# Collaborative Research: Integrating the eTextbook: Truly Interactive Textbooks for Computer Science Education

(TUES Phase I Proposal: May 27, 2011)

## Project Summary

While there is much evidence that Algorithm Visualizations and similar interactive courseware is pedagogically effective, survey data point to disappointing levels of adoption among CS instructors. This is due to many impediments that they experience when trying to add such courseware to their existing courses. By providing integrated solutions for semester courses in the form of complete textbooks combined with interactive content, we make it possible for faculty to adopt such solutions whole-sale, which is how instructors often approach teaching new courses or topics. Thus, the clear benefits of interactive courseware will become available to many more students once it is made available in the form of integrated eTextbooks. Once exemplars of such eTextbooks exist, momentum should build for a wider group of developers to join the process.

Our vision goes beyond simply creating an interactive eTextbook. We embrace a Creative Commons for textbook creation, not just in terms of licensing but in terms of allowing for community creation of artifacts. The goal is to allow instructors to modify existing eTextbooks by adopting major portions and then changing sections, or taking text and visualizations from different books and combining them. Such community construction of textbooks follows the vision of the ongoing Connexions project. We will work with the Connexions team to enhance or supplement their infrastructure so that Connexions is better able to handle the type of highly interactive eTextbook that we envision. An important component of the enhancement that integration can bring is the relationship between text book and assessment feedback. When the system provides frequent assessment questions, students and their instructors can better know that they are on track.

This integration of five concepts (online textbook content, interactive courseware, collaborative creation, open source, and online assessment) will benefit four different stakeholders – students, instructors, content authors, and algorithm visualization developers – by seamlessly allowing them to share and use materials. Together, these five elements define the *visualization-based computer science hypertextbook*, which we broadly define as an online learning resource merging traditional text-based descriptions of Computer Science concepts with pedagogically sound interactive visualizations and explorations that lead learners into a highly interactive level of engagement with the material. The emerging HTML5 standard makes the eTextbook more technically feasible, both in terms of creation and ease of adoption.

The outcome of this project will be a "storyboard" defining a detailed roadmap for an entire hypertextbook on data structures and algorithms, an end-to-end framework for the development process, and several complete prototype sections. This topic is especially suitable for an interactive textbook since it can take obvious advantage of interactivity to let students explore and manipulate the dynamic processes that are at the heart of data structures and algorithms instruction. The PIs have extensive text available already due to prior textbook-writing efforts, and extensive access to open-source visualizations written by ourselves and by various collaborators. We hope to rally the algorithm visualization developer community to collaborate on this project.

**Intellectual Merit:** This work will improve the teaching of courses for which we create relevant content, and will also impact the development of future eTextbooks by demonstrating successful ways to integrate text, interactivity, and assessment in a creative commons environment.

**Broader Impact:** Interactive hypertextbooks are valuable beyond Computer Science. Research results indicate that online instruction in many fields can be enhanced by student interaction with well-designed simulations. Our efforts will provide an exemplar of how collaborative, open-sourced workflows could be used to develop hypertextbooks for many disciplines.

## 1 Problem Statement

The dream of an electronic textbook (sometimes referred to as a hypertextbook) has been actively pursued for at least two decades. Goals include (i) improving exposition through a richer collection of technologies than are available through print textbooks, and (ii) increase student engagement with the material, in order to get them to learn at a higher level in Bloom's taxonomy [31]. Putting standard textbooks online is not itself a game changer for education. Having your textbook on a computer or eBook reader does make it more portable and more convenient, and makes it easier to update more frequently than print versions. But that is not enough to change fundamental student interactions. The next natural step has been to take advantage of multimedia, and to augment textbooks with movies, animations, and audio where appropriate. But while these additional information modes are important, this also is not sufficient to fundamentally change student behavior. What is lacking with both innovations (eBooks and multimedia) is that they do not change the passive relationship between students and textbooks.

The real game-changing potential that the computer brings to education is its ability to allow the student to interact in non-passive ways with information, and the ability of the computer to provide immediate feedback to the student based on these interactions. The computer can allow students to work with interactive simulations, to build their own models and test them, to manipulate virtual entities, and to structure their knowledge by creating representations for it. Instead of merely viewing material, we can hope to use frequent assessment (by asking questions) to get students to respond, and through interactive activities get them to change and construct virtual artifacts. See [36, 35] for background on efforts to define and implement this vision of a hypertextbook.

Data Structures and Algorithms as a topic can particularly benefit from the use of advanced technology. Much of the content that students must learn in such courses involves the detailed workings of dynamic processes, such as sorting and searching algorithms. Dynamic process is extremely difficult to convey using static presentation media such as text and images in a textbook. During lecture instructors typically draw on the board, trying to illustrate dynamic process through words and constant changes to the diagrams. Many students have a hard time understanding such explanations at a detailed level. For this reason, there has long been great interest among CS instructors in the use of animation and visualization to convey dynamic concepts. Our particular focus is on the use of algorithm visualization (AV) [45, 31] as a means both to deliver the necessary dynamic exposition, and to increase student interaction with the material.

## 2 Motivation

Periodic surveys on both instructor interest in AVs and their level of use have been conducted among attendees of education conferences and listservs for more than a decade. For instance, [31] reports on three surveys conducted in 2000. Collectively, the surveys indicate a strong positive view in favor of AV use (over 90%). However, only about 10% of respondents at that time indicated frequent use of AVs in data structures courses at their institutions, while about half to three quarters indicate occasional use of AVs in such classes.

At SIGCSE'10 (held in Milwaukee during March 2010) we conducted a new survey to determine instructor attitudes toward AVs and their use in the classroom. For details, see [44]. The survey was designed in the spirit of the 2000 surveys reported in [31], and the findings are consistent with those earlier results. In 2010, 41 of 43 respondents agreed or strongly agreed that AVs can help learners learn computing science, with two neutral. However, just over half had used AV in a class within the past two years. It is not clear that all of the "yes" answers from the present survey refer to what we might consider to be "AVs." But they probably all refer to some sort of dynamic software visualization run on a computer, as was the case in the 2000 survey. We did not ask about frequency of use of AVs in class.

Our third survey question asked what respondents see as the greatest impediments to using AVs in courses. Roughly half the responses relate to finding suitable AVs to use. We can hope that the presence of newer online resources such as the AlgoViz Portal (`http://algoviz.org`) will reduce this information gap, by making it easy for instructors to locate quality AVs. However, about half of the responses relate to issues in integrating AVs into the courses. These results are roughly the same as reported in [31, 27, 36]. These issues are much harder to deal with, and are representative of well-known problems for adoption of educational technology in general [15]. While it is easy to give students pointers to AVs as supplemental material, it is much harder to integrate AVs into lectures, labs, and assignments. This comes as a result of factors like lack of time on the instructor's part to make course changes, lack of time within the course to spend additional time on a given topic, and inconsistencies between the course textbook and the AV.

