UNIVERSITY OF PITTSBURGH

ISSP 2990
INDEPENDENT STUDY
SUMMER 2022

Augmenting Digital Textbooks with Smart Content

Author:
Arun-Balajiee
LEKSHMI-NARAYANAN

Supervisor:
Dr. Peter BRUSILOVSKY

A report submitted in fulfillment of the requirements for Independent study

in the

PAWS Lab
Intelligent Systems Program
School of Computing and Information

December 28, 2023

UNIVERSITY OF PITTSBURGH

Abstract

Intelligent Systems Program
School of Computing and Information

Augmenting Digital Textbooks with Smart Content

by Arun-Balajiee LEKSHMI-NARAYANAN

A powerful set of educational tools has emerged over the last decade with the rise in the adoption of online adaptive learning content. An increasingly popular tool in this space is the "intelligent textbook" as a platform to support and distribute content for e-learning, given its resemblance with real-life physical books. Existing efforts in this direction include the development of digital textbooks where both textual content and interactive learning activities (i.e., examples, problems, etc.) are carefully handcrafted by the authors so that they are perfectly placed to follow the knowledge acquisition-practice flow. However, this approach is very time-consuming, and it requires the work of high-expertise authors. In this work, we suggest and discuss a scalable solution: we take existing digital textbooks and augment them by using repositories of existing online learning material associated with the subject matter. We present our current work in this direction and discuss challenges and opportunities for the future work. This work was done in collaboration with Jordan Barria-Pineda.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

A gradual switch from paper-based to "electronic" textbooks (e-textbooks) opened an exciting opportunity to extend these classic learning tools with functionalities not previously available in paper format. Among the most appealing and popular ways to extend textbooks with new functionalities is converting examples and problems, a traditional component of textbooks in many domains, into interactive learning activities. This approach makes textbooks truly interactive and augments learning by reading with learning by doing.

One of the first domains to embrace this kind of interactive textbook was computer science education (CSE) where the development of interactive learning activities from algorithm animations to automatically-assessed programming problems was a popular research direction. The need to integrate interactive learning activities with online textbooks has been extensively discussed by the computer science education community for many years Rößling et al., 2006 and some best examples of interactive textbooks have been produced for computer science subjects. Among these examples are ELM-ART Brusilovsky, Schwarz, and Weber, 1997, the first adaptive textbook with interactive problems and examples for learning LISP, OpenDSA Fouh et al., 2014, the first open-source infrastructure for collaborative construction of e-textbooks with interactive animations and problems (originally developed for Data Structures and Algorithms course), and RuneStone books Ericson and Miller, 2020 a popular infrastructure for presenting online textbooks for programming augmented with interactive learning activities. These and other interactive textbooks have been extensively evaluated in various learning contexts and their effectiveness was convincingly demonstrated Weber and Brusilovsky, 2001; Ericson, Guzdial, and Morrison, 2015; Pollari-Malmi et al., 2017.

However, the current platforms for the development and delivery of interactive textbooks for CSE share the same problem: the "custom" nature of their production. These textbooks are expected to be developed "as a whole" for a specific purpose, with text and interactive problems developed and integrated together as a part of the authoring process. This approach allows developing excellent examples of interactive textbooks but doesn't support scaling up this process. For each "holistically developed" interactive textbook, there are dozens of professionally authored textbooks on the same subject that are not augmented with interactive content because this option has not been considered at the time of their creation. At the same, there are large repositories of interactive learning content of different types that could be used to augment these books. A missing piece in the infrastructure is the integration of an arbitrary textbook with its corresponding interactive content.

An important step towards building this infrastructure was done in OpenDSA project Fouh et al., 2014, which offered an opportunity to connect any LTI-compatible interactive content to OpenDSA textbooks. However, it is still focused on custom-built textbooks and doesn't support existing textbook. The project presented in this

paper attempts to take the next step in this direction and make both textbooks and integrative learning content reusable. Our goal is to build an infrastructure that allows turning any textbook available in electronic format (such as PDF) into an interactive textbook by augmenting it with interactive learning content from existing repositories. This paper presents an important component of this infrastructure - and interface that support augmentation of existing books with interactive content without breaking the structure of these textbooks. In the following sections, we present our current implementation of this interface, demonstrate the approach for integrating smart content into textbook structure, and discuss future work in this direction.

