whole structure indicating relational structures of test items and clusters a “relational structure”.

### 3 CONSTRUCTION OF THE RELATIONAL STRUCTURE

On the basis of the ordering relations derived from inconsistency rates and the relational structure in clusters, the order of understanding of the Java language is processed as follows.

Expression (1) is contradiction function introduced in the statistical analysis. Inconsistency rates are calculated by applying the probability  $f$  of “correct random guess” and the probability  $c$  of “careless mistake”. The relational structure of test items is constructed by ordering relations extracted with inconsistency rates smaller than a constant.

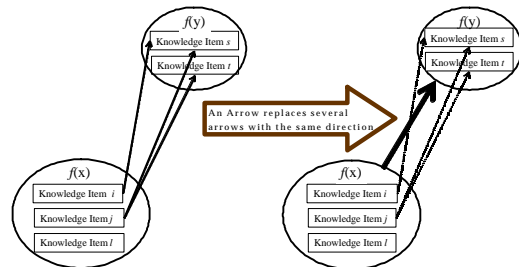
Knowledge items are merged to form several clusters by

$$P(N_{00}, N_{01}, N_{11}, f, c) = \frac{1}{N_{01} + 1} \sum_{k=0}^{N_{01}} B(k; N_{00} + k, f) B(N_{01} - k; N_{11} + N_{01} - k, c) \mathbf{1} \quad (1)$$

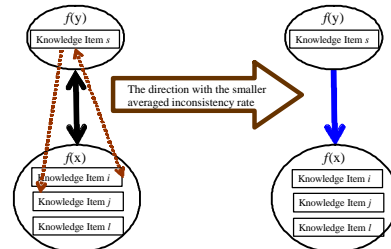
where  $B(k; n, q) = \sum_{x=0}^k \binom{n}{x} q^x (1-q)^{n-x} c$

using factor loadings calculated in the Factor Analysis of test results. Factors are also extracted.

The relational structure consisting of only knowledge



**Figure1** Simplification of the relational structure of knowledge items : ordering relations with a similar direction



**Figure2** Simplification of the relational structure of knowledge items : ordering relations with inconsistent directions

	correct answer of the test items	knowledge required for the answers
A 1	attribute	instance variable
A 2	enums class	super class
A 3	class variable	class variable
A 4	inheritance	inheritance
A 5	sub class	sub class
A 6	attribute	instance variable
A 7	enum class	super class
A 8	interface	interface
A 9	instance method	method
A 10	method	method
A 11	instance variable	instance variable
A 12	super class	super class
A 13	class method	class method
A 14	interface	interface

**Figure3** Simplification on the relational structure of knowledge items : transitive ordering relations

items, which is constructed in ., has two many arrows indicating an ordering relation. This structure is so complicated that capturing the order of understanding from this structure is difficult. Therefore, we simplify the relational structure by transforming the ordering relations of knowledge items to those of clusters. Figures 1,2, and 3 show how to form clusters to simplify the relational structure of knowledge items.

#### 4 EXPERIMENTS FOR EXTRACTING ORDERING RELATIONS

In the experiments for extracting ordering relations of the Java language, we gave a test to subjects. The test consists of 15 problems (A1 to A15) on the knowledge of the Java language. We adopt Multiple-choice questions because quantitative analysis is easier and more subjects can answer Multiple-choice questions. The answer to each test item is judged as "correct" or "error". The summary of the test (correct answers and knowledge items needed) is shown in Table 2.

A total of 96 subjects consists of three groups with different majors: 42 students majoring in cultural sciences (half a year (once a week)), 24 workers of an electric industry (intensive course (two days (7 hours a day))), and 30 graduate students majoring in information science (half a year (once a week)). A single teacher delivered lectures and exercise classes to all the three groups, in order to decrease the differences of teaching effects caused by the differences of teaching environment. In order to decrease the difference of difficulty between questions, we set basic questions for each knowledge item.

The tests are conducted as follows.

