CMPT 225

Lecture 22 – Tree Sort
Last Lectures

- We saw how to ...
  - Describe how a self-balancing binary search tree balances itself
  - Perform insertion into a self-balancing binary search tree, applying appropriate rotations when necessary
  - Express time efficiency of self-balancing binary search tree’s operations for various case scenarios
Learning Outcomes

At the end of the next few lectures, a student will be able to:

- Define the following data structures:
  - Binary search tree
  - Balanced binary search tree (AVL)
  - Binary heap
  as well as demonstrate and trace their operations
- Implement the operations of binary search tree and binary heap
- Implement and analyze sorting algorithms: tree sort and heap sort
- Write recursive solutions to non-trivial problems, such as binary search tree traversals
Today’s menu

- Our goal is to
  - Understand how tree sort works
  - Sort an array using tree sort
  - Analyze time/space efficiency of tree sort
Tree Sort

- Takes advantage of the ordering property of binary search trees (or AVL trees) to sort the data

- **Algorithm:**
  array = array of elements
tree = empty binary search tree (or AVL)
for index = 0 to (elementCount – 1) do
  insert array[index] into tree
for index = 0 to (elementCount – 1) do
  get the next inorder element from the tree
  store the element in array[index]
Let’s try!

- Phase 1
Let’s try!

- Phase 2
Time Complexity Analysis of Tree Sort

- Time complexity of Tree Sort algorithm:
  - building the BST tree
    - average case:
    - worst case:
  - building the AVL tree:
  - traversing tree and inserting back into array:
- Overall cost:
Space Complexity Analysis of Tree Sort

- How much space (memory) does tree sort require to execute?
  - Tree sort algorithm requires an additional $n$ space for the tree which can create a space problem when $n$ is large
- Therefore, its space efficiency is $O(n)$
Let’s compare all sort algorithms seen so far.
Learning Check

- We can now ...
  - Understand how tree sort works
  - Sort an array using tree sort
  - Analyze time/space efficiency of tree sort
Next Lecture

 Heap