ELL781: Software Fundamentals for Computer Technology

Assignments Quiz, Maximum marks: 12

(**Answer all questions on this question paper.**)

NB: Please read all questions carefully. There may be subtle differences between what the question is asking for and the context in which things have been done during the assignments, and these have to be taken into account whilst answering. Working/derivation for all your answers should be shown fully and clearly.

1. Consider the conversion of a tournament scheduling problem to a graph colouring problem, as carried out in Assignment 1 (recall that each team had to play every other team twice, once at home and once away). If the tournament has n teams, then:

   (a) What is the number of vertices in the corresponding graph? **[1]**

   (b) What is the number of edges in the correspoding graph? **[2]**

   (c) Would your implementation have given the optimal solution for $n = 3$? Why or why not? **[1.5]**

2. In general, Ford-Fulkerson with BFS works better than with DFS, because BFS systematically finds augmenting paths with the fewest edges. However, suppose that instead of running Ford-Fulkerson until the point that no augmenting paths remain in the residual network, we decide to terminate it after a fixed number of augmentations, and just accept whatever flow has been achieved as an approximate solution. In this scenario, is it possible for DFS to work better than BFS? Try to construct a simple example where this happens, and explain why. (Hint: A very simple flow network with just 3 vertices should suffice.) **[3]**

3. The basic array-based implementation of the `MERGE` operation for an `MFSET`, as discussed in class, takes $O(n)$ time, and hence for a sequence of $n-1$ `MERGE` calls, will take a total of $O(n^2)$ time. This needs to be reduced to $O(n \log n)$ in order to make Kruskal's algorithm faster than Prim's for sparse graphs.

   (a) Describe any one approach for doing so (informally, pseudocode not needed). What kind of data structure is used to represent the `MFSET` in your chosen approach? **[1.5]**

   (b) Suppose your `MFSET` currently consists of the following components (sets of vertices): $A = \{1, 2, 3\}, B = \{4, 5\}, C = \{6\}$. Draw a figure to show how this will be represented using the above-chosen data structure. **[1.5]**

   (c) Supposing you now call the `MERGE` operation on the components A and B above. Show schematically how the merger will happen, and the resulting state of the `MFSET`. **[1.5]**