A Tool for Teaching Advanced Data Structures to Computer Science Students

An Overview of the BDP System

Katrin Becker and Melissa Beacham

Abstract

In order to design and write effective, robust code using advanced data structures, it is crucial to achieve a thorough understanding of the algorithms used to manipulate these structures. One means of accomplishing the task is to provide students with a graphical, animated system that allows users to observe changes that the structure undergoes while it is being used. One such system has been developed which demonstrates B-Trees. Some preliminary testing is complete and some initial reactions of the students who have tried the system are outlined.

Introduction

Advanced data structures are a critical part of any computer science learning curriculum. These structures are used constantly in production level projects throughout the information technology industry, and in order to develop and maintain these products, a complete understanding of the manner in which these data structures work is a necessity. However, due to the complexity of these structures, it is often difficult to impart a working knowledge of the creation and operation of these constructs to students by using traditional communication and delivery methods. Students often have trouble translating notes on a page into a comprehensive understanding of what actually occurs in the data structure itself. Even the most detailed diagrams of these structures still leave some operations to the imagination, and the student is often left with the impression that somewhere between steps "a miracle happens" and the correct end result is achieved.

A better approach for demonstrating how these structures actually work exists through the use of graphical tools. These tools can provide a more detailed, step-by-step walkthrough of how operations are performed on the structure and allow the student to explore each step of the process at his own individual pace. A tool that allows students to manipulate data structures in many different ways can facilitate a deeper understanding of how the structure behaves than can be achieved by looking at a few preset (and usually conveniently chosen) examples and then being left to generalize to the rest of the structure. Many students learn best by employing a "visual" approach. When provided with a representation of the structure they can see and interact with rather than a picture of such a structure it is often easier to comprehend and understand than an abstract representation of a structure that is static and pre-determined.

Teaching Data Structures

The well-worn standard approach of lectures supplemented with static visuals is still the most common method of teaching data structures, and most courses at the university level follow this format. Classes in data structures are often supplemented with lab components, where the students have the opportunity to ask questions and attempt to understand concepts they may have had trouble grasping during the lecture period. Often lab instructors have access to the same source notes and tools as the instructor and only a blackboard or overhead projector by way of audio-visual equipment. Lab instructors will use what they have to attempt to sketch out the behaviours of the structure in question but static media don't lend themselves well to such dynamic structures. Even the most detailed diagrams can only handle a finite (and small) number of specific examples leaving the student to puzzle out the remaining "What if" questions. A utility that allows the student to interact with the diagram and try out various operations and then see the results in the structure when the algorithms required for its manipulation are *correctly* applied would allow students to achieve a deeper and more thorough understanding of the topic at hand than most traditional approaches currently do.

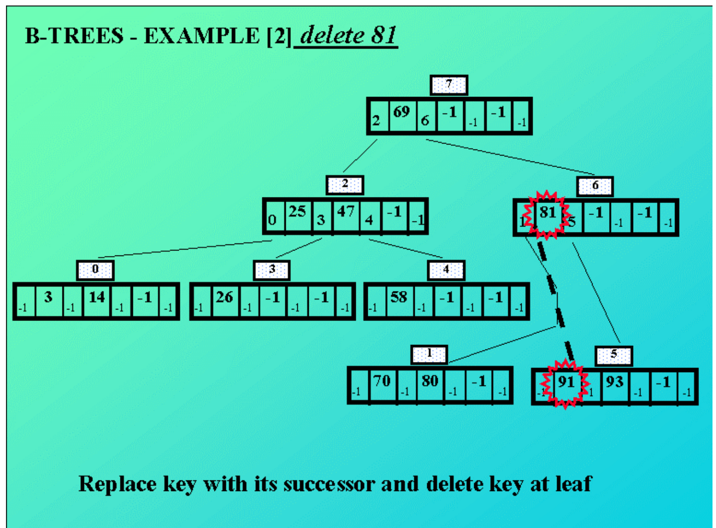

Figure 1 - Typical slide produced to supplement lectures.

The B-Tree Demonstration Program (BDP) is designed to assist students in understanding the B-Tree data structure as defined by Knuth [KNUTH98]. Students often experience difficulty understanding tree structures, partly because algorithms to manipulate them are often recursive and partly because small changes to the underlying list of data items can result in large changes to the structure. The BDP system provides users with an interactive version of a B-Tree that allows them to add and delete nodes at will and to watch how the tree reacts to these different situations. Students can choose keys to insert into the tree and see how the tree will change and balance. They can also delete keys and watch how the tree rearranges itself to conserve maximum efficiency. This capability is a much more dynamic means of conveying to the students how operations performed on a tree can affect the overall structure of the tree and provides users with an unlimited capacity to explore how different operations will impact the rest of the structure.

The system provides both students and instructors with an effective tool to demonstrate the workings of a B-Tree. Instead of relying on transparencies with concrete examples prepared prior to the lecture, instructors can respond to the students' questions about how a specific operation might affect a tree without attempting to draw the tree on the fly. The tool provides the answers as each change is entered into the system, leaving the instructor merely to explain why the action occurred as it did. The use of this tool will also save instructors valuable time that is often spent manually drawing the structure and then correcting the inevitable errors that appear the first time an example is tried while at the same time trying to answer students' questions. The tool can be relied upon to correctly implement the algorithms and effect the changes, leaving the instructor free to provide explanations as they happen.

