Computers in Human Behavior 41 (2014) 478–485

Contents lists available at ScienceDirect

Computers in Human Behavior

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

Exploring students learning behavior with an interactive etextbook in computer science courses

Eric Fouh *, Daniel A. Breakiron, Sally Hamouda, Mohammed F. Farghally, Clifford A. Shaffer

Department of Computer Science, Virginia Tech, United States

ARTICLE INFO

Article history:
Available online 20 November 2014

Keywords:
eTextbook
Learning behavior
Mobile learning
Computing education

ABSTRACT

We present empirical findings from using an interactive electronic textbook (eTextbook) system named OpenDSA to teach sophomore- and junior-level Computer Science courses. The web-based eTextbook infrastructure allows us to collect large amounts of data that can provide detailed information about students' study behavior. In particular we were interested in seeing if the students will attempt to manipulate the electronic resources so as to receive credit without deeply going through the materials. We found that a majority of students do not read the text. On the other hand, we found evidence that students voluntarily complete additional exercises (after obtaining credit for completion) as a study aid prior to exams. We determined that visualization use was fairly high (even when credit for their completion was not offered). Skipping to the end of slideshows was more common when credit for their completion was offered, but also occurred when it was not. We measured the level of use of mobile devices for learning by CS students. Almost all students did not associate their mobile devices with studying. The only time they accessed OpenDSA from a mobile device was for a quick look up, and never for in depth study.

1. Introduction

Recent interest in MOOCs (Massive Open Online Courses) and the popularity of practice systems like Khan Academy and Code Academy is just the latest in a trend toward increased use of interactive online course materials at all levels of education. MOOCs especially drive a need for scalable methods of assessment that do not require direct involvement of limited teaching resources. While some MOOCs attempt to address the need for assessment through crowd sourcing (especially to students in the class), automated assessment of exercises provides an excellent way to scale up assessment when it is possible. For many traditional courses, a longstanding problem is lack of sufficient practice exercises with feedback to the student. Again, automated assessment provides a way to increase the number of exercises on which students can receive feedback.

Online tutorials and interactive exercise systems provide the opportunity to automatically log massive amounts of user interaction data at fine detail, to the level of individual mouse events. This wealth of information might be used in a number of ways to improve the pedagogical value of online materials. Developers can hope to learn from interactive log data simple things like what platforms or devices are most often used by students, which exercises are taking an unreasonable amount of time, or which ones appear too easy because students never get them wrong. But careful analysis of log data can hope to deduce more complex behavior. For example, by examining the times of various interactions, we should be able to determine whether students are reading the materials before attempting the associated exercises. We can hope to tell whether students are viewing all of the parts of a given visualization, or skipping through it. We should be able to tell whether students are going back to the materials to use them to study for a test by the fact that they look at them even after receiving credit for completing them.

The OpenDSA project (Fouh et al., 2014; Shaffer, Karavirta, & Naps, 2011; Shaffer, Naps, & Fouh, 2011) provides a collection of online, open-source tutorials for Data Structures and Algorithms (DSA) courses. They combine textbook-quality text with algorithm visualizations (AV) and randomly generated instances of interactive examples and exercises to provide students with unlimited practice. OpenDSA collects log data for all user interactions occurring on an OpenDSA web page. Between Spring 2013 and Spring 2014, we collected and analyzed usage data from eight courses totaling about 700 students. We present an analysis that demonstrates how log data might be used to understand how students use online tutorial systems.

* Corresponding author at: 2000A Torgersen Hall, Department of Computer Science, Virginia Tech, Blacksburg, VA 24061, United States. Tel.: +1 (540) 231 4354.
E-mail addresses: efouh@vt.edu (E. Fouh), breakid@vt.edu (D.A. Breakiron), sallyh84@vt.edu (S. Hamouda), mfseddik@vt.edu (M.F. Farghally), shaffer@cs.vt.edu (C.A. Shaffer).

http://dx.doi.org/10.1016/j.chb.2014.09.061
0747-5632/

2. Materials and methods

The basic functional unit for OpenDSA materials is a module, which represents a single topic or part of a typical lecture, such as a single sorting algorithm. Each module is a complete unit of instruction and typically contains AVs, interactive assessment activities with automated feedback, and textbook quality text. Modules can be grouped together into chapters, such as might be found in traditional paper books. OpenDSA content is built using HTML5 and JavaScript, making it device and browser independent. AVs are built using the JavaScript Algorithm Visualization (JSAV) library (Karavirta & Shaffer, 2013). Many OpenDSA exercises are "algorithm simulations". These require that the student manipulate a data structure to show the changes that an algorithm would make on it, such as clicking to swap elements in an array or clicking on appropriate nodes in a tree or graph. The AVs and algorithm simulation exercises are specifically designed to be manipulated either through mouse and pointer interactions or touch interactions when using touchscreen devices. We generally refer to algorithm simulation exercises, as "proficiency exercises". This type of exercise were inspired and built in collaboration with the team that created the TRAKLA2 system (Malmi et al., 2004). We make use of the Khan Academy framework (http://github.com/Khan/khan-exercises) to provide support for multiple choice, T/F, and custom interactive exercises that we call "mini-proficiency" exercises. We will refer to these collectively as "KA exercises". All exercises are automatically graded and provide feedback to the user. Students can repeat exercises as many times as they want until they get credit, or even work them again after receiving credit as a study aid.

