Game-based Learning Approach to Binary Search Tree Operations

Süleyman Eken *, Kocaeli University, Department of Computer Engineering, TR-41380 Umuttepe, Kocaeli, Turkey.
Ahmet Sayar, Kocaeli University, Department of Computer Engineering, TR-41380 Umuttepe, Kocaeli, Turkey.

Suggested Citation:

Received December 05, 2012; revised February 10, 2013; accepted March 18, 2013.
Selection and peer review under responsibility of Prof. Dr. Fahrettin Sadikoglu, Near East University.
©2013 Academic World Education & Research Center. All rights reserved.

Abstract

Games perform a certain number of educational purposes. In this respect, they aim to motivate students to learn more effectively and encourage them to learn from their mistakes. Subjects of Data Structure and Algorithms course offered to computer science students are very virtual and are very difficult to interpret visually. In this study, we develop a computer game for teaching of basic operations (construction, insertion, deletion) of binary search trees (BST) including rotation operation to convert unbalanced BST to AVL tree. BST operations and rotation are mostly complex concepts encountered by IT students. We test effectiveness of the proposed system by an examination. An examination related to BST operations and rotation is applied to two groups of students in Kocaeli University Computer Engineering Department: one group takes classical education and the other group learns the subject by means of computer games. The findings show that the second group learns BST operations and rotation better than the first group.

Keywords: Binary search tree, data structure, computer education with games, game-based learning, educational computer game;

* ADDRESS FOR CORRESPONDENCE: Süleyman Eken Kocaeli, University, Department of Computer Engineering, TR-41380 Umuttepe, Kocaeli, Turkey, E-mail address: suleyman.eken@kocaeli.edu.tr / Tel.: +90-262-303-3596
1. Introduction

Data Structures and Algorithms are essential topics in computer science education. These topics are very difficult to understand and interpret virtually by students. Moreover, as courses on algorithms and data structures are mostly performed by code writing and are taught via classical education, students scarcely ever find it interesting. Games perform a certain number of educational purposes. In this respect, they aim to motivate students to learn more effectively and encourage them to learn from their mistakes. Games are preferred to all kinds of teaching methods and they are an innate behaviour of human beings. A baby prefers his/her meal with a game. A baby eats better when the food is served with a game. Similarly, students learn easier when the materials are taught with games.


In this paper, a game-based approach is developed to provide a helpful method for studying basic operations, which are performed on BST and AVL trees. BST operations (construction, insertion, and deletion) and rotation operation to convert an unbalanced BST to an AVL tree are mostly complex concepts to perceive easily. So, an application is developed in this context to learn BST and AVL trees more enjoyably. By this application, students, which studied theoretically, could be put into a visual practice environment that consolidates their knowledge. This application can also be used by teachers as an education material in the classroom. The operations examined in the application are adding a node, deleting a node, rotation and balancing the tree.

The remainder of this paper is organized as follows. In Section 2, BST operations, rotation and balancing are explained. In Section 3, methodology of the study is presented. Section 4 draws a conclusion and future works.

2. Interactive Graphics Tools for BST

Data structures are the ways of organizing and storing data sets in a computer to be able to use them efficiently. According to the algorithm, data need to be adjusted data structures such as arrays, stacks, queues, trees and graphs. Different applications require different kinds of data structures to get best results in terms of performance and usability. In a tree data structure, data held in a hierarchical structure are composed of nodes and branches. Parts of the data or themselves are kept at nodes, and branches indicate connection relationships between other nodes. Due to the nature of tree data structure, it is suitable solution or modelling of a lot of problem. For this reason, there are many varieties of tree. BST data structures are one of the most widely used varieties of trees. BST makes data kept in memory easily searchable. The search time becomes O(logN) for an n-node BST. The most important aspect of BST is searching on it.

BST has the following conditions and rules with respect to tree data structures. (i) Each node has at most two child nodes representing two sub-trees. Each tree is also a binary search tree. (ii) Every node’s key is larger than all keys in its left sub-tree. (iii) Every node’s key is smaller than all keys in its right sub-tree. (iv) There must be no duplicate nodes.
2.1. BST Operations

Operations on BST can be listed as follows: (1) Construction, building of a tree; (2) destroying, deleting all items in the tree, (3) empty-check, if the tree is empty or not. It returns true if the tree is empty, otherwise false. (4) Adding-inserting, inserting a new node into the tree, (5) deleting, deleting a node from a tree, (6) traversing, accessing and processing an item in the tree, (7) searching-searching an item in the tree.

