Computers in Human Behavior 30 (2014) 286–298

Contents lists available at ScienceDirect

Computers in Human Behavior

journal homepage: www.elsevier.com/locate/comphumbeh

Learning programming via worked-examples: Relation of learning styles to cognitive load

Siti-Soraya Abdul-Rahman [a,*], Benedict du Boulay [b,1]

[a] Department of Artificial Intelligence, Faculty of Computer Science and Information Technology, University of Malaya, Lembah Pantai, 50603 Kuala Lumpur, Malaysia
[b] Department of Informatics, University of Sussex, Falmer, Brighton, East Sussex BN1 9QJ, United Kingdom

ARTICLE INFO

Article history:
Available online 10 October 2013

Keywords:
Cognitive load
Learning styles
Worked-example strategies
Programming

ABSTRACT

This paper describes an experiment that compared learners with contrasting learning styles, Active *vs.* Reflective, using three different strategies for learning programming via worked-examples: Paired-method, Structure-emphasising, and Completion. The quality of the learners' acquired cognitive schemata was assessed in terms of their post-test performance. The experiment investigated variations in learners' cognitive load, taking both the learning strategies and the learners' learning styles into account. Overall, the results of the experiment were inconsistent. In comparing the effects of the strategies during the learning phase, the study found significant differences in cognitive load. Unexpectedly, no differences were then detected either in cognitive load or in performance during the post-test (post-test). In comparing the effects of the learning styles during the learning phase and the transfer phase, medium effect sizes suggested that learning style may have had an effect on cognitive load. However, no significant difference was observed in performance during the post-test.

1. Introduction

An important goal of Instructional Design is to devise materials for, and methods of, instruction that maximise the chance that learning is both straightforward for the learner and effective in its outcome. By "straightforward for the learner", we mean that as far as possible the complexity of what is to be learned is kept manageable so that the effort required by the learner is minimised. By "effective in its outcome" we mean that the learner develops the schemata to structure and encode what is learned so that when faced with future problems they are able both to deal with them correctly and with minimal effort. Many researchers have investigated these issues under the general heading of Cognitive Teaching Models (Wilson & Cole, 1996). Two particular areas of work involve the exploration of worked-examples in addition to solving problems as learning materials, and the development of strategies to maximise schema acquisition. This paper explores these two issues in relation to the learning of programming. Having students learn from worked-examples is an important pedagogic strategy to aid the acquisition and management of the cognitive skills and schemata underpinning problem-solving ability in general, and programming in particular (see e.g. Chi, Bassok, Lewis, Reimann, & Glaser, 1989; Pirolli & Anderson, 1985; Atkinson, Derry, Renkl, & Wortham, 2000; van Merriënboer & Paas, 1990). Various methods have been proposed to assist schema construction when learning from programming worked-examples. The first is the Completion Strategy where learners gradually build up a schema by completing partially presented worked-examples (van Merriënboer & de Croock, 1992). The second is the "Structure-emphasising Strategy", to use Quilici and Mayer's phrase (2002), in which learners are supported in identifying higher level schemata such as plans (e.g. Davidovic, Warrant, & Trichina, 2003). Within the programming education literature, programming plans have been regarded as one form of schema (Soloway & Ehrlich, 1984). Programming plans are generic program fragments that represent "stereotypic actions in a program" (Ehrlich & Soloway, 1984, p. 115). A plan is a chunk of programming knowledge which can be retrieved and applied in future problem solving (Rist, 1989). In summary, two good methods of getting learners to understand how programs work are by getting them to complete partially finished solutions, and by getting them to appreciate the higher level structure that holds a program together.

Cognitive load theory (CLT) has provided guidelines for the development of several instructional formats, including worked-examples (Kirschner, 2002). CLT implies that instructional design should take account of the learner's limited working memory capacity to maximise the chance of effective schema construction (Sweller, van Merriënboer, & Paas, 1998). During learning, many cognitive processes are restricted by working memory capacity.

* Corresponding author. Tel.: +60 3 7967 6333; fax: +60 3 7957 9249.
E-mail addresses: siti_soraya@um.edu.my (S.-S. Abdul-Rahman), B.du-Boulay@sussex.ac.uk (B. du Boulay).
[1] Tel.: +44 01273 678324.

0747-5632/$ - see front matter
http://dx.doi.org/10.1016/j.chb.2013.09.007

The basic tenet of the CLT is to optimise such loads so that greater cognitive resources are available for actual learning to take place. In other words, the freed resources can, in principle, be directed to the learning activities that are relevant to the process of schema construction. CLT defines three different types of cognitive load, namely intrinsic, extraneous, and germane load. Intrinsic load is generated by the inherent complexity of the material being learned in terms of the numbers of its elements and their interactions. Extraneous load is generated by the form and means through which the material is experienced. For example, easy material can be presented in a complex way and vice versa. According to Sweller (2010), germane load is of a different category and so differs from both intrinsic and extraneous load. Whereas the emphasis of the latter two loads (intrinsic and extraneous) is heavily dependent on the characteristics of the instructional material, germane load, is concerned with the working memory resources that the learner chooses to devote to deal with the element interactivity of the task at hand. To that extent intrinsic load is about the properties of the material to be learned whereas germane load is about the learner's reaction to that material. In summary, CLT would argue for the following three principles. First, find ways to manage the inherent complexity of what is being learned, e.g. by appropriate methods of sequencing the material. Second, make sure that further barriers are not placed in the way of learning e.g. by simplifying the way that the material is presented. Third, assist and encourage learners to devote as many of their cognitive resources to the learning task in hand e.g. by removing distractions.