Chapter 2

Related Work

Multiple research efforts have been carried out during the last decade in order to develop technology-enhanced textbooks. Electronic Textbooks (e-textbooks) support content distribution at scale in different formats and for different purposes. In recent years, there have been many discussions that project what technological enhancements could surround the use of intelligent e-textbooks in education Ritter et al., 2019. Among these discussions, a few noteworthy contributions such as the use of intelligent question-asking Koc-Januchta et al., 2020, intelligent tutoring Walker et al., 2017, and augmentation of assessment questions Dresscher, Chacon, and Sosnovsky, 2021 bring to the light the possibilities of interesting enhancements in smart digital textbooks. On the one hand, these technological enhancements could be implemented as artificially intelligent agents or systems in e-textbooks that deliver, recommend or scaffold the learner's needs while reading Xu and Warschauer, 2020. On the other hand, it could be possible to integrate reusable smart content that are adaptive to the needs of the user, without necessarily adapting or personalizing the system behaviour to the learner. In general, the main idea has been to maintain the affordances of physical textbooks combined with the capabilities of web pages. However, some efforts have been made to incorporate the design of the novel functionalities for students that could expand the potentialities of intelligent textbooks Walker et al., 2018.

In our work, we take this second route of augmentation of intelligent textbooks with adaptive, personalized learning material presented as Smart Learning Content (SLC) Brusilovsky et al., 2014. Typically, there are 5 different levels of SLCs Brusilovsky et al., 2014, namely,

1. **Level 1** SLCs are independent of the delivery platform for learning. In this setup, the variables used by the SLC to personalize content do not persist, once the session is closed and is stateless.

2. **Level 2** The SLC and the delivery platform for learning are integrated into a single system and the delivery platform saves the data produced by the SLC, which is used across several user sessions. A limitation of this setup is that the SLC would need to be developed specifically for the delivery platform and cannot exist outside of the system. For example, CodeAcademy [1], KhanAcademy [2], Brilliant [3]

3. **Level 3** All content in the SLCs is internal to the platform, but the delivery platform supports multiple SLCs. For example, OpenDSA Fouh et al., 2014

[1] http://codeacademy.com
[2] https://khanacademy.org
[3] https://brilliant.org

4. **Level 4** The platform supports multiple SLCs and allows the use of external content, using proprietary protocols to retrieve the external content. For example, BlueJ or Moodle with plug-in support, TestMyCode Vihavainen et al., 2013, A+ Karavirta, Ihantola, and Koskinen, 2013 and JavaGuide Hsiao, Sosnovsky, and Brusilovsky, 2010.

5. **Level 5** The platform supports multiple SLCs that use standard protocols, such as Learning Tools Interoperability (LTI), allowing for maximum flexibility. For example, LTI with Moodle [4].

Among these levels, our earlier discussions cover implementations that could be considered as level 2 SLCs Chacon et al., 2021. We also discussed implementations that utilize the benefits of Learning Tools Interoperability (LTI) Barria-Pineda, Akhuseyinoglu, and Brusilovsky, 2019 to integrate early implementations of level 5 SLCs in the intelligent textbooks. In this work, we explore possibilities for an implementation that could meet the gold standards discussed as levels 3 and 4 in our prior work, that is, to support multiple SLCs that are both native and external resource recommendations to the delivery platform. In this case, we set the platform content delivery to be the intelligent textbook.

[4] https://docs.moodle.org/400/en/LTI_and_Moodle

Chapter 3

Augmenting Reusable Smart Learning Content

We set the goals for level 3 and 4 SLCs, which we describe again here,

1. **Level 3** Integrate Multiple SLCs that are native to the platform such as OpenDSA Fouh et al., 2014
2. **Level 4** Support multiple SLCs on the platform such that external content can be integrated into the system.

