The University of Melbourne
Department of Computing and Information Systems
COMP90038 Algorithms and Complexity

Assignment 1, Semester 1 2016

Due date: Thur, April 14 9:00am

Marks available: 20 (Note: this corresponds to 10% of your final assessment)

Objectives

The objectives of this assignment are to:

- improve your understanding of the time complexity of algorithms and recurrence relations;
- develop skills in analysis and formal reasoning about complex concepts;
- improve written communication skills; in particular the ability to present algorithms clearly, precisely and unambiguously.

Problems

1. **(2 MARKS)** Consider the following pseudo-code:

```
$x \leftarrow 0$
for $k \leftarrow 1$ to $n$ do
    for $j \leftarrow k$ to $2n$ do
        $x \leftarrow x + 1$
```

   (a) How many times is the inner-most statement ($x \leftarrow x+1$) executed? Justify your answer using the summation notation and rules as described in Appendix 1 of Levitin.

   (b) Express the time complexity of the code segment using the appropriate Big $O/\Omega/\Theta$ notation.

2. **(2 MARKS)** Consider the following recurrence relation:

$$T(n) = \begin{cases} 0 & \text{if } n = 1 \\ 3T(n/3) + 2cn & \text{otherwise} \end{cases}$$

   where c is a positive constant. Assume that $n = 3^m$.

   Solve the recurrence. Your answer should show how you derive the closed form.

3. **(6 MARKS)** Consider an array of `n` distinct integers `A[0,...,n-1]`, sorted from smallest to largest value. How can you determine whether an item exists in the array at index position i such that `A[i] =` i?

   For example, `A=[-8,0,2,6,12]` has `A[2] =` 2, while `A=[-8,0,1,6,7,11]` does not have any index where `A[i] =` i.

   (a) Design an efficient algorithm to solve this problem (your algorithm should have time complexity better than linear).

   (b) What is the running time of your algorithm? Justify your answer.

4. **(4 MARKS)** Your colleague has partially implemented an algorithm to test if a strongly connected graph is bipartite (can be 2-colored). Pseudo-code for her algorithm is presented below.

   Your colleague has won the lottery and decided to leave university, so it is up to you to finish the algorithm. Unfortunately, she did not define the data structure for the variable X. However, you can tell from the documentation that the data structure was supposed to be either a *stack* or a *queue*.

```
function IsBipartite($G[0..n-1][0..n-1]$)
    //Input is an adjacency matrix $n \times n$
    NumSeen = 0
    Color[0..n-1] ← [-1, ⋯, -1]
    X ← EmptyQueueOrStack( )
    PushOrEnqueue(X, <0, Red>)
    while NumSeen < n do
        i, c ← PopOrDequeue(X)
        if Color[i] = -1 then
            NumSeen ← NumSeen + 1
        Color[i] ← c
        NextColor ← Red
        if c = Red then
            NextColor ← Blue
        for $j$ ← 0 to $n$ do
            if G[i][j] = 1 then
                if Color[j] = c then
                    return false
                PushOrEnqueue(X, <j, NextColor>)
    return true
```

   (a) Will a *stack* data structure or *queue* data structure produce the correct answer?

   (b) Provide an adjacency matrix for a small graph that causes the other data structure to either return the wrong result or not terminate. (Hint: a 3 node graph is sufficient)

5. **(2 MARKS)** In Section 4.2, Levitin suggests that a *decrease-and-conquer* approach can be used for the topological sorting problem. The idea is to repeatedly remove a source (a node with no incoming edges) and list the order in which such nodes were removed.

   What happens if this approach is run on a cyclic graph? Justify your answer in no more than a few sentences.

6. **(4 MARK)**

   Mutual Information (MI) is a dimensionless quantity that can be thought of as the reduction in uncertainty about one random variable given knowledge of another. High MI indicates a large reduction in uncertainty; low MI indicates a small reduction; and zero MI between two random variables means the variables are independent.

   Formally, the MI of two discrete random variables X and Y can be defined as:

$$I(X,Y) = \sum_{x \in X} \sum_{y \in Y} p(x,y) \log \left(\frac{p(x,y)}{p(x)p(y)} \right)$$

   where $p(x,y)$ is the joint probability distribution function of X and Y, and $p(x)$, $p(y)$ are the marginal probability distribution functions of X and Y respectively.

   Assume that X has n states $\{x_0, \cdots, x_{n-1}\}$ and Y has n states $\{y_0, \cdots, y_{n-1}\}$. Let C denote the joint probability distribution matrix: $C(i,j) = p(X = x_i, Y = y_j)$.

   (a) Design an algorithm to calculate the MI between X and Y. You may assume that the $\log(.)$ function is available (Note: when $\log_2$ is used, the MI is measured in *bits*).

   You should include a description of the input to your algorithm and return values/side effects where appropriate.

   (b) What is the time complexity of your algorithm?

Submission and evaluation

- Submit a PDF document via the LMS. Note: Microsoft Word submissions are not acceptable — if you use Word, create a PDF version for submission.
- Marks are primarily allocated for correctness, but elegance of algorithms and how clearly you communicate your thinking will also be taken into account. Where indicated, the complexity of algorithms also matters.
- We expect your work to be neat—parts of your submission that are difficult to read or decipher will be deemed incorrect. Make sure that you have enough time towards the end of the assignment to present your solutions carefully.

If you questions, email the Head Tutor, Toby Davis <`todavies@student.unimelb.edu.au`> or the Lecturer Michael Kirley <`mkirley@unimelb.edu.au`>, a precise description of the problem, bring it up at a lecture, or use the LMS discussion board.

Late submission will be possible, but **a late submission penalty will apply**: a flagfall of 2 marks, and then 1 mark per 12 hours late.