The instructional effectiveness of different worked-example designs that build on CLT has been widely researched. It is worth noting, however, that the majority of these studies have failed to provide consistent findings for the instructional effectiveness with regard to learning outcomes and cognitive load effects (Moreno, 2006; Paas & van Gog, 2006).

In addition to working memory capacity as a factor influencing learning, there is the issue of learning style. Various different learning styles have been identified, see (Felder & Silverman, 1988) for a review. Here we are interested in the active *vs.* reflective dimension. Active students have a tendency to learn by trying things out and experimenting, while reflective students have a tendency to learn by thinking things through (Felder & Silverman, 1988). Some students are "balanced" and can deploy either mode depending on the circumstances. From their review of the literature, Graf, Lin, and Kinshuk (2008) identified an *indirect relationship* between working memory capacity and learning style. That is to say, learners with low working memory capacity tend to prefer an active style of learning, whereas learners with high working memory capacity tend to prefer a reflective style of learning. However, in a recent study conducted by Graf, Liu, Kinshuk, Chen, and Liang (2009), they found another significant relationship. This was that learners with a strong preference for either an active or a reflective learning style on its own tend to have low working memory capacity whereas the more balanced the learning style is, the higher the working memory capacity tends to be. Thus, the relationship between working memory capacity and learning style needs further elucidation.

In conclusion, we argue that investigating the relationship between learning styles and teaching methods, and so the differential effects of cognitive load on active and reflective learners, is a promising line of investigation and that is the main focus of this paper.

The paper is organised as follows. The next subsection describes the goals of the work and Section 2 sets out the method employed in a study, including its design, materials, and methods of analysis. Section 3 provides the results, initially comparing the effects of the strategies and then the effects of the learning styles. Finally Sections 5 and 6 provide a discussion and conclusions.

1.1. Goals of the work

This research set out to explore strategies for learning programming via worked-examples aiming to promote schema acquisition and transfer. Learning style is a factor that influences how much effort learners will typically expend on understanding worked-examples. In particular, active learners tend to be more impulsive than reflective learners, and as a consequence of their reduced effort and may find themselves overwhelmed by detail. In view of that, we hypothesised that the two learning styles might interact with learners' cognitive load and would determine the quality of the acquired cognitive schemata and hence the transfer of learning. To answer this question, we investigated the differential effects of different worked-example strategies on cognitive load (i.e. extraneous and germane load), including learning efficiency measures, as well as post-test performance, taking into account learners' learning styles. This paper is based on the first author's thesis work (Abdul-Rahman, 2012). The ideal situation would therefore be a teaching method where (i) active and reflective learners would experience similar degrees of difficulty while using it despite the tendencies of active learners not to reflect on what they are doing, (ii) the quality of what is learned as measured by post-tests would also be similar, and (iii) the effort experienced by active and reflective learners while undertaking the post-test would be similar, thus suggesting the acquired schemata were equally effective.

2. Method

2.1. Sample and design

The experiment involved 117 participants from a Malaysian university all taking the same course, but involving a mix of nationalities: local students as well as international students. The majority of the participants were first year undergraduates taking a Programming course, using English as the medium of instruction, as part of their degree work. A small number of students (34 of them) repeating the course were also participants. The experiment was conducted as part of the course topic on loops and the transfer test was administered as the students' mid semester test. For the experimental part of the course, the students interacted with a system called LECSES, described later.

Prior to group allocation, a pre-test was administered to assess the learners' level of knowledge of programming. Learners were allocated equally and pseudo-randomly (based the learner's learning style) to one of three strategy groups, namely the Structure-emphasising, the Completion, and the Paired-method groups. In other words, the allocation attempted as far as possible to have equal proportions of Active, Balanced and Reflective students in each group. The Paired-method strategy was a mixture of the other two strategies, and was designed to try to overcome the difficulties associated with the other two strategies (Abdul-Rahman, 2012). Out of the 117 learners who took part in the experiment, only the data from the 110 learners who completed all the tasks required for the main phases of the experiment were used for the analysis.

In the Structure-emphasising group (in the following labelled as the *SE condition*; $n = 37$), learners worked with LECSES in which they had to explain and reflect on the underlying structure of the worked-examples. In the Completion group (in the following labelled the *CS condition*; $n = 36$), learners worked with LECSES that presented worked-examples with partial program solutions that had to be completed and modified. In the Paired-method group (in the following labelled the *PM condition*; $n = 37$), learners received a combination of both the other two worked-example