Sacred Heart University
DigitalCommons@SHU

School of Computer Science & Engineering Faculty Publications

School of Computer Science and Engineering

7-15-2019

Teaching Self-Balancing Trees Using a Beauty Contest

Samah Senbel
Sacred Heart University, senbels@sacredheart.edu

Follow this and additional works at: https://digitalcommons.sacredheart.edu/computersci_fac

 Part of the Systems Architecture Commons

Recommended Citation

Senbel, S. (2019). Teaching self-balancing trees using a beauty contest. Proceedings of the 2019 ACM *Conference on Innovation and Technology in Computer Science Education*. Scotland: Aberdeen. July 15. Doi: 10.1145/3304221.3325544

This Conference Proceeding is brought to you for free and open access by the School of Computer Science and Engineering at DigitalCommons@SHU. It has been accepted for inclusion in School of Computer Science & Engineering Faculty Publications by an authorized administrator of DigitalCommons@SHU. For more information, please contact ferribyp@sacredheart.edu, lysobeyb@sacredheart.edu.

Teaching Self-Balancing Trees Using a Beauty Contest

Samah Senbel
School of Computer Science & Engineering
Sacred Heart University
Fairfield CT USA
senbels@sacredheart.edu

ABSTRACT

Trees data structures and their performance is one of the main topics to teach in a data structures course. Appreciating the importance of tree structure and tree height in software performance is an important concept to teach. In this paper, a simple and amusing activity is presented. It demonstrates to students the importance of a well-balanced tree by comparing the height of a binary search tree to a balanced (AVL) tree build upon some personal data to find the "prettiest" tree (minimum height). The activity highlights the fact that, irrelevant of your data sequence, a balanced tree guarantees a height of O(log n) and everyone "wins" the beauty contest.

CCS CONCEPTS

• Social and Professional topics → Computing education programs

KEYWORDS: Self-balancing trees; AVL trees; Binary Search Trees; Tree Height; Search Complexity

ACM Reference format:
Samah Senbel. 2019. Teaching Self-Balancing Trees Using a Beauty Contest. In *Proceedings of the 24th Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE'19), 15-17/July, 2019, Aberdeen, Scotland, United Kingdom.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3304221.3325544

1 INTRODUCTION

Data structures is a significant part of the foundation of computer science for computer programming students. It introduces students to the different data structures and their uses, properties and implementation. This is a core part of software design and development [5]. At Sacred Heart University, the data structures course is taught in the second semester of the freshman year for all computer science and engineering students. It starts by covering the simpler linear data structures: arrays, linked lists, stacks, queues, and hashing, then trees of all types; binary search trees (BST), AVL trees [1] or red-black trees, heaps, and B-trees, and finally graphs are introduced towards the end of the course.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the Owner/Author.
ITiCSE '19, July 15–17, 2019, Aberdeen, Scotland UK
© 2019 Copyright is held by the owner/author(s).
ACM ISBN 978-1-4503-6301-3/19/07.
https://doi.org/10.1145/3304221.3325544

Students often have no problem understanding the concepts and uses of the different data structures, but struggle with the analysis, comparison, and implementation of the different structures. Therefore, it is important to introduce as many demonstrations, examples, memorable activities and interesting assignments as possible [4].

An important concept to teach in a data structures course is the different types of trees and their applications and performance measures. After studying the main concept of each tree type and its uses, we analyze and compare its performance to the others before starting on its implementation. The importance of the tree height in tree processing and performance cannot be understated [3].

Typically, the binary search tree and its implementation is covered first in its own lecture followed by a self-balancing tree: AVL trees and/or Red-Black trees. We introduce an exercise to be done at the beginning and middle of the self-balancing tree lecture.

2 ACTIVITY DESCRIPTION

The goal of this activity is to demonstrate the efficiency of AVL trees. Students build a BST from a set of personalized integers, check the height, and the student with the lowest height "wins". Then, an AVL tree is built from the same data, and now everyone has the same tree height and are all "winners" of the tree beauty contest.

The first issue is how big to make the data set to make sure all students get an AVL tree of the same height and "win". Figure 1 charts the eight range for both BST and AVL trees, as given in the following equations [2]:

$$\text{BST tree height:} \quad \log_2(n+1) \le h < n \qquad (1)$$

$$\text{AVL tree height:} \quad \log_2(n+1) \le h < 1.44 \log_2(n+2) - 0.328 \qquad (2)$$

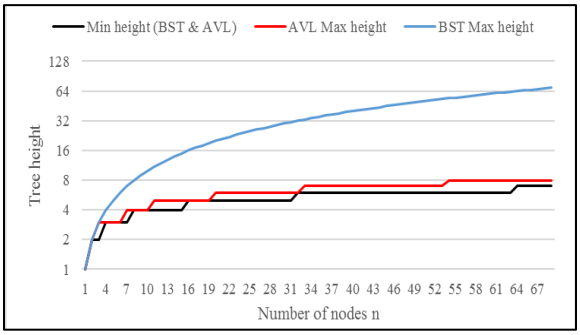

Figure 1: Height range of a BST and AVL tree

Based on figure 1, only 11 values of n give a fixed value of the AVL height: 4, 5, 6, 8, 9, 10, 16, 17, 18, 19, and 32. Any of these can be used in this activity, 16 was chosen as it is large enough to need a few rotations hopefully, and can be done in a short time.

Activity 1: (2 minutes) Each student is given a sheet with 16 questions that need to be answered with integer replies, such as "How many siblings do you have?" and "What is your dorm room/ home number?". The students are allowed to answer with wrong answers to protect their privacy. In case of a repeated number, they should change it to a new number.

