pseuCo Book: An Interactive Learning Experience

Felix Freiberger
Saarland University
Computer Science – Saarland Informatics Campus
Saarbrücken, Germany

ABSTRACT

Software tools and apps are pervading educational technology in many areas, not least in the field of computer science education. However, many of these tools have a very narrow scope and lack context, or only work as add-ons to textbooks and other course materials. This paper advocates an alternative: a truly interactive textbook experience where interactive demonstrations and exercises are interwoven with more classical textual elements. We present a framework that provides authors with the infrastructure for writing such books, and present pseuCo Book, an interactive book for teaching concurrent programming. The effectiveness of this approach is demonstrated by a user study encompassing a detailed empirical evaluation.

CCS CONCEPTS

• **Social and professional topics** → **Computer science education**; • **Human-centered computing** → **User studies**; *Web-based interaction*; • **Applied computing** → **Computer-assisted instruction**; **Interactive learning environments**; **E-learning**; *Extensible Markup Language (XML)*; • **Theory of computation** → **Concurrency**.

KEYWORDS

computer science education, concurrent programming, concurrency theory, interactive textbook

ACM Reference Format:
Felix Freiberger. 2022. pseuCo Book: An Interactive Learning Experience. In *Proceedings of the 27th ACM Conference on Innovation and Technology in Computer Science Education Vol 1 (ITiCSE 2022), July 8–13, 2022, Dublin, Ireland*. ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3502718.3524801

1 INTRODUCTION

Since its early beginnings, computer science education has always been an arduous task, with a significant focus on not just conferring knowledge, but facilitating understanding. Particularly throughout the past decade, significant research and engineering effort has been invested into making this process not just easier, but more successful. In many cases, this work has resulted in the creation of tools that use the capabilities of modern technology to provide an interactive learning environment centered around a specific topic.

ITiCSE 2022, July 8–13, 2022, Dublin, Ireland

ACM ISBN 978-1-4503-9201-3/22/07.
https://doi.org/10.1145/3502718.3524801

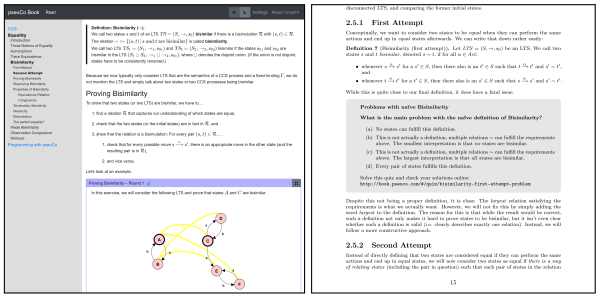
These tools take many different shapes. Some (e.g. [1]) focus on visualizing a particular algorithm, essentially serving as an interactive, highly visual replacement for what previously would have been one or two examples in a textbook. Others essentially provide an *integrated development environment* (IDE), an environment facilitating the development of programs by providing a comfortable editing experience and tools to help identify and eliminate errors, but are specifically adapted to teaching by using a programming language that is tailored to the topic in question (e.g. [2]), generally friendly towards beginners (e.g. *Snap!* [6]), or even adaptable to the user's programming skills [7]. Some include additional education features like intelligent tutoring through automatically provided hints [12] or encouragement [9]. There are also platforms that combine an IDE, documentation, and cloud-based, containerized testing environments.[1] Some tools extend the "IDE for education" concept beyond programming into areas like proof systems while avoiding the complexity associated with fully-featured proof assistants [3]. Another class of tools, like Automata Tutor [4, 5], provides exercises for students in conjunction with an environment where they can enter their solution and receive feedback from an autograder.

In most cases, these tools provide significant *depth*, giving students ample opportunity to practice a small set of exercise types or to deepen their understanding of a particular topic. On the other hand, these tools are often *narrow*, focusing on a single topic. Therefore, courses using these tools often also have large parts of their syllabus lacking such tool support, or need to switch tools mid-course. Also, these tools typically are *incomplete*, providing demonstrations or exercises to be used alongside a lecture or textbook, and depend on at least basic previous knowledge. This makes them unsuitable for standalone learning and can cause additional friction, e.g. when students need to consult a definition from the lecture which is located in separate lecture notes to solve an exercise.

The goal of this paper is to explore the possibilities of building an educational tool with less depth but more breadth, making it possible to overarch and support the entire syllabus of a computer science lecture and to fully replace lecture notes. We build one such tool, *pseuCo Book*, delivered in the form of an interactive book that integrates its interactive elements into static textbook content.

