Available online at www.sciencedirect.com
ScienceDirect
ELSEVIER
Electronic Notes in Theoretical Computer Science 178 (2007) 61–68
Electronic Notes in Theoretical Computer Science
www.elsevier.com/locate/entcs

Inductive Reasoning and Programming Visualization, an Experiment Proposal

Andrés Moreno[2] Niko Myller[1,2] Erkki Sutinen[1,2]

Department of Computer Science
University of Joensuu
Joensuu, Finland

Taiyu Lin[3] Kinshuk[4]

Advanced Learning Technologies Research Centre
Department of Information Systems
College of Business
Massey University
Palmerston North, New Zealand

Abstract

We lay down plans to study how Inductive Reasoning Ability (IRA) affects the analyzing and understanding of Program Visualization (PV) systems. Current PV systems do not take into account the abilities of the user but show always the same visualization independently of the changing knowledge or abilities of the student. Thus, we propose IRA as an important skill when comprehending animation, which can be used to model the students and thus to adapt the visualization for different students. As an initial step we plan to check if IRA correlates with ability to answer program related questions during program visualization. We discuss the possible benefits of using IRA modeling in adaptive PV.

Keywords: Program visualization, Inductive reasoning, User modelling, Adaptive systems.

1 Introduction

Inductive reasoning is one of the important characteristics of human intelligence: Thurston regarded inductive reasoning as one of the seven primary mental abilities [23] that are accounted for intelligent behaviors. Merriam-Webster on-line dictionary defined induction as an "inference of a generalized conclusion from particular

[1] Part of this work was carried out during authors' visit to Massey University, Palmerston North, New Zealand.

[2] Email: firstname.lastname@cs.joensuu.fi

[3] Email: t.lin@massey.ac.nz

[4] Email: kinshuk@ieee.org

1571-0661/$ – see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.entcs.2007.01.032

instances". Pellegrino and Glaser [19] noted that the induction reasoning ability can be extracted in most aptitude and intelligent tests and is the best predictor for academic performance. Harverty et al. [7, p. 250] cited several other researches that viewed inductive reasoning as an significant factors for problem solving, concept learning, mathematic learning, and development of expertise, and Heller et al. [8] research showed that inductive reasoning is necessary ability for extracting the knowledge of problem solving in physics.

Despite its recognized importance underlying the learning process, little effort is spent on research to support the inductive reasoning process taking place in virtual learning environments [13]. This paper describes a project to incorporate modeling and support of learner's inductive reasoning ability in Jeliot - a program visualization system to help learning Java. If an inductive reasoning ability proves to be a good indicator of programming learning, program visualization tools could adapt their contents according to the students inductive reasoning ability. The paper is a continuation of the road map laid down in [2] and can be seen as an initial step towards an adaptive or smart program visualization system.

This paper starts with an introduction to the research done in inductive reasoning ability (Section 2. Following in Section 3 there is a description of Jeliot 3 and the modifications done to support the experiment described in Section 4. We end the paper with a discussion of the possible implications of the findings we may come across.

2 Inductive Reasoning Ability

The term induction is derived from the Latin rendering of Aristotle's epagoge that is the process for moving to a generalization from its specific instances [20]. Bransford et al. [4] pointed out that generalizations aimed at increasing transferability (ability to apply to a different context) can result in (mathematical) models, or global hypothesis in terms of Harverty et al. [7]. These can be later applied to a variety of contexts in an efficient manner.

Zhu and Simon [25] pointed out that the learners have to induce how and when to apply the problem solving method in worked-out examples. In reality, the learners also need to induce where to apply with a contextual awareness. The induction here requires the learners to (1) recognize the similarities and differences of the parameters in the current and the experienced contexts, (2) recognize and match pattern of current context to the experienced context(s), and/or (3) recognize/create the hypothesis/method that can be applied to solve the problem.

Point (1) requires information filtering, encoding and classification. In the early stage of an inductive reasoning task, one needs to see what attributes are relevant to the task at hand. Selected attributes are then encoded into meaningful chunks so that classifications or categories can be created. It is an iterative process to proceed from the selection of attributes to the forming of categories. In situations where a classification is sought, e.g., when solving a well-defined problem, the meaningfulness of classification is already a sufficient condition for one to stop this iteration, whereas if one is not confident with what the classifications should be, e.g. exploration of a novel domain, one may need to extend this iterative process to include hypothesis testing.