OpenDSA has been used to teach the second semester fundamental data structures and algorithms course (CS2), and also a more advanced data structures, algorithms, and analysis course (CS3) at a total of five higher education institutions in the US, Egypt, and Finland. Different instructors have used OpenDSA in different ways, but typically they used OpenDSA exercises for graded homework, and/or used AVs from OpenDSA as a lecture aid. During Spring 2013, OpenDSA was used in three course sections: two offerings of Virginia Tech's version of CS3 (we will refer to these as C1 and C2), and one offering of a CS3 course at Alexandria University in Egypt (which we will refer to as C3). OpenDSA was used as the primary resource to teach topics related to Hashing in C3, followed by a midterm that included questions on hashing. Course C1 used OpenDSA as the primary source of material for sorting and hashing (about three weeks of material). For this section, AVs (slideshows) were not given credit. There was a single assignment to complete the exercises for the two chapters, so all exercises were due at the same time. In contrast, course C2 used another textbook for the initial presentation of material on sorting. However, OpenDSA exercises were then assigned after the lecture period on sorting was complete. C2 used OpenDSA as the primary source for Hashing. C2 students received a small amount of credit for completing slideshows. We deliberately made this distinction in slideshow credit between the two sections in order to study how that affected student learning behavior, as described below.

We relied on the logged data to infer students' behavior. The OpenDSA system records about 200 different types of events. The events can be grouped into the following categories:

- Registration and login interactions (all actions such as student registration, logging into and out of the system).
- Static Content interactions (when a student loads a module page, follows a hyperlink, or navigates to another page using the navigation menu or the table of contents).
- Interactive Activities interactions (when the student clicks to advance a slideshow, clicks within an AV, etc.).
- Assessment Activities interactions (all interactions involving loading an exercise, submitting an answer, completing a step of a proficiency exercise, etc.); and
- Gradebook interactions (when students load the gradebook page to check their score).

All events are recorded with a timestamp. To assess the relationship between the use of OpenDSA and students performance, we used both log data and performance data (students' scores in written tests) from the Fall 2013 offering of a CS3 course at VT (which we refer to as C4). OpenDSA was used as the main course material, and the instructor regularly used the visualization in the classroom as lecture aide. The students performed regular mandatory OpenDSA homework and took two midterms and one final examination. OpenDSA-based homework accounted for 20% of the course final grade.

3. Student learning behavior

We sought to measure the prevalence of several behaviors that we collectively term as "credit-seeking". We believe this is different from (and possibly in conflict with) "learning" behavior. Examples of "credit-seeking" behavior include jumping straight to exercises without reading the associated text, clicking through slideshows (such as shown in Fig. 1) as quickly as possible, skipping to the end of slideshows to get credit, and using the results from AVs to complete exercises.

3.1. Not reading

We observed two general patterns of behavior related to reading the text. Ideally, students would read most or all of the text, completing exercises as they encounter them within the module. But students often skip directly to the exercises, only reading as required to get exercise credit. Note that proficiency and KA exercises are placed "behind a button" by default, in that they require that the student clicks on an interface button to reveal the exercise.

We measured the total time that a module was open before a button was clicked to reveal an exercise. We assumed that once a student clicked the button to reveal an exercise, they attempted

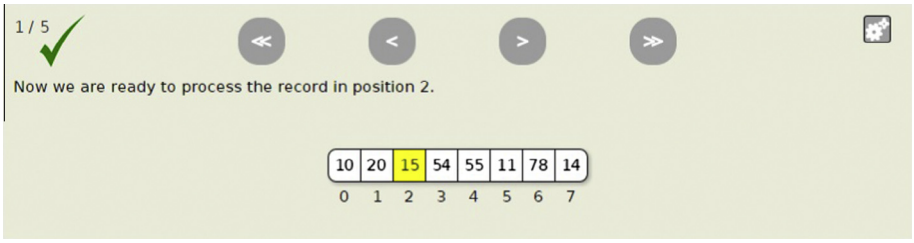

Fig. 1. Example of an OpenDSA slideshow. Standard controls allow the user to advance the slideshow by one slide, back up one slide, jump back to the beginning, or jump to the end.