Algorithm Visualization: A Report on the State of the Field

Clifford A. Shaffer
Department of Computer Science
Virginia Tech
Blacksburg, VA 24061
shaffer@cs.vt.edu

Matthew Cooper
Department of Computer Science
Virginia Tech
Blacksburg, VA 24061
macooper@cs.vt.edu

Stephen H. Edwards
Department of Computer Science
Virginia Tech
Blacksburg, VA 24061
edwards@cs.vt.edu

ABSTRACT

We present our findings on the state of the field of algorithm visualization, based on extensive search and analysis of links to hundreds of visualizations. We seek to answer questions such as how content is distributed among topics, who created algorithm visualizations and when, the overall quality of available visualizations, and how visualizations are disseminated. We have built a wiki that currently catalogs over 350 algorithm visualizations, contains the beginnings of an annotated bibliography on algorithm visualization literature, and provides information about researchers and projects. Unfortunately, we found that most existing algorithm visualizations are of low quality, and the content coverage is skewed heavily toward easier topics. There are no effective repositories or organized collections of algorithm visualizations currently available. Thus, the field appears in need of improvement in dissemination of materials, informing potential developers about what is needed, and propagating known best practices for creating new visualizations.

Categories and Subject Descriptors

E.1 [**Data Structures**]; E.2 [**Data Storage Representations**]; K.3.2 [**Computers and Education**]: Computer and Information Science Education

General Terms

Algorithms, Measurement, Design

Keywords

Data Structure and Algorithm Visualizations, Algorithm Animation, Courseware

1. INTRODUCTION

Data structure and algorithm visualizations and animations (hereafter referred to generically as *algorithm visualizations*) have a long history in computer science education. While the 1981 video "Sorting out Sorting" by Ronald Baeker was the first well-known visualization, ad hoc visualizations existed long before. The first recognized system for creating algorithm animations was BALSA [2] in 1984. Since then, hundreds of algorithm visualizations have been implemented and provided freely to educators, and scores (or hundreds) of papers have been written about them.

It is widely perceived that algorithm visualizations can provide a powerful alternative to static written presentations (from textbooks) or verbal descriptions supported by illustrations (from lectures). There has been some debate in the literature as to whether algorithm visualizations are effective in practice. Some studies have shown the classic dismissal that is the downfall of most technological interventions in education: "no significant difference" [6, 9, 11]. Other studies have shown that algorithm visualizations can indeed improve understanding of the fundamental data structures and algorithms that are part of a traditional computer science curriculum [13, 3, 7]. Certainly, many visualizations exist and are widely (and freely) available via the Internet. Unfortunately, the vast majority of those currently available serve no useful pedagogical purpose.

So we see that (a) many algorithm visualizations exist, yet relatively few are of true value, and (b) algorithm visualizations can be demonstrated to have pedagogical value, yet it is also quite possible to use them in ways that have no pedagogical effect. These facts seem to imply that creating and deploying effective algorithm visualizations is difficult. There is a small body of literature that investigates how to create pedagogically useful algorithm visualizations (for example, [10, 15]). Yet, there is still much to be done before we are at the point where good quality visualizations on most topics of interest are widely available.

The purpose of this paper is to provide a summary of preliminary findings resulting from our efforts to survey the state of the field of algorithm visualization. To bound the content area, we focused our attention on topics commonly taught in undergraduate courses on data structures and algorithms. We seek an understanding of the overall health of the field, and present a number of open research questions that we and others can work on in the future. Some examples of the questions we seek to address are:

- What visualizations are available?
- What is their general quality?
- Is there adequate coverage of the major topic areas covered in data structures and algorithms courses?
- How do educators find effective visualizations?
- Is the field active, and improving?
- Is there adequate infrastructure for storing and disseminating visualizations?

SIGCSE'07, March 7–10, 2007, Covington, Kentucky, USA.

