Assessment of Students' Conceptual Understanding: Unleashing the Power of the Interactive Textbook

Carrie Lindeman
Ithaca College
Department of Computer Science
953 Danby Rd, Ithaca NY, 14850
1-607-274-3583
clindem1@ithaca.edu

Toby Dragon
Ithaca College
Department of Computer Science
953 Danby Rd, Ithaca NY, 14850
1-607-274-3583
tdragon@ithaca.edu

ABSTRACT
Online, interactive textbooks are continually gaining popularity in computer science courses. We present a system for automated analysis that can harness the power of these textbook/practice systems in order to provide information about high-level conceptual understanding to educators. This information can be used to support individual students and to improve course content. The basis of our system is a Concept Graph, an artifact representing the concepts to be taught during the course, and their interrelations. We offer our working version of this Concept Graph for an Introductory Computer Science course, both as an artifact for consideration by the community, and as a basis for understanding our system. We describe the manner in which our system uses these Concept Graphs to provide useful information to educators.

CCS Concepts
• Social and professional topics ~ Computer science education •Applied computing ~ E-learning •Applied computing ~ Computer-assisted instruction •Applied computing ~ Computer-managed instruction

Keywords
Computer Science education; Intelligent Tutoring Systems (ITS); Expert Knowledge base (EKB); ELearning; Automated Assessment

1. INTRODUCTION
Online, interactive textbooks are gaining popularity in computer science education. These systems provide the benefit of allowing students to practice concepts as they are presented. The increase in out-of-classroom instruction can potentially allow for a blended learning environment, where students learn many of the basic principles from the textbook and practice environment. Educators can then focus more class time on collaborative work, discussion, and more challenging subjects. However, the educator gains less information about their students' grasp on these core concepts that are learned at home.

Often, these online interactive systems are accompanied by a minimal set of tools that allow educators to see the submitted answers to questions, and aggregated information about the class as a whole. However, the systems rarely offer the educator any indication of students' understanding of high-level concepts, as the systems do not generally include any type of information that would facilitate that process.

SIGCSE '16, Month 1–2, 2016, City, State, USA.

DOI: http://dx.doi.org/10.1145/12345.67890

We present a system that offers information about students' understanding of high-level concepts based on their work within an online interactive system. This information is not a magic solution, but rather a system that allows educators to specify the important concepts, their interrelation, and their relation to the exercises that students complete. In this way, we combine the educator's understanding of the course material with the data from the online practice environment. This tool supports educators in their attempts to help individual students and to improve their curriculum overall.

We now present a scenario to offer the reader a glimpse into the potential uses of the tool (Section 2). We then present the underlying technology that provides the functionality described (Section 3). We discuss interesting implications of this work, including another scenario demonstrating the use in improving course content (Sections 4 and 5). We discuss our grounding in ELearning theory (Section 6), and conclude by describing our plans for using and improving the system (Section 7).

2. ASSESSING AND SUPPORTING STUDENTS
Here we describe a hypothetical scenario to demonstrate our system's potential. This scenario describes the expected context, manner of use, and productive outcomes of the system, allowing the reader to see the purpose of the tool and the manner in which the tool supports assessment of student performance.

Imagine Professor Sinclair, the instructor of an Introductory Computer Science course, grading quizzes from the past week. She notices that her student, Jeffery, who has generally performed well, failed this week's quiz. She decides to consult our automated textbook assessment tool to see if she can identify Jeffery's misconception. When she opens the tool, she focuses on the visualization of Jeffrey's knowledge about the current topics (Figure 1).

This quiz was focused on the use of if statements and the professor notices that Jeffery's assessment reflects a mediocre score for the concept *If Statement*. However, she also notes that the associated concept for *Boolean Expressions* has an even worse rating. The system reflects the relationship between these two concepts, in that a problem with the concept of boolean expressions contributes to a problem with the concept of if statements. Therefore the rating on *Boolean Expressions* (red)