These survey results lead us to the conclusion that it is easier to integrate a complete block of instruction (either a complete topic or even a semester course) than it is to fit a piece of instruction such as an AV into existing presentation on that topic. Instructors are used to the concept of adopting a new textbook for new courses that they teach, and often welcome lecture notes and other class support artifacts. Even for courses taught previously, instructors will adopt new textbooks and new presentations for various topics in the course. A key aspect is that adopting a new chunk of content allows the instructor to completely replace or populate a block of course time, as opposed to squeezing new content or presentation techniques on top of existing material. In the past, most AV developers have implicitly taken the approach that their AV will be integrated into existing presentations. A typical example might be an AV presenting a Quicksort algorithm, with no tutorial explanation of the algorithm. The idea seems to be that this visualization can be slipped into lecture or used by students to supplement the textbook for self-study. But this approach has led to the problems noted above. In contrast, a complete unit of instruction (including AVs) can more easily replace an unsatisfactory existing presentation of the topic. In this way, instructors are able to bring into their courses the new pedagogy provided by AVs' better presentation of dynamic processes.

Important technological factors contribute to making AVs easier to use in classes than in prior years. More AVs are available than ever before [45], backed up by improved research studies and improved access both through general Internet search and at the AlgoViz Portal. Increased access to the Internet by students and instructors, both in and out of the classroom, makes AV use more practical. For example, at Virginia Tech, while it has been "possible" to project Internet material from a computer in class for over a decade, it has only been in the past couple of years that such access has become both ubiquitous and reliable in all of our classrooms. This makes a huge difference in the confidence of mainstream instructors for using such technology, in contrast to early adopters [15].

Another potential factor in favor of the use of AVs and hypertextbooks is ubiquitous availability of laptops and mobile devices. For example, all Engineering majors at Virginia Tech are required to own a tablet PC, and most also have mobile devices including smartphones, eBook readers, or iPads. However, there is also a downside to the non-PC devices, in that they have various technology limits on how eTextbook content can be provided. So while there is ubiquitous access to the Internet among our target audience, there are still limits. For example, Java applets cannot be displayed on most mobile devices. However, the new HTML5 standard helps Internet content providers to more easily support the broad range of mobile devices.

## 3 Prior Work

Programmed Learning, Programmed Instruction, Keller plan, and Integrated Learning System are concepts that formed the foundation for Computer Based Training (CBT) in the early to mid 1980s. They are now marketed by Thomson/Course Technologies as the NETg series (National Education Training Group). These are non-facilitated models of instruction used by industry (HR departments) for employee training and by the ASTD for employee professional development. The concept was originally based on B. F. Skinner's work. CBT is also a forerunner to Human Performance Technology (HPT).

The bulk of effort by CS educators involved with interactive AVs has focused on development of the AV technologies and the AVs themselves. At best there have been small, focused hypertextbook modules that incorporate AVs into a particular topic or small set of topics within a course. For example, [13] reports on a set of web pages integrated with applets that were used in parts of a theory of computation course. Many instructors who teach theory now use Susan Rodger's JFLAP [12] throughout much of their course and Rodger has published a hard copy guide to using JFLAP [33]. In the preface Rodger warns the reader "our book assumes that the reader has read briefly about these topics first in an automata theory textbook or a compiler textbook".

[34] describes efforts at building a Perl-based infrastructure for creating hypertextbooks. The infrastructure relies on the Dreamweaver toolkit. Only a few chapters (three out of seven) of a theory of computing and a biology book has been produced using this technology. The system requirements for these hypertextbooks also hinder their wide adoption due to browser restrictions, use of Java applets, and specific screen resolutions.

[38] reports on a technology that integrates AV into Moodle-based lessons, but the emphasis is again on a technology that would support visualization-based hypertextbooks in Moodle, and to our knowledge little progress has been made on the actual writing of such a textbook.

Titterton, Lewis, and Clancy [49] have used a lab-centric mode of instruction for introductory CS courses at UC-Berkeley. Their current work is being done in Moodle and uses a small amount of AVs along with many other "check point" exercises that students must complete as they progress through a set of material presented in Moodle. It is built upon an earlier technology called UC-WISE that was developed and used exclusively at UC-Berkeley. Their materials are designed for introductory CS courses that aimed mostly at developing students' programming skills. Hence they make only limited use of AVs. Alharbi, Henskens, and Hannaford [2] are using eXe, an open source authoring system that allows instructors to create academic-related web content using XML and HTML (`http://www.http://exelearning.org/`). Their efforts intend to help teach Operating Systems using visualization. Learners can interact with the AVs, and can take quizzes and tests online. They used the Sharable Content Object Reference Model (SCORM: `http://www.adlnet.gov/`) to support integration of digital teaching materials with a CMS (Moodle in their case).

Another effort to integrate AVs into hypertext is being performed by Karavirta [16]. His solution is based on HTML and JavaScript, allowing the hypertextbook to be viewed in any browser without additional plug ins. The learner can interact with the animation and draw annotations on it, but the current system does not store the annotation, nor does it support quizzes and tests.

Largescale use of educational hypermedia in South Korea provides interesting feedback about the challenges of eTextbooks. [18] identifies usability, portability, interactivity, and feedback as major elements to consider while designing such systems. Learners should be able to ask questions and receive help, as well as control, manipulate, search and browse the eTextbook content. They also advocate for the development of models to support group collaboration.

To our knowledge, no one has written a complete electronic textbook, tightly integrated with AVs, that could be used as the primary learning resource in a semester-long computer science course. This is perhaps surprising because Marc Brown's groundbreaking dissertation on AV from 1988 [9] issued the following caution:

> Much of the success of the BALSA system at Brown [at the time Brown's thesis was written] is due to the tight integration of its development with the development of a textbook and curriculum for a particular course. BALSA was more than a resource for that course – the course was rendered in software in the BALSA system.

Why have CS educators involved with AV not heeded Brown's warning and authored textbooks that effectively incorporated interactive AV learning resources? Why have they instead focused on development of the AV technologies themselves? We suspect the answer is something known to anyone who has written a textbook or an AV: it consumes huge amounts of time. Combining the complications of integrating AVs into a textbook project can make the task seem daunting. It becomes easier to write AV software than it is to write a hypertextbook that integrates the software into a complete courseware package that can be disseminated broadly to the CS education community. Another impediment has been technical. Java, JavaScript, and Flash collectively were a big step forward in providing cross-platform interactivity. HTML5 provides a further lowering of technical hurdles, making the development task less daunting.

