CURTIN UNIVERSITY OF TECHNOLOGY
Department of Computing

Design and Analysis of Algorithms
Semester 1, 2016

ASSIGNMENT—Due 4pm, 16th May 2016

QUESTION ONE (20 marks)

a) **(10 marks).** Use the Master method to solve the following recurrence function:

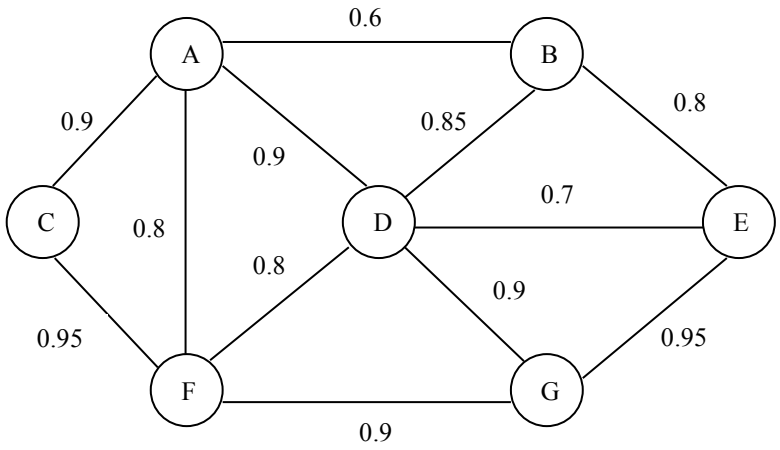
$$T(n) = 3T(\sqrt[2]{n}) + log_2\, n$$

Hint: The function is not in a form suitable for the master method. However, you can convert it to the required form for master method by using another variable $k = \log_2 n$. For details, read page 86 of the textbook (third edition).

b) **(10 marks).** Consider the recurrence function $T(n) = 2\ T(n - 1) + 1$. State the upper bound time complexity of the function and use induction to prove that the time complexity is correct.

QUESTION TWO (40 marks)

Consider the following communication network that is represented by a weighted graph $G = (V, E)$ in which the non-negative number $r_{u,v}$ represents the *operational probability* or *reliability* of link $(u, v) \in E$, for $0 \le r_{u,v} \le 1.0$. Recall that a path $P_{a,b}$ is a sequence of links from a given source node a to its destination node b. The reliability of a path (called *path reliability*), $R_{a,b}$, is computed by multiplying the reliability of each link in path $P_{a,b}$. For example, the reliability of path $P_{A,E}$ = (A, D, B, E) from source node A to destination node E is $R_{A,E} = (0.9 * 0.85 * 0.8) = 0.612$. We define *the most reliable path* from a source node s to a destination node t as the path with the highest reliability among all possible paths from s to t, i.e., the maximum $R_{s,t}$.



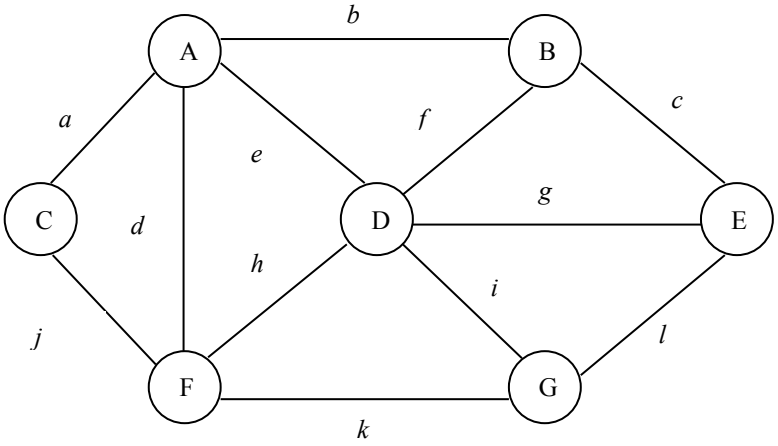
a) **(25 marks).** Design a *greedy* algorithm that generates the most reliable path from a source node *s* to every destination node *t* in the network. Show your algorithm in a concise and clear pseudo-code. Further, you must explain in detail each line of your pseudo-code and show how to implement your algorithm so that it has the best possible time complexity.

Hint. Is it possible to modify Dijkstra's algorithm here?

b) **(15 marks).** Use your algorithm in part a) to generate the most reliable path from node A to every other node for the given graph. List the most reliable paths and their corresponding path reliabilities.

QUESTION THREE (40 marks)

Consider an undirected graph $G = (V, E)$, where V is a set of nodes and E is a set of links. As an example, consider the following graph, where each link is labeled by a lower case letter, e.g., link *a* connects nodes A and C. As defined in Chapter 23 of the textbook (Introduction to Algorithms by Cormen, et al), a cut $(S, V - S)$ is a partition of nodes in V. Further, a link $(u, v) \in E$ *crosses* the cut $(S, V - S)$ if **either** node $u \in S$ and $v \in (V - S)$ **or** node $u \in (V - S)$ and $v \in S$; i.e., one of its end points is in S and the other is in $V - S$. As an example, $S_1 = \{C\}$ and $S_2 = V - S_1 = \{A, B, D, E, F, G\}$ is a cut. The weight of a cut is defined as the number of links **crossing** the cut. As an example, the weight of the cut (S_1, S_2) is two; there are two crossing links in the cut. The **maximum cut** (called **Max-Cut**) is a cut with the **maximum weight**. The problem of finding a maximum cut in a graph is known as the **Max-Cut Problem**, a well known NP-complete problem.



a) **(5 marks).** Generate all possible cuts in the given graph, and determine its maximum cut.

b) **(20 marks).** Design a greedy algorithm to solve the Max-Cut problem. As part of your solution, you must state your greedy criteria. Further, show your algorithm in a concise but clear pseudo-code. You must explain in detail each line of the pseudo-code and show how to implement the algorithm so that it has the best possible time complexity.

c) **(10 marks).** Use your algorithm in part b) to generate the Max-Cut of the given graph. Does your algorithm generate an optimal result?

d) **(5 marks).** Give a counter example to show that your greedy algorithm does not always generate an optimal result.

Instruction for submission

1. Assignment submission is **compulsory**. NO LATE SUBMISSION WILL BE ACCEPTED. If your assignment is incomplete due to illness or other circumstances on or near the submission date you must submit the partial solution you have completed. Any justified exceptional circumstance (e.g., due to illness supported with a Medical Certificate) will be taken into consideration.

2. Your answer **must be neatly typed** and clearly presented. Use a cover page that includes the words "Design and Analysis of Algorithms Assignment", and your name and student ID as recorded in the student database. Submit the hardcopy of your assignment to the unit box, and its soft copy to the unit's Blackboard.

3. Due date may only be altered with the consent of *all* students enrolled in the unit and with the consent of the lecturer.

Failure to meet these requirements may result in the assignment not being marked.