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

Assignment 1, Semester 1 2018

Released: 29 March. Deadline: 16 April 9:00am

Objectives

To improve your understanding of the time complexity of algorithms and recurrence relations. To develop problem-solving and design skills. To improve written communication skills; in particular the ability to present algorithms clearly, precisely and unambiguously.

Problems

1. **[3 marks]** Consider the recursive function listed below. The function takes one positive integer n as input, and a, c and k are numerical constants.

   function F(n)
   if $n = 1$ **then**
   return a
   return F($n - 1$) + $cn + k$

   (a) Set up the recurrence relation for the function's value, F(n), and solve it to derive the closed-form. You must show working for how the closed-form was derived.

   (b) Set up the recurrence relation for the number of multiplications made by the algorithm, M(n). Then find the closed-form for it. You must show working.

   (c) Finally, state the Θ expression for the time complexity of the algorithm.

2. **[2 marks]** Consider two different sorting algorithms. Algorithm A has time complexity $\Theta(n^2)$, while Algorithm B has time complexity $\Theta(n \; log \; n)$.

   Can we say which of the two algorithms will sort an array of 1000 elements faster? Briefly justify your answer.

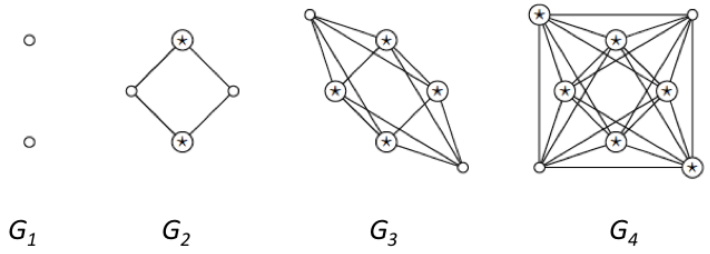
3. **[1 mark]** Consider a decrease-and-conquer approach for solving the topological sorting problem. The idea is to repeatedly remove a *source* (a node with no incoming edges) and list the order in which such nodes were removed.

   What happens if this approach is run on a cyclic graph? Justify your answer in no more than a few sentences. You should include a graph to help illustrate your argument.

4. **[2 marks]** Consider a set of graphs $G : \{G_1, G_2, \ldots, G_m\}$ where each G_i is defined in the usual manner, that is, $G_i =< V, E >$ where V (the set of vertices) and E (the set of edges) are defined for graph G_i.

   G_1 consist of two vertices ($V = 2$) and no edges (E is the empty set $\emptyset$).

   For every $k \geq 2$, G_k consists of a copy of G_{k-1} plus the addition of two vertices. There is also an edge from each of the additional vertices to each vertex in the copy of G_{k-1}. For example, the figure below shows G_1, G_2, G_3 and G_4 from left to right. Here, the $*$ in the node indicates that the node was created in the graph G_{i-1}.



   (a) Create a table showing the number of vertices V and the number of edges E in G_k, for $k = 1$ to 6

   (b) Derive a formula for the number of vertices V_k

   (c) Write a recurrence (with base case) for E_k

   (d) Solve the recurrence to get a closed-form expression for E_k

5. **[5 marks]** A cattle station in outback Australia supplies a significant proportion of cattle destined for the international markets. For management reasons, the cattle station is sub-divided into independent *feed lots*. The network of feed lots can be represented using a graph structure, where the nodes represent the feed lots with unit distance between each pair of directly connected feed lots (i.e., each edge is of length 1).

   As a worker on the cattle station, you need to check the status of the feed lots regularly. Starting from a given feed lot u, you would like to calculate the number of possible paths to feed lot v that have the minimum number of edges as you move from u to v.

   Using pseudocode, write the **function** CountFeedLotPaths($G < V, E >$, u, v) that returns the correct number of paths between u and v meeting the criteria described above. The input G is a valid graph representing the cattle station consisting of V feed lots with E edges between the feed lots, and u, $v \in V$ represent the starting and finishing feed lots.

   Your algorithm should have $O(V + E)$ time complexity.

   Hint: consider adapting a graph traversal algorithm introduced in the lectures.

   Your pseudocode should use appropriate variables, conditional statements and loops and be formatted using appropriate indentation where necessary.

6. **[2 marks]** A common programming (or algorithm design task) is to write code or a function to return a value for a mathematical calculation. In this question, you will do this for the following scenario:

   Consider a simplified neural network consisting of a collection of n neural units described by an activation level x_i (a real number), for $i = 0, 1, \ldots, n-1$. The updating of the activation level of each neural unit x_i at each time step t is controlled by the following equations:

$$x_i^{t+1} = x_i^t + \Delta x_i \tag{1}$$

$$\Delta x_i = -x_i^t + p\ tanh(x_i^t) + q \sum_{j \neq i}^{n-1} G_{ij}\ tanh(x_j^t) \tag{2}$$

   In equation (2), the second term on the right side describes the neural unit 'local' dynamics, which has a strength determined by the parameter p (a real value). The last term on the right side reflects the random unit-to-unit interactions in the network that has an impact on the value of x_i. Interactions between the individual neural units are controlled by the elements of the $n \times n$ connection matrix G, which are drawn independently from a Gaussian distribution with mean 0 and variance $1/n$, where q is a scaling parameter (a real value).

   Your task: complete the function below in pseudocode to update the activation level of each of the n neural units. You may assume that the n neural unit activation values x_i at time t are stored in an array `A[0,..,n-1]`, that is x_i^t=`A[i]`. Assuming that the activation levels of all neural units at time step t are valid, your function should calculate the new neural unit activation levels at time $t+1$. You may call the maths 'library' function `tanh( )` in your pseudocode.

   function UPDATE($A[0,..,n-1], G, p, q$)

   // Input: A an array of neural unit activation levels x_i in the neural network at time t
   G the neural unit interaction graph in adjacency matrix ($n \times n$) format

   // Output: A an updated array containing the activation levels for each neural unit x_i in the neural network at time $t+1$

   // – add pseudocode here –

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.
- 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. 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, Anagha Madhusudanan <`amadhusudana@student.unimelb.edu.au`> or the Lecturer, Michael Kirley <`mkirley@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 2 marks, and then 1 mark per 12 hours late.

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.