1. Show that when BFS is performed on an undirected graph, all the edges of the graph are classified as either tree edges or cross edges, and that the cross edges connect vertices at the same or adjacent levels in the BFS tree.

Hint: show that any other type of edge would contradict the fact that the tree path from the root $r$ of the BFS tree to any vertex $v$ is a shortest path from $r$ to $v$.

2. Consider a directed graph whose edges are labeled with probabilities, i.e., numbers between 0 and 1. The reliability of a path is defined as the product of the probabilities of the edges along the path.

Design and analyze an $O((|V| + |E|) \log |V|)$-time algorithm that finds the reliability of the most reliable path from a given vertex $r$ to all other vertices in the graph.

Hint: do not start from scratch; rather, show how to efficiently transform a given instance of this problem into another problem that can be directly solved by Dijkstra’s algorithm, and how a solution to the latter problem can be used to solve the former problem.

Note: intuitively, this models the following scenario: if the edges represent links in a network, and the probability on an edge represents the probability that the given link is functional, and if links fail independently of one another (that’s a big “if”), then the reliability of a path is the probability that all links in the path are functional.

3. Consider the least-cost paths problem on a directed graph that has non-negative edge costs, where the maximum cost of any edge is $C$.

(a) Show that during the execution of Dijkstra’s algorithm, at the end of any loop iteration in which a vertex $v$ is removed from $U$, if $D = d[v]$, then for any $w \in U$, we have $D \leq d[w] \leq D + C$ or $d[w] = \infty$. Hint: use (among other facts) the lemma proved in class.

(b) Suppose further that the edge costs are non-negative integers (bounded by $C$). The result from part (a) suggests the following modification of Dijkstra’s algorithm: maintain $C + 1$ lists, where each list contains those vertices whose tentative distance values are the same, so that every vertex is either on exactly one list or has tentative cost $\infty$. Expand these ideas to obtain a variation of Dijkstra’s algorithm that runs in time $O(C \cdot |V| + |E|)$. In particular, if $C$ is a constant, the running time is $O(|V| + |E|)$.

4. The shortest paths problem for DAGs is easier than for general graphs. Consider the following linear-time algorithm, which takes as input a directed acyclic graph $G$ on $n$ vertices, together with a cost function $c$ on edges (we can even allow negative costs), and an initial vertex $r$.

Initialize $d[v] = \infty$ for $v = 1 \ldots n$
$d[r] \leftarrow 0$

Topologically sort the graph $G$
for each vertex $v$, taken in topological order, do
  for each neighbor $w$ of $v$ do
    $d[w] \leftarrow \min(d[w], d[v] + c(v, w))$

Show that when this algorithm terminates, $d[v]$ is the minimal edge cost of any path from $r$ to $v$. Hint: mimic the proof given in class of the correctness theorem for Dijkstra’s algorithm, and make use of the essential property of a topological sort, namely, that if there is a path from $s$ to $t$ in the graph, then the main loop in the above algorithm will process $v = s$ before $v = t$. 