The pattern finding in point (2) is detection of co-variation from a stack of samples or past experiences as explained by Holland et al. [9]. In the research done by Heller et al. [8], the instructors believed that the attainment of problem solving skill in physics comes from reflective practices to extract knowledge from previous experiences of working on other problems or from sample problem solutions. Point (3) corresponds to the hypothesis generation activity in categorization of Harverty et al. [7]. Hypothesis generation differs from mere guessing in an important ways - having a rationale behind the hypothesis. The rationale of the hypothesis is primarily derived from the observed pattern. Use of analogy also plays an important role for inductive reasoning.

3 Jeliot 3

Jeliot 3 [15] is a program visualization system that animates the execution of Java programs. It has been used as an lecturing aid by teachers and a learning aid by students. Jeliot is currently used in classrooms [3] and in distance education [10]. Different problems have been identified in these contexts.

In classrooms, the mediocre students seem to benefit from the utilization of the program visualization tool the most. However, for the weakest and strongest students, Jeliot 3 in its current form might not be that helpful and useful. On the one hand, the visualization in Jeliot is still too complex and hard to grasp for the weakest students. On the other hand, the strongest students do not need the visualization as much and would like to use it only as needed or their learning might be even harmed because of this.

In distance education, students are studying alone and can have problems with learning and motivation. For visualization tools, this means that the use of the system should be both motivating and give students support in order to overcome barriers or even become a partial replacement of a teacher. As there is no teacher explaining the program visualization and the related programming concepts for the students, the tool should assume this role and provide explanations and hints to students as needed.

In order to know how to support students during the learning process on individual basis, we plan to use adaptation based on both students performance and cognitive traits.

3.1 Question generation in Jeliot

Students can be engaged and motivated to explore and make sense of it by asking prediction questions during the visualization [5,17]. Furthermore, interaction and question answering during multimedia learning have a positive influence on problem-solving ability in the domain [6].

We have implemented an automatic prediction question generation facility into

Jeliot 3 [16]. Question generation is achieved by analyzing the program trace (MCode) before it is visualized because the correct answer is also available. During the visualization the question is popped up before the evaluation of the selected expression has started. The question display and processing is based on the work of Rößling and Häussge [22], namely *AVInteraction*. Students' answers are saved into a database together with the information related to the question such as related programming concepts. We aim to use the information collected about student's performance to analyze the skill development in different programming concepts and model student's cognitive traits in order to adapt the visualization to the abilities of the student.

3.2 Modeling the student in Jeliot

Student modeling provides the means to track the students' activities while using a learning tool and analyze students behavior and possible abilities and known concepts. Our devised system will contain both a domain model, and a student model. The domain model will contain a reduced set of Java concepts derived from the Java Learning Object Ontology (JLOO) [12]. This set of concepts will still retain the "pre-requisite" relationships amongst concepts suggested in the JLOO.

To record the student's performance, the system will use an overlay user model: the model will consist of pairs containing a concept from the domain model and the student's ability to understand it. Abilities will be measured as the number of times the student has been asked about a certain concept, and number of correct and incorrect answers.

In the system, when a student answers correctly two or more questions related to the same Java concept, the user model will mark that concept as understood. We believe that already the second correct answer will minimize the "chance factor", nevertheless it does not necessarily mean that the concept is fully understood. Only concepts that have their "pre-requisite" concepts understood will be available for the system to generate questions.

4 Experiment Design

In this section, we propose an experiment to investigate the relationship between the students' IRA determined with psychometric test and their learning performance while using the previously described program visualization environment, Jeliot 3.

The students of a Programming I course in ViSCoS project, a 2 year-long web-based Computer Science study program, will be the experiment participants. At the beginning of the course students will complete Web-IRA, a task to analyze the IRA of the students. Web-IRA consists of questions which are divided into three categories: (1) series extrapolation where students are asked to find out what is the next item in a series - testing the ability to generate hypothesis, perform comparison [1]; (2) analogical reasoning where students are often given a pair of related items and are asked to use this as an analogy to find another pair of related items - testing the ability to filter attribute relevance and map relevant attribute from one context to another [24], and (3) exclusion where the students are asked to detect irregularity among the items - testing the ability to classify and test hypothesis [11].

The programming course is divided in weekly lessons, and concepts are distributed across them. For the experiment purposes, user model concepts will be also tied with the week they are explained, i.e. before a concept is learned students are asked to answer questions related to the concept (pre-test) and after the students have learned about the concept they will be presented with questions related to the same concept (post-test).

On week 2, they will be introduced to Jeliot 3 and the experiment. Each students will use a web start version of Jeliot, which will connect to an on-line version of the student model. Jeliot 3 will be updating the user model as the questions are answered.