We apply pseuCo Book to the *Concurrent Programming* course at Saarland University. This course teaches significant parts of the *Parallel and Distributed Computing* knowledge area of CS2013 [8], educating students to safely use message-passing and shared-memory concurrency. The introduction to this practical part is done using pseuCo, a concurrency-focused academic programming language featuring a variety of teaching tools, including pseuCo.com [2], a web IDE for CCS and pseuCo. However, the lecture starts off with the knowledge unit "Formal Models and Semantics", teaching students how to write, analyze, and compare process models using

[1] e.g. Codio (https://www.codio.com/)



(a) Web app (b) PDF file

Figure 1: Screenshots of excerpts from the Bisimilarity section of pseuCo Book, showing the web version (a) and PDF version (b). The web version shows a definition, some static content, and an interactive exercise; the PDF version shows some static content and a quiz.

Milner's Calculus of Communicating Systems (CCS) [10, 11] and notions of equality like *trace equivalence*, *bisimilarity*, and *observation congruence*. Many students consider this to be the hardest part of the course, making it a fitting starting area for the development and deployment of pseuCo Book.

PseuCo Book was used in the 2021 edition of this course. To evaluate its effectiveness, we conducted a user study quantifying student perception of pseuCo Book by means of surveys. In addition, we measured student performance in the lecture's assignments and exams to empirically evaluate the impact of pseuCo Book.

Structure of This Paper. The remainder of this paper is structured as follows. Section 2 describes a general framework for interactive books. Section 3 describes pseuCo Book, a concurrency textbook built upon this framework. Section 4 describes supporting infrastructure for in-lecture use of pseuCo Book. The user study is described in Section 5. Finally, Section 6 concludes the paper.

2 HYBRID DOCUMENT FORMAT – INTERACTIVE TEXTBOOKS MADE EASY

To help authors develop interactive books, we have developed an XML-based markup language called *Hybrid Document Format (HDF)*.[2] It features text and lists with basic formatting and predefined formats for definitions, lemmas, and proofs, as well as structural elements like chapters and sections. It allows for footnotes, references, inline code and code listings, and LaTeX-formatted math. Both arbitrary PDF files and standalone TeX files can be included as images. Finally, it provides means to include interactive elements.

There are two kinds of interactive elements: Simple quizzes and hand-crafted interactive elements. Quizzes support both single and multiple choice questions and are fully defined within the HDF file, with full support for formatting, including math mode. Both types of interactive elements can optionally hide subsequent blocks of explanatory text behind a click-through spoiler warning.

As demonstrated in Figure 1, HDF books can be exported in two different formats: As an interactive web app or as a static PDF file.

[2] see https://depend.cs.uni-saarland.de/~freiberger/hdf/

To create the web app (cf. Figure 1a), the HDF file is parsed, auxiliary files are processed (e.g. included TeX images are compiled), and the result is exported as a set of JSON and SVG files that can be rendered by a custom reader web app written in React. It includes a dark mode and an automatically-enabled offline mode allowing reading anywhere after the initial download. By allocating URLs at paragraph granularity, students and instructors can easily share links to any point within pseuCo Book. These features are implemented in a framework that can be combined with the hand-crafted interactive elements and any other customizations (e.g. a landing page) to build a reading application for a specific HDF book. Aside from basic support for state persistency provided by the framework, the hand-crafted interactive elements are generic, arbitrary React components. The resulting web app is the main way for students to access the book and its interactive elements.

To create the PDF version, the HDF compiler outputs an intermediary LaTeX file that is then typeset. The result, shown in Figure 1b, contains all static content and quizzes. In this version, interactive demonstrations and exercises are substituted by a placeholder text and link to the corresponding element in the web version.

The HDF compiler and the reader web app framework are published under an open source license.

3 PSEUCO BOOK

3.1 A First Look at pseuCo Book

PseuCo Book is a work-in-progress interactive book intended to become a full replacement of lecture notes for a concurrency lecture, built using HDF. At the start of the 2021 edition of the Concurrent Programming lecture, a single chapter of pseuCo Book was available, introducing different notions of equality. While an additional chapter was made available during the lecture, this paper focuses on the equality chapter.

