CS240, Spring 2022
Assignment 4: Question 1

Q1) Specify all of the following in big-O in terms of f, n, and t (as necessary). What is the maximum height of T? What is the total space usage of T and all A(u)? What time is needed to search for a key k in T ?

Since each node contains a pointer to every element in our alphabet $\{0, 1, ..., 2^t - 1\}$, this means we need to determine how many elements in the alphabet fit inside of x_i. In other words our maximum number of nodes needed to store x_i

$$\text{number of nodes needed to store } x_i = 1 + \log_{2^t}(x_i)$$

Since we are given that:

$$x_i <= n^f$$

This means that the maximum number of nodes we will ever need to store (aka the height of the tree is given by):

$$\begin{aligned} \text{height} &<= \log_{2^t}(n^f) \\ &<= f \times \log_{2^t}(n) \end{aligned}$$

Using log rules we can simplify the log equation to get:

$$\begin{aligned} \text{height} &<= 1 + f \times \log_{2^t}(n) \\ &<= 1 + f \times \frac{\log_2(n)}{\log_2(2^t)} \\ &<= 1 + f \times \frac{\log_2(n)}{t} \\ &<= 1 + \frac{f}{t} \times \log_2(n) \\ &\in O(\frac{f}{t} \times \log_2(n)) \end{aligned}$$

Conversationally in order to find a node within our tree, at most we would need to traverse through the height of the tree thus we get:

$$\text{Time to find a child } = O(\frac{f}{t} \times \log_2(n))$$

Since each node has 2^t children this implies that the total number of elements that exist at level i (where i is an integer greater then or equal to 0) is given by:

$$\text{number of elements at level i in the tri} = (2^t)^i$$

Therefore the total number of nodes is given by:

$$\begin{aligned}
\#\text{ of nodes } &= \sum_{i=0}^{1+\frac{f}{t}\times\log_2(n)} (2^t)^i \\
&= \frac{(2^t)^{\frac{f}{t}\times\log_2(n)} - 1}{2^t - 1} \\
&= \frac{(2^f)^{\log_2(n)} - 1}{2^t - 1} \\
&= \frac{n^f - 1}{2^t - 1} \\
&\in O(\frac{n^f}{2^t})
\end{aligned}$$

Since each node contains an array of A(U), where the size of A(U) is $2^t + 1$, our total space complexity is:

$$\text{space complexity} \in O(\frac{n^f}{2^t} \times (2^t + 1))$$

$$\text{space complexity} \in O(n^f)$$