Following weeks will introduce concepts at a fast pace. Students will be given a brief introduction to them and asked to visualize the program using Jeliot 3 and answer the questions. To motivate students to perform as well as they can points are granted based on the correct answers. As said before, only a reduced set of concepts will be used in the domain model. For the experiment, we will have two or three concepts that will make Jeliot generate questions about them. For example, in week 4 *Logical Statements* and *Comparison Operations* will be the chosen concepts.

At the end of the course, the Web-IRA could be completed again by the students, to test the reliability of the test itself.

The experiment is designed in a way that the student's behavior can be analyzed in terms of inductive reasoning ability. However, two major sources of bias can be predicted: student's prior knowledge and the case when a student does not make mistakes. If the student knows a concept and commits no mistake we can not analyze the effort required to induce/learn the concept. During the analysis, the first bias can be removed statistically by data (grade) from other courses students had taken. The analysis will not include data gathered from student who made no mistake.

After removing the potential biases, the following procedure will be carried out in the analysis:

(i) Data gathered in Web-IRA will be standardized; and student's data gathered in Jeliot 3 will be matched with information from data gathered in Web-IRA.

(ii) The standardized Web-IRA will allow separation of students into groups, for example, groups of low, medium, or high IRA.

(iii) Pattern of behaviors in different groups can be searched by either statistical mean (average number of mistakes), decision-tree based classification algorithm such as ID-3 [21], or neural network.

5 Discussion and Future Work

It can be argued that *deductive reasoning* plays a more important role when learning programing than inductive reasoning. In programming lectures, teachers provide students with a set of rules about what programming statements mean and their effects in the execution of a program. This set of rules can be seen as the special structure for deductive reasoning on how to program. However, learners find problems to match these rules without any previous knowledge. This fact can hinder learning as it produces *belief bias*, i.e., they are more likely to consider an argument as false when they are unable to relate it to their previous knowledge. As students test the examples and program the exercises, the special structure starts to consolidate. We argue that it is the inductive reasoning that produces such consolidation.

Co-relating IRA and programming visualization comprehension is just another step in acknowledging the importance of cognitive skills in programming. Pea and Kurland [18] summarized the work on programming aptitude tests and pointed how inductive skill was a good indicator of future "programming success (p. 45). If the link between IRA and programming abilities can be determined, it could have several implications in the Computer Science field. One of the possible implications of the result is that Web-IRA could be used to predict the success for students taking programming courses. Moreover, it could be interesting to further study how much Jeliot 3 could help those students with a low IRA.

But the results could imply as well that if reasoning patterns can be found in Jeliot 3, they can be built into the pedagogical module of Jeliot 3 which would then be able to classify students into groups (low, medium, or high IRA). It entails that Jeliot 3 alone will be able to model students' IRA without the need of data from Web-IRA.

5.1 Personalization of Program Visualization tools

The experiment described here is part of a bigger effort to make program visualization tools aware of personal differences. A investigation done on Jeliot 3 [14] showed that students could not always fully understand the animations. To help students understanding the animations and the concepts behind, it was suggested to add explanations and questions to the animation. This will require a proper modeling of the student's ability and previous knowledge. Otherwise, the expanded animation will not cover the needs of a specific student. For this experiment, we will implement the student model that will gather the information about a student's knowledge, and decide which questions are more suitable for the students. A further iteration in the development of the tool will add textual explanations of the animations and the programming concepts involved in them.

If a link between IRA and program visualization comprehension were to be found, animations, questions and explanations could be tailored to the student's IRA. It is hypothesized that students with lower IRA will benefit of further explanations or hints. For example, animations of certain concepts could be ignored after a number of times the student has seen it. The number of repetitions should be linked to the student's IRA; a student with high IRA should not require the same information repeated as many times as a student with lower IRA. However, this will require further experimentation to fully prove that hypothesis.

References

[1] Bailenson, J. N., M. S. Shum, S. Atran, D. L. Medin and J. D. Coley, *A bird's eye view: biological categorization and reasoning within and across cultures*, Cognition **84** (2002), pp. 1–53.

[2] Bednarik, R., A. Moreno, N. Myller and E. Sutinen, *Smart program visualization technologies: Planning a next step*, in: *Proceedings of the 5th IEEE International Conference on Advanced Learning Technologies*, Kaohsiung, Taiwan, 2005, pp. 717–721.

[3] Ben-Bassat Levy, R., M. Ben-Ari and P. A. Uronen, *The Jeliot 2000 program animation system*, Computers & Education **40** (2003), pp. 1–15.

[4] Bransford, J. D., A. L. Brown, R. R. Cocking, M. S. Donovan and J. W. Pellegrino, "How people learn: Brain, mind, experience, and school," National Academy Press, Washington, US, 2000.