- . The teacher delivers a lecture on basic knowledge from materials related to the Java language.
- . The learners do programming exercises such as revision,

compiling and execution of a program.

. After the exercise, the learners take a written test on the Java language.

#### 5 THE CONSTRUCTION OF RELATIONAL STRUCTURE

We construct a relational structure following the construction process described in Section 3.

Inconsistency rate is the proportion of applying contradiction function P to test data. 15 knowledge items are merged to form clusters using factor loadings. In this paper, Factor

Material #	@	Material # A	Material # B	Material # C
11,168	Variable	47	50,120	Variable
43,124	Method	49	114	Class
124	Variable	135	121	Super Class
163	Super Class	67	125	Sub Class
163	Sub Class	69	125	Method
197	Interface	80	166	Interface

exists if inconsistency rates are less than 0.1 (the inconsistency rates less than 0.1 are highlighted).

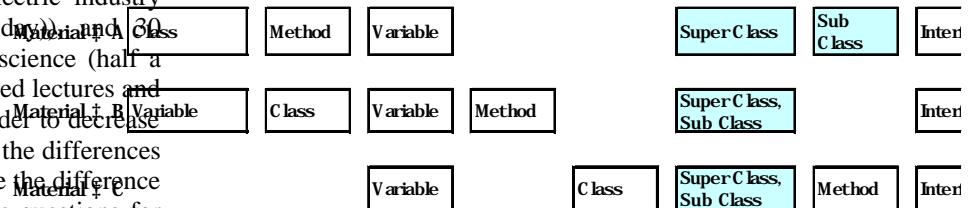
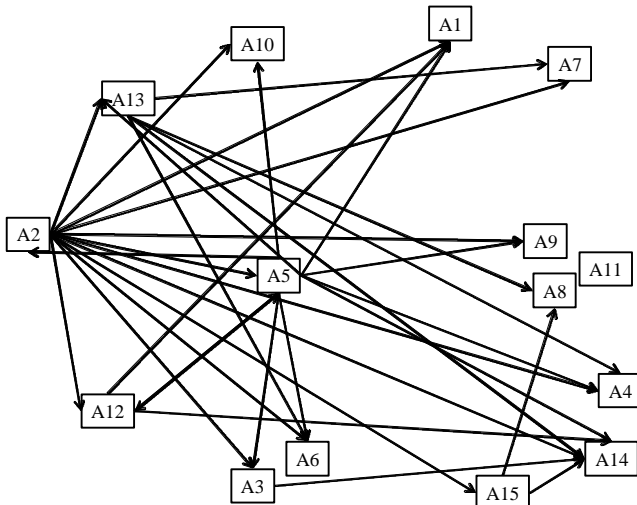