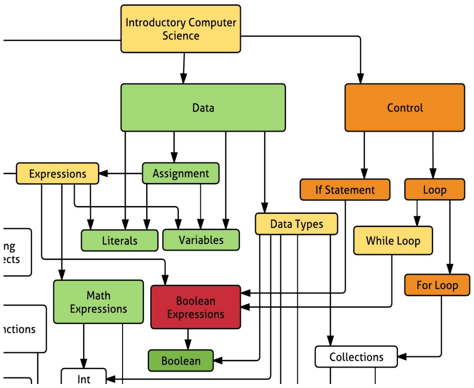

Figure 1. The information shared with the instructor as the she examines a specific student assessment.

lowers the rating of *If Statements* (orange) and eventually also lowers the rating of *Introductory Computer Science* (yellow). She clicks the *Boolean Expressions* node to see with which related textbook exercises Jeffrey has struggled. A pattern in Jeffrey's answers reveals a clear misconception of boolean expressions. After this quick and pointed use of the assessment tool, Professor Sinclair is able to give effective feedback to Jeffrey about his specific misconception.

This basic scenario describes one use case of our assessment tool, and demonstrates a practical application for an educator. The tool provides the ability to quickly see an overview of the concepts presented in a given class, along with assessment of a given student's theoretical understanding on those concepts based on real data from student performance. We now describe in detail the technology and the content development necessary to make this tool function as described in this scenario. We can then discuss the additional benefits and alternative uses of this system.

3. Underlying Technology

Figure 1 represents a partial graph of certain concepts covered in an Introductory Computer Science course and their interrelations. Our system requires such a graph of the concepts covering the entire course in question. We refer to this graph as a *Concept Graph*, which is a type of Expert Knowledge Base (EKB). EKBs are a common tool used in Intelligent Tutoring Systems (ITSs) research [1]. We use this Concept Graph to automatically analyze the information collected from an online textbook in order to produce assessment of high-level concepts based on low-level information about student performance. This high-level information can be used by educators to quickly and easily offer pertinent feedback to students, as described in the scenario above. There have been other attempts at similar assessment techniques, particularly Butz et. al. [2]. However, to our knowledge these attempts have been left undeveloped and unused in real classrooms.

We now present the details of our system in order to clarify the inner workings of its assessment mechanism. We must consider the online learning tool from which it takes low-level student performance information, the structure and content of the Concept Graph, and their interconnection. Then we can consider the analysis system that aggregates data and offers assessment of student performance based on the relevant Concept Graph.

3.1 The Learning Environment

The learning environment used by the students is an online, interactive textbook system called Runestone Interactive[1] [3]. This system presents mixed media for instruction (text and video) interleaved with practice exercises (Figure 2). Exercises include multiple choice questions, drag and drop questions, a live Python debugging tool, and text areas where Python code can be written and executed.

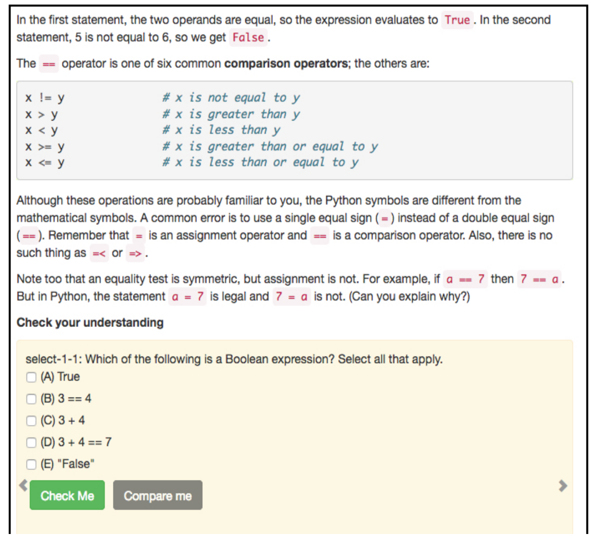

Figure 2. A screenshot showing the learner's view of the Runestone Textbook.

Specifically, we are focused on the flagship textbook of this system "How to Think Like a Computer Scientist," which presents introductory programming in Python. This textbook system is used in our curriculum to promote blended learning, where students work independently online as well as in the classroom. Students are expected to read and complete exercises that introduce a topic for homework. They attend class where the topic is covered in more detail, and finally return to the text to complete more challenging exercises for practice and to help cement ideas.

