Wilber 1 is not good enough. Suppose we have a BST (such as Tango trees) whose running time on an access sequence is upper bounded by some function \( T(m, n, k) \) of the number \( m \) of accesses, the number \( n \) of keys, and the number \( k \) of interleaves incurred by each access. (Assume for simplicity that every access incurs the same number of interleaves.) Determine the number \( S(m, n, k) \) of different access sequences with \( m \) accesses, \( n \) keys, and where each access incurs exactly \( k \) interleaves. Conclude that \( T(m, n, k) = \Omega(\lg S(m, n, k)) \), and compute the resulting lower bound on \( T \).

Link-cut trees with LCA. Recall that the least common ancestor (LCA) of two nodes \( u \) and \( v \) in a rooted tree \( T \) is the deepest node in \( T \) that is an ancestor of both \( u \) and \( v \). Describe how to modify link-cut trees in order to support efficient LCA(\( u, v \)) queries: given two nodes \( u \) and \( v \), find their LCA. You should support LCA queries in \( O(\lg n) \) amortized time per operation, while preserving the \( O(\lg n) \) amortized cost per link, cut, and findroot.