Wilber 1 is not good enough. Consider a path in the perfect BST $P$ that incurs $k$ interleaves. Such a path can be of length between $k$ and $\lg n$, and must change the preferred child of $k$ of its nodes. There are therefore $\sum_{i=k}^{\lg n} \binom{i}{k} = \binom{\lg n + 1}{k + 1}$ options for each access in the sequence. So

$$S(m, n, k) = \left(\frac{\lg n + 1}{k + 1}\right)^m.$$  

We can assume that our tree behaves differently on different request sequences (to see this, imagine requiring a special output symbol right after we find an element). Therefore, at least $\lg S(m, n, k)$ decisions are needed in order to distinguish between the different $S(m, n, k)$ access sequences. So

$$T(m, n, k) = \Omega(\lg S(m, n, k)) = \Omega(mk \frac{\lg n}{k}).$$

Notice that for such sequences, if $k$ is a constant then $T(m, n, k)$ is within a factor of $\lg \lg n$ from Wilber 1 and is tight with Tango trees.

Link-cut trees with LCA. The basic idea is: $access(u)$ then $access(v)$ and output the last node reached via parent pointers (when switching between auxiliary trees). The only problematic case (where there will be no parent pointers) is when $LCA(u, v) = v$. So if no parent pointer is traversed we output $v$. Notice that if $LCA(u, v) = u$ then we are fine because $access(u)$ makes all of $u$’s children unpreferred.