2. WHAT'S OUT THERE?

Since Spring 2006, we have made a significant effort to catalog as many existing algorithm animations as we could. The results can be found at our Data Structure and Algorithm Visualization Wiki [18]. In this time, we have developed the most extensive collection of links to algorithm visualizations currently available. While it is by no means complete, it does serve as a representative sample of the total population of visualizations accessible from the Internet. We are still collecting visualizations, and many of the following assessments are based on preliminary data.

2.1 How did we find them?

We used a number of techniques to locate algorithm visualizations. We began with a list of all visualization systems that we were aware of from our general knowledge of the field. We developed a topic list based on our experiences teaching relevant courses. We considered what search terms would be most productive for locating visualizations via Internet searches (this is discussed further in Section 3). Based on our topic list, we then performed searches using Google to find whatever we could. Once we had generated a base of visualization links, we then examined these pages to try to locate other visualizations, since developers of a given visualization often have others available. Sometimes we could find these other visualizations from direct references on the pages we already had, and other times we could deconstruct the URLs to find more visualizations. Whenever we stumbled across a page that had links to collections of visualizations, we would follow those links to capture any new ones not yet in our collection.

2.2 How many are available?

We have collected links to over 350 visualizations. Many of these are individual applets or programs, but a significant fraction appear as parts of integrated visualization collections (typically, individual applications that include 5-20 distinct visualizations, or toolkits that distribute a collection of 5-20 distinct visualizations as a unit). If a given program contains multiple visualizations (for example, a single Java applet that embodies separate visualizations for both stacks and queues), then we count it multiple times, once for each distinct visualization. We speculate that we have so far captured roughly half of what is publicly available, and most likely we have captured the vast majority of the visualizations that are easily found and better known. We are still actively collecting new links.

2.3 How are they implemented?

Since Java's introduction in the mid-1990s, virtually all algorithm visualizations and algorithm visualization toolkits have been implemented in Java. Probably over half of available visualizations are provided as applets directly from web pages. However, a noticeable minority are Java applications that must be downloaded and opened locally. These numbers are somewhat biased. There is a tendency for us to search for applets, since this turns out to be easier to do (as explained in Section 3). Visualizations that are directly available in web pages will typically get more attention from potential users, since they need not go through the additional step of downloading and unpacking a visualization or visualization system. We do a significant amount of cross checking by capturing links from visualization link collections that we find (as described in Section 2.1) to insure that our data collection process does not focus unfairly on individual applets.

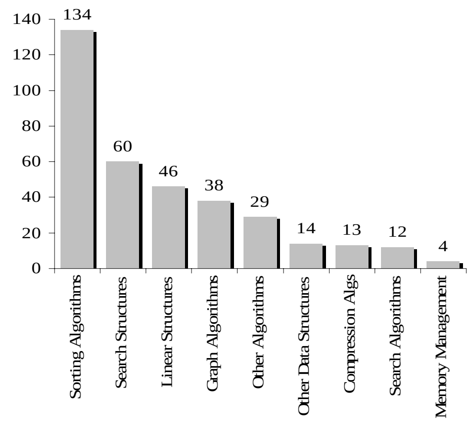

Figure 1: Histogram of major categories in Table 1

2.4 How are they disseminated?

Almost none of the visualizations that we have found are available from large, organized repositories. We discuss the status and effect of various courseware repositories on the algorithm visualization community in Section 3. Many visualizations are cataloged by link sites, meaning sites that (like our wiki) attempt to link to collections of visualizations that the site managers have considered worthy. However, most of these link sites are small in scale, perhaps linking to 20 favorite visualizations for some course or textbook. There have been a small number of efforts to produce comprehensive catalogs of visualizations (for example, the Hope College collection [8] in 2001), but our own wiki appears to be the only effort to do this that is currently active.