PseuCo Book consists of traditional textbook elements like introductory materials, definitions, explanations, and examples. In addition, it contains interactive elements that form an integral part of the content of pseuCo Book: They are placed throughout the sections, with the text after them typically assuming that students have solved (and understood) them. These interactive elements comprise quizzes (as provided by the hybrid document format) and two kinds of hand-crafted interactive elements:

Demonstrations are hand-crafted playgrounds designed to help students understand a particular detail of the presented material. An example is discussed later in Section 3.2.

Exercises are hand-crafted interactive elements asking students to perform a task, like proving bisimilarity of two systems, or minimizing a labeled transition system (LTS) up to observation congruence by executing a previously-described algorithm by hand. Students are asked to work directly within the exercise and their solutions are checked automatically. The amount of guidance given to students along the way differs across exercises.

Some interactive elements fall in-between the demonstration and exercise categories.

Many exercises are parametric and are re-used multiple times throughout a chapter. For example, all exercises related to proving two LTS to be bisimilar, weakly bisimilar, or observation congruent,

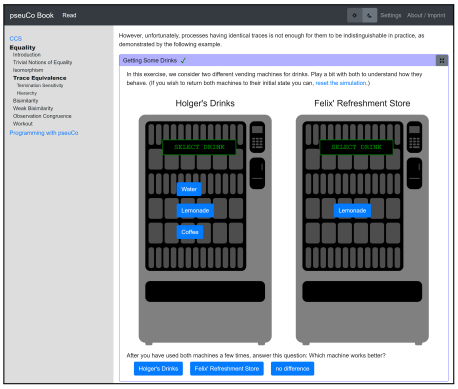


Figure 2: A screenshot of the interactive demonstration "Getting Some Drinks" shown inside the pseuCo Book web app

plus a demonstration teaching students how a bisimilarity proof for non-bisimilar systems fails, share the same implementation, parametrized by the LTS and flags configuring the type of equality and mode of operation. While pseuCo Book and pseuCo.com [2], the web IDE for pseuCo and CCS, are separate projects, the interactive elements of pseuCo Book borrow some of pseuCo.com's functionality, e.g. for LTS display, through a shared library.

The equality chapter of pseuCo Book is split into eight sections. After an introduction, it discusses the concept of equivalence classes, then introduces isomorphism, trace equivalence, bisimilarity, and weak bisimilarity, before finally describing observation congruence. "Workout", a final section added in response to student feedback, contains additional training exercises.

While this chapter of pseuCo Book uses only 3 quizzes, it contains a total of 25 hand-crafted demonstrations and exercises, 6 of which are workout exercises.

3.2 Deep Dive: An Interactive Demonstration

To exemplify the use of interactive elements in pseuCo Book, this section describes a single demonstration, located in the section on trace equivalence, in detail.

At this point within the book and lecture, students have gained an understanding that *actions* of processes are observable to the external world whereas *states* are internal, and that processes are considered to be *trace equivalent* if their sets of *traces*, i.e., sequences of actions processes can perform from their initial state, are identical. Based on this, many students assume that trace equivalence perfectly captures what a process truly *is*, in the sense that it fully describes what can be observed about the process from the outside.

This perception is incorrect as processes can observably differ in their branching behavior despite being trace equivalent. To help students *experience* and understand this (instead of teaching that insight purely theoretically by discussing process interactions on an LTS level), at this point, pseuCo Book contains the demonstration named "Getting Some Drinks", shown in Figure 2.

The demonstration shows two virtual vending machines. Initially, the students have no insight into the machines' internals, but can interact with them using buttons. Both machines show very similar behavior: They wait for the student to press "Insert Coin", then ask them to select a drink – water, lemonade, or coffee – using a button-push. Then, the selected drink is dispensed, and students see a "Take Water", "Take Lemonade", or "Take Coffee" button which reverts the machine to the initial state.

While the left machine, *Holger's Drinks*, works as expected, the right one, *Felix' Refreshment Store*, has a twist: While it offers all three types of drinks in principle, whenever a coin is inserted, it *nondeterministically* (in this demonstration: randomly) chooses one drink to offer exclusively. In Figure 2, the machine is depicted in this state, with lemonade having been chosen.

Students are asked to interact with both machines, then select which one works "better" (in an informal sense), forcing them to look for this (rather obvious) difference. Once this question has been answered correctly, a follow-up question asks students to compare the trace sets of both machines. To allow students to gain confidence that they have explored the full behavior of the machines, and to more easily track the possible traces, at this point in time, the underlying process models are revealed: The LTS are being superimposed on the vending machines, with the current states being highlighted as students can continue interacting with the machines. Once students have correctly indicated that the sets of traces are identical, the demonstration is marked as solved.

