DA5300: Data Structures for Data Science(Worksheet-2)

Questions are from the reference textbook "Data Structures and Algorithms in Python", GoodRich, Tamassia, GoldWasser

September 8, 2024

1 Tree

- The maximum number of nodes in a binary tree of height H is.
- The maximum number of leaf nodes in a binary tree of height H is
- The maximum number of internal nodes in a binary tree of height H is
- Two ordered trees T and T are said to be isomorphic if one of the following holds:

  1. Both T and T'' are empty.
  2. The roots of T and T'' have the same number $k \geq 0$ of subtrees, and the i^{th} such subtree of T is isomorphic to the i^{th} such subtree of T for $i = 1, \ldots, k$.

  Design an algorithm that tests whether two given ordered trees are isomorphic. What is the running time of your algorithm?
- Show that there are more than $2n$ improper binary trees with n internal nodes such that no pair are isomorphic.
- If we exclude isomorphic trees, exactly how many proper binary trees exist with exactly 4 leaves?
- Given a proper binary tree T, define the reflection of T to be the binary tree T' such that each node v in T is also in T', but the left child of v in T is v's right child in T' and the right child of $v$in T is v's left child in T. Show that a preorder traversal of a proper binary tree T is the same as the postorder traversal of T's reflection, but in reverse order.
- Design algorithms for the following operations for a binary tree T :

  1. preorder next(p): Return the position visited after p in a preorder traversal of T (or None if p is the last node visited).

2. inorder next(p): Return the position visited after p in an inorder traversal of T (or None if p is the last node visited).
3. postorder next(p): Return the position visited after p in a postorder traversal of T (or None if p is the last node visited).

What are the worst-case running times of your algorithms?

- Let T be a binary tree with n positions. Define a Roman position to be a position p in T , such that the number of descendants in p's left subtree differ from the number of descendants in p's right subtree by at most 5. Describe a linear-time method for finding each position p of T , such that p is not a Roman position, but all of p's descendants are Roman.
- Let T be a tree with n positions. Define the lowest common ancestor (LCA) between two positions p and q as the lowest position in T that has both p and q as descendants (where we allow a position to be a descendant of itself). Given two positions p and q, describe an efficient algorithm for finding the LCA of p and q. What is the running time of your algorithm?
- Let T be a binary tree with n positions, and, for any position p in T, let d_p denote the depth of p in T. The distance between two positions p and q in T is $d_p + d_q - 2d_a$, where a is the lowest common ancestor (LCA) of p and q. The diameter of T is the maximum distance between two positions in T. Describe an efficient algorithm for finding the diameter of T. What is the running time of your algorithm?
- Suppose each position p of a binary tree T is labeled with its value $f(p)$ in a level numbering of T . Design a fast method for determining $f(a)$ for the lowest common ancestor(LCA), a, of two positions p and q in T, given $f(p)$ and $f(q)$. You do not need to find position a, just value $f(a)$.

2 Heap

- What does each remove min call return within the following sequence of priority queue ADT methods: add(5,A), add(4,B), add(7,F), add(1,D), remove min(), add(3,J), add(6,L), remove min(), remove min(), add(8,G), remove min(), add(2,H), remove min(), remove min()?
- At which positions of a heap might the third smallest key be stored?
- At which positions of a heap might the largest key be stored?
- Is there a heap H storing seven entries with distinct keys such that a preorder traversal of H yields the entries of H in increasing or decreasing order by key? How about an inorder traversal? How about a postorder traversal? If so, give an example; if not, say why.

- Let H be a heap storing 15 entries using the array-based representation of a complete binary tree. What is the sequence of indices of the array that are visited in a preorder traversal of H? What about an inorder traversal of H? What about a postorder traversal of H?
- Bill claims that a preorder traversal of a heap will list its keys in non-decreasing order. Draw an example of a heap that proves him wrong.
- Hillary claims that a postorder traversal of a heap will list its keys in non-increasing order. Draw an example of a heap that proves her wrong.
- Show how to implement the stack ADT using only a priority queue and one additional integer instance variable.
- Show how to implement the FIFO queue ADT using only a priority queue and one additional integer instance variable.
- Suppose two binary trees, T1 and T2, hold entries satisfying the heap-order property (but not necessarily the complete binary tree property). Describe a method for combining T1 and T2 into a binary tree T , whose nodes hold the union of the entries in T1 and T2 and also satisfy the heap-order property. Your algorithm should run in time O(h1 + h2) where h1 and h2 are the respective heights of T1 and T2.