Activity 2: (3 minutes) Using the exact sequence of answers, they draw a BST of their personal data. The resulting tree would have height of 5 to 16, depending on their answers.

Activity 3, Group discussion: (5-10 minutes) Students are asked for their BST height, and the student(s) with the shortest heights are declared the winner(s) and given a small reward (Sticker & Candy). Figure 2 shows a sample answer.

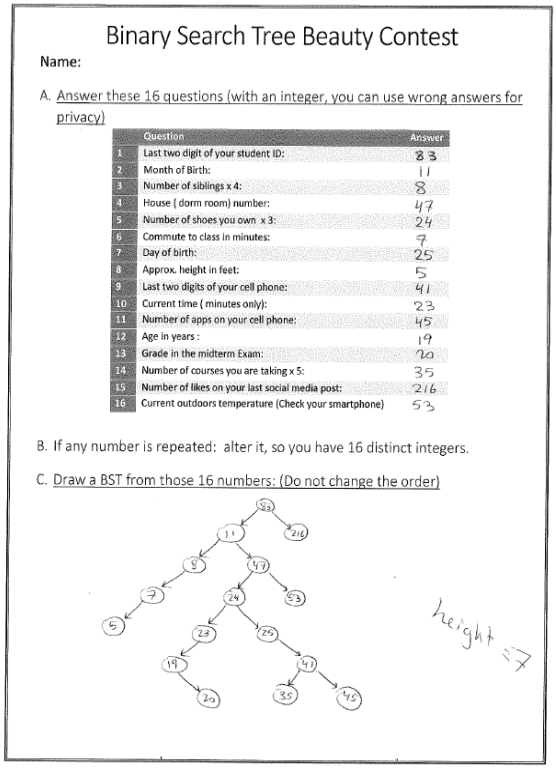

Figure 2: Sample BST answer for the contest

Having done these activities, the concept of self-balancing trees is explained and how the AVL trees does that by performing single and double rotations as needed. Several examples are solved to demonstrate the technique. Tree operations are briefly explained, as well as tree height (figure 1) and how it affects performance. This takes about 45-50 minutes. Before going into the details of actually coding the AVL tree, the following activities are performed:

Activity 4: (5 minutes) Using the same data set, the students build up their individual AVL tree. A sample is shown in figure 3.

Activity 5: (5 minutes): Students are asked for their AVL tree height, and as expected, all the students get a height of 5, and are all "winners" now. The same small reward is given to the entire class as they show their trees, and any errors found are fixed.

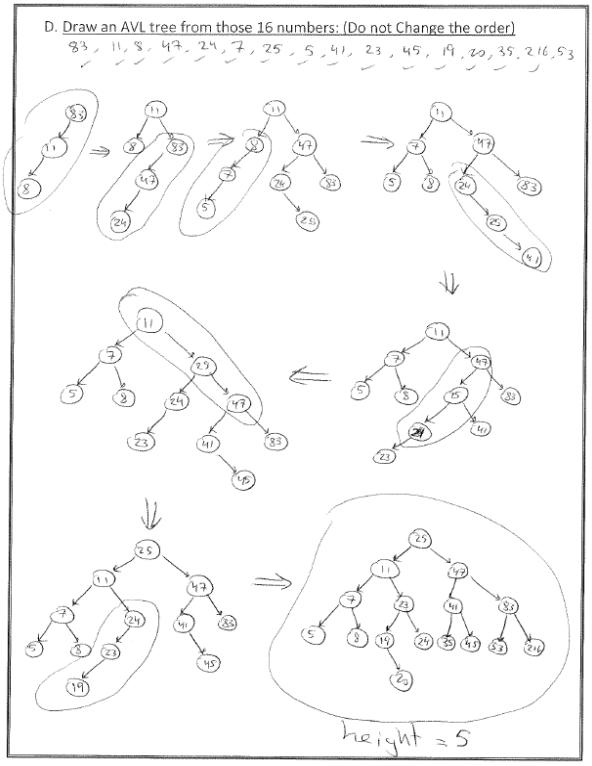

Figure 3: Sample AVL tree answer

Having set the mood of the class into a more positive mood, the students get a 15-minute break, and then we start on the intricacies of the actual coding of the AVL tree and its operations (about 75 minutes). Total class time is 2 hours and 45 minutes. This exercise has been practiced with freshman students over the past four years and proved to be an effective way to teach tree performance.

REFERENCES

[1] Adelson-Velsky, G. and Landis, E. 1962. An algorithm for the organization of information. In *Proceedings of the USSR Academy of Sciences (in Russian). Vol 146: pp. 263–266.* English translation by Myron Ricci in *Soviet Mathematics.* Doklady, vol 3: pp.1259–1263.

[2] Donald Knuth. 2000. Sorting and searching. Addison-Wesley, Boston, Massachusetts, USA. ISBN 0-201-89685-0.

[3] Koffman, E. and Wolfgang, P. 2015. Data Structures abstraction and design. Chapter 9, John Wiley Publishers, Hoboken, New Jersey, USA. ISBN-10: 1119355214.

[4] Lawrence, R. 2004. Teaching Data Structures Using Competitive Games. In *IEEE Transactions on Education, Vol. 47, no. 4.* IEEE.

[5] Liu, X., Wang, X. and Wang, R. 2013. Application of Blended Learning in Data Structures and Algorithms Course Teaching, In *Proceedings of the International Conference on Education Technology and Information System (ICETIS '13).* Atlantis Press, Paris, France.