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

Assignment 2, Semester 1 2020

Deadline: Sunday June 7, 11:59pm 30 marks (15% of final assessment)

Objectives

To improve your understanding of the time complexity of algorithms. To develop problem-solving and design skills. To improve written communication skills; in particular the ability to present algorithms clearly, precisely and unambiguously. In order to get full marks algorithms should be optimal in terms of time complexity or, when specified, should strictly run at the time complexity required. Python code is not acceptable.

Problems

1. **[10 marks]** Consider the function *SumKSmallest(A[0..n − 1], k)* that returns the sum of the *k* smallest elements in an *unsorted integer array A* of size *n*. For example, given the array *A*=[6,-6,3,2,1,2,0,4,3,5] and *k*=3, the function should return -5.

    a. **[3 marks]** Write an algorithm in pseudocode for *SumKSmallest* using the brute force paradigm. Indicate and justify (within a few sentences) the time complexity of your algorithm.

    b. **[3 marks]** Write another algorithm in pseudocode for *SumKSmalleast* using the transform & conquer paradigm. Your algorithm should strictly run in *O(n log n)* time. Justify the time complexity of your algorithm.

    c. **[4 marks]** Explain with details, how we could implement another *SumKSmalleast* that strictly runs in less than *O(n log n)* time, considering *k*<<*n*. Demonstrate the time complexity of your approach as a function of *n* and *k*.

2. **[6 marks]** Let *T* be an arbitrary binary tree. Let us call a given node of *T* an *AVL node* if its balance factor (*i.e.*, the difference between the heights of the left and right subtrees) is -1, 0 or 1. Consider the function *markNonAVLNodes(T)* that "marks" all nodes in *T* that do not satisfy the balance factor condition.

    a. **[2 marks]** Write a *recursive* algorithm in pseudocode for *markNonAVLNodes(T)*.

    b. **[3 marks]** Write an *iterative* algorithm in pseudocode for *markNonAVLNodes(T)*.

    c. **[1 marks]** Demonstrate the time complexity of the developed algorithms.

3. **[4 marks]** Let A be an unsorted integer array with no duplicates. Write an algorithm in pseudocode to generate an *AVL tree* from A in $O(n \log n)$ time.

4. **[10 Marks]** Consider an unweighted directed acyclic graph $G = <V, E>$

   a. **[8 marks]** Design an efficient algorithm in pseudocode to find the number of different directed paths from node $s \in V$ to node $t \in V$ using any other node in G as a stepping stone. You may assume that $s \neq t$.

      The output of your algorithm should be an array with size $|V|$ where the i-th entry in the array contains the number of different directed paths from node s to node t using the i-th node as the stepping stone. Your algorithm should run in $O(|V|+|E|)$ time in order to receive full marks.

      Hint: your algorithm can make a call to an available function TopologicalSort().

   b. **[2 marks]** Explain how your algorithm generates the correct output. You may illustrate by drawing a graph with appropriately labelled nodes, edges and path(s).

Submission and evaluation

- You must submit a PDF document via the LMS. Note: handwritten, scanned images, and/or 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.

- Please write any pseudo code following the format suggested in the examples provided in the sample lecture slides and/or the textbook. Take care with indentation, loops, if statements, initialisation of variables and return statements. Cormen et al. (available as an e-book in the library) provide some guidelines for pseudo code (pages 20–22).

- Make sure that you have enough time towards the end of the assignment to present your solutions carefully. Time you put in early will usually turn out to be more productive than a last-minute effort.

- You are reminded that your submission for this assignment is to be your own individual work. For many students, discussions with friends will form a natural part of the undertaking of the assignment work. However, it is still an individual task. You should not share your answers (even draft solutions) with other students. Do not post solutions (or even partial solutions) on social media. It is University policy that cheating by students in any form is not permitted, and that work submitted for assessment purposes must be the independent work of the student concerned.

- Please see https://academicintegrity.unimelb.edu.au

If you have any questions, you are welcome to post them on the LMS discussion board. You can also email the Head Tutor, Lianglu Pan <lianglu.pan@unimelb.edu.au> or the Lecturer, Douglas Pires <douglas.pires@unimelb.edu.au>. In your message, make sure you include COMP90038 in the subject header. In the body of your message, include a precise description of the problem.

Late submission will be possible, but a **late submission penalty will apply**: a flagfall of 3 marks, and then 3 marks per day.

Extensions will only be awarded in extreme/emergency cases, assuming appropriate documentation is provided – simply submitting a medical certificate on the due date will not result in an extension.