Yang, J. C. et al. (Eds.) (2018). Proceedings of the 26th International Conference on Computers in Education. Philippines: Asia-Pacific Society for Computers in Education

A Learning Support System for Understanding Pointers in C Language Based on Program Behavior Visualization

Satoru KOGURE[a*], Yun YE[a], Koichi YAMASHITA[b], Yasuhiro NOGUCHI[a], Tatsuhiro KONISHI[a] & Yukihiro ITOH[c]
[a]*Faculty of Informatics, Shizuoka University, Japan*
[b]*Faculty of Business Administration, Tokoha University, Japan*
[c]*Shizuoka University, Japan*
*kogure@inf.shizuoka.ac.jp

Abstract: Much research has been conducted on systems that facilitate learners' understanding of program behavior through program behavior visualization. In this study, we focus on program education by constructing a system to support the understanding of pointers in C language. The system requires the learner to answer questions about the assignment statement of the pointer. The learner responds by clicking on a conceptual view in which the behavior of the algorithm is visualized. We conducted an evaluation experiment using 50 first year students in engineering department as subjects. Our results indicate that the learning system was effective.

Keywords: Programming learning support system, C language, visualization, pointer

1. Introduction

It is highly important to visualize program behavior to promote learner understanding. However, it is often difficult for beginner programmers to accurately understand such behavior. Thus, there are many support tools for visualizing the behavior of program and algorithms alike (Rößling & Freisleben, 2002; Moreno et al., 2004; Yu et al., 2014; Kogure et al., 2014; Yamashita et al., 2016; Yamashita et al., 2017). C language beginners may have difficulty understanding pointers because program code and behavior in the target world do not sufficiently correspond. We thus developed TEDViT (Kogure et al., 2014; Yamashita et al., 2016; Yamashita et al., 2017) to enable learners to manipulate conceptual views to express their level of pointer understanding. Study participants ranged from beginners (i.e., those with no knowledge of pointers) to middle level learners (i.e., those with little pointer knowledge). The system proposed in this research facilitates learner understanding of program behavior when using pointers correctly in statement units. To work in the system, learners were required to answer questions about the meaning of each proposed statement according to the provided guidance. They also used GUI operations to determine how each statement would affect objects in the conceptual view. We provided this system to 50 beginner-level university students in an actual classroom setting. Our results suggested that the system is an effective way to learn C pointers.

2. TEDViT: Teacher Explaining Design Visualization Tool

In previous study, we constructed a program learning support system that visualizes program behavior in both implementation view (including memory image) and conceptual view (algorithmic appearance). TEDViT contains a feature through which the teacher can freely describe the visualization method of the conceptual view. First, the teacher prepares the correct C program and the visualization rule set. The prepared files are then input into the system and the system

reconstructs the program behavior using variable info and visualization info automatically generated.

3. System Requirement and Design Policy

In this study, we focused on C language pointer education. Especially, the support system assists the learner into understanding the assignment statement, including the pointer variable. We thus summarized the assignment statement variation including the pointer as Table 1.

Table 1

Assignment Statement Types

Left \ right	Primary	Address	Pointer	Pointer with offset	*Pointer	*Pointer with offset m
Primary	(A) a = t[i]	---	---	---	(B) a = *q	(C) a = *(q+m)
Pointer	---	(D) p = &t[i]	(E) p = q	(F) p = q+m	---	---
*Pointer	(G) *p = t[j]	---	---	---	(H) *p = *q	(I) *p= *(q+m)
*Pointer with offset n	(J) *(p+n) = s	---	---	---	(K) *(p+n) = *q	(L) *(p+n)=*(q+m)

We explain the exercise flow used to understand the C language pointer assignment statement we proposed in this research. There are two available paths according to learning proficiency. The learning path is different for intermediates and beginners. The system distinguishes between beginner and intermediate learners based on whether they can accurately answer 1.0 and 2.0 question. Intermediate and expert learners who correctly answer these questions are given the main path, while beginners who do not provide correct answers are given the detour path. In the 1.X and 2.X questions prepared for the beginner, the learner answers stepwise by fragmenting either the left or right-side expression. For example, if the right-side expression is *(p+3), the question is divided into three stages. In the first step, the system asks the learner “Which area does ‘p’ indicate?” (2.1 question). In the second step, the system asks the learner “Which area does ‘p+3’ indicate?” (2.2 question). In the third step, the system asks the learner whether “Is ‘*(p + 3)’ an address or a value?” (2.3 question). In the fourth and final step, the system asks the learner “What is the value of ‘*(p + 3)’?” (2.4 question). The learner responds to these questions by clicking on a value assigned to the variable or a box assigned to the array or pointer variable in the conceptual view.