Currently, we implemented two ways to integrate SLC into a textbook: a list of recommended videos (see the red tab in Fig. 3.1) and a list of statically attached interactive exercises (see the green tab in Fig. 3.1). The video interface was developed for an information retrieval textbook. It shows recommended videos using thumbnails, which work as links to related content on YouTube[1] and multimedia sharing websites. These implementations could consider a ranking and rating-based approach to listing the content to allow the factor of "human-in-the-loop" recommendations to support and enhance intelligent recommendations to the users of these systems and their students in these courses.

The second way (a list of interactive programming exercises) was developed for e-textbooks on introductory programming. The list of available exercise types with descriptions is provided in Table 3.1. These exercises range from simple problems that test the student's understanding of the inputs or outputs of a given program to puzzles that can be solved in several steps. Interactive and animated examples could make the process of reading and understanding the code more engaging for the reader. This could scaffold a student's learning in their process of understanding a course on introductory concepts in programming. Such an SLC integration could possibly turn the mundane process of reading a textbook into a rich, interactive experience that offers possibilities for hands-on content experimentation. Further, students who are curious learners can explore the concepts discussed on a page with a related live, interactive examples to keep them engaged.

3.1 Proof-of-Concept Implementations

We collected a set of SLCs from various sources, in order, to offer a wide range of online learning activities to students. The available content ranges from a low level of interactivity (i.e., educational videos and worked-out examples) to more interactive activities (i.e., parsons problems and coding-from-scratch problems). In this paper, we will focus on two courses as study cases:

[1] https://www.youtube.com

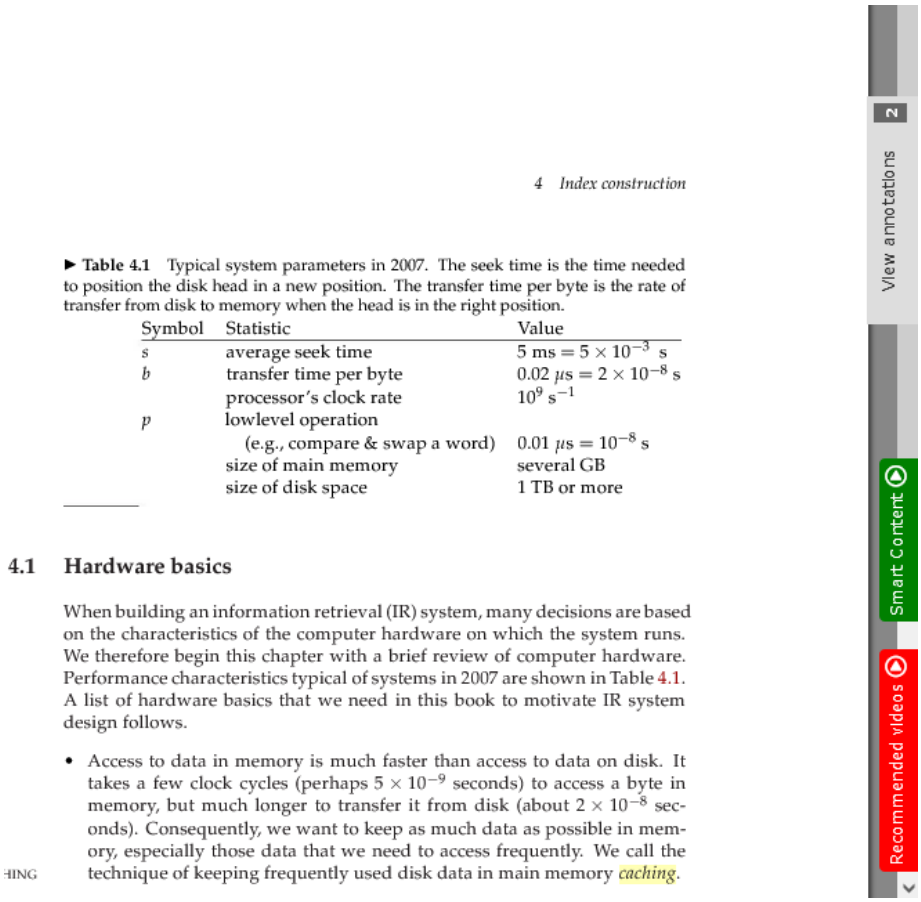

FIGURE 3.1: Smart Content (green tab) along with the Video Recommendations (red tab) in an information retrieval textbook

1. A Graduate course on Information Retrieval based on an open source textbook
2. An Undergraduate course on Programming in Python with the main textbook for reference, "Python for Everybody"[2].

In this Independent Study, we only focus on integrating Python textbook with smart content.

3.1.1 A Python Programming Textbook with Smart Content

To support a Python Programming course we augmented a popular textbook "Python for Everybody" with smart content. This textbook is available in several formats, including as a PDF[3]. It consists of several chapters that cover the basics of Python programming. The textbook starts with delivering the course content from scratch, going into sequential topics, and keeping the target audience as novice or beginner-level programmers in Python. We use this setup to experiment with SLC implementations to help practise programming in Python with a set of worked out examples and programming problems. We target SLCs that cannot be directly covered within the text. We think that the programming exercises could be presented as a list of short problems related to the material being read in a given page, section or chapter of the book.

[2]https://www.py4e.com
[3]https://www.py4e.com/book

TABLE 3.1: Types of Systems for Integrated Python Programming Exercises

System	Types of Exercises
QuizPET Brusilovsky et al., 2018	Parameterized code tracing problems for Python with automatic assessment
PCEX Hosseini et al., 2020	Program construction examples in an engaging, interactive form in order to increase motivation
WebEX Brusilovsky, 2001	Web-based programming examples in Python & other languages
2D Parsons Ihantola and Karavirta, 2011	A 2D version of Parson's puzzles for Python
PCRS Zingaro et al., 2013	Programming problems that provide incomplete skeleton code and checks the answers using a set of tests
Jsvee Sirkiä, 2018	Animated programming examples to visualize the program steps

To test our current infrastructure, we attached a range of smart learning content for Python to various sections of the textbook as shown in Fig. 3.2. When a link to an SLC item from the list of entries for smart content is clicked, it launches a dialog instance with the specific programming example or problem. For example, Fig. 3.3 shows a code tracing problem from QuizPET system (Quizzes for Python Educational Testing) Brusilovsky et al., 2018. Another kind of programming exercises for Python that we made available in the book are interactive worked examples of program construction from PCEX system with step-wise program explanations and walk-through (Fig. 3.4. These exercises allow for the student to focus on specifics of a given program).

In total, we demonstrated the ability to connect six types of SLC worked examples and problems listed in Table 3.1. A more detailed description of these SLC types could be found in Hosseini et al., 2020. When augmenting the book with SLC, we considered Python programming activities that are related to the topics covered in the text of a particular page of section in a chapter. These programming activities use the knowledge or concepts covered in the textbook up to that point and avoid the concepts that will be covered later in the textbook.

In the future version, we hope to provide a smart textbook authoring system for course instructors, which will allow them to augment the same textbook with SLC that they want to use in their classes. A prototype of this authoring system with learning analytics support can be found in Albó et al., 2022. We also plan to support the authoring process with instructor-focused content recommender system Chau, Barria-Pineda, and Brusilovsky, 2018. This could be considered as our long term goal for smart content, but in the current work of smart content design for programming exercises, we only focus on the interface for delivering SLC to students through a textbook.

