Data Structures
Other Stuff

CS284
Shortest path in unweighted graph

Figure: Caption
BFS

// from https://www.geeksforgeeks.org/shortest-path-unweighted-graph/
public boolean BFS(ArrayList<ArrayList<Integer>> adj, int src, int v, int[] pred, int[] dist) {
    LinkedList<Integer> queue = new LinkedList<Integer>();
    boolean[] visited = new boolean[v];

    for (int i = 0; i < v; i++) {
        visited[i] = false;
        dist[i] = Integer.MAX_VALUE;
        pred[i] = -1;
    }
    visited[src] = true;
    dist[src] = 0;
    queue.add(src);
    
    while (!queue.isEmpty()) {
        int u = queue.poll();
        for (int v : adj.get(u)) {
            if (!visited[v]) {
                visited[v] = true;
                pred[v] = u;
                dist[v] = dist[u] + 1;
                queue.add(v);
            }
        }
    }
    return true;
}
Shortest path in unweighted graph

```java
while (!queue.isEmpty()) {
    int u = queue.peek();
    queue.poll();
    for (int i = 0; i < adj.get(u).size(); i++) {
        int ith_item = adj.get(u).get(i);
        if (visited[ith_item] == false) {
            visited[ith_item] = true;
            dist[ith_item] = dist[u] + 1;
            pred[ith_item] = u;
            queue.add(ith_item);

            if (ith_item == dest)
                return true;
        }
    }
}
return false;
```
Theorem (copied from http://staff.ustc.edu.cn/csli/graduate/algorithms/book6/chap12.htm). Given an open-address hash table with load factor $\alpha = n/m < 1$, the expected number of probes in an unsuccessful search is at most $1/(1 - \alpha)$, assuming uniform hashing.

Proof. In an unsuccessful search, every probe but the last accesses an occupied slot that does not contain the desired key, and the last slot probed is empty. Let us define $p_i = \Pr[\# \text{ probs} = i]$ Thus, the expected number of probes is

$$1 + \sum_{i=0}^{\infty} ip_i$$
To evaluate the above equation, define $p_i = \Pr[\# \text{ probs } \geq i]$ Thus

$$\sum_{i=0}^{\infty} i p_i = \sum_{i=0}^{\infty} q_i$$

Now let’s estimate $q_i$. The probability that the first probe accesses an occupied slot is $q_1 = n/m$, whereas $q_2 = \frac{n}{m} \times \frac{n-1}{m-1}$, thus

$$1 + \sum_{i=0}^{\infty} q_i = 1 + \alpha + 2 + \cdots = \frac{1}{1 - \alpha}$$
Theorem. In a hash table in which collisions are resolved by chaining, a successful search takes time \((1 + \alpha/2)\)

Theorem. We assume that the key being searched for is equally likely to be any of the \(n\) keys stored in the table. To find the expected number of elements examined, we therefore take the average, over the \(n\) items in the table, of \(1\) plus the expected length of the list to which the \(i\)th element is added. The expected length of that list is \((i- 1)/m\), and so the expected number of elements examined in a successful search is

\[
\frac{1}{n} \sum_{i=1}^{n} \left(1 + \frac{i - 1}{m}\right) = 1 + \frac{\alpha}{2} - 1/2m
\]
Huffman tree

There are 8 characters, their frequencies are E: 120, U: 37, D: 42, L: 42, C: 32, Z: 2, K: 7, M: 24. Count the number of bits you need to encode the document.
There are 8 characters, their frequencies are E: 120, U: 37, D: 42, L: 42, C: 32, Z: 2, K: 7, M: 24. Count the number of bits you need to encode the document.

**Answer.** $6 \times 2 + 6 \times 7 + 5 \times 24 + 4 \times 32 + 3 \times 42 + 3 \times 42 + 3 \times 37 + 1 \times 120 = 785$