Project PIs Shaffer and Naps have a nearly unique set of resources and experience to make this project feasible. We have each in the past authored a textbook related to Data Structures and Algorithms [42, 43, 29, 24, 32]. We both have rights to reuse the contents of these textbooks for this project, so effectively we already have available a complete set text and image resources right from the beginning. We have both created many AVs in the past. Naps is the director of the JHAVÉ project [26], so he is not only responsible for developing many AVs, but also has great insight into developing an AV-building system. Shaffer is director of the AlgoViz project [1], a gathering place for users and developers of algorithm visualizations and animations (AVs). It is a gateway to AV-related services, collections, and resources. Naps has coordinated two ITiCSE Working Groups on Algorithm Visualization [31, 27] and was involved in another ITiCSE Working Group on hypertextbooks [36]. Thus, between us we have active collaborations or extensive interactions with nearly all of the major AV developers in the world. As described below, we hope to leverage this extensive network of developers to help with this project.

## 4 A Case Study

In this section we briefly describe an online hashing tutorial and a study conducted to determine its pedagogical efficacy. These results have helped us to make progress toward defining the requirements for a more comprehensive electronic textbook. The tutorial itself has proved to be popular, with the tutorial homepage visited (based on a sampling of our log files) about 600-700 times per month from users outside of Virginia Tech.

In 2008 and repeated again in 2009, students in separate sections of a sophomore level course on data structures and algorithms at Virginia Tech were taught about hashing using two separate treatments. One section was given standard lecture and textbook for one week, similar to what had been done in previous offerings of the course. The other section spent the class time working through an online tutorial combined with algorithm visualizations to present the same material. The tutorial used text content taken from the course textbook, so that it was an exact match to the material being presented in the control section. However, the online tutorial heavily supplemented this text with algorithm visualizations. The tutorial can be found online at `http://research.cs.vt.edu/AVresearch/hashing`.

In each of the trials, the two sections were given a quiz on hashing at the conclusion of the week of instruction. The results were positive: significant differences in performance were obtained in favor of those who used the online tutorial versus standard lecture and textbook. This means that not only can an online tutorial be as effective as an instructor (which has important implications for distance learning), but that providing proper interactivity allows computerized instruction to be better than lecture-based (passive) instruction. However, there is at least one major consideration that might influence the results of this particular study: How much impact did the controlling structure of coming to class and doing the tutorial in "lab" setting have on the results? The outcome could be quite different for a student just reading the material and working through the visualizations on their own, where self-discipline might well not be sufficient to provide the necessary amount of time and attention. Likewise, the controlled environment of attending lecture before reading the textbook on one's own is also likely to have a major difference compared to just reading the book on one's own.

We hypothesize that, if we want to have a viable self-contained electronic textbook that supports self-study of the material, the system needs to build in some equivalent to the controlled pacing and feedback that is encountered when attending classes or labs. This means that assessment and an objective measure of "progress" through the material needs to be an integral part of the system (whether purely self assessment or with results submitted to an instructor). This means a much tighter integration of material, interactive exercises, and assessment than is provided by a system as simple as the Hashing Tutorial.

## 5 Solution Overview: Technical Considerations

Based on our interpretation of the results of the 2010 survey [44], our experiences with the Virginia Tech Hashing Tutorial, and the continuous improvements in online technologies that we observe, we think that the time is ripe to create an entire online hypertextbook with integrated visualizations for an undergraduate course on Data Structures and Algorithms. Depending on the exact topics to be covered, this could be beneficial at any level from CS2 through senior algorithms.

When considering the full breadth of content that would be contained in a complete course textbook on the subject, we conclude that three distinct forms of content presentation are desirable. First, no matter how dynamic and interactive the topic, text and images as are now found in typical textbooks continue to have their place as part of the exposition. Some content simply is not visual or dynamic and so is most efficiently transferred via words and images.

Second, some content is essentially expository (i.e., at that point in the presentation there is no need for student constructivist interaction), but the content concerns a dynamic process or is otherwise conducive to visual presentation. This includes most algorithm description, such as how a particular sorting algorithm works. Since this presentation does not involve exploration or decision making, or (at this point) demonstration of proficiency, the prime concern is what techniques provide the clearest explanation. This could best be handled by a presentation that lies heavily on diagrams and simple animation, with pacing controlled by the the reader [39]. In essence, an animated slide show is adequate for such presentations.

Third, there are topics that do require or benefit from student interaction. This includes activities as simple as probing a calculation, (e.g., trying different inputs to a simple simulation or calculation). An example from our hashing tutorial is the famous Birthday Problem: How many people need to be in a room before the odds are greater than even that two share a birthday? Another need for interaction comes with demonstrating proficiency with an algorithm, such as the interactive exercises for tree insertions that are part of TRAKLA2 [21, 20]. These are best handled by something like a Java applet to support user interaction and dynamic on-the-fly calculations/processing of the algorithm. Performance comparisons are also of this nature, where the student is invited to define and run built-in simulations of multiple data structures or variants to see how they perform. Binding all of this together is a steady stream of assessment activities, to make sure that students stay “on track” and to keep them engaged even during otherwise passive exposition. This can be done with simple pop-up questions (depending on the desired control model, they might or might not require a correct response to continue) and end-of-section quizzes whose successful completion might be required to demonstrate competence needed to continue to the next section.

As we progress from the first to the third of these presentation approaches, the development cost goes up greatly. Text and images can be developed relatively quickly. In contrast, it took several student-years of effort (and over two actual years) to develop the Hashing Tutorial (equivalent to a single week of instruction), mainly due to the effort involved in developing a handful of Java applets. A properly animated slideshow takes longer to create than equivalent text and images, but should be significantly faster to implement than a fully interactive exercise.

A number of technologies are available for developing the three exposition types just described. Text and images can certainly be done with any traditional web development tool. The second component, which we characterize as an animated slide show, can be done (as the characterization suggests) using a variety of presentation tools such as MicroSoft Powerpoint, LaTeX’s Beamer package, OpenOffice Impress, or Apple Keynote. However, while presentations can be created in these tools, the resulting presentations cannot so easily be integrated with the rest of the electronic textbook. Browsers can support some of these tools as plugins, but not universally across a range of devices. The presentations can generally be converted to PDF format, but only Adobe’s Reader can actually display the animations (at least, evince and xpdf, popular alternative PDF readers, do not). Nor do the PDFs well integrate with non-PDF portions of the whole.

Flash is another popular tool for developing animations, and is rich enough to support the interactive aspects of the third component of presentation as well. However, Flash requires a plugin in most browsers, and so is not compatible with devices such as the iPad.

