Principles of Distributed Computing
Exercise 12: Sample Solution

1 Concurrent Ivy

a) The three nodes are served in the order $v_2, v_3, v_1$.

b) Figure 1 depicts the structure of the tree after the requests have been served. Since $v_1$ is served last, it is the holder of the token at the end.

![Figure 1: Tree after the requests have been served.](image)

2 Tight Ivy

In order to show that the bound of $\log n$ steps on average is tight, we construct a special tree which is defined recursively as follows. The tree $T_0$ consists of a single node. The tree $T_i$ consists of a root together with $i$ subtrees, which are $T_0, \ldots, T_{i-1}$, rooted at the $i$ children of the root, see Figure 2.

First, we will show that the number of nodes in the tree $T_i$ is $2^i$. This obviously holds for $T_0$. The induction hypothesis is that it holds for all $T_0, \ldots, T_{i-1}$. It follows that the number of nodes of $T_i$ is $n = 1 + \sum_{j=0}^{i-1} 2^j = 2^i$.

We will show now that the radius of the root of $T_i$ is $R(T_i) = i$. Again, this is trivially true for $T_0$. It is easy to see that $R(T_i) = 1 + R(T_{i-1})$, because $T_{i-1}$ is the child with the largest radius. Inductively, it follows that $R(T_i) = i$.

By definition, when cutting of the subtree $T_{i-1}$ from $T_i$, the resulting tree is again $T_{i-1}$. Let $C: T_i \mapsto T_{i-1}$ denote this cutting operation. For all $i > 0$, we thus have that $C(T_i) = T_{i-1}$. We will now start a request at the single node $r$ with a distance of $i$ from the root in $T_i$. On its path to the root, the request passes nodes that are roots of the trees $T_1, \ldots, T_i$. All of those nodes become
children of the new root \( v \) according to the Ivy protocol. The new children lose their largest “child” subtree in the process, thus the children of node \( v \) have the structures \( C(T_1), \ldots, C(T_i) = T_0, \ldots, T_{i-1} \). Hence, the structure of the tree does not change due to the request and all subsequent requests can also cost \( i \) steps. Since \( n = 2^i \), each request costs exactly \( \log n \).