SAUMYA GUPTA
210949

PAGE NO.
DATE / /

Solution of Problem 3

We can modify Djikstra's algorithm to solve this problem.

Firstly, create a priority queue in the form of min-heap, named, Q.

Now, create a vector "dist" which store minimum bandwidth to reach each switching center from a. Initialise all entries in "dist" to INT-MAX (i.e, ∞)

Enqueue a into Q with priority 0

While Q is not empty:
- → Dequeue a switching center, v from Q with minimum priority (i.e., element at top of min-heap).
- → If v is b, we can stop the algorithm and return the minimum bandwidth stored in dist[b] as max bandwidth of path from a to b.
- → For each neighbour w of v calculate minimum bandwidth from a to w as min of bandwidth of the edge between v and w and min bandwidth stored in dist[v]
- → If this minimum bandwidth is less than minimum bandwidth stored in dist[w], update dist[w] with this new minimum bandwidth and w into Q with priority new.

If the algo reaches this point, it means there is no path from a to b.

SAUMYA GUPTA
210944

PAGE NO.
DATE / /

Solution of Problem-4

Initialise an empty set R to store transitive reduction edges.

For each vertex u in V, do the following:
→ Perform DFS starting from u, visiting all reachable vertices.
→ During DFS for each ~~vertex~~ visited vertex v, mark v as visited.
→ While perform DFS, if we encounter an edge (u, v) where u is current vertex and v is a visited vertex, remove the edge (u, v) from E.
→ Add edge (u, v) to the set R.

Create a new graph G' with same vertices as G and edges in R.

Time Complexity:
Of DFS → $O(|V|+|E|)$
~~So, after~~ Since, we have performed DFS for V vertices,
T. C of algorithm = $\boxed{O(|V| \times (|V|+|E|))}$