2.5 Who makes them?

While these numbers are quite preliminary, we have developed a picture of who is providing visualizations. Perhaps 10-20% of algorithm visualizations are essentially single efforts by their respective authors. Perhaps 30-40% are provided by "small shops" that have created typically 5-15 visualizations, mostly by hand as individual Java applets. These might have each been created by the same individual over some number of years (typically a faculty member who is teaching relevant courses), or they might have been developed by a small number of students working for a given faculty member. Perhaps 50% of visualizations are provided by teams that have also created visualization systems of one sort or another (and the visualizations are related to the system). We catalog such systems in addition to cataloging the individual visualizations.

2.6 What is the content distribution?

We have restricted our study to topics that are typically covered in undergraduate courses in data structures and algorithms. While this concerns mostly lower division materials, we also considered some upper division topics like computational geometry, algorithms for $\mathcal{NP}$-complete problems, dynamic programming, and string matching. Our top-level categories for grouping visualizations, along with their current population counts, are shown in Table 1.

As Figure 1 shows, there is wide variation in coverage.

Linear Structures	**46**
Lists	13
Stacks & Queues	32
Search Structures	**60**
Binary Search Trees	13
AVL Trees	7
Splay Trees	9
Red-Black Trees	8
B-Trees and variants	10
Skiplist	3
Other Data Structures	**14**
Heap/Priority Queue	8
Search Algorithms	**12**
Linear/Binary Search	2
Hashing	10
Sorting Algorithms	**134**
Sorting Overviews	6
Insertion Sort	18
Selection Sort	18
Quicksort	23
Mergesort	28
Heapsort	9
Radix Sort	10
Graph Algorithms	**38**
Traversals	10
Shortest Paths	11
Spanning Trees	9
Network Flow	3
Compression Algorithms	**13**
Huffman Coding	10
Memory Management	**4**
Other Algorithms	**28**
Computational Geometry	6
$\mathcal{NP}$-complete Problems	3
Algorithmic Techniques	7
String Matching	8
Mathematical Algorithms	4

Table 1: Major categories for visualizations and their counts for 352 visualizations collected. Each major category shows its total, and significant subcategories are also shown.

Nearly 15% of all visualizations are on linear structures such as stacks and queues, even though these probably present less difficulty to students than many other topics. Over a third of all visualizations that we have found are on sorting algorithms. Sorting is an important topic for undergraduate data structures and algorithms courses, but it is certainly being given too much attention by creators of visualizations. Further, many of the sorting visualizations are variations on the classic "Sorting out Sorting" video, and just show bars being swapped. In contrast, most specialized and advanced topics are poorly covered. There is certainly room for new visualizations.

2.7 What is their quality?

A majority of the visualizations that we have encountered either cannot be made to work easily, or appear to have no pedagogical value. When we say "have no pedagogical value," we mean that they give the user no understanding whatsoever of how the data structure or algorithm being "visualized" works, and so will be of little use in the classroom.

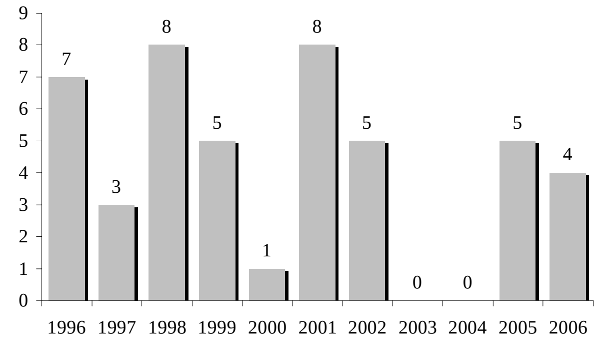

Figure 2: Last-change dates for sorting visualizations. Counts are by project.

Only a small fraction (certainly less than a quarter) would we feel comfortable recommending for use, either as a lecture aid, as the basis for a lab exercise, or for self-study of a topic. Another quarter of visualizations fall in between: they are potentially useful but severely limited. Even the better visualizations tend to have serious deficiencies. Roughly half of all algorithm visualizations are actually animations. For many of these, users are relegated to being passive observers with no control over pacing (aside from animation speed), the data being processed, or the operations being conducted. A different type of deficiency often occurs with visualizations of tree structures, which are popular subjects for visualization. Typically, these will show the tree that results from a user-selected insert or delete operation. But rarely do they illustrate at all, let alone effectively, how the insert or delete operation actually works. Similar deficiencies occur in many visualizations for simple structures such as lists, stacks and queues. Only the outcome of an insertion or deletion is displayed, with no explanation of how the result was obtained.