The content of the textbook (including the exercises) is defined by text files, and therefore developing and changing content is technologically simple. The system logs all user actions to a database, and provides these data in XML format upon request. We use these data as the input to our automated analysis system.

3.2 The Concept Graph

The purpose of the Concept Graph is to make explicit all concepts that are expected to be taught in class, as well as the relationships between these concepts. These concepts could be at any level of granularity that the educator chooses. The end product is a Directed Acyclic Graph (DAG). Below is the current version of our concept graph designed for our Introductory Computer Science course in Python (Figure 3). Nodes in the graph are concepts taught in the course, and edges represent roughly the relationship "is a part of." For example, the node for *If Statement* has an edge connected to *Boolean Expression* because an understanding of boolean expressions is a part of an understanding of if statements.

[1] http://runestoneinteractive.org

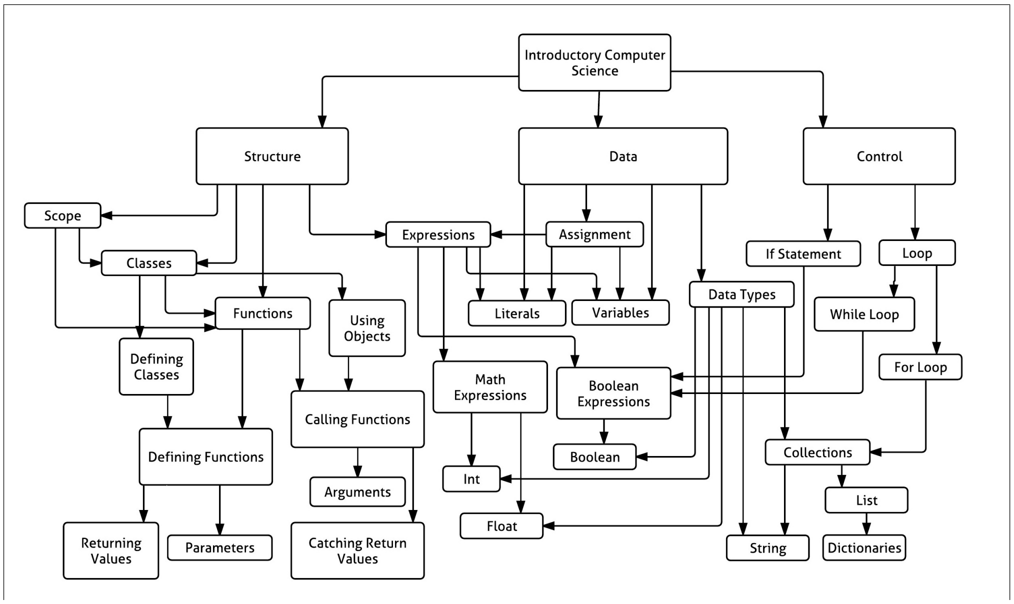

Figure 3. The Concept Graph the subject Introductory Computer Science.

Nodes without parents are generally high-level, abstract concepts, and nodes without children are generally low-level, concrete concepts. The graph transitions from concrete to theoretical because the graph is assessing a student's understanding of an abstract concept based on their estimated understanding of lower-level concepts. These estimates originate from actual student performance data provided by the textbook. The means by which the textbook data is connected to the concept graph are discussed in Section 3.3.

Note that we refer to this artifact as the "current" version of the concept graph. The creation of such an artifact is obviously subjective and prone to debate, even within a single team or by a single author. The concepts can be at any level of granularity, and the decision of when such a graph is "correct" or "complete" is difficult. This reflects the flexibility, subtlety, and challenge that every educator must face when deciding the content and organization of a given course. However, there is no need for this artifact to be perfect, and it can be iteratively improved with ease. The graph itself is defined in a text file (in JSON format) by a list of nodes and a list of edges, and therefore is easily editable. The real demonstration of a "correct" graph is the usefulness when an educator employs the system in the realistic setting of their class.

Considering this complex authoring task, any educator considering the creation of this graph might be (rightly) concerned about the effort involved and the ambiguity of its form. To address this concern and offer a simple example solution, a default graph can be automatically built from the textbook source files of any text authored in the Runestone system. A textbook is inherently structured into chapters and subsections. A graph can then be created simply by using the names of these chapters and subsections to create the graph structure (Figure 4).