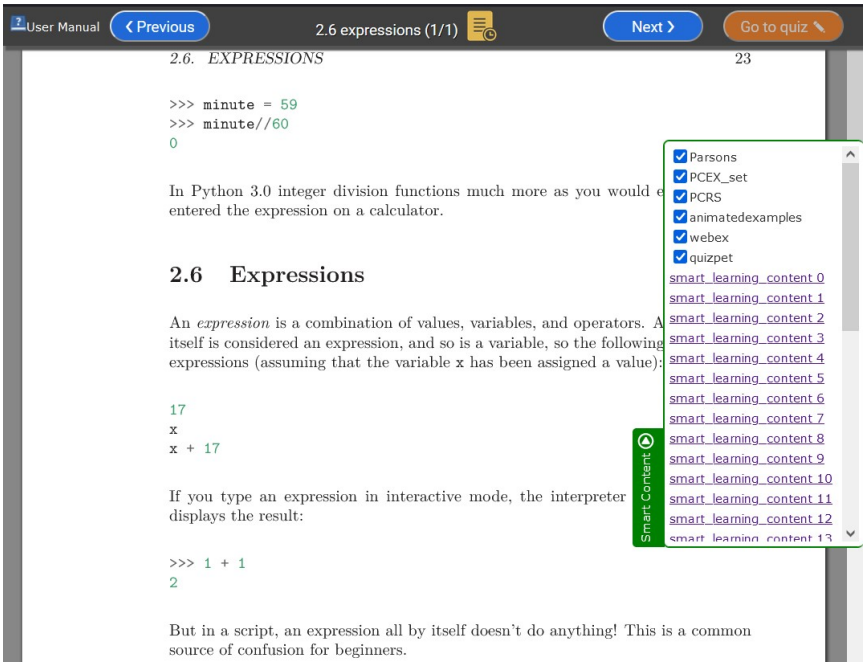

FIGURE 3.2: List of smart exercises displayed to practice Python programming. These are links to the different programming exercises hosted on external repositories, to support Level 4 smart learning content

FIGURE 3.3: Smart Content Modal with a Programming Exercise in Python. In this example, the student has to type the correct output to the program in the textbox below for system to evaluate

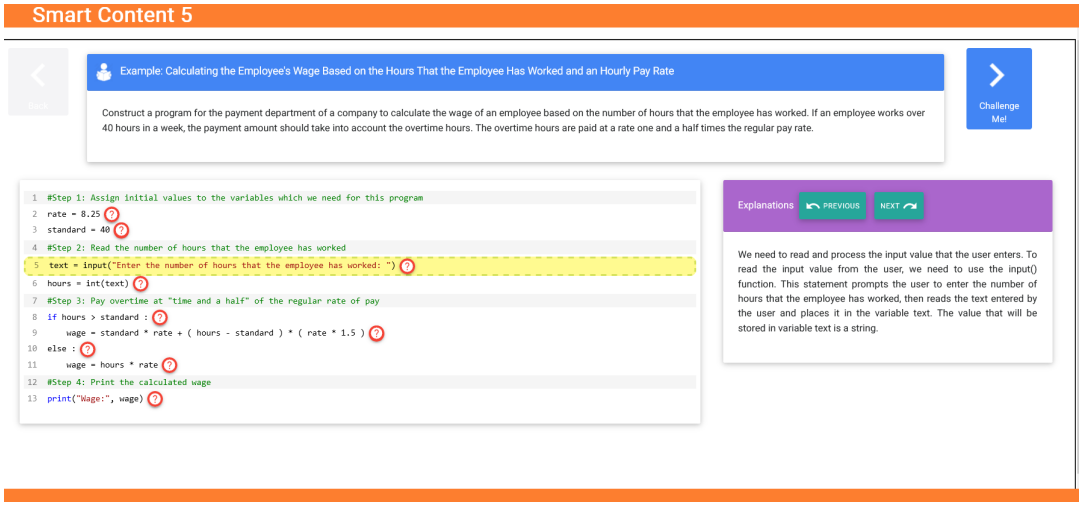

FIGURE 3.4: Smart Content Modal with a programming exercise in Python. In this figure, the student is asked to explain the different lines in a given program, broked down into smaller steps and compare them with the standard explanations that the system expects to be appropriate at those lines