Description of the System

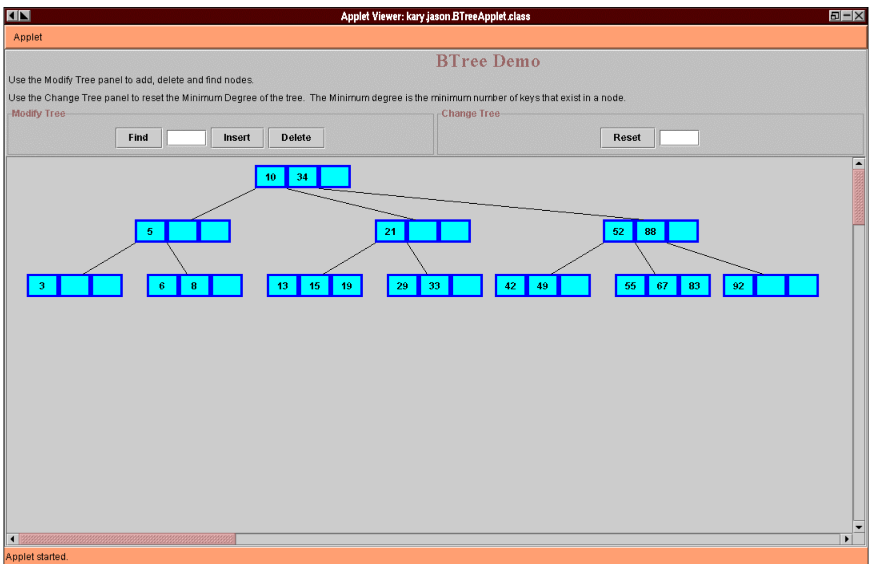

Figure 2 - Initial configuration of BDP

The BDP system is designed to run from any web browser, using a Java 1.2 plug-in to access the powerful capabilities of the language to animate the structure within an applet window. Both Internet Explorer 5.0 and Netscape Communicator 4.6 will support the applet and run the program within the browser window. The system was designed using the Java 1.2 language standard (commonly referred to as Java 2). This standard provides the system with the speed and graphical capabilities to redraw the tree quickly after an operation is performed and make the necessary changes to the internal structure of the tree to calculate the appropriate changes and display them.

The system was implemented on a web-browser to minimize the knowledge of Java and applet design required in order to run it. Accessibility from a web browser virtually ensures that all students in a data structures class will possess the necessary knowledge to make use of the system. In addition, the use of HTML documents surrounding the system applet window allows the designers to display instructions for the system's use as well as to seamlessly post updates to the system as improvements are added.

The system consists of a single pane that contains the B-Tree as it is currently implemented and above the tree are buttons that control the automation. The tree exists as a scrollable window that enables the user to view different parts of the tree by scrolling both left-to-right and up-and-down. This capability is necessary since the trees can grow very wide as the tree attempts to maintain a balanced arrangement of keys within the nodes. The original tree displayed is an order-4 B-Tree, meaning that each node within it contains a maximum of three keys and four pointers to subsequent nodes. The minimum degree of the tree, defined as the maximum number of keys that can exists within a node before the node will be combined with another node, is one for this example.

User's Manual

There are three basic operations that the user can perform on the B-Tree displayed on the screen. The user may insert a key into the tree, delete an existing key from the tree, or search for a key within the tree.

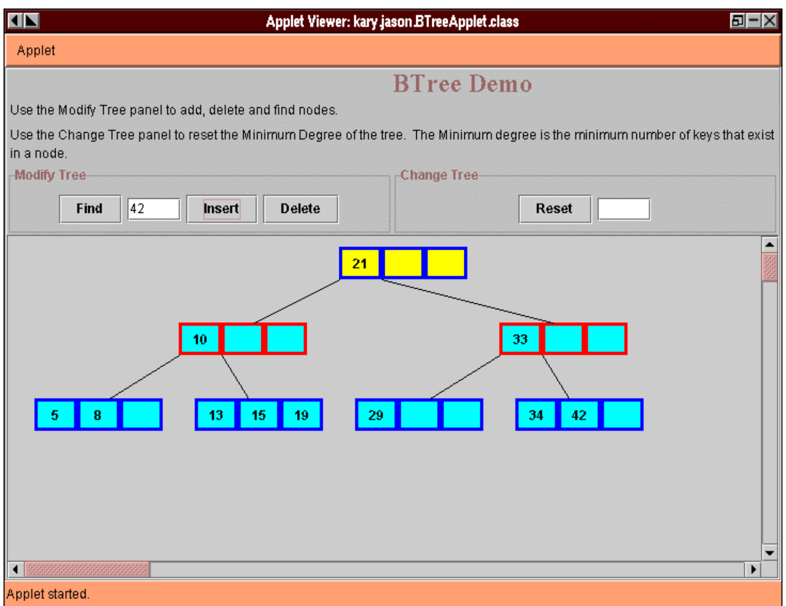

Figure 3 - After insertion

Insert Key

To insert a key into the tree, the user enters the value of the key into the next field at the top of the screen and presses the Insert button to place the key in the appropriate spot. The BDP system is designed to accept numeric keys between 0 and 1000, however it will accept higher value keys. Negative keys are rejected. After pressing the Insert button, the system inserts the key into the tree and highlights any nodes that were split to place the key in a different colour. This colouring of nodes that are split assists the student in determining how the tree handles overflow within a node by creating new nodes.