This automation offers default functionality of the assessment system without need of further authoring work. The default structure can then also be modified by the author to suit their needs. The trade-offs of using an automatically constructed Concept Graph versus editing or constructing one's own Concept Graph are discussed in Section 4.1.

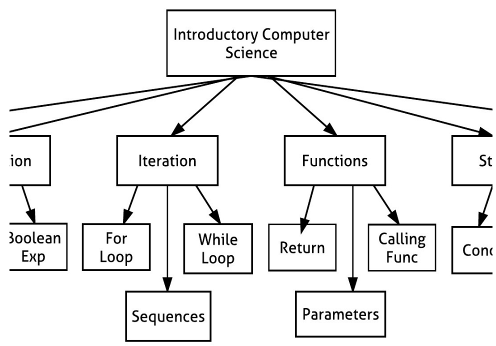

Figure 4. A portion of the automatically generated concept graph from the Runestone Textbook.

3.3 Connecting Concept Graph to Data

As previously stated, the exercises in the learning environment are defined by text files, and data from these exercises are recorded in a database and retrievable in an XML format. These data must now be linked to nodes in our Concept Graph to assess the student's understanding of high-level concepts. This connection can be made in two different ways.

The first and simplest way is to rely on the automated Concept Graph created from the textbook source files. When parsing the source files of subsections, the questions within those subsections can automatically be linked to the corresponding nodes. In this way, a fully functional Concept Graph including connections to the data source can be automatically created.

When a Concept Graph differs from the textbook source, the connection from exercises to the nodes in the graph must be specified. This is accomplished by tagging the exercises in the text source with the names of concepts from the Concept Graph. Authoring then is the rather intuitive task of tagging each exercise in the text with the concepts from the graph that are relevant to that exercise. The exercises are generally connected with lower level concepts, but the system allows for exercises to be connected to any node in the graph. Questions can even be tagged with high-level concepts such as data types. For example, this might be appropriate for a question that asks students to differentiate many different data types.

Again, the trade-offs between automation and manual authoring are discussed in section 4.

3.4 The Automated Assessment System

Now that the data source, the Concept Graph, and their connection are established, we consider the manner in which we use this structure to offer automated assessment. We employed, altered, and extended open-source software originally developed for an unrelated project, the Metafora Project[2]. This software was developed to conduct automated analysis across many different computer-based learning tools [4], and therefore provides a basis for collecting and analyzing individual actions in XML format [5]. It offers a web-based framework (built on Google Web Toolkit, GWT[3]) in which analysis can be coded in java on the server-side and all available data can be viewed, filtered, and explored in a web client. These data range from the low-level actions by the student to high-level analysis performed by the system.

Algorithmically, the system reads the concept graph structure from either JSON or the book source (as described in Section 3.2) and creates a representation in memory that includes leaf nodes for the specified exercises related to the graph (as described in section 3.3). The system collects XML data from the book about a given student's performance on exercises and populates the leaf nodes with these data. The system then uses a post-order traversal of the concept graph to aggregate actual data in a bottom-up fashion from the lowest levels of the graph to all nodes in the graph. Finally, the system uses a pre-order traversal of the graph to distribute the aggregated information in a top-down fashion, offering estimated knowledge on concepts about which real data has not yet been collected. Currently, these calculations are simple weighted averages. Improved techniques using mathematical modeling and machine learning techniques to produce more accurate results are a matter of current research.

These calculations result in a numerical value for each node, which we then convert into a color code to be presented to the educator. Using Google Charts[4] tools, we display an org chart where each node is labeled with its concept title, has a colored border to represent the node score, and can be clicked to display all related data. Color codes simplify the overwhelming amount of data available to the educator. Such a simplified interface has been shown to allow teachers to identify needed information in a short amount of time [6]. The web interface does allow for teachers to further analyze the source of these color codes, as described in Section 2.

Taking a holistic view of the system, one can now see the manner in which this tool utilizes the textbook system, the data from student use, and the defined Concept Graph to offer the behavior described in the scenario presented in section 2. This system provides educators a powerful tool to peruse the estimated understanding of any given student based on their work within the online textbook for the class, and to drill down into that estimate to understand the source of the estimation.