One technology that appears to be robust enough to implement all desired dynamic and interactive components is HTML5 incorporating JavaScript. HTML5 integrates its dynamic components well with standard text and images, and easily ports between PC browsers and mobile devices. Potential concerns include ease of use for content developers (as compared to, for example, PowerPoint when developing animated slide shows), and level of penetration of the necessary browser technology. Given that alternatives such as Flash and Java are at least as problematic in terms of a typical user having access (since they require plugins), the problem of penetration seems to be low and quickly receding, in that we can expect that college-level students tend to have access to moderately up-to-date browser technology. Thus, we currently advocate use of HTML5 technology for developing the dynamic components of the electronic textbook.

Students as stakeholders should be allowed to interact with these materials in one of two modes – exploratory and logging mode. In exploratory mode, student interactions are not tracked in any way. Instead they are practicing with the material, often making mistakes but doing so in a way that gradually leads to understanding. In logging mode, student interactions are tracked so that they and their instructors may evaluate how well they are achieving their learning objectives. "Tracked" in this sense might simply mean that the answers to assessment questions are logged and either graded automatically (e.g., progress on this unit can be checked off), or forwarded to the instructor (or some combination of automatic grading and information provided to the instructor).

The developers of hypertext materials are an important group of stakeholders in the process. Content authors should be able to write the hypertextbook learning modules using a set of workflows that (1) make it easy to add existing (compatible) AVs into the text they are writing and (2) allow others to copy and modify the materials under a set of open source attribution standards. The second of these workflows provides an opportunity for an instructor who may want to tailor some materials to their particular course to morph into a content author in the sense of modifying materials originally developed by someone else. While this instructor uses the materials in their course, the modified materials may remain "private", only visible to the students in the particular course. However, there should be a process that allows for such modified materials to be reviewed and published as an alternative to or improvement upon the original materials.

## 6 Proposed Work

Developing a full textbook's worth of AVs and related interactive exercises is a huge undertaking. In fact, it is far more than is possible within the scope of a TUES Type 1 project. The immediate goals of this project are to:

1. Settle all the infrastructure issues and develop a complete workflow process for developing the full hypertextbook.

2. Develop a first version "storyboard" for the envisioned integrated hypertextbook that links together existing AVs (from many sources) with the existing text, or documents where missing visualizations need to be created. We note that these AVs will be incompatible with the final system.

3. Develop a small set of prototype AVs within the proposed infrastructure as proof-of-concept complete lessons.

4. Develop a set of prototype assessment questions and activities within the proposed assessment infrastructure.

5. Establish connections with potential AV content providers for a future production phase of the project.

### 6.1 Integrating with the Connexions Infrastructure

The Connexions Project (`http://cnx.org`) was founded in 1999 by co-PI Baraniuk and is now one of the world's largest and most popular open education resources, with over 2 million unique users per month from over 190 countries [4, 5, 3, 41, 40]. The STEM content alone in Connexions has been used over 80 million times since 2003. Connexions users exploit the platform to form communities for creating new, interactive educational content; to publish it in a variety of formats (from online to print to the spoken word); to translate content into languages other than English; and to encourage a flow of ideas and applications between the academy and the "real world." Just as importantly, Connexions makes possible with educational materials what engineers and computer scientists do on a daily basis: iterate and innovate. Connexions content is improved upon through a series of built-in feedback loops, engaging both learners and teachers. Today, Connexions hosts over 18000 modules comprising over 1100 course collections. Semantic XML markup of modules makes content use and presentation easy and flexible. Connexions modules can be displayed as an individual web page, woven seamlessly into many different courses, converted to PDF for printing or EPUB for viewing on an iPad or iPhone, or even processed through a speech synthesizer to accurately read material to the blind or illiterate. All content in Connexions is licensed under the Creative Commons Attribution (CC-By) license, which makes it easy for authors to reuse and combine pieces of textbooks, or to make their own altered version of an existing textbook. This supports the collaborative development model that is a key part of our vision. Connexions has also already worked out significant policy concerns regarding editorial validation for publications through the use of post publication review and lenses (see `http://cnx.org/lenses`). The platform features quality control lenses that enable users to preferentially locate and view materials endorsed by third parties, such as professional societies, educational organizations, or publishers. The IEEE Signal Processing Society has deployed a lens for signal processing content based on a peer review model similar to the one it applies to its conferences and journals.

The infrastructure underlying Connexions, known as Rhaptos (`http://rhaptos.org`), is an open source project. In 2010, Connexions announced the release of Enterprise Rhaptos (`http://enterpriserhaptos.org`) which makes it easy for organizations and individuals to install and run an instance of the system locally on their own servers.

For the above reasons, Connexions will supply a major part of the infrastructure necessary to develop and deliver a 21st century hypertextbook. However, since our project is sufficiently different from standard electronic textbook efforts, there are a number of integration concerns that must be worked through to insure a smooth workflow for full-scale production of the hypertextbook. Hence, a major goal of this project is to resolve these issues through a combination of modifications or additions to the Connexions infrastructure (many of which are already underway as part of the Connexions Roadmap), third party solutions, and compromises on the current vision.

- Connexions does not presently support HTML5 or have native support for JavaScript. Currently, we anticipate that HTML5/JavaScript will be our implementation platform of choice for the interactive portions of the project. We anticipate eventual native support within Connexions for these elements. In the meantime, equivalent effects can be achieved with calls from Connexions pages to JavaScript elements hosted at Virginia Tech.

- Integration of assessment within the hypertextbook, which is addressed in Section 6.4.

- Improved cloning of content: While it is possible to share a module within Connexions, we hope to build on the current "cloning" process to make it easier for instructors to modify or combine existing works. A related issue is a more hierarchical model for module collections (sections building chapters building books) that makes sharing and mixing content easier. Much of this is currently in the Connexions roadmap.

- User (student) tagging, rating, and commenting of sections of the textbook content has been supported in the past by the Connexions infrastructure, but is not presently due to lack of use by the students combined with maintenance issues. We plan to revisit this issue, perhaps in the context of integrating with course forums where students will be more motivated to participate. Also related are efforts to use data mining techniques on log data of student use of the eTextbook (clickthroughs, times, question results) for (i) intelligent tutoring and (ii) deducing sections of the book where students have trouble, and indicating a need for improvement.
- Distributed collection: To distribute server burdens, we will investigate allowing other sites to run an instance of Connexions (via Enterprise Rhaptos), and house their own material. This material would be integrated into the main Connexions repository index. This would make it feasible, for example, to distribute the server load for the hypertextbook to Virginia Tech servers, while having the book appear in all ways as part of the Connexions collection.
- Book cross-referencing support for numbering of images, tables, figures, section, etc. across a collection of Connexions modules is desired for our project. We also seek improvements in support for table of contents, bibliography, and indexing.

