Mason Legere
230128712
mlegere@unbc.ca

Assignment 2
CPSC 482

2021-02-25

Given an undirected and unweighted graph $G = (E, V)$ we wish to find the shortest path between two given vertices $u, v \in V$. To accomplish this we used a natural modification of Dijstra's path finding algorithm with the caveat of storing the number of minimal maths to a given node at each step in a breadth-first manner.

Algorithm 1: Counting Dijastra's Algorithm

Result: Number of shortest paths between S and F
$G \leftarrow$ graph with vertex set V, and edge set E represented as adjacency matrix ;
$q \leftarrow$ queue of vertices initially containing S ;
$paths \leftarrow$ array of length $|V|$ initialized with 0 ;
$dists \leftarrow$ array of length $|V|$ initialized with ∞ ;
$seen \leftarrow$ array of length $|V|$ initialized with `False` ;
$seen$[S] $\leftarrow$ `True`;
$paths$[S] $\leftarrow 1$;
$dists$[S] $\leftarrow 0$;
while *!empty(q)* **do**
  $v \leftarrow q$.pop() ;
  for $h \in N_G(v)$ **do**
    if *!*$seen[h]$ **then**
      append h to q ;
      $seen[h] \leftarrow$ `True`
    end
    if $dists[h] > dists[v] + 1$ **then**
      $dists[h] \leftarrow dists[v] + 1$;
      $paths[h] \leftarrow paths[v]$
    end
    else if $dists[h] = dists[v] + 1$ **then**
      $paths[h] \leftarrow paths[h] + paths[v]$
    end
  end
end

As the assignment does not ask directly for correctness we will "prove" it in an informal manner. Suppose we are starting at a node $u \in V$. Then we note that the number of shortest path to any other node $v \in V - \{u\}$ is equal to the number of shortest paths is equal to the number of shortest paths to a subset of the neighbours of u, namely $N_G(u)$ such that these neighbours have minimal distance to u. This is exactly what the algorithm computes, as if $dists[h] = dists[v] + 1$ that means that h is a potential minimal neighbour and we sum the number of paths. Likewise if $dists[h] > dists[v] + 1$ we have instead found a new shortest path to v through the neighbour h and therefore h is a potential minimal neighbour and all previous neighbours were not minimal.

In terms of performance complexity, nothing has changed against breadth first search for shortest paths. We iterate over each node only once but we also check each of its neighbours. Therefore the complexity is given $\mathcal{O}(|E| + |V|)$ which isn't that great considering that $|E|$ can naturally be quite large for highly connected graphs.