[2] http://www.metafora-project.org/

[3] http://www.gwtproject.org/

[4] https://developers.google.com/chart/

4. AUTHORING VERSUS AUTOMATION

Throughout section 3 we explained the manner in which the Concept Graph and its connections to individual questions could be created in two different fashions: either automatically or manually. It is worth noting again that an author need not choose only one or the other, but can begin with an automated Concept Graph and edit it to suit. We now discuss this decision in more detail to highlight underlying interesting implications of our system, namely: the contrast of a graph versus a tree structure, and the contrast of hand-crafting a Concept Graph versus using automation.

4.1 Graph Versus Tree Structure

A key difference between a hand-crafted Concept Graph and one that is automatically generated from the book is that the auto-generated graph will always be a tree structure (each node having only one parent), as seen in Figure 4. This is because no question is featured in multiple subsections and no subsection is featured in multiple chapters. In contrast, the nodes in a handcrafted Concept Graph might have multiple parents. For example, in Figure 3, the node for *Boolean Expression* has both *Expressions* and *If Statements* as parent nodes. This difference in structure has potentially profound effects on the resulting calculations, as it can reveal a learner's progress over time. The node for *Boolean Expression* may have a score that fluctuates based on the learner's current work. The learner may not understand boolean expressions at first, but gain confidence when working with if statements. In a handcrafted Concept Graph the progression is recognized in the visualization because many different sections of the text might be related to Boolean Expressions. In an auto-generated Concept Graph, there is only one section of the book that affects this node, making a more static timeline of their performance.

Our automated analysis system was designed to handle any DAG, so calculations for assessment can function regardless of whether the graph is a tree or not. However, a clean and understandable automated display of a DAG is a much more difficult task than automated display of a tree structure. Off-the-shelf free software (in our case Google Charts), offers useful web visualization of tree structures, but we have not found similar software for displaying a DAG.

We have mitigated this display problem by creating a function that converts a DAG into a tree structure. This works by duplicating and incrementing an ID number for any nodes with multiple parents. This tree structure is used only for display purposes, and allows organized visualization of our Concept Graphs (Figure 5).

As can be seen in the figure, this tree is useful, but can easily become large and repetitive, making this solution less than ideal. Better software for DAG visualization is an area of active investigation.

4.2 Improving Content by Authoring

Setting aside the graph vs. tree aspect discussed in Section 4.1, there are still trade-offs to consider with automated graph creation. The convenience of having an automatically generated Concept Graph from the textbook structure is apparent. However,

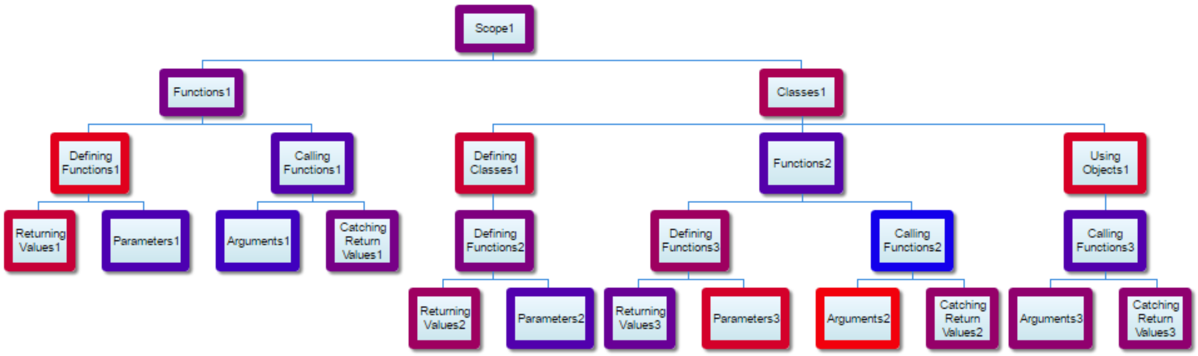

Figure 5. A subsection of our EKB Concept Graph converted to a tree and displayed in our assessment tool.

we have found that hand crafting a Concept Graph has the potential to improve the overall course content. The process of creating a concept graph and linking each exercise in the book helps flesh out the content presented to students and the learning goals for the course. Creating the concept graph can reveal an imbalance of exercises to concepts, even potentially revealing the absence of certain desired concepts from the textbook entirely. In this way, making a concept graph is an exercise in polishing course content and ensuring that the content is driven by desired learning outcomes. We now offer a final scenario to exemplify this potential iterative improvement of course content.