### 6.2 Developing the Storyboard

Given the standard Connexions infrastructure and the existing material embodied in Shaffer and Naps' prior textbook efforts, the text for a hypertextbook can be quickly uploaded into Connexions. It is then a relatively short step to identify locations where interactive activities would benefit the text. In many cases, there are existing Java implementations of various forms that cover these topics. In some cases, links to these can be placed in the pages, in other cases readers can be directed to open a standalone visualization at suitable points. This is clearly an unsatisfactory way to experience a hypertextbook, but it gives us an initial "storyboard" for developing the final product. Many of the the available AVs are not really satisfactory presentations of the material, but they might provide ideas for use during the AV content development phase. These development specifications would be made part of the storyboard, hopefully to be picked up by future developers. Once created, the storyboard can then be gradually converted to the true hypertextbook that we envision using the defined framework.

### 6.3 Developing Interactive Activities

As described earlier, HTML5 with JavaScript appears to be the best compromise technology for developing interactive activities for this project. It will be native to all browsers, and provides all necessary computational and interaction capability. As we begin developing AVs in this language, we can quickly develop standard libraries based on our past extensive experience with developing AVs in Java.

Our goal is to develop AVs to support three to five key sections of the textbook. These sections would be selected primarily for their ability to span the range of different AV activity types. Some reasonable examples include one or more sorting visualizations (these are relatively straightforward to produce), one or more tree visualizations (with interactive construction in the style of TRAKLA2 exercises [21, 20]), and one or more graph algorithms. These are intended as exemplars to be used by other developers who agree to support our project in the future.

### 6.4 Support for Assessment

A primary goal of a hypertextbook is to maximize engagement by the reader. Additional engagement can be created by having the student respond to interactive questions at frequent intervals. Although many AVs include questions, few do so in a way that allows an instructor to monitor their students' progress. More typically the student's interaction with the system produces immediate feedback to the student, but the assessment of that interaction is not recorded in a way that the instructor can access. Nor do the students' answers provide a persistent record for the student that a section has been mastered, or integrate with navigation through the content such as provided by an "intelligent tutor". Two AV systems that do support this sort of integrated assessment are TRAKLA [20, 21] and JHAVÉ [26].

Although TRAKLA and JHAVÉ support online assessment of students using the visualizations, they both do so in a unique, non-portable way. We seek an approach that can work within a variety of presentations as well as within the electronic textbook structure itself (not all questions will come within AVs).

An on-going subproject within Connexions is the development of an integrated assessment infrastructure to enable instructors to search, select, and assign problems to their students from a question and answers database (`http://quadbase.org`). Students will have access to immediate feedback on their answers as well as pointers to community-generated remedial and enrichment materials. The Connexions team is working with cognitive scientists from Duke University to ensure that the infrastructure design maximizes student learning outcomes. Our hypertextbook project would be an early adopter of this implementation, and thus we would have input into its features.

Issues that must be address with respect to the assessment system include:

- How to represent a wide range of assessment question types. Quadbase uses the proposed IMS Question and Test Interoperability specification (QTI: `http://www.imsglobal.org/question/qtiv1p2/imsqti_oviewv1p2.html`).
- How to generate and present questions within the hypertextbook. In some cases, questions are generated from the existing database. In some cases, questions are generated on-th-fly using data generated randomly from within an AV.
- Developing effective strategies for assessing student responses. Assessing multiple-choice, multiple-selection, and fill-in-the-blank questions is straightforward. But automated assessment of textual responses is clearly much more difficult.
- How to use the results of the assessment to guide the next steps in learning.
- How to store student responses and student progress in a manner that makes it easy for instructors to analyze their students' results.
- How to store student responses and student progress in a manner that respects their privacy.

### 6.5 Building a Hypertextbook Developer Community Effort

An important consideration is the broader community that can become involved in development of the hypertextbook. Such a project is a huge undertaking. As evidence of this (besides our experiences with the extraordinary amount of time that it takes to develop high-quality AVs), consider that there exist few examples of the type of artifact that we seek to create, as discussed in Section 3. Ideally, a broader community can be encouraged to contribute to the project, much in the style of an open source software development effort.

Our discussion of the "storyboarding" effort suggests temporary use of various existing Java-based AVs. In many cases, these AVs were created by collaborators or contacts of ours. We hope that we can get support from them to re-implement versions of their AVs for the hypertextbook effort. Letters of support from three major European AV developers are included with this proposal. Since our materials are distributed with a GPL or Creative Commons license, intellectual property rights will be less of an issue than if a commercial publisher were involved.

## 7 Dissemination Plan

By its nature, this project is entirely about dissemination. By hosting the materials developed by this project as free, open resources at Connexions, we will make the results widely available to the educational community. Furthermore, this provides a natural mechanism for advertising availability of the materials through Connexions' normal procedures. PI Shaffer's textbook is currently online, with a number of instructors from around the world who use the PDF version for their classes (logfile data indicate that the PDF documents were downloaded nearly 10,000 times per month during July to November, 2010, and over 11,000 times during March 2011). Through this site we can encourage current users to consider adopting the hypertextbook version of the text. We can also disseminate information about the project through the AlgoViz Portal.

To increase visibility of our educational infrastructure and encourage broader use of the framework, we will present our project to the CS educational community through informal channels such as the SIGCSE listserv and contact colleagues who can use the framework in their classes. More formally, we will give a tutorial on using our framework at a major educational conference such as SIGCSE. More generally, the topic of eTextbooks, especially as relates to community involvement by instructors and students in created the product, are likely to be viable options for special sessions. The Signal Processing Education Network, co-founded by co-PI Baraniuk and funded by the NSF CI-TEAM program (`http://dsp.rice.edu/spen`), will provide a useful model for actively engaging an expanding community of computer science educators.

Additional dissemination of results and knowledge from this work will come in the form of publications and workshops/tutorials at national or international conferences. Appropriate venues to publish the results of this research include SIGCSE, ITiCSE, FIE, CCSC and/or ASEE, all recognized conferences that cover advances in computer science education. Relevant journals include *ACM Transactions on Computing Education* (previously, *JERIC*), Elsevier's *Computers & Education: An International Journal*, *IEEE Transactions on Education*, and Taylor & Francis' *Computer Science Education.* We will prepare tutorials and technical reports, and make all papers and source code available on the Internet.