This example thus provides a gentle introduction of the fact that the branching structure of a process, and therefore its observable behavior, cannot be fully captured by only representing its traces.

4 PSEUCO BOOK LIVE SESSIONS

The exercises in pseuCo Book, like most traditional exercises, are generally designed to be used by students studying on their own. However, there are of course cases where these exercises may be useful in a more interactive setting within a course.

For example, to increase student engagement in the remote-learning context, in addition to presenting the material in detail in prerecorded videos, the instructor of the Concurrent Programming lecture held *live sessions*, online video conferences in which she briefly summarized the material and presented small exercises to be solved (and discussed) by the participating students.

Of course, the exercises in pseuCo Book can be used in such a context without specific technical support. However, their integration into pseuCo Book can help to solve a problem particularly prevalent in remote learning: The lack of feedback to the instructor. With students often being shy about asking questions or even sharing updates about their progress, it is hard for an instructor to judge whether students are mostly finished with an exercise, need more time, or require additional help to solve the exercise.

To provide this feedback when used in a live instruction context, pseuCo Book features *pseuCo Book Live Sessions*. They enable instructors to create a link that allows students to join their session within pseuCo Book (in addition to a video conference or in-person meeting). While in a live session, the instructor can send students to a specific exercise that is then opened automatically on all participating students' devices. Students are free to navigate pseuCo Book while in a live session, but can always return to the instructor-provided exercise with a single click.

When students work on an exercise as part of a pseuCo Book Live Session, their progress is tracked and shared with the instructor in

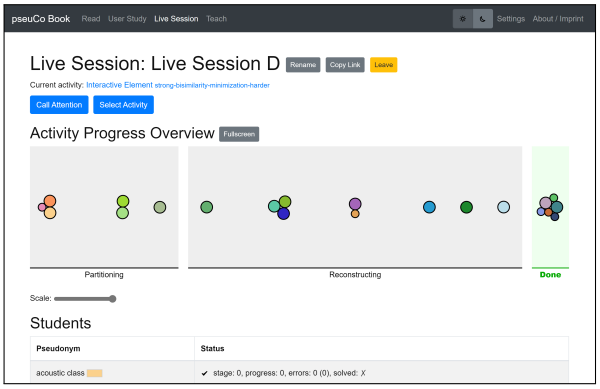


Figure 3: Screenshot of the teacher's view of a pseuCo Book Live Session. The overview shows each students' progress visually, with each dot representing one student. Smaller dots indicate a student is not actively working on the exercise.

real-time. The instructor view, shown in Figure 3, contains an animated overview of the student population indicating their progress throughout the exercise, grouping them into exercise-defined buckets. For example, for the exercise used in Figure 3, asking students to execute a two-phase LTS minimization algorithm by hand, it shows which phase each student is in and how many of the steps they have executed already.

For privacy reasons, and to encourage student participation, the information is shared in pseudonymized form, with each student being assigned a random, pseudonymous color. If the instructor decides to share the overview with their class, students can identify their own dot on this graph through its color as every student is shown their own pseudonymous color in a footer below the exercise. This may provide additional motivation to some students by letting them know when they are ahead of their peers.

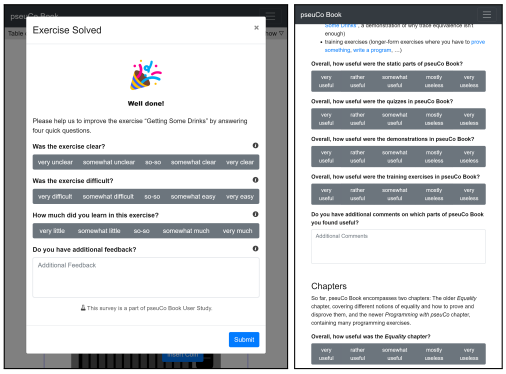
5 PSEUCO BOOK USER STUDY

To evaluate the effectiveness of pseuCo Book as an educational tool, i.e. to assess the educational effectiveness empirically as well as to measure the perceived usefulness of pseuCo Book, especially its interactive parts, by the students, a user study was conducted as part of the 2021 edition of the Concurrent Programming course at Saarland University, with participation open for the 185 students registered for the course. Due to the SARS-CoV-2 pandemic, the course was taught in a remote learning setting with the materials mainly being presented in prerecorded videos.