5. Content Improvement Scenario

Imagine again Professor Sinclair, who has been teaching Introduction to Computer Science at her institution for many years. On recent exams, she has noticed that almost all of her students get one question wrong, a question involving dictionaries. While she is aware of her limited time covering this subject, she has always relied on the textbook chapter to fill in gaps for the students. She decides to investigate how she can improve the student understanding of dictionaries.

She decides to look at the class average Concept Graph for last year's class. Since most students got the dictionary question wrong, she assumed there would be a poor score for the *Dictionaries* concept in the Concept Graph. However, there was a positive average score. She is confused as to how this is possible, and therefore decides to compare the question she used on the final with the questions used for the book assessment. She finds that the textbook does not feature any questions that require the student to initialize and populate a dictionary. In contrast, the question on the final asks students for precisely this task. Professor Sinclair now realizes that she has been testing students on something that she had never had them practice. With this specific knowledge she is able to adjust the course content, Concept Graph, and exercises in the textbook for future learners. This theoretical improvement can then be tested in subsequent versions of the course.

This scenario describes a realistic use of our system to aid the educator in refining course content, rather than aiding an individual student. Our system can provide average student data across an entire course in this Concept Graph form, allowing for assessment on a macro-level. This approach can potentially aid in improving the course itself, and thereby improving all students' classroom experience.

6. ELearning Theory

While we consider the rationale behind our system intuitive, it is important to note that we ground our work in ELearning educational theory. Specifically, we refer to the work of educational theorist Robert Gagné and his nine events of ELearning [7]. Our learners are more likely to be successful because we have molded our approach to follow proven principles. We briefly describe the manner in which our system follows these nine events in Table 1.

Table 1. Table captions should be placed above the table

Event	Our System
Gaining attention	The textbook has engaging videos, live code windows, and interactive exercises.
Stimulating recall of prior knowledge	Viewing the concept graph shows the learner what concepts they have already successfully completed.
Presenting information	The textbook itself is the format for presenting content.
Providing guidance	Our future work includes having the textbook suggest exercises and chapters to review based on performance.
Elicit performance	The exercise questions and practice problems prompt the learner to engage.
Providing feedback	Each question gives feedback and explanations based on the learner's selection.
Assessing performance	The system calculates representations of their understanding of different concepts based on their performance and provides visualization in the form of a concept graph that is color coded to reflect level of understanding.
Enhancing retention and transfer	The assessment tool prompts the instructor to make adjustments to the course based on class-wide misunderstandings and the assessment prompts the learner to change their path based on performance.

7. CONCLUSIONS AND FUTURE WORK

Our system utilizes an online, interactive textbook to offer high-level assessment of student understanding. The specific examples given demonstrate the ways in which this system can aid educators in assisting individual students and improving the overall course content. The system is developed to the point where it is useful for the scenarios described above, and we look forward to putting it in the hands of real educators for use with their courses in the coming year.

However, we also see this work as the foundation of a much larger system that can provide further support for Computer Science education. Future directions include increasing the system's ability to understand different types of exercises, providing automated intervention, and creating dynamic collaborative groups based on students' current understanding. We explain each of these briefly, to give the reader a concept of the overall potential of this system.

A clear next step for the project is to expand the sources of information about student performance. Our current system is limited in the type of exercises from which it can glean information. The first version of the system relies mainly on multiple choice and other easily graded question types. However, the textbook also has interactive code windows and programming exercises that are currently hand graded by the instructor. We are aware of the body of research in automated assessment of code snippets and full programs e.g., [8]. Ideally, we plan to integrate this type software with our system via additional XML messages. Receiving assessment on these more open-ended tasks has the potential to drastically improve our estimates of a student's conceptual and procedural understanding.

