COMP108 Data Structures and Algorithms

Week 11 Tutorial Exercises

Due: 5 May 2023, 5:00pm

(Late submission accepted until Monday 9:00am)

Information

- Handwrite/typeset your answers and make it into a single pdf file. See this guide on how to scan documents to pdf.
- Submission: Submit a file named **COMP108W11.pdf** on Canvas
  Late submission is only accepted until Monday 9:00am.
- Submission of lab/tutorial exercises contributes to 10% of the overall module mark. Submission is marked on a pass/fail basis - you will get full marks for submitting a *reasonable attempt*.
- Individual feedback will not be given, but solutions will be posted promptly after the deadline has passed.
- These exercises aim to give you practices on the materials taught during lectures and provide guidance towards preparation of examination.
- Relevant lectures: **Lectures 26-30**
- Turn to next page for the questions.

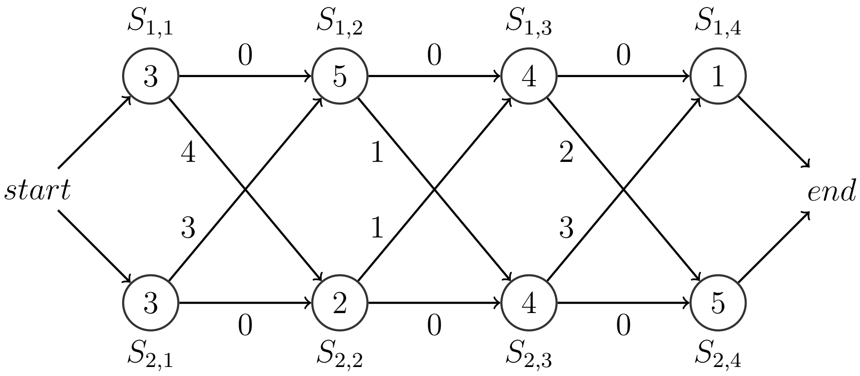
1. (a) Suppose we are given the following numbers: 40, 23, 11, 35, 15, 30, 18, 25. If we apply merge sort to sort these numbers into ascending order, the final merge step would be to merge the two sorted sequences:

$$(11, 23, 35, 40) \text{ and } (15, 18, 25, 30)$$

Draw figures to show this final merge step. Use two pointers to point to the two sorted sequences. Show step by step how the pointers move to obtain the final merged sequence.

(b) Merge sort makes use of an algorithm to merge two sorted sequences. Assume that the given two sequence are already in **descending order**. Modify the pseudo code Merge() in the lecture notes to merge two descending ordered sequences into one descending ordered sequence.

2. Suppose there are two assembly lines each with 4 stations, $S_{i,j}$. The assembly time is given in the circle representing the station and the transfer time is given next to the arrow from one station to another.



(a) Using dynamic programming, fill in the table of the minimum time $f_i[j]$ needed to get through station $S_{i,j}$. You should also show **all** the intermediate steps in computing these values, e.g., in computing $f_1[2]$, you need to specify that $f_1[2] = \min\{\_\_, \_\_\}$.

Intermediate steps:

j	$f_1[j]$	$f_2[j]$
1		
2		
3		
4		

(b) What is the minimum time f^* needed to get through the assembly line?

(c) Which stations should be chosen to achieve the minimum time?

3. Consider the following recurrence.

$$T(n) = \begin{cases} 1 & \text{if } n == 0 \text{ or } n == 1 \\ 2 & \text{if } n == 2 \\ T(n-1) + T(n-3) & \text{if } n > 2 \end{cases}$$

(a) Design and write a pseudo code for a recursive procedure to compute $T(n)$.

(b) Draw the execution tree for $T(7)$.

(c) Using the concept of dynamic programming, rewrite your recursive procedure into a non-recursive one.