The study was conducted online as part of the students' regular usage of pseuCo Book. It was introduced in the lecture, and students were reminded to enroll in the study when they used pseuCo Book.

The study consisted of two parts:

- Participants of the study were shown two kinds of surveys:
  - Whenever a student had solved a hand-crafted interactive element for the first time, they were shown a brief questionnaire (see Figure 4a) asking for feedback.
  - A larger survey (see Figure 4b) was conducted after completion of the course. Participation in this survey was incentivized by a giveaway.



(a) Interactive element survey **(b) Final survey**

Figure 4: Screenshots of the two surveys shown to participants of the user study.

- For all students enrolled in the study, detailed logs of their usage of pseuCo Book were collected and stored. These logs include events when students opened or closed pseuCo Book, backgrounded or restored their pseuCo Book browser tab, navigated between different parts of the application, entered or left interactive elements, made progress or errors in interactive elements, or solved interactive elements. These logs allow computing the time students spent working with pseuCo Book and each interactive element with high accuracy.

The data collected in both parts was correlated with the students' marks and grades obtained in the Concurrent Programming lecture. In total, 77 users enrolled in the user study.

Ethics & Privacy. Data was collected based on the free, informed consent of all participants. Students were informed about the study design within the lecture and during enrollment. Participation in the study did not affect availability of materials or grading of the lecture. The study was approved by the Ethical Review Board of the Faculty of Mathematics and Computer Science at Saarland University and the university's data protection officer.

5.1 Surveys

In total, 582 questionnaires shown after solving an interactive element have been submitted, spread over 53 unique users, with 24 submitting feedback for at least half of all exercises. Multiple submissions were possible if users signed in on multiple devices or reset pseuCo Book. 13% of submissions were such duplicates.

The larger final survey was submitted by 40 users in total (and a limit of one submission per user was enforced for this survey).

5.1.1 Interactive Elements. In the questionnaires submitted per interactive element (cf. Figure 4a), users were asked four questions: (1) "Was the exercise clear?", (2) "Was the exercise difficult?", (3) "How much did you learn in this exercise?", and (4) "Do you have additional feedback?"

Questions 1 to 3 were shown with a five-point scale from "very unclear / easy / little" (1) to "very clear / difficult / much" (5).[3]

[3]For the second question, the options were displayed in reverse order, from "very difficult" to "very easy".

Property	N	Min.	Max.	Mean	Median	SD
clarity	546	1	5	4.40	5	0.97
difficulty	550	1	5	2.22	2	1.05
learning	543	1	5	3.53	4	1.09

Table 1: Aggregate ratings of all interactive elements. The number of answers varies as no questions were mandatory.

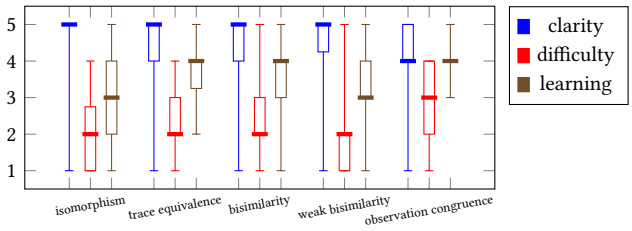


Figure 5: Ratings of interactive elements by topic

Usefulness of…	N	Min.	Max.	Mean	Median	SD
static content	40	1	5	3.92	4	0.83
quizzes	40	3	5	4.32	4	0.69
demonstrations	40	3	5	4.52	5	0.68
exercises	39	3	5	4.69	5	0.52
chapter (overall)	40	3	5	4.72	5	0.55

Table 2: Evaluation of the final survey

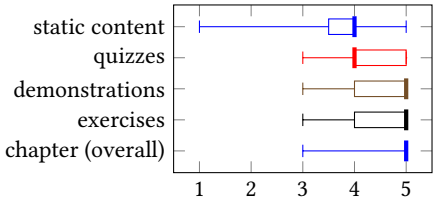


Figure 6: Overall usefulness ratings

The results are shown in Table 1. Overall, students perceived the exercises as generally clear and somewhat easy. Their perceived learning effect was good, but leaves room for improvement.

A more detailed analysis, splitting the answers by the topic handled in the interactive elements, is shown in Figure 5, with topics presented in chronological order. It shows that clarity was perceived as high throughout the topics, with a minor dip towards the end. The perceived difficulty slowly rises, but even for the last topic, it centers around "so-so". The perceived learning effect is also higher for the later topics. We believe these differences can be explained by students generally struggling more with topics building upon previous knowledge.