Beyond collecting more data to improve estimates, our system also has the potential to provide automated adaptation and/or feedback based on these estimates. This would bring our system fully into the realm of ITS. As described above, the first version of our system relies on the instructor to provide intervention. A future goal is to automate certain types of intervention, easing the load on the educator and helping students when an educator is not readily available. The system could directly interact with students, offering suggestions of content to review and exercises to practice. The Concept Graph already contains the necessary mapping to provide these suggestions. The system identifies problem areas for a student (labeled in orange or red on their Concept Graph). The next step is to traverse the graph to reach appropriate leaf nodes, which represent content that the student should review. The greater task at hand is to create a user interface that integrates with the online textbook to present these suggestions in a useful and intuitive manner. In a similar fashion, we could consider macro-adaptation, where the book's content is altered in order to present only the information for which the user is most prepared. A large body of research in such prior work is available to leverage e.g., [9].

As a final direction of planned future work, we also see potential for our system in the realm of Computer Supported Collaborative Learning (CSCL). Particularly, we recognize the importance and challenge of group formation [10], and we see the potential to offer interesting results in the subfield of dynamic group formation. Specifically, we plan to use the information available from the Concept Graphs of different students in order to pair students that could work productively together [11]. We can also use the specific portion of the Concept Graph that prompted the pairing to suggest exercises that would be productive for the pair. This grouping concept is an active area of research for our team, and we plan to pilot test pairings within the year.

8. ACKNOWLEDGMENTS

<Anonymous>

9. REFERENCES

[1] Woolf, B. P.: Building intelligent interactive tutors: Student-centered strategies for revolutionizing e-learning. Morgan Kaufmann. (2010)

[2] Butz, C. J., Hua, S., & Maguire, R. B.: A web-based bayesian intelligent tutoring system for computer programming. Web Intelligence and Agent Systems: An International Journal, 4(1), 77-97 (2006)

[3] Miller, B. N., & Ranum, D. L.: Beyond PDF and ePub: toward an interactive textbook. In: Proceedings of the 17th ACM annual conference on Innovation and technology in computer science education, pp. 150-155. ACM. (2012)

[4] Dragon, T., Mavrikis, M., McLaren, B. M., Harrer, A., Kynigos, C., Wegerif, R., & Yang, Y. (2013). Metafora: A web-based platform for learning to learn together in science and mathematics. *IEEE Transactions on Learning Technologies*, *6*(3), 197-207.

[5] Dragon, T., McLaren, B. M., Mavrikis, M., & Geraniou, E. (2011, July). Scaffolding collaborative learning opportunities: integrating microworld use and argumentation. In *International Conference on User Modeling, Adaptation, and Personalization* (pp. 18-30). Springer Berlin Heidelberg.

[6] Mavrikis, M. and Gutierrez-Santos, Sergio and Poulovassilis, Alexandra (2016) Design and evaluation of teacher assistance tools for exploratory learning environments. In *Proceedings of the Sixth International Conference on Learning Analytics & Knowledge (LAK'16).* ACM, pp. 168-172. ISBN 9781450341905.

[7] Gagné, R. M., Briggs, L. J., & Wager, W. W. (1992). Principles of instructional design (4th ed.). Forth Worth, TX: Harcourt Brace Jovanovich College Publishers

[8] Pieterse, V. (2013, April). Automated assessment of programming assignments. In *Proceedings of the 3rd Computer Science Education Research Conference on Computer Science Education Research* (pp. 45-56). Open Universiteit, Heerlen.

[9] Hsiao, I. H., Sosnovsky, S., & Brusilovsky, P. (2010). Guiding students to the right questions: adaptive navigation support in an E-Learning system for Java programming. *Journal of Computer Assisted Learning*, *26*(4), 270-283.

[10] Dillenbourg, P. (2002). Over-scripting CSCL: The risks of blending collaborative learning with instructional design. *Three worlds of CSCL. Can we support CSCL?*, 61-91.

[11] Dragon, T., Lindeman, C., Wormsley, C., & Lesnefsky, D. (2016). Better than Random: Can We Use Automated Assessment to Form Productive Groups on the Fly? In *the 13th International Conference on Intelligent Tutoring Systems (ITS-2016), Workshop on Intelligent Support for Learning in Groups*.