This exercise system contains two features. The first is ``Program behavior visualization function.'' This exercise system can automatically visualize behavior on conceptual view by pointer assignment statement by utilizing the function of visualization of program behavior of conventional TEDViT. The second is ``Dynamic learning path function according to learner proficiency.'' The beginner uses this function to incrementally understand the statement starting with the smallest fragment.

4. Experimental Evaluation

The evaluation experiment was conducted to determine whether the system was effective for pointer learning. To confirm this effectiveness, we constructed the two hypotheses. The first hypothesis is that *this system can improve pointer comprehension for learners (this evaluates the learning effect)*: (Hypohesis 1). The second hypothesis is that *this system is sufficient for learning pointers (this evaluates system interface usability)* :(Hypohesis 2).

Study subjects consisted of first-year university students belonging to engineering departments (i.e., 50 first-year engineering university students taking “programming foundation” subjects). Because of class experiments, we did not compare experimental and control groups. Learners used our system on Firefox in Ubuntu 16.04 OS running in Virtual Box for normal exercises. The exercise consisted of a pre-test, system use, post-test, and questionnaire. In the pre and post-tests, we provided a program involving the assignment statement of the pointer to learners, who then answered questions about the program’s execution result. We let students answer questions about the execution results of three programs during these tests.

We obtained a pretest average of 31.0 points, and a posttest average of 57.2 points (possible scores ranged from 0.0 to 90.0). We also examined self-evaluation scores regarding pointers in both the pre and post-tests. We obtained a pre-test self-evaluation average of 2.37 points and a post-test self-evaluation average of 2.94 (possible scores ranged from 1.0 to 4.0). These favorable results suggested that hypothesis 1 was correct.

We then provided learners with a questionnaire regarding their overall impressions of the system, the effectiveness of the visualization function, and whether gradual learning was effective for beginners. We obtained 3.18 points regarding overall impressions of the system, 3.47 points regarding the effectiveness of the visualization function, and 3.27 points regarding whether gradual learning was effective for beginners (possible scores ranged from 1.0 to 4.0). Finally, learners were asked whether the system or a textbook was more useful for pointer learning. A total of 36 students answered that the system was more useful, while 11 students answered that the methods were similarly effective, and two students answered that a textbook is more useful (Figure 6). These results suggest that hypothesis 2 was correct.

5. Conclusion

This study constructed a system to help learners understand pointers in C language. The system asked learners questions about pointer assignment statements. Learners were then required to respond to these questions by clicking on the conceptual view in which the algorithmic behavior was visualized. We then conducted an evaluation experiment with 50 first-year university students in engineering departments. Our results indicating that the system was effective for learning. We will consider using the functions and structures of pointers in future research.

Acknowledgements

This study was supported by JSPS KAKENHI Grant Numbers JP16K01084 and, JP18K11567.

References

Kogure, S., Fujioka, R., Noguchi, Y., Yamashita, K., Konishi, T., & Itoh, Y. (2014). Code reading environment by visualizing both variable’s memory image and target world’s status. *Proceedings of ICCE2014*, 343-348.

Moreno, A., Myller, N., & Sutinen, E. (2004). Visualizing programs with jeliot3. *AVI04: Proceedings of the Working Conference on Advanced Visual Interfaces*, 373-376.

Rößling,, G., & Freisleben, B. (2002). ANIMAL: A System for Supporting Multiple Roles in Algorithm Animation. *Journal of Visual Languages & Computing*, *13*(3), 341-354.

Yamashita, K., Fujioka, R., Kogure, S. Noguchi, Y., Konishi, T., & Itoh, Y. (2016). Practices of algorithm education based on discovery learning using a program visualization system. *Research and Practice in Technology Enhanced Learning (RPTEL), 11*(15), 1-19.

Yamashita, K., Fujioka, R., Kogure, S., Noguchi, Y., Konishi, T., & Itoh, Y. (2017). Classroom Practice for understanding pointers using learning support system for visualizing memory image and target domain world. *Research and Practice in Technology Enhanced Learning (RPTEL), 12*(17), 1-16.

Yu, Y., Nakano, H., Hara, K., Suga, S., & Aiguo, H. (2014). A C programming learning support system and its subjective assessment. *Proceedings of International Conference on Computer and Information Technology*, 561-566.