5.1.2 Final Survey. The rating questions from the final survey, asking students to describe the usefulness of different parts of pseuCo Book on a 5-point scale from "very useful" (5) to "very useless" (1), are evaluated in Table 2 and Figure 6. All questions yielded high ratings, with averages around 4 ("rather useful") or 5 ("very useful"). Notably, the students rank the static content (text and figures) of pseuCo Book lowest, followed by quizzes, and rank demonstrations and exercises as the most useful parts. Another detail of note is that the whole equality chapter is rated as slightly more useful than any of its parts, which may indicate that students see additional value in the tight integration between these parts.

We also analyzed the answers to five freeform questions: (1) "Do you have additional comments on which parts of pseuCo Book you found useful?", (2) "Did you have any technical issues while using pseuCo Book?", (3) "What did you like about pseuCo Book?", (4) "What did you dislike about pseuCo Book?", and (5) "Do you have any further suggestions for improving pseuCo Book?"

Of the 40 submissions of this survey, 23 included freeform answers, which were classified manually by the author. The classification was validated by an external contributor.

In total, 10 students explicitly described pseuCo Book as "helpful" or "useful". Only a single student reported a technical issue.

21 students highlighted details they liked. The most liked parts were the exercises ($n = 12$) and the interactivity in general ($n = 6$), followed by the interactive demonstrations ($n = 3$) and the step-by-step guidance within exercises ($n = 3$). Other items given ($n = 2$) were the feedback given by exercises, the overall presentation / usability, and the structure of pseuCo Book.

Overall, 13 students made suggestions.[4] The most popular ($n = 6$) suggestion was to extend pseuCo Book to cover more parts of the lecture. Similarly, the second most popular suggestion ($n = 4$) was to add more exercises.[5] Other suggestions given repeatedly ($n = 2$) were improvements to navigation (like a search feature), more options to reset exercises or whole sections, and an option to show sample solutions for exercises.

A total of 15 students expressed dislikes. The most common ($n = 5$) was a lack of topics covered in pseuCo Book with respect to the lecture syllabus. Other repeated dislikes included pseuCo Book not being a full replacement for the lecture notes ($n = 3$), some exercises not preparing students well enough for a pen-and-paper scenario ($n = 2$), and the spoiler hiding mechanism ($n = 2$).

5.2 Empirical Evaluation

To empirically evaluate the effectiveness of pseuCo Book, we measured its usage by study participants and correlated that with their performance in the Concurrent Programming lecture.

The built-in tracking mechanism allowed us to calculate the time participants spent in pseuCo Book and in specific interactive elements. Timers were started when the user opened pseuCo Book or any interactive element, and were stopped automatically when the user navigated away or their browser was no longer visible. In addition, a timeout of 5 minutes ensured that e.g. users simply leaving their machines were not counted in forever. For technical reasons, offline usage was not counted.

We correlated our measurements against two performance indicators from the Concurrent Programming lecture:

[4] This includes suggestions given outside of question 5.
[5] This suggestion was already partially implemented during the lecture by the addition of a "Workout" section containing additional instances of the exercises presented previously throughout the chapter.

#	Time	Points	Slope β_1	F Statistic	p-Value
1	book	admission	1.05	4.64	0.017
2	book	exams	1.59	4.86	0.015
3	interact.	exam ex. 1	−0.02	0.02	—
4	interact.	exam ex. 2	0.12	0.09	0.380
5	interact.	exam ex. 3	0.67	12.20	0.000 43
6	interact.	exam ex. 4	0.28	1.30	0.129
7	interact.	exam ex. 5	0.09	0.73	0.198

Table 3: Comparison of pseuCo Book usage (total time in book / in interactive elements) to performance in lecture

- The lecture had an exam admission requirement based on assignment sheets (submitted in groups of generally three students). We collected the total number of points students received on these sheets.
- The lecture featured four exams: Two main exams, one for the theoretical part of the lecture and one for the practical part, plus one re-exam each. We collected the results from these exams on a per-exercise granularity.

Overall, the 77 participants of the study generated 187100 tracking events over 1719 sessions in pseuCo Book. In total, these sessions encompassed 260 hours of usage, averaging 3.38 hours per user. 52 participants had at least one hour of usage time.