2.8 When were they made?

A significant number of systems were developed in the early 1990s for creating algorithm animations and visualizations. However, most of these are now no longer available, or so difficult to install and run due to changes in computer operating systems, that they are not currently a factor in education. If we consider the development of visualizations since the mid 1990s (i.e., Java), we have observed a decline over time in the creation of new visualizations, particularly after 2002. Figure 2 shows a histogram of the last-change dates for the sorting visualizations currently in our collection. These counts are "by project" rather than by individual visualization, in that if a given visualization or visualization system provided visualizations for multiple sorting algorithms, then we only count it once in the histogram. We can see that, while there are still active projects, overall activity does not appear to be as extensive today as it was previously. Since we are measuring the "last change" date for the various visualizations, the better numbers at the end of the histogram merely indicate that there still exist active projects, as opposed to a recent rise in activity.

As discussed above, the recent decline in the creation of new visualizations certainly cannot be explained by saturation of either topic coverage or content quality, since both are sadly lacking. Ongoing projects appear no more attuned to the topical gaps in coverage. We speculate that students were readily available to create visualizations during the Java boom of the mid to late 1990s, when Java and applets were new and the "in thing" for students to learn. But now there are other "in things" competing for students' attention, and Java is considered old hat. The nation-wide drop in computer science enrollments might also mean that there are fewer students available who want to do such projects.

2.9 Will we find them again?

Like everything on the Internet that is maintained by individuals, algorithm visualizations have a high turnover rate in their accessibility. To get some measure of this, we have been tracking the status of the visualization links on our wiki. Each week, we take a snapshot of how many links on any of our wiki pages are still accessible. As of May 29, 2006, we had 46 active (working) links in our wiki. Of those 46, after three months we had one whose host machine was no longer on the Internet, three which returned "page not found" (two of these three were at the same site), and two with permanent redirection pages (indicating that our links need to be updated in the wiki or eventually we will lose them). In other words, over a span of three months, 6 out of 46 links were either lost to us or moved. One was subsequently found at a new location. On July 31, 2006, we had 219 unique working links. Of these, 15 were no longer available on October 31, 2006.

3. SEARCHING FOR VISUALIZATIONS

There are essentially two distinct ways to go about finding an algorithm visualization on a given topic. One is to "google for it"—that is, use your favorite Internet search engine to search for whatever keywords you believe will best find a useful visualization on the topic you need. The other is to look in an algorithm visualization collection, a courseware repository, or a curated link collection of which you might be aware.

Let us assume that there exists on the Internet a suitable visualization on a specific topic. In that case, we can hope that standard Internet search technology will allow educators to find it. If so, this might alleviate the need to create and maintain specialized repositories or link collections for courseware in general, and visualizations in particular. Unfortunately, whether any given instructor will be able to find an existing artifact depends a lot on the instructor's ability to supply the right keywords to yield successful results. Like any Internet search, keywords need both to capture the desired artifacts, *and* to avoid burying the user in overwhelming numbers of false-positive responses. Therefore, keywords must identify both the *type* of material desired (visualizations as opposed to explanations), and the topic or content area desired. Some keywords, while technically specific, might lead to a wealth of non-related information. For example, looking for "Huffman Trees" is likely to give results related to the data structure, while looking for "lists" or "queues" is likely to return information unrelated to computer science. Unfortunately, some data structures have common, everyday names.