## 8 Evaluation Plan

This project is primarily proof-of-concept development for an eventual complete online hypertextbook for courses in data structures and algorithms. As such, its main products are a working plan for complete end-to-end infrastructure for creating the full set of materials, the "storyboard" for the complete hypertextbook, prototype materials for selected key sections, and the community agreements necessary to complete materials development. Evaluation metrics for these efforts include:

- Number of outside collaborators who commit to providing materials to the project;
- Number of specific interactive exercises developed, both by project team members and by outside contributors;
- Completeness of the "storyboard";
- Feedback from the broader AV development community on the proposed development framework;
- Success of the assessment system in terms of its ability to represent a range of questions, number of questions developed, ability to store and process student assessments, and willingness of the broader community to adopt it.

Pedagogical assessment of sample materials at the proof-of-concept level is also important. We have already conducted preliminary assessments of similar materials (see Section 4). During the course of this project, we will incorporate the materials that we develop into existing courses (such as CS3114: Data Structures and Algorithms, that is taught by PI Shaffer at Virginia Tech). We will collect evaluation statistics in comparative experiments between treatment sections and control sections in accordance with standard techniques for conducting such evaluations. One prior evaluation experience is described in Section 4.

## 9 Work Plan

Our work plan and an estimated time frame is given below. Times add up to more than 12 months, since the various tasks will be interleaved and shared between participants.

Year 1 (2012):

- Define the overall framework for producing hypertextbook sections with interactive AVs. January–August.
- Design how the assessment system will integrate into the hypertextbook. May–December.
- Implementation work as needed for the framework. May-August
- Develop the "Storyboard" from the existing text and existing Java-based AVs available on the web. May–December.
- Develop sample prototypes for AVs using the framework. August–December.

Year 2 (2013):

- Develop more prototypes. Ongoing throughout year.

- Develop quiz items. Ongoing throughout year.
- Use prototype sections in classes, and with assessment. Spring and Fall semesters.
- Presentation/workshop at SIGCSE 2013: March.
- Disseminate information about the project. Ongoing
- Build coalition of contributors for needed AVs. Ongoing

## 10 Broadening Participation: Minorities and Undergraduates

This project is a natural for recruiting undergraduate students for research projects. We have had many students work on AV-related activities over the years. PI Naps has managed a site REU project since 2009 that supports 5–10 undergraduates each summer as they develop JHAVÉ modules. PI Shaffer has had over 10 undergraduate independent study students and summer interns over the past four years. With the "Storyboard" in place giving guidance on specific AVs that need to be developed, and the likely interest on the part of students in developing skills related to HTML5, we anticipate that many undergraduates will choose to do independent projects related to this effort.

At Virginia Tech, we have had success in broadening participation in computing research by minorities. The PI has a strong track record working with female and minority students at VT (including two of the undergraduates doing independent study in the past four years on AV-related projects and two minority summer interns). The Department of Computer Science holds an annual research symposium for undergraduate research students. This is part of the Virginia Tech Undergraduate Research in Computer Science (VTURCS) program. The PI often requires that undergraduate students that work with us participate in VTURCS with a poster presentation.

The VT CS department generally has had success recruiting African-American female graduate students from neighboring Historically Black College and Universities (HBCU). We have three recent female African American PhD's in computing, (Tracy Lewis, at Radford University, Cheryl Seals at Auburn University, and Jamika Burge at Penn State). We currently are recruiting African-American students from Auburn University, as part of our collaboration with Dr. Juan Gilbert. The success of women in Computer Science in the department is reflected by a high percentage of Ph.D. graduates. 10 out of 50 PhD graduates from Spring 2008 to Spring 2011 have been female, which is higher than the national average. We currently have four Latino students in our graduate program (2 PhD students and 2 Masters). One of them, Ricardo Quintana-Castillo, is a recipient of an NSF Graduate fellowship. Dr. Shaffer's NSF-funded AlgoViz projects have supported four graduate students: A.J. Alon (minority), Monika Akbar (female), Michael Stewart (first-generation college graduate), and Eric Fouh (minority).

## 11 Key Staff

**PI:** Cliff Shaffer, Professor of Computer Science at Virginia Tech, has interests in algorithms and data structures, visualization, problem-solving environments, computational biology, and educational use of computers. He was the PI and lead designer for the highly successful **Project GeoSim** in the mid-1990s, which developed a series of simulations for geography education. He has extensive experience with designing problem-solving environments for a number of scientific and engineering applications. He is the designer and course coordinator for CS2606: Data Structures and File Processing, and wrote the textbook used in that course. Role: Dr. Shaffer will provide overall project management, lead efforts to define the details of the development process that will lead to the hypertextbook, lead efforts to create the detailed storyboard, contribute to the question bank, and work to build the consortium of AV developers that will create the bulk of the AVs needed to complete the hypertextbook.

**Co-PI:** Tom Naps, Professor of Computer Science, University of Wisconsin Oshkosh, has taught a broad range of computer science courses, authored computer science textbooks in data structures and algorithms, and pursued work in algorithm visualization both from the perspective of an instructor who wants to design visualizations of particular algorithms to help his students and as the developer of the GAIGS and JHAVÉ AV systems. Role: Dr. Naps will help author textual materials, incorporate interactive visualizations into those materials, and work on the quizzing system design and implementation.

**Co-PI:** Richard Baraniuk, Victor E. Cameron Professor of Electrical and Computer Engineering, Rice University. Founder and Director of the Connexions project. Baraniuk has served as the PI of over 30 NSF and DOD projects in sensing and signal processing over the past 17 years, including an ARO MURI and several DARPA programs. He has been actively involved in open education in general and Connexions in particular since founding the project in 1999. His own signals and systems e-textbook in Connexions has been accessed over 5 million times. Role: Liaison with Connexions, our intended development platform for the hypertextbook.

**Graduate, undergraduate, and intern students** The Virginia Tech portion of the project will hire one graduate research assistant to perform mission-critical development. The University of Wisconsin-Oshkosh portion of the project includes funds for hourly wages for undergraduate students. Both departments have historically had great success in recruiting undergraduate students to contribute on educational research projects. At Virginia Tech, this is formalized formalized through our Undergraduate Research in Computer Science (VTURCS) program. Both departments have ongoing REU summer internship programs. We anticipate that a large number of undergraduates at both schools will be interested in doing independent study projects to support this effort, as a vehicle for learning HTML5-related skills.

## 12 Results from Prior NSF Support

**NSF CCLI Phase 1 Award (DUE-0836940)** *Building a Community and Establishing Best Practices in Algorithm Visualization through the AlgoViz Wiki.* PIs: Clifford A. Shaffer and Stephen H. Edwards. $149,206 for 01/2009 – 12/2010.

**NSF NSDL Small Project (DUE-0937863)** *The AlgoViz Portal: Lowering Barriers for Entry into an Online Educational Community.* PIs: Clifford A. Shaffer and Stephen H. Edwards, $149,999 for 01/2010 – 12/2011.