We analyzed multiple pairings of timing measurements from pseuCo Book and performance measurements from the course. For each pairing, we fitted a linear regression (with slope β_1), performed an ANOVA analysis, and used a one-sided T-test to check for relevance of our model (i.e., null hypothesis H_0: $\beta_1 \leq 0$, alternative H_a: $\beta_1 > 0$). Results are shown in Table 3, summarized for brevity.

Scenario 1 compares the total time a student spent in pseuCo Book with their total admission points from assignment sheets. Scenario 2 compares the same timing with the point sum of the exams (with re-exams replacing the corresponding main exam scores as appropriate). In these scenarios, we see a significant ($p < 0.05$) correlation between total usage of pseuCo Book and student performance, with each hour of time spent corresponding to roughly 1.1 admission points (out of 96) and 1.6 exam points (out of 120).

This correlation is expected as it can be explained by students benefiting from using pseuCo Book, but also by at least one likely confounder: Students spending more time in pseuCo Book are more likely to be motivated students who invest more time into the lecture and therefore perform better overall. To analyze this further, Scenarios 3 through 7 compare the total time a student spent in interactive elements in the equality chapter with their marks in the 5 individual exercises in the theoretical exam (not considering the re-exam as there is no direct equivalence between the exercises):

Exercise 1 was a warmup exercise, not directly related to the equality chapter, only handling some prerequisites.

Exercise 2 was a congruence proof for bisimilarity, considering a new CCS operator.

Exercise 3 asked students to apply trace equivalence, weak bisimilarity, and observation congruence to transition systems.

Exercise 4 required students to model a system using CCS.

Exercise 5 contained multiple puzzles, many related to CCS.

If students profited from the knowledge they acquired in pseuCo Book directly, we would expect to see strong correlations between learning time in pseuCo Book and strongly related exercises, and weak or no correlation to other exam exercises.

Indeed, we find no significant correlation in Scenarios 3, 6, and 7, which considered exercises with no or little direct relation to notions of equality. In Scenario 5, covering exercise 3, which tested the students' ability to apply three of the notions of equality handled in pseuCo Book to examples, we see a *highly* significant ($p < 0.001$) correlation between learning time and exam performance.

Curiously, for exercise 2 (Scenario 4), we see no significant correlation. This is likely explained by the fact that while this exercise revolved around bisimilarity, a core topic of pseuCo Book, it mostly tested the ability of students to handle abstract proofs, not their understanding of bisimilarity itself. An expansion of pseuCo Book to handle this topic in more detail seems desirable (and was, in fact, suggest by one student in the evaluation survey).

In summary, we believe this analysis shows strong evidence of students gaining understanding of the topics covered in pseuCo Book through the included interactive elements.

5.2.1 Limitations. As this analysis is based on a non-controlled study, it only proves correlation. While we believe the most important confounder, differing student motivations, has been controlled for by the analysis at exercise granularity, some possible confounders remain. For example, students spending more time in pseuCo Book may place higher relative weight on the topic of equality, spending more time preparing for related exam questions using other materials. Moreover, while the very strong correlation shown seen in Scenario 5 and the analysis of students' perception through surveys provide some evidence for the specific effectiveness of pseuCo Book, this study did not compare pseuCo Book with other teaching tools, e.g. the lecture notes used previously. Additional evaluation through a controlled study would be desirable.

6 CONCLUSION

In this paper, we presented pseuCo Book, a tool to assist in teaching difficult-to-grasp concepts of theoretical computer science by combining traditional textbook-like content with directly embedded interactive exercises and demonstrations. Through a multi-faceted analysis of data collected in a user study, we have shown pseuCo Book's effectiveness in teaching students these concepts to the point where they can apply them effectively.

The underlying authoring tool, the Hybrid Document Format, and the corresponding reader application are available under an open source license, allowing others to easily create interactive books covering other topics of computer science.

ACKNOWLEDGMENTS

This work was partially funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – project number 389792660 – TRR 248 – CPEC, see https://perspicuous-computing.science. The author would like to thank Holger Hermanns for his contributions, Verena Wolf for her help in conducting and evaluating the study, Gregory Stock for his help in evaluating the study, and the anonymous reviewers for their constructive feedback.