[5] Byrne, M. D., R. Catrambone and J. T. Stasko, *Evaluating animations as student aids in learning computer algorithms*, Computers & Education **33** (1999), pp. 253–278.

[6] Evans, C. and N. J. Gibbons, *The interactivity effect in multimedia learning* (2006), accepted to Computers & Education.

[7] Harverty, L. A., K. R. Koedinger, D. Klahr and M. W. Alibali, *Solving inductive reasoning problems in mathematics: No-so-trivial pursuit*, Cognitive Science **24** (2000), pp. 249–298.

[8] Heller, P., K. Heller, C. Henderson, V. H. Kuo and E. Yerushalmi, *Instructors' Beliefs and Values about Learning Problem Solving* (2001), www-page, from http://groups.physics.umn.edu/physed/Talks/Heller%20PERC01.pdf (Retrieved 15 Oct, 2003).

[9] Holland, J. H., K. J. Holyoak, R. E. Nisbett and P. R. Thagard, "Induction: Processes of inference, learning, and discovery," The MIT Press, London, UK, 1987.

[10] Kannusmäki, O., A. Moreno, N. Myller and E. Sutinen, *What a novice wants: Students using program visualization in distance programming course*, in: A. Korhonen, editor, *Proceedings of the Third Program Visualization Workshop (PVW 2004) Research Report CS-RR-407* (2004), pp. 126–133.

[11] Laumann, L. L., "Adult age differences in vocabulary acquisition as a function of individual differences in working memory and prior knowledge," Ph.D. thesis, West Virginia University (1999).

[12] Lee, M.-C., D. Y. Ye and T. I. Wang, *Java learning object ontology*, in: *Proceedings of the 5th IEEE International Conference on Advanced Learning Technologies*, Kaohsiung, Taiwan, 2005, pp. 538–542.

[13] Lin, T., Kinshuk and A. Patel, *Cognitive trait model for persistent student modelling*, in: D. Lassner and C. McNaught, editors, *Proceedings of EdMedia 2003* (2003), pp. 2144–2147.

[14] Moreno, A. and M. Joy, *Jeliot 3 in a Demanding Educational Setting*, in: *Proceedings of the Fourth International Program Visualization Workshop*, Florence, Italy, 2006, pp. 48–53.

[15] Moreno, A., N. Myller, E. Sutinen and M. Ben-Ari, *Visualizing Program with Jeliot 3*, in: *Proceedings of the International Working Conference on Advanced Visual Interfaces, AVI 2004*, Gallipoli (Lecce), Italy, 2004, pp. 373–380.

[16] Myller, N., *Automatic Prediction Question Generation during Program Visualization*, in: *Proceedings of the Fourth International Program Visualization Workshop*, Florence, Italy, 2006, pp. 89–93.

[17] Naps, T. L., *JHAVÉ – Addressing the Need to Support Algorithm Visualization with Tools for Active Engagement*, IEEE Computer Graphics and Applications **25** (2005), pp. 49–55.

[18] Pea, R. J. and D. M. Kurland, *On the cognitive prerequisites of learning computer programming on the cognitive prerequisites of learning computer programming on the cognitive prerequisites of on the cognitive prerequisites of learning computer programmin*, Technical Report 18, Bank Street College, Center for Children and Technology (1983).

[19] Pellegrino, J. W. and R. Glaser, *Analyzing aptitudes for learning: Inductive reasoning*, in: R. Glaser, editor, *Advances in instructional psychology (Vol. 2)*, Lawrence Erlbaum Associates, Hillsdale, NJ, 1982 .

[20] Rescher, N., "Induction," University of Pittsburgh Press, Philadelphia, US, 1980.

[21] Ross, P., *Rule induction: Ross quinlan's id3 algorithm* (2000), www-page, from http://www.dcs.napier.ac.uk/~peter/vldb/dm/node11.html (Retrieved 23 Oct, 2003).

[22] Rößling, G. and G. Häussge, *Towards Tool-Independent Interaction Support*, in: A. Korhonen, editor, *Proceedings of the Third International Program Visualization Workshop, Warwick, England*, 2004, pp. 110–117.

[23] Selst, M. V., *Intelligence* (2003), www-page, from http://www.psych.sjsu.edu/~mvselst/courses/psyc235/lecture/chapter14intelligence.pdf (Retrieved 14 Oct, 2003).

[24] Sternberg, R. J., “Intelligence, information processing, and analogical reasoning: the componential analysis of human abilities,” Wiley, New York, 1977.

[25] Zhu, X. and H. A. Simon, *Learning mathematics from examples and by doing*, Cognition & Instruction **4** (1987), pp. 137–166.