Delete Key

To delete a key from the tree the user enters the value of a currently existing key and presses the Delete button. The system removes the key from the tree and rearranges the remaining keys and nodes within it to maintain the minimum degree of the tree.

Search Key

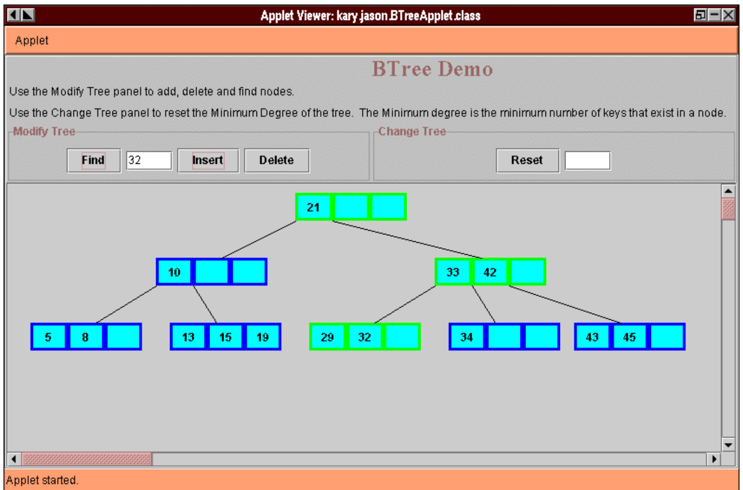

Figure 4 - Nodes visited during search

Finally, the user may search for a key in the tree and see how many nodes are inspected before the key is found. There are two methods of searching employed by the system: a recursive search following pointers down the tree to the next level and a linear search for a key within a node to find either the target key or a pointer to the next node that might contain the target key. Since a linear search presents few problems for most intermediate students the system does not show this process. It does highlight every node that is "touched" during this process in a different colour. This helps the user follow along with the steps in the algorithm implemented to find a key and provides a better understanding of why a B-Tree provides an efficient means of storing large amounts of data for quick retrieval.

Future Additions

Future revisions of the BDP program will introduce new features to increase the tool's effectiveness. One such improvement would be to allow a student to "step through" each portion of an insertion and deletion to see the steps that produce the final tree as displayed. By allowing students to view each "decision" that the system makes, the process of understanding why a tree places a key in particular position is explicitly outlined for the student. Allowing the user to specify a different order for the tree would also further expanding the utility of the system.

Student Reaction

The BDP system was presented to current members of a File Architecture class at the University of Calgary for the purpose of obtaining casual feedback related to the system's design and utility. The system was made available towards the end of the semester and as a result feedback has been limited. Comments that were received were encouraging.

Conclusion

The BDP system is an attractive and convenient alternative to traditional methods of illustrating the B-Tree data structure that can be actively manipulated in lectures, laboratory sessions, as well as by the students themselves out of class. The system provides a reliable demonstration of a B-Tree structure in practice and affords instructors an opportunity for a multitude of examples of all typical operations that may be performed on such a tree. This dynamic ability to respond to various situations frees instructors to spend a greater proportion of their lecture time discussing the algorithms themselves and does not restrict them to a few specific examples prepared in advance. In addition, it provides a tool that students can explore at their leisure outside of class time. This allows instructors to focus on the more difficult questions regarding the B-Tree structure as opposed to the basic concepts supplemented with simplistic examples. The combination of availability during lectures, labs, and personal study time gives students an opportunity to develop a consistent knowledge base upon which they can build and later create more complex, effective, and robust programs. The BDP system will prove to be a valuable asset to students and instructors alike.

References

[KNUTH98] Knuth, Donald E. (1998) The Art of Computer Programming Volume 3 Second Edition, Addison-Wesley Reading, MA. Ch. 6.2.4

Web-Based Resources

http://www.cs.princeton.edu/~ah/alg_anim/gawain-4.0/ - describes the animation of geometric algorithms using Java 1.1 and a program called Gawain used primarily to animate sorting algorithms

http://cse.uta.edu/~holder/courses/cse2320/lectures/applets/rbt/RedBlackTreeExample.html - very interesting animation of a Red-Black Tree

http://rowlf.cc.wwu.edu:8080/~n9442097/bt.html - a binary tree viewer written in an undefined version of Java

http://techunix.technion.ac.il/~itai/ - presents a program to animate splay trees with many user controls