A Note on Erdős-Szekeres Theorem

Matthew Romig

Abstract

Erdős-Szekeres Theorem is proven. The proof is very similar to the original given by Erdős and Szekeres. However, it explicitly uses properties of binary trees to prove and visualize the existence of a monotonic subsequence. It is hoped that this presentation is helpful for pedagogical purposes.

1 Introduction

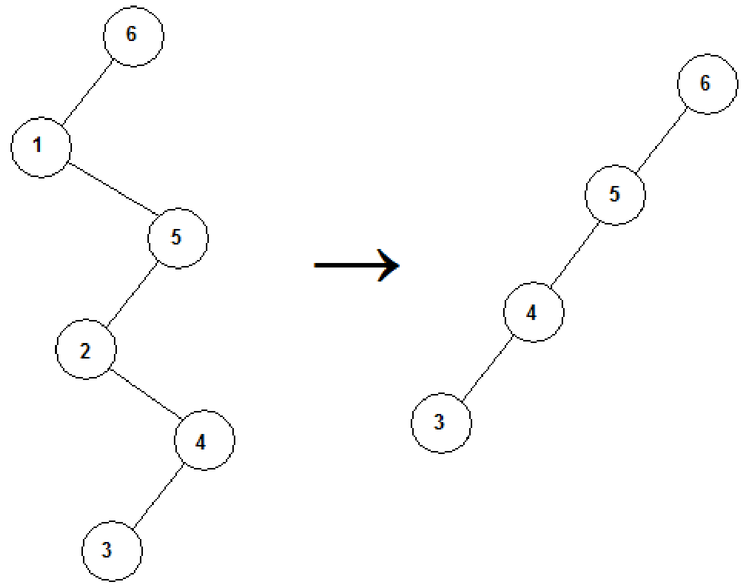
As in [1], the identity $n^2+1 = ((n-1)^2+1)+(2n-1)$ motivates the proof. Assume there is only one increasing or decreasing subsequence of size n in the first $(n-1)^2+1$ terms. Take the last term of the subsequence and remove it, leaving $(n-1)^2$ terms. Each of the other $2n-1$ terms then serves as the last term of a size n monotonic subsequence formed with the $(n-1)^2$ terms. A binary search tree is built from these $2n$ terms. Either the tree has a node with two children, or it has a depth of $2n-1$ and a single leaf. In either case, a monotonic subsequence of size $n+1$ may be found.

2 Proof

Lemma 2.1. *Given a binary tree having only one leaf and depth* $2n-1$*, there are nodes that can be deleted to make a new tree of depth* n *where either all child nodes are left children or they are all right children.*

Proof. Let l denote the number of nodes which have a left child and let r denote the number which have a right child. Because the depth is $2n-1$, it must be that $l+r = 2n-1$, as there is only one leaf and each parent can be paired bijectively with the edge connecting it to its only child. We must have either $l \geq n$ or $r \geq n$, as otherwise $l+r \leq 2n-2$. Without loss of generality, suppose $l \geq n$. We delete every node having a right child by the following procedure. Starting at the root, determine the type of child each non-leaf node has. If the node has a left child, do nothing. If the node has a right child, it is deleted and replaced with its right child. Specifically, the deleted node's parent (if it has one) takes the deleted node's child as its own child, see [2]. In both cases, repeat the procedure on the child node until the leaf is reached. When the leaf is reached, the only remaining non-leaf nodes must be those which originally had a left child. The procedure guarantees that each non-leaf node must have a left child. There must be n or greater such nodes as $l \geq n$. Since the leaf wasn't deleted, the tree has at least $n+1$ nodes. Finally, the tree still has only one leaf as each node had at most one child originally and deletion replaces that child with another. As a result, we end up with a tree which has all left children and a depth of n. See Figure 1 for a visual example. □

Figure 1: Lemma 1 for a tree with $n = 3$



Theorem 2.2. *Given a sequence of $n^2 + 1$ distinct real numbers, $n \geq 1$, there exists a monotonic subsequence of size $n + 1$.*

Proof. We use induction. For $n = 1$, the result is clear. Now suppose the theorem holds for integers $1, 2, \ldots, n - 1$. Given a sequence of $n^2 + 1$ distinct real numbers, we want to find an increasing or decreasing subsequence of size $n + 1$. Let the terms of the sequence be $a_1, a_2, \ldots, a_{n^2+1}$. By the induction hypothesis, there is at least one increasing or decreasing subsequence of size n in the first $(n - 1)^2 + 1$ terms. List the last terms $p_1, p_2, \ldots, p_s$ of each increasing or decreasing subsequence of size n. If a term is the last term for more than one such subsequence it is included only once, and it is associated with only one of the subsequences. Furthermore, the terms have been listed with respect to their order in the original sequence, that is if $1 \leq i < j \leq s$, $p_i = a_s$, and $p_j = a_t$ then $s < t$.