Chapter 4

Discussion & Future Work

With our implementation, we show that it is possible to integrate and augment e-textbooks with multiple SLCs. They are available as non-intrusive sidebars for the reader to explore material relevant, without losing focus on the main text content of the intelligent textbook. Dynamic SLC recommendations provided at different levels of granularity by relevance (given section, chapter, page or paragraph) is our next step to explore. Determining the most appropriate granularity and difficulty level of the programming exercises in the smart content to act as useful recommendations that scaffold students' learning. Further, adaptations could model the patterns of user or student interactions with the system to fine-tune the recommended SLC, further governed by the reader's control on the ("human-in-the-loop") curation of the SLCs listed with rating and ranking features. SLC of other types could take no inputs from the user, but present as passive recommendations in relation to the concepts covered at a page or section. Learner-sourced approaches to recommend questions Ni et al., 2021; Huang et al., 2020 is another interesting research opportunity to explore and address the challenge of dynamic content allocation. Questions that are most relevant to a page, section or chapter could be dynamically curated in the side panel. The learner-sourced SLC could be generated by peers taking the course or students who took the course. This material could be rated and ranked by the current users to improve the recommendations provided. Reusing resources in this manner could potentially open the doors to exemplary SLCs integrated into intelligent textbooks for other learner content delivery systems. These opportunities can be realized by overcoming a few challenges discussed below.

4.0.1 Challenges

In our implementation towards integrating multiple SLCs, we find that allocating the right content could be a potential challenge. While it is possible to support personalized, integrable and adaptive SLCs for specific chapters or sections within intelligent textbooks on different topics, it is a challenge to make it scale up to different topics and courses in these system implementations. Another challenge is that for instructors teaching these programming courses, as discussed by Chau et. al. Chau, Barria-Pineda, and Brusilovsky, 2018, the content allocation for all the intended concepts in the course may not be possible and this makes this implementation potentially static. Further, a scope to explore would be smart content allocation that adapts to the teachers' understanding of the course topics Sosnovsky and Brusilovsky, 2015 as another challenge. It would especially be interesting to include a smart content that is modeled by the topic and the knowledge levels of the course instructors. Hence, the presented content would then augment their understanding of the topics covered in the course. All these three challenges consider a "human-in-the-loop" implementation. A final challenge is to integrate an SLC to augment the content

presented in the digital textbook as a recommender system that involves less human intervention to improve its personalization. An example to support such an integration would be recommendations to external web resources like Wikipedia with additional information is not native to the content available within the text, but augments the information provided without much scope for the user to rate or rank these recommendations to match their personal choices. An implementation in this direction Rahdari et al., 2020 sets possible paths for us to explore as augmented SLC in our future work and as a means to overcome this challenge. In the light of understanding and overcoming these challenges, we will be able to explore and support more types of dynamic SLCs in our future iterations of intelligent textbooks.

4.1 Conclusions

In this work, we present an interesting perspective on integrating smart learning content (SLCs) in intelligent textbooks as the delivery platform. Along with the possibility of supporting multiple SLCs, they could be native and external resources using open, proprietary protocols (levels 3 and 4 SLCs) for retrieval. This meets our goal that we set forth of building an infrastructure that could turn any ordinary e-textbook into an intelligent, adaptive and interactive textbook. Although not our goal to begin with, since our implementation uses the resources that are not *native*, but external to the intelligent textbooks framework (we benefit from using the SLC repositories developed by others), we present a system that is flexible, suggesting that the learning content delivery platform can be interchangeable (level 5). The seamless integration of the SLCs into intelligent textbooks, allows for the possibility of interactive and engaging learning content delivery platforms for curious learners. In the long term, this allows for better adoption of enhanced intelligent e-textbooks. Finally, we discuss the challenges encountered while making scalable integration of SLCs into the deliver platform. We discuss existing solutions that could allow us to overcome these challenges. Technical advancements in the not so distant future could help address these challenges with efficient protocols for seamless augmentation of smart learning content without breaking the structure of the e-textbooks.