In this study, the operations that examined for BST are adding a node, deleting a node. Inserting a new node into a BST need to follow the left or right branch to down the tree until null sub-tree is found. There are four possible cases when we delete a node: (i) the node (leave node) to be deleted has no children. (ii) The node to be deleted has only a right sub-tree. (iii) The node to be deleted has only a left sub-tree. (iv) The node to be deleted has two sub-trees.

2.2. AVL Tree and Rotation Operation

AVL tree (height-balanced BST) is a search tree in which the heights of the sub-trees differ by no more than one. Let $H_L$ represent the heights of left sub-trees and $H_R$ represent the heights of right sub-trees. Absolute value of AVL sub-trees height must equal or less than 1 ($|H_L - H_R| \leq 1$). Rotation is a transformation process to convert unbalanced binary search tree to AVL tree. Unbalanced BST falls into one of four cases [10]:

- Left of left (LL): Left sub-tree is higher than the right sub-tree, before and after adding an element.
- Right of right (RR): Right sub-tree is higher than the left sub-tree, before and after adding an element.
- Right of left (RL): Initially left sub-tree is higher, after adding an element right sub-tree is higher.
- Left of right (LR): Initially right sub-tree is higher, after adding an element left sub-tree is higher.

2.3. User Interface and Use Cases

Developed game is a web application running on the client side. The game can be operated as desktop application by copying to any computer or can be accessed via the Internet by installing to a server. We have prepared our game with HTML5 and oCanvas that is open source JavaScript library [11]. The library enables working with objects instead of working with pixels. Figure 1 shows interface of our game.

When the user opens the game, the main menu showing BST basic operations and rotation operation is displayed. The user chooses one of the nine operation sections and starts playing. When any section is selected, one of the trees of the selected section is displayed. We consider status of AVL balancing (LL-RR case) as an example (see Figure 2). The operation/operations expected to perform by the user is shown at the bottom of screen. Here, the operations are “Add the given node 71 and balance the tree”. The node is to be added or deleted is placed on operation explanation. User throws the nodes to basket in deleting operation. The user can work freely on the tree. The tree can be taken shape in desired form by dragging or dropping nodes. Spots appearing on the screen help the user for placing nodes. When the user finishes the operation, he/she pushes “OK” button to check if the process is correct or not. Also, the user can view the correct result of the operation by clicking “Solution” button. If the user wants to initiate the game at anytime, the user can click “Reset” button. If the user wants to play for different tree of same operation, the user can click “New” button.
3. Methodology

This section presents the methodology of the study on game based learning infrastructure for BST basic operations. The main objective of this research is to measure the success of the proposed study by comparing it with the classical education.

To test effectiveness of the proposed system by an examination, we arrange a sample space among the second year IT students. An examination related to BST operations and rotation is applied to two
groups of students in Kocaeli University Computer Engineering Department: one group takes classical education and the other group learns the subject by means of computer prepared game.

BST and AVL education is given to two groups over a period of two weeks. During the classical education, just verbal and written education materials are provided with the first group of students. After a brief explanation about BST operations and rotation operation, prepared game is given to second group to play and learn the topic. We test effectiveness of the proposed system by an examination. An examination related to BST operations and rotation. Learning performance of the students is measured by the following questions:

- Add or delete node ’X’ on a given binary search tree.
- Perform LL rotation for the given node and balanced tree.

4. Results and Future Works

The findings of the examinations show that the second group learns BST operations and rotation better than the first group. The average grade is about 60 for the first group; 87 for the second group. So, learning BST operations and rotation operation with game-based leaning is very helpful for students and timesaving method for both students and teachers. In the future, we will create game-based learning tools for some other data structures such as arrays, stacks, queues, heaps, and sets. Also, we will increase the visual aspect of the application with AJAX and Web 2.0 technologies.

References