Table3 Inconsistency rate and clusters

			£(1)				£(2)		£(3)			£(4)	£(5)	£(6)	£(7)		
			(A11)	(A4)	(A13)	(A9)	(A8)	(A12)	(A5)	(A1)	(A7)	(A6)	(A3)	(A2)	(A10)	(A15)	(A14)
£(1)	Instance variable	(A11) Instance variable	0.00	0.39	0.14	0.57	0.77	0.41	0.19	0.76	0.73	0.58	0.23	0.14	0.50	0.57	0.83
		(A4) Inheritance	0.52	0.00	0.07	0.55	0.82	0.13	0.05	0.64	0.57	0.46	0.46	0.02	0.40	0.43	0.73
		(A13) Class method	0.62	0.47	0.00	0.65	0.84	0.16	0.03	0.65	0.66	0.58	0.48	0.02	0.27	0.56	0.79
		(A9) Instance	0.54	0.40	0.14	0.00	0.76	0.32	0.06	0.52	0.63	0.54	0.32	0.03	0.34	0.52	0.71
£(2)	Super Class, Sub Class	(A8) Class	0.16	0.21	0.01	0.14	0.00	0.41	0.16	0.30	0.26	0.24	0.20	0.10	0.14	0.06	0.53
		(A12) Super class	0.73	0.53	0.16	0.72	0.90	0.00	0.01	0.61	0.72	0.66	0.41	0.04	0.35	0.65	0.80
£(3)	Attribute	(A5) Sub class	0.72	0.56	0.15	0.67	0.89	0.08	0.00	0.67	0.74	0.65	0.46	0.01	0.33	0.64	0.83
		(A1) Attribute	0.74	0.53	0.14	0.52	0.79	0.08	0.06	0.00	0.28	0.19	0.41	0.03	0.26	0.57	0.74
£(4)	Class Variable	(A7) Super class	0.68	0.33	0.08	0.58	0.76	0.24	0.12	0.18	0.00	0.42	0.42	0.08	0.36	0.39	0.69
		(A6) Attribute	0.58	0.31	0.07	0.57	0.79	0.22	0.05	0.24	0.53	0.00	0.60	0.03	0.25	0.30	0.73
£(5)	Thought	(A3) Class variable	0.55	0.60	0.27	0.63	0.85	0.18	0.08	0.66	0.71	0.74	0.00	0.02	0.37	0.63	0.78
£(6)	Method	(A2) Super class	0.74	0.60	0.21	0.70	0.90	0.29	0.06	0.70	0.76	0.68	0.45	0.00	0.47	0.63	0.84
£(7)	Package	(A10) Method	0.71	0.59	0.10	0.66	0.85	0.18	0.04	0.62	0.70	0.59	0.42	0.05	0.00	0.52	0.81
		(A15) Package	0.62	0.38	0.13	0.61	0.77	0.30	0.10	0.65	0.57	0.39	0.46	0.03	0.24	0.00	0.68
		(A14) Interface	0.62	0.10	0.01	0.26	0.67	0.03	0.02	0.34	0.30	0.28	0.09	0.01	0.18	0.05	0.00

In the computation of inconsistency rates, the probability  $f$  of “correct random guesses” and the probability of “careless mistakes” are set as follows. Probability  $f$  is set as 0.2 because there are 5 choices in multiple-choice questions. Probability  $c$  is set as 0.071, which is computed from data measuring the levels of understanding in three learning phases (before learning, after learning with materials, after learning with exercises) [14].

Figure 4 shows the relational structure of test items constructed with the inconsistency rates in Table 3. Figure 5 shows the relational structure of clusters.



## 6 DISCUSSION ON THE LEARNING PROCESS

Figure 6 shows an example of an efficient learning process derived from the simplified relational structure following rule a and rule b. Rule a states that clusters adjacent to each other in the relational structure should not be separated. Rule b states that clusters not adjacent to each other are arbitrarily ordered.  $f(5)$ :Thought is excluded from the learning process because it refers to a factor related to

thought. The following process is derived with  $f(2)$ :Super class as a starting point.

$f(2)$ :Super class, Sub class .  $f(3)$ :Attribute .  $f(6)$ :Method .  
 $f(4)$ : Class Variable .  $f(7)$ :Package .  $f(1)$ : Instance Variable

In this process, the learning of  $f(2)$ :Super class, Sub class is followed by the learning of  $f(4)$ : Class Variable and  $f(1)$ : Instance Variable. In this process, after the learning of the concept of “class”, the learning proceeds to knowledge items requiring compound knowledge. For this reason, this process allows an efficient learning of the Java language.