REFERENCES

[1] Rachid Anane and Mohammad T. Alshammari. 2020. A Dynamic Visualisation of the DES Algorithm and a Multi-faceted Evaluation of Its Educational Value. In *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education, ITiCSE 2020, Trondheim, Norway, June 15-19, 2020*, Michail N. Giannakos, Guttorm Sindre, Andrew Luxton-Reilly, and Monica Divitini (Eds.). ACM, Trondheim, Norway, 370–376. https://doi.org/10.1145/3341525.3387386

[2] Sebastian Biewer, Felix Freiberger, Pascal Leo Held, and Holger Hermanns. 2017. Teaching Academic Concurrency to Amazing Students. In *Models, Algorithms, Logics and Tools - Essays Dedicated to Kim Guldstrand Larsen on the Occasion of His 60th Birthday (Lecture Notes in Computer Science, Vol. 10460)*, Luca Aceto, Giorgio Bacci, Giovanni Bacci, Anna Ingólfsdóttir, Axel Legay, and Radu Mardare (Eds.). Springer, Switzerland, 170–195. https://doi.org/10.1007/978-3-319-63121-9_9

[3] David M. Cerna, Martina Seidl, Wolfgang Schreiner, Wolfgang Windsteiger, and Armin Biere. 2020. Aiding an Introduction to Formal Reasoning Within a First-Year Logic Course for CS Majors Using a Mobile Self-Study App. In *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education, ITiCSE 2020, Trondheim, Norway, June 15-19, 2020*, Michail N. Giannakos, Guttorm Sindre, Andrew Luxton-Reilly, and Monica Divitini (Eds.). ACM, Trondheim, Norway, 61–67. https://doi.org/10.1145/3341525.3387409

[4] Loris D'Antoni, Martin Helfrich, Jan Kretínský, Emanuel Ramneantu, and Maximilian Weininger. 2020. Automata Tutor v3. In *Computer Aided Verification - 32nd International Conference, CAV 2020, Los Angeles, CA, USA, July 21-24, 2020, Proceedings, Part II (Lecture Notes in Computer Science, Vol. 12225)*, Shuvendu K. Lahiri and Chao Wang (Eds.). Springer, Los Angeles, CA, USA, 3–14. https://doi.org/10.1007/978-3-030-53291-8_1

[5] Loris D'Antoni, Matthew Weavery, Alexander Weinert, and Rajeev Alur. 2015. Automata Tutor and what we learned from building an online teaching tool. *Bull. EATCS* 117 (2015), 144–158. http://eatcs.org/beatcs/index.php/beatcs/article/view/365

[6] Dan Garcia, Brian Harvey, and Tiffany Barnes. 2015. The beauty and joy of computing. *Inroads* 6, 4 (2015), 71–79. https://doi.org/10.1145/2835184

[7] Felienne Hermans. 2020. Hedy: A Gradual Language for Programming Education. In *ICER 2020: International Computing Education Research Conference, Virtual Event, New Zealand, August 10-12, 2020*, Anthony V. Robins, Adon Moskal, Amy J. Ko, and Renée McCauley (Eds.). ACM, New Zealand, 259–270. https://doi.org/10.1145/3372782.3406262

[8] Joint Task Force on Computing Curricula, Association for Computing Machinery (ACM) and IEEE Computer Society. 2013. *Computer Science Curricula 2013*. ACM Press and IEEE Computer Society Press, New York, NY, United States. https://doi.org/10.1145/2534860

[9] Samiha Marwan, Ge Gao, Susan R. Fisk, Thomas W. Price, and Tiffany Barnes. 2020. Adaptive Immediate Feedback Can Improve Novice Programming Engagement and Intention to Persist in Computer Science. In *ICER 2020: International Computing Education Research Conference, Virtual Event, New Zealand, August 10-12, 2020*, Anthony V. Robins, Adon Moskal, Amy J. Ko, and Renée McCauley (Eds.). ACM, New Zealand, 194–203. https://doi.org/10.1145/3372782.3406264

[10] Robin Milner. 1980. *A Calculus of Communicating Systems*. Lecture Notes in Computer Science, Vol. 92. Springer-Verlag, Berlin. https://doi.org/10.1007/3-540-10235-3

[11] Robin Milner. 1989. *Communication and concurrency*. Prentice Hall, New York.

[12] Thomas W. Price, Yihuan Dong, and Dragan Lipovac. 2017. iSnap: Towards Intelligent Tutoring in Novice Programming Environments. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education, SIGCSE 2017, Seattle, WA, USA, March 8-11, 2017*, Michael E. Caspersen, Stephen H. Edwards, Tiffany Barnes, and Daniel D. Garcia (Eds.). ACM, Seattle, WA, USA, 483–488. https://doi.org/10.1145/3017680.3017762