The runtime should be O(nlogn).

Worst case for rebuild function is O(n), since we rebuild from the root. The rebuild function takes n nodes out of the subtree and return back in the form of a balanced search tree.

So the worst case for the insertion is O(n^2) if we rebuild at the root in every insertion, which will never happen. So the runtime should be smaller than O(n^2).

And because we should check a node in every height if never rebuild, the runtime should be longer than O(n)

So the runtime is O(nlogn)

Maximum height:

T(n)<nlogn

Prove:

T(n) means the height of a subtree with n-1 node inserting a node

T(1)=1 T(2)=T(1)+2

Root: node 2

T(3)=T(1) T(4)=T(2)+2^2

Root: node3

T(5)=T(2) T(6)=T(3) T(7)=T(4) T(8)=T(5)+2^3

So T(n)=T()+

So T(n)=p1\*2+p2\*2^2+p3\*2\*3+…+pk\*2^k, which n=2^k