CONCORDIA UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE AND SOFTWARE ENGINEERING

COMP 6651 NN: Algorithm Design Techniques

Winter 2022

Term Project: Grading Scheme of part II

1. **3 points.** Provide a clear definition of the terminology with respect to edge and link: convention used in the lectures was: edge (undirected), link (directed). We note some confusion in several projects, and no clear definition of an edge vs. a link.

2. **5 points.** Algorithm **guarantees** providing a (directed) path with the required color **for each node pair** of $\mathcal{SD}_1$ (non-COVID set) and $\mathcal{SD}_2$ (COVID set). For their computation, see below.
   If you used Ford-Fulkerson's algorithm, you need to explain how you set the capacity values in such a way as to guarantee finding a path for every node pairs in $\mathcal{SD}_1$ and $\mathcal{SD}_2$. For instance, if you set a uniform capacity of 1 for all links (having in mind to get link-disjoint paths), then you will not be able to find paths for all node pairs in $\mathcal{SD}_1$ and $\mathcal{SD}_2$.

3. **32 points.** Algorithms have been provided for the following tasks

   (a) **9 points.** Orientation of the edges $\rightsquigarrow$ (directed links) links

   (b) **9 points.** Identification of the bridges (unavoidable shared edges/links)

   (c) **5 points.** Explain how you handle the bridges in your algorithms for graph orientation and definition of the paths for the requested node pairs. Following the explanations provided in the slides about bridges which are dead ends, it was expected that you would discuss them and distinguish them from those which are not dead-ends. Your proposed algorithm(s) should be explicit about how bridges are taken care.
   Dead-end bridges: no choice, you have to use them in both directions, while for the other bridges, it may be possible to use them so that you only share links in the same direction.
   In other words, if you have an algorithm to identify bridges, but you never talk again about bridges, you get no point here.

   (d) **9 points.** Paths for $\mathcal{SD}_1$ and $\mathcal{SD}_2$: explanations on how you compute them.

4. **20 points.** Algorithms take into account the objectives: throughput (10 points) and link sharing (10 points), meaning some discussions are made with respect to the objectives
5. **10 points.** Quality of the writing (English).
6. **5 points.** For each algorithm, input and output are clearly described.
7. **10 points.** Plan of the paper follows the recommendations.
8. **5 points.** Figures help for the understanding of the algorithms.
9. **5 points.** Literature review goes beyond provided references.
   Literature should cite published work, i.e., conference and journal papers, and avoid as much as possible blogs, wikipedia, ...
10. **5 points.** Meaningful conclusion (i.e., a conclusion is not a summary of the work)

Some comments:

- Identification of bridges can be done in linear time using Tarjan's algorithm [1]. You need to go to paper/book checks for the claims made on websites such as, blogs, GeeksForGeeks, which are not always reliable.
- Maximum flow algorithm of Ford and Fulkerson works on a **directed** graph: if you apply it on an undirected graph.
- What is a heuristic? A heuristic algorithm is used to design a solution for a problem as quickly as possible. It may not produce the best solution, but gives a near-optimal solution in a short time.
- What is a greedy algorithm? A greedy algorithm is any algorithm that follows the problem-solving heuristic of making the locally optimal choice at each stage. It usually does not produce an optimal solution, but a greedy heuristic can yield locally optimal solutions that approximate a globally optimal solution in a reasonable amount of time.
- Minimizing the number of shared links (same or opposite directions): rather than computing shortest paths with Dijkstra's algorithm or with the help of Ford Fulkerson's algorithm, note that computing a Hamiltonian path or a cycle (Hamiltonian path with the option of going several times through the same nodes) is a better mechanism for reducing the number of shred links!
- Ford and Fulkerson's algorithm **does not compute paths**. You need to add explanations on how you derive paths from the usage of Ford and Fulkerson's algorithm.

- DFS/BFS are not known as graph orientation algorithms: you need to provide explanations on how you use these exploration scheme algorithms to deduce graph orientations.

References

[1] R.E. Tarjan. A note on finding the bridges of a graph. *Information Processing Letters*, 2(6):160 – 161, 1974.