Acknowledgements

We acknowledge the help offered by our colleagues in the implementation of parser and smart learning content allocation in our intelligent textbook implementation. Also, the work of one of the authors was funded by CONICYT PFCHA/ Doctorado Becas Chile/ 2018 - 72190680.

Bibliography

Albó, Laia et al. (2022). "Knowledge-Based Design Analytics for Authoring Courses with Smart Learning Content". In: *International Journal of Artificial Intelligence in Education* 32, pp. 4–27.

Barria-Pineda, Jordan, Kamil Akhuseyinoglu, and Peter Brusilovsky (2019). "Learning Content Integration into an Electronic Textbook for Introductory Programming". In.

Brusilovsky, Peter (2001). "WebEx: Learning from Examples in a Programming Course". In: *WebNet*.

Brusilovsky, Peter, Elmar Schwarz, and Gerhard Weber (1997). "Electronic textbooks on WWW: from static hypertext to interactivity and adaptivity". In: *Web Based Instruction*. Ed. by Badrul H. Khan. Englewood Cliffs, New Jersey: Educational Technology Publications, pp. 255–261.

Brusilovsky, Peter et al. (2014). "Increasing Adoption of Smart Learning Content for Computer Science Education". In: *Proceedings of the Working Group Reports of the 2014 on Innovation amp; Technology in Computer Science Education Conference*. ITiCSE-WGR '14. Uppsala, Sweden: Association for Computing Machinery, 31–57. ISBN: 9781450334068. DOI: 10.1145/2713609.2713611.

Brusilovsky, Peter et al. (2018). "An integrated practice system for learning programming in Python: design and evaluation". In: *Research and Practice in Technology Enhanced Learning* 13.

Chacon, Isaac Alpizar et al. (2021). "Integrating Textbooks with Smart Interactive Content for Learning Programming". In: *iTextbooks@AIED*.

Chau, Hung, Jordan Barria-Pineda, and Peter Brusilovsky (2018). "Learning Content Recommender System for Instructors of Programming Courses". In: *AIED*.

Dresscher, Lucas, Isaac Alpizar Chacon, and Sergey A. Sosnovsky (2021). "Generation of Assessment Questions from Textbooks Enriched with Knowledge Models". In: *iTextbooks@AIED*.

Ericson, Barbara J., Mark J. Guzdial, and Briana B. Morrison (2015). "Analysis of Interactive Features Designed to Enhance Learning in an Ebook". In: *Proceedings of the 11th International Conference on International Computing Education Research*. ACM. DOI: 10.1145/2787622.2787731. URL: https://doi.org/10.1145%2F2787622.2787731.

Ericson, Barbara J. and Bradley N. Miller (2020). "Runestone: A Platform for Free, On-Line, and Interactive Ebooks". In: *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*. New York, NY, USA: Association for Computing Machinery, 1012–1018. ISBN: 9781450367936. URL: https://doi.org/10.1145/3328778.3366950.

Fouh, Eric et al. (2014). "Design and architecture of an interactive eTextbook – The OpenDSA system". In: *Science of Computer Programming* 88, pp. 22–40. ISSN: 0167-6423. DOI: https://doi.org/10.1016/j.scico.2013.11.040.

Hosseini, Roya et al. (2020). "Improving Engagement in Program Construction Examples for Learning Python Programming". In: *International Journal of Artificial Intelligence in Education* 30, pp. 299–336.

Hsiao, Ihan, Sergey A. Sosnovsky, and Peter Brusilovsky (2010). "Guiding students to the right questions: adaptive navigation support in an E-Learning system for Java programming". In: *J. Comput. Assist. Learn.* 26, pp. 270–283.

Huang, Alice et al. (2020). "Selecting Student-Authored Questions for Summative Assessments". In: *bioRxiv*.