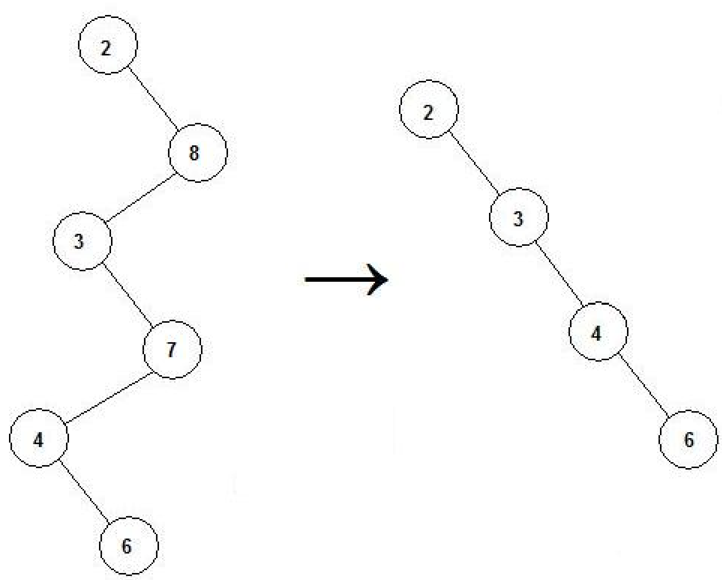
Now insert nodes corresponding to each term p_i into a binary search tree. See [2] for a description of binary search tree insertion. We just emphasize here that if $i < j$, then p_i is inserted before p_j. If the tree has $2n$ nodes, we are finished. Otherwise, remove the s terms from the original sequence. Again consider the first $(n-1)^2+1$ terms and repeat this procedure until the tree has $2n$ nodes.

We are guaranteed a binary search tree with $2n$ nodes by the following reasoning. If there is only one monotonic subsequence among the first $(n-1)^2+1$ terms, only one term is removed which forms the root of the tree. The remaining $(n-1)^2$ terms are then considered along with the first term $a_{(n-1)^2+2}$ of the last $2n-1$ terms of the original sequence. Since the number of terms considered is $(n-1)^2+1$ and there was only one monotonic subequence of size n which depended on the removed element, there must be at least one monotonic subsequence of size n whose last term is $a_{(n-1)^2+2}$. Similarly, when this term is removed $a_{(n-1)^2+3}$ which follows it becomes the last term of a size n monotonic subsequence. In this way, each of the last $2n-1$ terms is found to be the final term of a monotonic subsequence of size n.

If there was more than one monotonc subsequence among the first $(n-1)^2+1$ terms, some of the $2n-1$ terms won't be needed to build the tree. Assume there were s unique last terms from the subsequences. As described above, each of these terms is inserted into the tree. These s terms are removed and replaced with the next s terms not yet considered. If from this point on only one monotonic subsequence is found on each iteration of the procedure, the tree will still have $2n$ nodes as there are $2n-s$ terms that will be used. Finally, note that there can be more than one monotonic subsequence on any iteration of the procedure. Suppose the tree already has r nodes and $s>1$ unique last terms are found. A similar argument shows the tree will have $2n$ nodes as at least $2n-r-s$ terms will be used.

Each given node corresponds to the last term of a monotonic subsequence. If a node is a tail of an increasing subsequence and it has a right child, then we have found an increasing subsequence of length $n+1$, where the right child is last term. Likewise, if the node has a left child and is the tail of a decreasing subsequence, then we have found a decreasing subsequence of length $n+1$. In both cases this completes the proof. If for every node neither case holds, we must have built a tree that has only one leaf. Therefore, we have constructed a tree with $2n$ nodes of depth $2n-1$. By Lemma 2.1, we can delete some of the nodes to make a tree of depth n with either all left children or all right children. Reading off the values of each node from the root to the leaf thus gives a decreasing or increasing subsequence of length $n+1$. An example of a tree built and trimmed for a size 10 ($n=3$) sequence is shown in Figure 2.

Figure 2: Procedure applied to the sequence $10, 1, 9, 2, 8, 3, 7, 4, 6, 5$



□

References

[1] P. Erdős and G. Szekeres. A combinatorial problem in geometry. *Compositio Math.*, 2:463-470, 1935.

[2] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein. *Introduction to Algorithms, Third Edition*, 3rd ed. The MIT Press, 2009.

M. Romig
E-mail address: `mwr645@gmail.com`