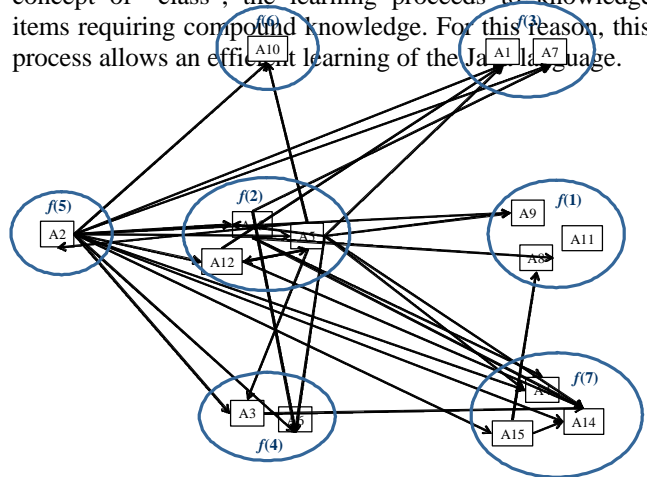


Figure 5 Relational structure of clusters

Next, we discuss the differences between the relational structure of knowledge items and the positions of the learning items written in a learning material . shown in Table 4 [8], [9], [10], [13]. In the relational structure of Figure 7, Sub class is the only knowledge item to Method. Also in the material, Sub class is followed by Method. This means that our relational structure follows the same relational order as the learning material for this point. On the other hand, knowledge item Interface requires compound knowledge. Therefore, our relational structure

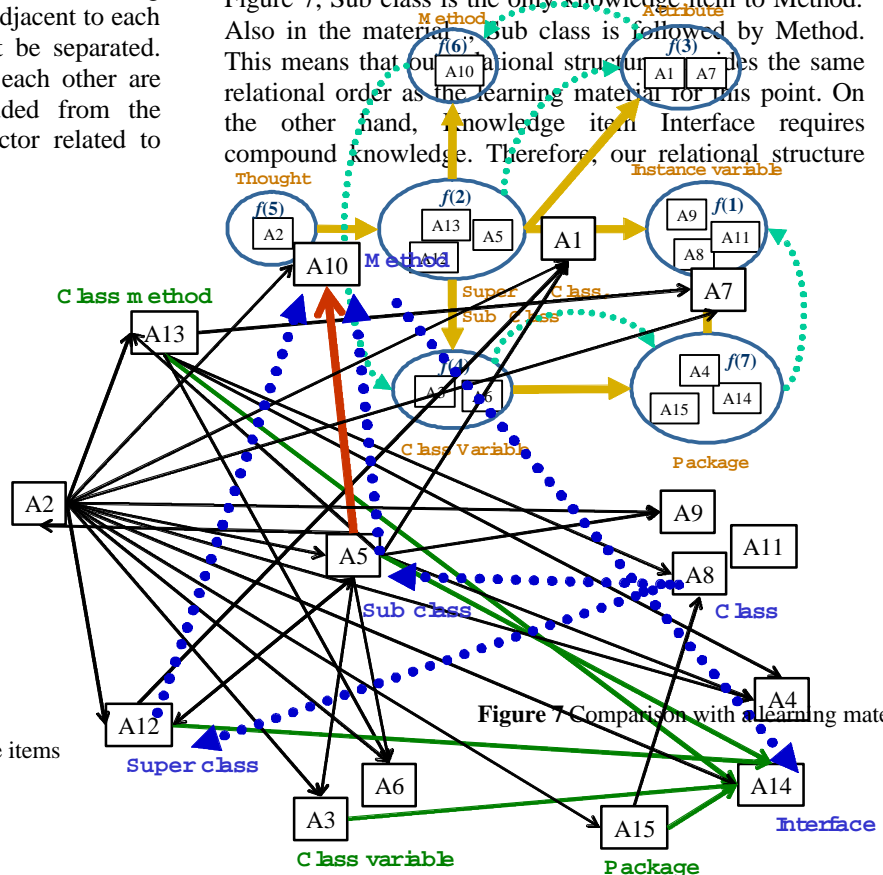


Figure 7 Comparison with a learning material .

Figure 4 Relational structure of knowledge items

contains several ordering relations from other knowledge items to Interface, and also in the learning material ., Interface is positioned at a later part of the learning process. Thus, the ordering relations of basic knowledge and compound knowledge are consistent with the result of the experiment. However, other learning items are positioned differently. Therefore, learning items for which the level of understanding cannot be easily captured depend on the experience and knowledge of the author of the learning materials.

