Basic Algorithms, Assignment 13
Solutions

1. Suppose that we are doing Dijkstra’s Algorithm on vertex set \( V = \{1, \ldots, 500\} \) with source vertex \( s = 1 \) and at some time we have \( S = \{1, \ldots, 100\} \). What is the interpretation of \( \pi[v], d[v] \) for \( v \in S \)?

**Solution:** \( d[v] \) is the minimal cost of a path from \( s \) to \( v \) and \( \pi[v] \) will be the vertex just before \( v \) on that path.

What is the interpretation of \( \pi[v], d[v] \) for \( v \not\in S \)?

**Solution:** \( d[v] \) is the minimal cost of a path \( s, v_1, \ldots, v_j, v \) where all the \( v_1, \ldots, v_j \in S \). \( \pi[v] \) will be the vertex just before \( v \) in this path, here \( v_j \).

Which \( v \) will have \( \pi[v] = \text{NIL} \) at this time.

**Solution:** Those \( v \) for which there is no directed edge from any vertex in \( S \) to \( v \).

For those \( v \) what will be \( d[v] \)?

**Solution:** Infinity

2. (Extra from last week!) You may use Agarwal/Kayal/Saxena but, if so, mark clearly how it is used.

   (a) Call a positive integer \( n \) **TONG** if it has at least one prime divisor \( p \) of the form \( p = 10k + 7 \). Show \( \text{TONG} \in \text{NP} \).

   **Solution:** Oracle gives the value \( p \). Verifier must check that
   i. \( p \) divided by 10 gives a remainder of 7
   ii. \( p \) is prime – using AKS
   iii. \( p \) divides \( n \)

   (b) (harder!) Call a positive integer \( n \) **LAYLA** if it has exactly one prime divisor \( p \) of the form \( p = 10k + 7 \). Show \( \text{LAYLA} \in \text{NP} \).

   **Solution:** Oracle gives the prime factorization (possibly with repetition) \( n = p_1 \cdots p_r \) with \( p_1 \) of the form \( 10k + 7 \). Verifier must check that
   i. \( p_1 \) divided by 10 gives a remainder of 7
   ii. All other \( p_i \) divided by 10 do not give a remainder of 7
   iii. All \( p_i \) are prime – using AKS.
   iv. \( n = p_1 \cdots p_r \)

   (Note: As all \( p_i \geq 2 \) the number of factors \( r \leq d, d \) the number of digits of \( n \). So if AKS takes \( O(d^c) \) applying AKS to each factor takes \( O(d^c+1) \), still polynomial.)
3. Let $G$ be a DAG on vertices $1,\ldots,n$ and suppose we are given that the ordering $1\cdots n$ is a Topological Sort. Let $\text{COUNT}[i,j]$ denote the number of paths from $i$ to $j$. Let $s$, a “source vertex” be given. Give an efficient algorithm (appropriately modifying the methods of §24.1) to find $\text{COUNT}[s,j]$ for all $j$.

**Solution:** Let $s = 1$ (we can ignore the earlier vertices, if any) and write $\text{COUNT}[j]$ for $\text{COUNT}[1,j]$. We set $\text{COUNT}[1] = 1$. The key is that $\text{COUNT}[1,j]$ is the sum, over all $i < j$ with $i,j$ a directed edge, of $\text{COUNT}[1,i]$. Why? Well, every path from 1 to $j$ will have a unique penultimate point $i < j$ and given $i$ there will be precisely $\text{COUNT}[i]$ such paths. One way to implement this is to make a reverse adjacency list, create for every $j$ a list $\text{Adjrev}[j]$ of those $i$ with a directed edge from $i$ to $j$. This can be done in time $O(E)$ by going through the original adjacency lists and when $j \in \text{Adj}[i]$ adding $i$ to $\text{Adjrev}[j]$. Then we can implement this sum. The total time (assuming addition takes unit time) is $O(E)$. 