These projects seek to developed online infrastructure (a website and related community development efforts) to promote use of algorithm visualization (AV) in computer science courses. This is achieved by building a community of AV users and developers. So far, two journal papers [45, 10] and four conference papers [46, 48, 44, 47] have been produced relating to this work.

**NSF CCLI Phase 1 Awards (DUE-0341148 and DUE-0126494)** *Integrating Algorithm Visualization into Computer Science Education.* PIs: Thomas L. Naps, Scott Grissom, Myles McNally, $71,993 for 2001–2002 and $197,118 for 2003–2007.

These two grants resulted in significant development on the JHAVÉ algorithm visualization system. Naps and his co-PI's worked with eight undergraduate student research assistants during the course of these grants. A website was established at `http://jhave.org` to disseminate the visualization-based materials that were produced for teaching data structures and algorithms. Nine peer-reviewed research papers were published as a result of the grant, including three that had undergraduate students as co-authors [11, 19, 30, 22, 25, 28, 14, 23, 37].

**REU grant (CNS-0851569)** *Exploring Open Source Software: Development and Efficacy of Online Learning Environments in Computer Science*, PIs: Thomas L. Naps and David Furcy, $261,167 for 2009–2012.

This project has as its primary goal providing student participants with extensive exposure to open source software for the delivery of instructional algorithm visualizations, both as developers and educational researchers.

**NSF Partnerships for Innovation Program (05-38934)** *Building Communities and Sharing Knowledge in Engineering Education: A University/Industry Partnership*, PI: Richard Baraniuk, $600,000, 2006–10.

**NSF Cyberinfrastructure-TEAM Award (1041396)** *Signal Processing Education Network (SPEN)*, PI: Richard Baraniuk, $868,225, 2010–13.

This university/industry partnership between Rice's Connexions Project (cnx.org) and National Instruments (ni.com) developed a new approach to building and sustaining virtual educational communities around active content and applied the results to the spectrum of engineering education venues: university undergraduate and graduate courses, industrial training and continuing education, just-in-time learning on the job, and high-school laboratories. A component of the broader Connexions project, SPEN is developing a network of signal processing education champions around Connexions to develop new open education content and encourage its adoption at institutions nationwide and worldwide. Publications from these project include [6, 8, 17, 7]. Specific outcomes include:

- Built a world-wide community of educators, students, and practitioners in DSP led by Rice faculty and NI technology evangelists who developed and refined a critical mass of free Connexions DSP course materials.
- Enriched these materials with interactive LabVIEW visualizations to make the concepts come alive and encourage experimentation, exploration, and design.
- Translated the materials into a number of languages, including Spanish, to reach both local and worldwide audiences.
- Studied the marketing and business issues associated with growing and sustaining the project into a win-win outcome for both the university and industrial partners.

## References

[1] AlgoViz.org. The AlgoViz Portal, 2011. `http://algoviz.org`.

[2] A. Alharbi, F. Henskens, and M. Hannaford. Integrated standard environment for the teaching and learning of operating systems algorithms using visualizations. In *Fifth International Multi-Conference on Computing in the Global Information Technology (ICCGI)*, pages 205–208, September 2010.

[3] S. Appadweula, R. Baraniuk, M. Berry, M. Butala, H. Choi, M. Haun, D. Jones, M. Kramer, D. Mousa, L. Potter, D. Sachs, B. Wade, and R. Wagner. Open content signal processing laboratories in Connexions. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 777–780, 2003.

[4] R. Baraniuk, C. Burrus, B. Hendricks, G. Henry, A. Hero, D. Johnson, D. Jones, R. Kusuma, R. Nowak, J. Odegard, L. Potter K. Ramchandran, R. Reedstrom, P. Schniter, I. Selesnick, D. Williams, and W. Wilson. Connexions: DSP education for a networked world. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages IV–4145 – IV–4147, 2002.

[5] R. Baraniuk, C. Burrus, D. Johnson, and D. Jones. Connexions – Sharing knowledge and building communities in signal processing. *IEEE Signal Processing Magazine*, 21(5):10 – 16, 2004.

[6] R.G. Baraniuk. Challenges and opportunities for the open education movement: A Connexions case study. In T. Iiyoshi and M.S. Vijay Kumar, editors, *Opening Up Education – The Collective Advancement of Education through Open Technology, Open Content, and Open Knowledge*. MIT Press, 2008.

[7] R.G. Baraniuk. How open is open education? *Domus*, March 2009.

[8] R.G. Baraniuk and C.S. Burrus. Global warming toward open educational resources. *Communications of the ACM*, 51(9):30–32, September 2008.

[9] M.H. Brown. *Algorithm Animation*. MIT Press, Cambridge, Massachussets, 1988.

[10] E. Fouh, M. Akbar, and C.A. Shaffer. The role of visualization in computer science education. *Submitted to Computers in the Schools*, 2011.

[11] D. Furcy, A. Jungwirth, and T. Naps. Blocktree–pedagogical information visualization for heuristic search. In *Proceedings of the Twentieth International Florida Artificial Intelligence Research Society Conference, May 7-9, 2007, Key West, Florida, USA*. AAAI Press, 2007.

[12] E. Gramond and S.H. Rodger. Using JFLAP to interact with theorems in automata theory. *ACM SIGCSE Bulletin*, 31(1):336–340, 1999.

[13] M.T. Grinder, S.B. Kim, T.L. Lutey, R.J. Ross, and K.F. Walsh. Loving to learn theory: Active learning modules for the theory of computing. In *Proceedings of the 33rd ACM SIGCSE Technical Symposium on Computer Science Education (SIGCSE 2002)*, pages 371–375, Cincinnati, Kentucky, 2002.

[14] S. Grissom, M.F. McNally, and T.L. Naps. Algorithm Visualization in CS Education: Comparing Levels of Student Engagement. In *SoftVis '03: Proceedings of the 2003 ACM symposium on Software visualization*, pages 87–94, New York, NY, USA, 2003. ACM Press.

[15] K. Hew and T. Brush. Integrating technology into K12 teaching and learning: current knowledge gaps and recommendations for future research. *Educational Technology Research and Development*, 55:223–252, 2007.

[16] V. Karavirta. Towards seamless merging of hypertext and algorithm animation. In *Proceedings of the Fifth Program Visualization Workshop, PVW'08*, volume 224 of *Electronic Notes in Theoretical Computer Science*, pages 105–114, 2009.

[17] C.M. Kelty, C.S. Burrus, and R.G. Baraniuk. Peer review anew: Three principles and a case study in post-publication quality assurance. *Proceedings of the IEEE*, pages 1000–1011, June 2008.

[18] J.H.-Y. Kim and H.-Y. Jung. South Korean digital textbook project. *Computers in the Schools*, 27(3 & 4):247–265, 2010.

