UNIVERSITY OF BRISTOL

January 2014 Examination Period

FACULTY OF ENGINEERING

Second Year Examination for the Degrees of BSc, BEng and MEng

COMS21103J
Data Structures and Algorithms

TIME ALLOWED:
2 hours

This paper contains three questions:
All questions carry 20 marks each
All three questions will be used for assessment
The maximum for this paper is 60 marks

Other Instructions
1. Calculators must have the Engineering Faculty seal of approval.

TURN OVER ONLY WHEN TOLD TO START WRITING

Q1. This question involves the graph in Figure 1.

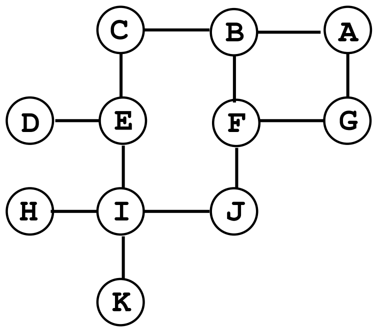

Figure 1: Graph for problem Q1.

a) Argue that it is not possible to transform the graph into a tree by removing a single edge. *[2 marks]*

b) List the labels of the vertices in the order they would be visited when performing depth first search starting from A. Assume that neighbours of the same node are visited in alphabetical order. *[3 marks]*

c) Provide the same list for when the graph is visited using breadth first search. *[3 marks]*

d) Write down pseudocode for the Breadth First Search algorithm for traversing a graph and state its complexity. *[3 marks]*

e) Consider an $n \times n$ matrix A with entries 0 or 1. We call an entry of A *empty* if it contains 0 and *full* otherwise. We call two entries of the matrix adjacent if they are in consecutive positions in either the same row or the same column of A.

   i) Design an algorithm which, given as input a matrix A as above, determines if there exists a way to go from $A[1, 1]$ to $A[n, n]$ only through adjacent positions that are empty. Assume that $A[1, 1]$ and $A[n, n]$ are empty.

   ii) Discuss the complexity of your algorithm.

[9 marks]

Q2. Recall that a disjoint-set data structure maintains a collection of disjoint subsets of some overall universe and supports the operations MakeSet, FindSet and Union.

a) Describe how a disjoint-set data structure can be implemented using an array of linked lists. Include a brief description of how each of the above operations is implemented. *[4 marks]*

b) Imagine you are required to extend the disjoint-set data structure by implementing a new operation Count, where Count(x) should return the size of the subset containing x. Describe how to modify the information stored in the implementation from part (a), and the operations MakeSet and Union, so that Count can be perfomed in time $O(1)$. You may assume that integer addition can be performed in time $O(1)$. *[4 marks]*

c) State a heuristic which allows any sequence of n MakeSet operations, followed by $n-1$ Union operations, to be executed in time $O(n \log n)$. *[1 mark]*

d) Assuming n is an integer power of 2, informally describe a sequence of n MakeSet operations, followed by $n-1$ Union operations, which requires time $\Omega(n \log n)$ to be executed on a disjoint-set structure implemented using an array of linked lists. Your description should cover any n that is an integer power of 2, not just one specific n of this form. *[4 marks]*

e) Write down pseudocode for Kruskal's algorithm for finding a minimum spanning tree in a connected graph. State the time complexity of the algorithm if it is based on a disjoint-set structure implemented using an array of linked lists. *[5 marks]*

f) A maximum spanning tree of a weighted undirected graph G is a spanning tree of G such that the total weight of the edges in the tree is as large as possible. Describe how to modify Kruskal's algorithm to find a maximum spanning tree of a graph. You need not prove correctness of your modification. *[2 marks]*

Q3. a) Given three polynomials of degree n of the form $f_i(x) = a_0 + a_1x + ... + a_{n-1}x^{n-1}$, the polynomial multiplication $h(x) = f_1(x) \times f_2(x) \times f_3(x)$ needs to be calculated. State the complexity of the operation using the basic school-book long polynomial multiplication method. *[2 marks]*

b) Explain (including a diagram) the steps for using the Fast Fourier Transform for polynomial multiplication $h(x) = f_1(x) \times f_2(x)$. For each step, clarify the input, output and complexity. *[4 marks]*

c) For the polynomial $f(x) = x - 4x^2$, apply the Fast Fourier Transform to evaluate the point-value representation for $f(x)$ using roots of unity as evaluation points, and the Halving Lemma. Show the intermediate steps in your evaluation. For a reminder, the FFT algorithm is shown here:

```
FFT(A, N)
begin
    if N = 1 then
        return A
    else
        ω_N ← e^(2πi/N)
        ω ← 1
        A^[0] ← (a_0, a_2, a_4, ..., a_{N-2})
        A^[1] ← (a_1, a_3, a_5, ..., a_{N-1})
        y^[0] ← FFT(A^[0], N/2)
        y^[1] ← FFT(A^[1], N/2)
        for k = 0 upto N/2 − 1 step 1 do
            y_k ← y_k^[0] + ω · y_k^[1]
            y_{k+N/2} ← y_k^[0] − ω · y_k^[1]
            ω ← ω · w_N
        return y
end
```

[6 marks]

d) Given the alphabet $\{0, 1, 2\}$, draw a finite state automaton that can find the first occurrence of a string that satisfies the regular expression $0^+12^+0^*1$. *[5 marks]*

e) Judith is trying to minimise the number of hours she spends studying a course, but still achieve at least 60% in the course. She looked at the course's webpage and realised there are 2 courseworks (CW1, CW2) and one final exam that account for 10%, 30% and 60% of the final grade respectively. She wishes to spend at most half of her studying time working on her own. She estimated that she can spend 40% of the time for CW1, 30% of the time for CW2 and only 20% of the exam studying time working within a group. Judith also wants to make sure that at least 50% of her studying time is enjoyable, and estimated she can enjoy 80%, 70% and 20% of her time studying for CW1, CW2 and the exam.
When checking with a student from last year, she was told that getting full marks in CW1 and CW2 requires 20 hours of studying each, while the exam requires 60 hours of studying to get full marks.
Assuming Judith's mark on each of the assessments would exactly match $\frac{x}{n}$ where x is the number of hours she spends and n is the total number of hours

needed to get full marks, write Judith's minimisation problem in the standard form for a linear programme. *[3 marks]*

END OF PAPER