## 7 CONCLUSION

In this paper, we have extracted the order of understanding of the Java language, which has previously been estimated by the experience of teachers in learning. For this purpose, we adopted a statistical analysis based on the Ordering Theory to extract the ordering relations of knowledge items. We also constructed a relational structure.

The data on 15 test items obtained in the experiment were analyzed with the Factor Analysis. Knowledge items were merged and they formed 7 clusters according to the factor loadings. The complicated relational structure of knowledge items was simplified to the relational structure of clusters.

We discussed the difference between the learning process derived from the relational structure and the learning process from learning materials. We conclude that this difference can be used to create a systematic learning process.

**figure 6** Learning process derived from a relational structure

The statistical analysis method used in this paper depends on the problems in tests. The meaning of each factor in the Factor Analysis must be manually determined. Therefore, the application of the proposed method is limited to a certain domain of the Java language. However, if problems for tests are prepared and the meaning of each factor is appropriately determined, the proposed method may be applied to many other domains.

## REFERENCES

1. R. J. Brachman, H. J. Levesque: Reading in Knowledge Representation, San Mateo, CA, *Morgan Kaufmann Publishers*, 1980.
2. E. Davis: Representation of Commonsense Knowledge, San Mateo, CA., *Morgan Kaufmann Publishers*, 1990.
3. S. Hasegawa, T. Yamazumi, and S. Koike: Imaging and understanding on control structure in programming education. *Journal of Information Processing Society of Japan* 39(4): 1180-1183, 1998.
4. N. Iwane, A. Takeuchi, and S. Otsuki: A network type intelligent educational system for arithmetic word

problems, *the Transactions of the Institute of Electronics, Information and Communication Engineers* J80-D-II(4): 915-924, 1997.

5. H.Kudo, Y.Sugiyama, M.Fujii, and K.Torii: Quantifying a design process based on experiments, *The journal of system and software* 9, pp.129-136, 1989.
6. T. Kuwabara: Relationship between complexity of information and difficulty of example oriented textbook. *the Transactions of the Institute of Electronics, Information and Communication Engineers* J80-D-II(11): 3039-3047, 1997.
7. N Matsuda, A. Kashihara, T. Hirashima, and J. Toyoda: A tutoring for behavior-based recursive programming, *the Transactions of the Institute of Electronics, Information and Communication Engineers* J80-D-II(1):326-335, 1997.
8. K. Muto: DOKUSYU-Java, SHOEISHA Co., Ltd., pp. 1-76, 1999.
9. S. Nakayama: Java Programming TETTEI-ENSYU, NIKKAN KOGYO SHIMBUN,Ltd, pp.1-184, 1998.
10. Y. Ootani, K. Muto: HAZIMETE-NO-Java, Gijutsu-Hyohron Co., Ltd. , pp.1-131, 1996.
11. R.C.Schank: Conceptual Information Processing, *New York : North-Holland*, 1975.
12. E.Soloway and K.Ehrligh: Empirical Studies of Programming Knowledge, *IEEE Transactions on Software Engineering*, SE-10 (5), pp.595-609, 1984.
13. Sun Microsystems, Inc.: Java Programming KOUZA, ASCII Co., Ltd, pp.12-79, 1998.
14. Y. Takemura, K. Shima, K. Matsumoto, K. Inoue and K. Torii: Factor analysis of comprehension states in the learning phases of a programming language, *Proceedings*

*of the 6th Asia-Pacific Software Engineering Conference (APSEC'99)*, pp.136-143, Dec. 1999.

15. Y. Takemura: An Analyzing Method about the Visualization for Understanding Structure of Java Language, *Journal of Osaka University of Junior College*, No.26, pp.143-153, 2002.

16. A. W. Airasian and W. M. Bart: Ordering Theory: A new and useful measurement model, *Education Technology*, pp.56-60, 1973.