[19] J. Lucas, T.L. Naps, and G. Rößling. Visualgraph: a graph class designed for both undergraduate students and educators. *ACM SIGCSE Bulletin*, 35(1):167–171, 2003.

[20] L. Malmi, V. Karavirta, A. Korhonen, J. Nikander, O. Seppälä, and P. Silvasti. Visual algorithm simulation exercise system with automatic assessment: TRAKLA2. *Informatics in Education*, 3(2):267–288, September 2004.

[21] L. Malmi and A. Korhonen. *Active Learning and Examination Methods in a Data Structures and Algorithms Course*, pages 210–227. Number 4821 in LNCS. Springer-Verlag, 2008.

[22] M. McNally, T.L. Naps, D. Furcy, S. Grissom, and C. Trefftz. Supporting the rapid development of pedagogically effective algorithm visualizations. *Journal of Computing Sciences in Colleges*, 23(1):80–90, 2007.

[23] T. Naps and S. Grissom. The Effective Use of Quicksort Visualizations in the Classroom. *J. Comput. Small Coll.*, 18(1):88–96, 2002.

[24] T.L. Naps. *Introduction to Data Structures and Algorithm Analysis, 2nd ed.* West Publishing Co., St. Paul, MN, USA, 1992.

[25] T.L. Naps. JHAVÉ – Supporting Algorithm Visualization Engagement. *IEEE Computer Graphics and Applications*, 25(5), September/October 2005.

[26] T.L. Naps. Jhavé: Supporting algorithm visualization. *IEEE Computer Graphics and Applications*, 25:49 – 55, September 2005.

[27] T.L. Naps, S. Cooper, and twelve more authors. Evaluating the educational impact of visualization. In *ITiCSE-WGR '03: Working Group Reports from ITiCSE on Innovation and Technology in Computer Science Education*, pages 124–136, 2003.

[28] T.L. Naps, M. McNally, and S. Grissom. Realizing XML Driven Algorithm Visualization. In *Proceedings of the Fourth Program Visualization Workshop*, pages 1–5, Florence, Italy, June 2006.

[29] T.L. Naps and G.J. Pothering. *Introduction to Data Structures and Algorithm Analysis with Pascal, 2nd ed.* West Publishing Co., St. Paul, MN, USA, 1992.

[30] T.L. Naps and G. Rößling. JHAVÉ – More Visualizers (and Visualizations) Needed. In *Proceedings of the Fourth Program Visualization Workshop*, pages 112–117, Florence, Italy, June 2006.

[31] T.L. Naps, G. Rössling, and nine more authors. Exploring the role of visualization and engagement in computer science education. In *ITiCSE-WGR '02: Working Group Reports from ITiCSE on Innovation and Technology in Computer Science Education*, pages 131–152, 2002.

[32] G.J. Pothering and T.L. Naps. *Introduction to data structures and algorithm analysis with C++.* West Publishing Co., 1995.

[33] S.H. Rodger and T.W. Finley. *JFLAP-an interactive formal languages and automata package.* Jones & Bartlett Learning, 2006.

[34] R.J. Ross. Hypertextbooks and a hypertextbook authoring environment. In *ITiCSE '08: Proceedings of the 13th annual conference on Innovation and technology in computer science education*, pages 133–137, Madrid, Spain, 2008. ACM.

[35] R.J. Ross and M.T. Grinder. Hypertextbooks: Animated, active learning, comprehensive teaching and learning resources for the web. In S. Diehl, editor, *Software Visualization*, number 2269 in Lecture Notes in Computer Science, pages 269–284. Springer, 2002.

[36] G. Rössling, T. Naps, and nine more authors. Merging interactive visualizations with hypertextbooks and course management. In *ITiCSE-WGR '06: Working Group Reports from ITiCSE on Innovation and Technology in Computer Science Education*, pages 166–181, 2006.

[37] G. Rößling and T.L. Naps. A Testbed for Pedagogical Requirements in Algorithm Visualizations. In *ITiCSE '02: Proceedings of the 7th annual conference on Innovation and technology in computer science education*, pages 96–100, New York, NY, USA, 2002. ACM Press.

[38] G. Rößling and T. Vellaramkalayil. First steps towards a visualization-based computer science hypertextbook as a moodle module. In *Proceedings of the Fifth Program Visualization Workshop, PVW'08*, volume 224 of *Electronic Notes in Theoretical Computer Science*, pages 47 – 56, 2009.

[39] P. Saraiya, C.A. Shaffer, D.S. McCrickard, and C. North. Effective features of algorithm visualizations. In *SIGCSE '04: Proceedings of the 35th SIGCSE Technical Symposium on Computer Science Education*, pages 382–386, New York, New York, USA, March 2004. ACM Press.

[40] J. Selingo. Connecting the dots. *ASEE Prism*, 13(4):34–37, 2003.

[41] J. Selingo. The Connexions project: Promoting open sharing of knowledge for education. *Syllabus*, pages 1–11, 2003.

[42] C.A. Shaffer. *A Practical Introduction to Data Structures and Algorithm Analysis*. Prentice Hall, second edition, 2001.

[43] C.A. Shaffer. A practical introduction to data structures and algorithm analysis, edition 3.2, 2011. Published online at `http://people.cs.vt.edu/~shaffer/Book/`.

[44] C.A. Shaffer, M. Akbar, A.J.D. Alon, M. Stewart, and S.H. Edwards. Getting algorithm visualizations into the classroom. In *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education (SIGCSE'11)*, pages 129–134, 2011.

[45] C.A. Shaffer, M.L. Cooper, A.J.D. Alon, M. Akbar, M. Stewart, S. Ponce, and S.H. Edwards. Algorithm visualization: The state of the field. *ACM Transactions on Computing Education*, 10:1–22, August 2010.

[46] C.A. Shaffer, M.L. Cooper, and S.H. Edwards. Algorithm visualization: A report on the state of the field. In *SIGCSE '07: Proceedings of the 38th SIGCSE Technical Symposium on Computer Science Education*, pages 150–154, March 2007.

[47] C.A. Shaffer, T.L. Naps, and E. Fouh. Truly interactive textbooks for computer science education. In *submitted to the Proceedings of the Sixth Program Visualization Workshop, PVW'11*, June 2011.

[48] C.A. Shaffer, T.L. Naps, S.H. Rodger, and S.H. Edwards. Building an online educational community for algorithm visualization. In *Proceedings of the 41st ACM technical symposium on Computer science education - SIGCSE'10*, pages 475–476, Milwaukee, Wisconsin, USA, March 2010.

[49] N. Titterton, C.M. Lewis, and M.J. Clancy. Experiences with lab-centric instruction. *Computer Science Education*, 20(2):79–102, 2010.