Visualizations constitute only a small part of the courseware materials available on the web. Far more artifacts exist that can be described as content presentations (lecture materials or tutorials) or projects or exercises (assignments). Thus, to find a given visualization, some sort of restrictive keywords are necessary for searches on most topics. Unfortunately, the providers of algorithm "visualizations" and "animations" often do not label them as either, so these search terms will not find them. Nor is there any standard alternative synonym that we know of that can capture the typical visualization. Since it turns out that the vast majority of visualizations written since the mid 1990s were written in Java, and many of these are delivered as applets, "applet" is often a successful choice of keyword. Unfortunately, this will tend to leave out those visualizations that exist within a visualization system, since they are not typically presented to the world labeled as "applets." Using "applet" as a keyword also results in generating a self-fulfilling prophecy that if you search for applets, then the only presentation mechanism you will find will be applets. Initially, this skewed the balance of the materials found in our wiki away from projects with integrated applications, since non-applets were intrinsically harder for us to find. But we have since made a conscious effort to catalog non-applets, and found that these make up a significant fraction of the total (they are still written in Java, however).

The main alternative to keyword-based Internet search is to look in visualization or courseware repositories or link collections. Unfortunately, there currently is little in the way of good repositories for data structures and algorithm visualizations. Within the computer science community, the most likely candidates are the collection of materials submitted to JERIC [12], and the CITIDEL repository [4]. JERIC and its contributed courseware are indexed as part of the ACM Digital Library [1]. While the ACM DL and CITIDEL are both huge collections, neither appears to have large amounts of courseware in general or visualizations in particular. Equally important, neither provides good search tools for courseware or visualizations. The bulk of their materials are papers, and they tend to organize by content topic (CITIDEL) or publication venue (ACM DL). Neither supports browsing for courseware separate from the (overwhelming) body of non-courseware content. SIGCSE maintains a collection of courseware links which includes a small number of visualizations [16]. Broader courseware repositories include SMETE [17], and Connexions [5]. None of these are well known within the CS education community, and none have large collections of visualizations. Further, none of the repositories mentioned here have much "web presence" with respect to algorithm visualization. In all the hours that we have spent conducting google searches for various visualizations, not a single visualization within any of these repositories was discovered by that means.

Courseware repositories have the potential to be far superior to a basic Internet search when looking for visualizations, for at least four reasons.

- Internet-based keyword searches do not help the instructor who is searching for good ideas. By browsing a list of visualizations and courseware for lower-division data structures and algorithms topics, the instructor might come upon different treatments and approaches than she might have found by listing topics and searching explicitly for them. She might even discover visualizations that encourage her to present new topics.
- Visualizations maintained by individuals are prone to loss of access. The Internet is well known for high turnover of material, either because URLs change, or because the material is no longer made available. Both keyword-based search of the Internet and collections of visualization links are susceptible to this problem. Only curated repositories hosted by stable providers

can give any assurance for long-term sustainability.

- A large fraction of existing visualizations are unusable, either because they simply do not work, or because they are poorly conceived and implemented. A repository, or a curated site of visualization links, could provide editorial guidance to educators regarding the quality of the visualizations. Such sites might pick and choose which visualizations to provide, or they might supply ratings information.
- A repository or curated site that developed its web presence could allow educators to actively participate in the site, using social bookmarking schemes, like `del.icio.us`, or user ratings, like `Amazon.com`, or features of other social networking tools. By allowing users to comment on, rate, and cross-link resources, educators would be able to add value to the entire collection over time just by using it. MERLOT [14] is one repository moving in this direction.

4. CONCLUSIONS AND FUTURE PLANS

While many good algorithm visualizations are available, the need for more and higher quality visualizations continues. There are many topics for which no satisfactory visualizations are available. Yet, there seems to be less activity in terms of creating new visualizations now than at any time within the past ten years. On the other hand, the theoretical foundations for creating effective visualizations appear to be steadily improving. More papers are appearing regarding effective use of visualization in courses, and a body of work has begun to form regarding how to develop effective visualizations in the first place. Collectively, the community is learning how to improve. While more fundamental research on how to develop and use algorithm visualizations is necessary, we also simply need more implementors to produce more quality visualizations. And we need to encourage them to provide visualizations on under-represented topics.