Ihantola, Petri and Ville Karavirta (2011). "Two-Dimensional Parson's Puzzles: The Concept, Tools, and First Observations". In: *Journal of Information Technology Education* 10, 119–132. URL: https://jite.org/documents/Vol10/JITEv10IIPp119-132Ihantola944.pdf.

Karavirta, Ville, Petri Ihantola, and Teemu Koskinen (2013). "Service-Oriented Approach to Improve Interoperability of E-Learning Systems". In: *2013 IEEE 13th International Conference on Advanced Learning Technologies*, pp. 341–345. DOI: 10.1109/ICALT.2013.105.

Koc-Januchta, Marta Maria et al. (2020). "Engaging With Biology by Asking Questions: Investigating Students' Interaction and Learning With an Artificial Intelligence-Enriched Textbook". In: *Journal of Educational Computing Research* 58, pp. 1190 – 1224.

Ni, Lin et al. (2021). "DeepQR: Neural-based Quality Ratings for Learnersourced Multiple-Choice Questions". In: *ArXiv* abs/2111.10058.

Pollari-Malmi, Kerttu et al. (2017). "On the Value of Using an Interactive Electronic Textbook in an Introductory Programming Course". In: *Proceedings of the 17th Koli Calling International Conference on Computing Education Research*. Koli Calling '17. Koli, Finland: Association for Computing Machinery, 168–172. ISBN: 9781450353014. DOI: 10.1145/3141880.3141890. URL: https://doi.org/10.1145/3141880.3141890.

Rahdari, Behnam et al. (2020). "Knowledge-Driven Wikipedia Article Recommendation for Electronic Textbooks". In: *EC-TEL*.

Ritter, Steven et al. (2019). "What's a Textbook? Envisioning the 21st Century K-12 Text". In: *iTextbooks@AIED*.

Rößling, Guido et al. (2006). "Merging Interactive Visualizations with Hypertextbooks and Course Management". In: *SIGCSE Bull.* 38.4, pp. 166–181. ISSN: 0097-8418. DOI: 10.1145/1189136.1189184.

Sirkiä, Teemu (2018). "Jsvee Kelmu: Creating and tailoring program animations for computing education". In: *Journal of Software: Evolution and Process* 30.2. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/smr.1924.

Sosnovsky, Sergey A. and Peter Brusilovsky (2015). "Evaluation of topic-based adaptation and student modeling in QuizGuide". In: *User Modeling and User-Adapted Interaction* 25, pp. 371–424.

Vihavainen, Arto et al. (2013). "Scaffolding Students' Learning Using Test My Code". In: *Proceedings of the 18th ACM Conference on Innovation and Technology in Computer Science Education*. ITiCSE '13. Canterbury, England, UK: Association for Computing Machinery, 117–122. ISBN: 9781450320788. DOI: 10.1145/2462476.2462501.

Walker, Erin et al. (2017). "EMBRACE: Applying cognitive tutor principles to reading comprehension". In: *International Conference on Artificial Intelligence in Education*. Springer, pp. 578–581.

Walker, Erin et al. (2018). "Balancing student needs and learning theory in a social interactive postdigital textbook". In: *End-user considerations in educational technology design*. IGI Global, pp. 141–159.

Weber, Gerhard and Peter Brusilovsky (2001). "ELM-ART: An adaptive versatile system for Web-based instruction". In: *International Journal of Artificial Intelligence in Education* 12.4, pp. 351–384.

Xu, Ying and Mark Warschauer (2020). "Exploring Young Children's Engagement in Joint Reading with a Conversational Agent". In: *Proceedings of the Interaction Design and Children Conference*. IDC '20. London, United Kingdom: Association for Computing Machinery, 216–228. ISBN: 9781450379816. DOI: 10.1145/3392063.3394417.

Zingaro, Daniel et al. (2013). "Facilitating Code-Writing in PI Classes". In: *Proceeding of the 44th ACM Technical Symposium on Computer Science Education*. SIGCSE '13. Denver, Colorado, USA: Association for Computing Machinery, 585–590. ISBN: 9781450318686.