Due 23:59 April 7 (Sunday). There are 100 points in this assignment.
Submit your answers (**must be typed**) in pdf file to CourSys
`https://coursys.sfu.ca/2024sp-cmpt-307-d1/`.
Submissions received after 23:59 will get penalty of reducing points: 20 and 50 points deductions for submissions received at $[00:00, 00:10]$ and $(00:10, 00:30]$ of April 8, respectively; no points will be given to submissions after $00:30$ of April 8.

1. 10 points (Ex 22.1-3 of text book)

   For any pair of nodes u and v in an edge-weighted digraph G with no negative cycles, let $P_{uv} = \{u \rightsquigarrow v\}$ be the set of shortest paths from u to v and p^*_{uv} be the path in P_{uv} with the minimum number of edges. Let $r(p^*_{uv})$ be the number of edges in p^*_{uv} and $r = max_{u,v \in V} r(p^*_{uv})$. Give a simple change to the Bellman-Ford algorithm that allows it to terminate in $r + 1$ passes, even if r is not known in advance; prove the correctness of the revised algorithm.

2. 20 points (Ex 24.1-3 of text book)

   In the Bellman-Ford shortest path algorithm discussed in class, the structure of a shortest path of at most i arcs from a source node s to a destination node v in a graph G is defined based on a shortest path of at most $i-1$ arcs from s to v and a shortest path of at most $i-1$ arcs from s to every node w with an arc (w, v) plus the arc (w, v). Distance-vector shortest path algorithm (a distributed version of Bellman-Ford algorithm) is used by an internet routing protocol to compute routing tables at routers. In the distance-vector algorithm, the structure of a shortest path of at most i arcs from s to v is defined based on a shortest path of at most $i - 1$ arcs from s to v, and for every arc (s, w), the arc (s, w) plus a shortest path of at most $i - 1$ arcs from w to v. Let G be a digraph of n nodes and m arcs without any negative cycle. For the distance-vector algorithm, give the structure of a shortest path of at most i arcs from s to v, Bellman equation and pseudo code to compute the shortest distance from s to every other node of G and the running time of the pseudo code (express the time in n and out-degree $\deg(s)$ of source node s) (**Hint:** Since shortest paths from multiple nodes (e.g., s, w) to v are used, you may need to use $\text{opt}(i, x, y)$ to explicitly express the optimal solution of at most i arcs from node x to node y. You can assume $\text{opt}(i-1, w, v)$ is given for every arc (s, w) when computing $\text{opt}(i, s, v)$ for every i.)

3. 15 points

   Write two programs for Dijkstra's single source shortest paths algorithm discussed in class, one implements the algorithm with the input graph G represented by adjacent list and the other implements the algorithm with G represented by adjacent matrix. Run your programs on the graph in Figure 1 and show the shortest paths found. Run your programs on G with $n = 100, 200, 400, 800$ nodes, for each n, with $m \approx 3n, n^{1.5}, n(n-1)/2$ edges, and each edge is assigned a random number from $[1, a]$ for some $a > 1$. Report the running times of your programs on a computer. Write a procedure to create input graphs for both implementations and exclude the running time of the procedure from the times of your implmentations for Dijkstra's algorithm. Please do not submit the codes. (Hint: For each n, first create a connected base graph of nodes $v_1, .., v_n$, e.g., a cycle, then add $m - n$ edges $\{u, v\}$ with u, v randomly selected from $\{v_1, .., v_n\}$ to the base graph.)

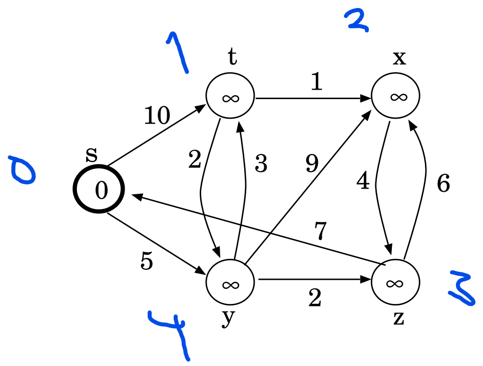

Figure 1: Input Graph.

(5 points)

(5 points) for each of the reported results from the implementations.

4. 15 points (Ex 22.3-7 of text book)

   Let G be a digraph representing a communication network with each edge $e = (u, v)$ assigned a weight $0 \leq r(e) \leq 1$ representing the reliability of the communication link from u to v. The reliability of a path P consisting of edges $e_1, e_2, .., e_k$ is $r(P) = \Pi_{1 \leq i \leq k} r(e_i)$ (e.g., for $P = e_1, e_2, e_3$, $r(P) = r(e_1) \times r(e_2) \times r(e_3)$). The reliability of a path with 0 edge (path from a node u to u) is 1. Give an efficient algorithm to find a most reliable path from a source node s and every other node in G; prove the correctness and analyze the running time of the algorithm.

5. 10 points (Ex 23.1-8)

   Let $Q^{(m)}$ be an $n \times n$ array that $Q^{(m)}[i, j]$ contains the predecessor of node j on a shortest path from node i to j that has at most m edges. Modify the Extend-Shortest-Paths algorithm and Slow-All-Pairs-Shortest-Paths algorithm to compute the matrices $Q^{(1)}, Q^{(2)}, .., Q^{(n-1)}$ as the matrices $L^{(1)}, L^{(2)}, .., L^{(n-1)}$ are computed.

6. 15 points (Ex 23.2-8 of text book)

   Give an $O(n(m + n))$ time algorithm for computing the transitive closure of a directed graph G of n nodes and m edges; prove the correctness and analyze the running time of the algorithm.

7. 15 points

   Professor Michener claims that there is no need to create a new source node in line 1 of Johnson's shortest path algorithm for the all pairs shortest paths problem, and suggets to use $G' = G$ and let s be any vertex of G'. Assume that $\infty - \infty$ is undefined, especially, it is not 0. Give an example of an edge weighed directed graph G to show that the professor's modification on Johnson's algorithm does not give a correct answer. Prove if G is strongly connected, then the professor's modification on Johnson's algorithm is correct.