Principles of Distributed Computing
Exercise 7: 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.

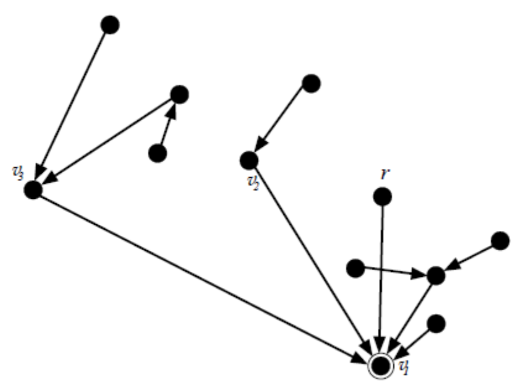

Figure 1: Tree after the requests have been served.

2 Tight Ivy

a) 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 $\mathcal{T}_0$ consists of a single node. The tree $\mathcal{T}_i$ consists of a root together with i subtrees, which are $\mathcal{T}_0, \ldots, \mathcal{T}_{i-1}$, rooted at the i children of the root, see Figure 2.

First, we will show that the number of nodes in the tree $\mathcal{T}_i$ is 2^i. This obviously holds for $\mathcal{T}_0$. The induction hypothesis is that it holds for all $\mathcal{T}_0, \ldots, \mathcal{T}_{i-1}$. It follows that the number of nodes of $\mathcal{T}_i$ is $n = 1 + \sum_{j=0}^{i-1} 2^j = 2^i$.

We will show now that the radius of the root of $\mathcal{T}_i$ is $\mathcal{R}(\mathcal{T}_i) = i$. Again, this is trivially true for $\mathcal{T}_0$. It is easy to see that $\mathcal{R}(\mathcal{T}_i) = 1 + \mathcal{R}(\mathcal{T}_{i-1})$, because $\mathcal{T}_{i-1}$ is the child with the largest radius. Inductively, it follows that $\mathcal{R}(\mathcal{T}_i) = i$.

By definition, when cutting off the subtree $\mathcal{T}_{i-1}$ from $\mathcal{T}_i$, the resulting tree is again $\mathcal{T}_{i-1}$. Let $\mathcal{C} : \mathcal{T}_i \mapsto \mathcal{T}_{i-1}$ denote this cutting operation. For all $i > 0$, we thus have that $\mathcal{C}(\mathcal{T}_i) = \mathcal{T}_{i-1}$.

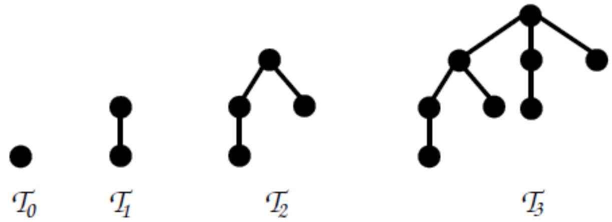

Figure 2: The trees $\mathcal{T}_0, \ldots, \mathcal{T}_3$.

We will now start a request at the single node v with a distance of i from the root in $\mathcal{T}_i$. On its path to the root, the request passes nodes that are roots of the trees $\mathcal{T}_1, \ldots, \mathcal{T}_i$. 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 $\mathcal{C}(\mathcal{T}_1), \ldots, \mathcal{C}(\mathcal{T}_i) = \mathcal{T}_0, \ldots, \mathcal{T}_{i-1}$. 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 $\log n$.

b) The access pattern we described above already has the property that each node requests the object in sequence. We can show this inductively over i for the trees $\mathcal{T}_i$.

First we introduce some additional notation. We consider a tree $\mathcal{T}_i$, for any $i > 0$, as two parts: The left subtree $\mathcal{L}(\mathcal{T}_i)$, which has the structure of $\mathcal{T}_{i-1}$, and the rest of the tree $\mathcal{R}(\mathcal{T}_i)$, which also has the same structure has $\mathcal{T}_{i-1}$. We then write $\mathcal{T}_i = \mathcal{L}(\mathcal{T}_i) \to \mathcal{R}(\mathcal{T}_i)$ to indicate that $\mathcal{T}_i$ is the tree obtained by rooting $\mathcal{L}(\mathcal{T}_i)$ as the left-most child of the root in $\mathcal{R}(\mathcal{T}_i)$. We note that with this notation one iteration of Ivy handling a request from the highest-depth leaf performs a tree rotation that can be described recursively as $\mathsf{Rot}(\mathcal{T}_i) = \mathsf{Rot}(\mathcal{R}(\mathcal{T}_i)) \to \mathcal{L}(\mathcal{T}_i)$. We further write $\mathsf{Rot}^k(\mathcal{T}_i) = \mathsf{Rot}(\mathsf{Rot}^{k-1}(\mathcal{T}_i))$ with $\mathsf{Rot}^0(\mathcal{T}_i) = \mathcal{T}_i$.

We can now show this inductively over i for the trees $\mathcal{T}_i$. We will start with $\mathcal{T}_1$ as the base case since our notation only works for $i > 0$ and the case for $\mathcal{T}_0$ is trivial. In the first iteration on $\mathcal{T}_1$ the leaf node requests the object, after that the edge is switched and the previous root node requests the object. For the inductive step we observe that over i iterations of the access pattern above Ivy accesses the highest-depth leaves of the trees $\mathcal{T}_i, \mathsf{Rot}(\mathcal{T}_i), \ldots, \mathsf{Rot}^{2^i-1}(\mathcal{T}_i)$. Unwinding the definition of Rot we see that these correspond to the highest-depth leaves of $\mathcal{L}(\mathcal{T}_i), \mathsf{Rot}(\mathcal{L}(\mathcal{T}_i)), \ldots, \mathsf{Rot}^{2^{i-1}-1}(\mathcal{L}(\mathcal{T}_i))$ on even (zero-indexed) iterations, and $\mathsf{Rot}(\mathcal{R}(\mathcal{T}_i)), \mathsf{Rot}^2(\mathcal{R}(\mathcal{T}_i)), \ldots, \mathsf{Rot}^{2^{i-1}}(\mathcal{R}(\mathcal{T}_i))$ on odd (zero-indexed) iterations. According to the inductive hypothesis these iterate through the 2^{i-1} nodes of the two subtrees $\mathcal{L}(\mathcal{T}_i)$ and $\mathcal{R}(\mathcal{T}_i)$. Thereby the alternation of the two iterates over all 2^i nodes of $\mathcal{T}_i$.