Given the existence of effective visualizations, there is still a major gap in terms of a credible national or international repository for visualizations and other courseware. Some repositories exist now (CITIDEL and the JERIC/ACM DL, for example), but they need to mature before they can claim to provide an adequate solution to this problem. Thus, a major concern for computer science educators should be how the gains made in knowledge and artifacts can be disseminated effectively so as to insure continued improvements.

Our own future plans include developing our wiki into a resource that can provide value to the community. We continue to expand our collection of links to visualizations. We are developing a bibliography of the research literature, a catalog of projects and algorithm visualization systems, and a directory of researchers. Much of the data reported here was hand generated. We are creating data collection tools to allow us and the CSE community to data mine our database. We have developed evaluation forms and a mechanism for collecting them, and tools for capturing public comments and ratings. If we can enlist significant community help, we plan to provide evaluations of existing visualizations, and thereby better pinpoint what is available and what is lacking, both in terms of quantity and quality. In these ways, we can make progress toward the day when computer science educators have a rich collection of high-quality visualizations that span all the traditional topics that we teach.

5. REFERENCES

[1] Association for Computing Machinery. The ACM digital library. `http://portal.acm.org`, 2006.

[2] M. H. Brown and R. Sedgewick. A system for algorithm animation. In *SIGGRAPH '84: Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques*, pages 177–186, New York, NY, USA, 1984. ACM Press.

[3] M. D. Byrne, R. Catrambone, and J. T. Stasko. Do algorithm animations aid learning? Technical Report GIT-GVU-96-18, Georgia Institute of Technology, 1996.

[4] CITIDEL: Computing and information technology interactive digital educational library. `http://www.citdel.org`, 2006.

[5] Connexions scholarly content repository. `http://cnx.org`, 2006.

[6] J. S. Gurka and W. Citrin. Testing effectiveness of algorithm animation. In *Proceedings, IEEE Symposium on Visual Languages*, pages 182–189, 1996.

[7] S. R. Hansen, N. H. Narayanan, and D. Schrimpsher. Helping learners visualize and comprehend algorithms. *Interactive Multimedia Electronic Journal of Computer-Enhanced Learning*, 2, 2000.

[8] Hope College. Complete collection of algorithm visualizations. `http://www.cs.hope.edu/~dershem/ccaa/ccaa`, 2006.

[9] C. Hundhausen and S. Douglas. Using visualizations to learn algorithms: should students construct their own, or view an expert's? In *Proceedings, IEEE Symposium on Visual Languages*, pages 21–28, 2000.

[10] C. Hundhausen, S. A. Douglas, and J. T. Stasko. A meta-study of algorithm visualization effectiveness. *Journal of Visual Languages and Computing*, 2002.

[11] D. J. Jarc, M. B. Feldman, and R. S. Heller. Assessing the benefits of interactive prediction using web-based algorithm animation courseware. In *SIGCSE '00: Proceedings of the Thirty-First SIGCSE Technical Symposium on Computer Science Education*, pages 377–381, New York, NY, USA, 2000. ACM Press.

[12] JERIC: Journal on Educational Resources in Computing. `http://www.acm.org/pubs/jeric`, 2006.

[13] A. W. Lawrence, J. Stasko, and A. Badre. Empirically evaluating the use of animations to teach algorithms. In *Proceedings, IEEE Symposium on Visual Languages 1994*, pages 48–54. IEEE Computer Society, 1994.

[14] Multimedia Educational Resource for Learning and Online Teaching. `http://www.merlot.org`, 2006.

[15] P. Saraiya, C. Shaffer, D. 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, Norfolk, VA, March 2004.

[16] SIGCSE educational links. `http://sigcse.org/topics`, 2006.

[17] SMETE digital library. `http://www.smete.org`, 2006.

[18] Virginia Tech Data Structures and Algorithm Visualization Research Group. Data structures and algorithm visualization wiki. `http://web-cat.cs.vt.edu/AlgovizWiki`, 2006.