Pre-tutorial questions

Do you know the basic concepts of this week's lecture content? These questions are only to test yourself. They will not be explicitly discussed in the tutorial, and no solutions will be given to them.

1. Running times
   (a) What is an algorithm's worst case running time?
   (b) What does it mean when we say that the algorithm runs in polynomial time?
   (c) What does it mean mean we say that an algorithm is efficient?
   (d) Look at the table with the running times in the slides. Can you explain the reason for these numbers?
2. Asymptotic analysis
   (a) Can you explain the ideas of the $O(\cdot)$, $\Theta(\cdot)$ and $\Omega(\cdot)$ functions? Why do we use these functions?
   (b) What does it mean that these functions are transitive and additive?
3. Basic data structures (revision)
   (a) What are linked lists, queues, stacks and balanced binary trees? What sort of operations are usually supported by these structures?
   (b) What is the height of a balanced binary tree containing n elements?
4. Graph terminology (revision)
   (a) What is a graph $G(V, E)$?
   (b) Graphs are usually represented either as an adjacency matrix or as an adjacency list. Can you explain the two representations?
   (c) What are the advantages/disadvantages between the two different representations?
   (d) What is a simple path in a graph?
   (e) What is a cycle in a graph?
   (f) What is a tree? If a tree has n vertices, how many edges does it have?
   (g) What is the difference between a rooted tree and an unrooted tree?
   (h) What is a bipartite graph?
5. Graph traversals (revision)
   (a) What is the difference between BFS and DFS?
   (b) Explain the two search algorithms BFS and DFS.

Tutorial

Problem 1

Sort the following functions in increasing order of asymptotic growth

$$n, n^3, n\log n, \frac{n}{\log n}, \frac{n}{\log^2 n}, \sqrt{n}, \sqrt{n^3}$$

Problem 2

Sort the following function in decreasing order of asymptotic growth

$$n^{1.5}, 2^n, \frac{n}{2^n}, \frac{2^n}{n^{10}}, n!, 1.5^n, 2^{\log_2 n}, n^{\frac{1}{\log_2 n}}$$

Problem 3

Which of the following is largest asymptotically

$$\log_3 n, \log_2 n, \log_{10} n$$

Problem 4

Use induction to prove that the sum $S(n)$ of the first n natural numbers is $n(n+1)/2$.

Problem 5

1. For each of the following pseudo-code fragments, give an upper bound for their running time using the big-Oh notation.
2. Typically, our goal is to prove an upper bound on the worst case running time of an algorithm. Sometimes, we want to prove a lower bound instead. Such a lower bound would show that there are "bad" instances which would cause the algorithm to take a long time. More formally, we say that an algorithm is $\Omega(f(n))$ if there exists constants $n_0, c > 0$ such that for every $n \geq n_0$, there exists an instance of size n such that the algorithm takes *at least* $cf(n)$ steps.

   For the second algorithm shown below, give a lower bound for the running time.

Algorithm 1 Stars

```
1: for i = 1, ..., n do
2:     print "*" i times
3: end for
```

1. $y=\sqrt{n}$, $\frac{n}{\log^2 n}$, $\frac{n}{\log n}$, n, $n\log n$, $\sqrt{n^3}$, n^3

2. $n!$, 2^n, $\left(\frac{2^n}{n^{10}}, 1.5^n\right)$, $n^{1.5}$, $2^{\log_2 n}$

$n^{\frac{1}{\log_2 n}}$ $\quad$ $\frac{2^n}{1.5^n n^{10}} = \frac{\left(\frac{2}{1.5}\right)^n}{n^{10}} \Rightarrow \infty$

3. all of them (dividing them gives a constant)

4. assume for k sum is $\frac{k(k+1)}{2}$

for $k+1$ prove still hold

$$\text{sum} = \frac{k(k+1)}{2} + k+1$$

$$= \frac{k^2+k+2k+2}{2}$$

$$= \frac{k^2+3k+2}{2}$$

$$= \frac{(k+1)(k+2)}{2}$$

5. 1. $O(n^2)$

2. $O(mn)$ $\Omega(nm)$

Algorithm 2 CHECK NUMBERS

```
1: procedure CheckNumbers(A,B)
        ▷ A and B are two lists of integers with |A| = n ≤ |B| = m
2:     count = 0
3:     for i = 1, ..., n do
4:         for j = i ... m do
5:             if A[i] ≥ B[j] then
6:                 count = count +1
7:                 break
8:             end if
9:         end for
10:    end for
11: end procedure
```

Problem 6

An undirected graph $G = (V, E)$ is said to be bipartite if its vertex set V can be partition into two sets A and B such that $E \subseteq A \times B$. Design an $O(n + m)$ algorithm to test if a given input graph is bipartite using the following guide:

1. Suppose we run BFS from some vertex $s \in V$ and obtain layers $L_1, \ldots, L_k$. Let (u, v) be some edge in E. Show that if $u \in L_i$ and $v \in L_j$ then $|i - j| \leq 1$.
2. Suppose we run BFS on G. Show that if there is an edge (u, v) such that u and v belong to the same layer then the graph is not bipartite
3. Suppose G is connected and we run BFS. Show that if there are no intra-layer edges then the graph is bipartite
4. Put together all the above to design an $O(n + m)$ time algorithm for testing bipartiteness.

Problem 7

Given an array A consisting of n integers $A[0], A[1], \ldots, A[n-1]$, we want to compute the upper triangle matrix

$$C[i][j] = \frac{A[i] + A[i+1] + \cdots + A[j]}{j - i + 1}$$

for $0 \leq i \leq j < n$. Consider the following algorithm for computing C:

Algorithm 3 summing-up

```
1: function SUMMING-UP((A))
2:     for i = 0, ..., n − 1 do
3:         for j = i, ..., n − 1 do
4:             add up entries A[i] through A[j] and divide by j − i + 1
5:             store result in C[i][j]
6:         end for
7:     end for
8:     return C
9: end function
```

1. Using the O-notation, upperbound the running time of SUMMING-UP.

6. BFS, alternate layers make up bipartite graph

7. $O(n^3)$, $\Omega(n^3)$

$i < \frac{1}{4}n$

$j > \frac{3}{4}n$

$\frac{1}{4}n \times \frac{3}{4}n \times \frac{1}{2}n$

$= n^3$

$\frac{1}{2}n$

2. Using the Ω-notation, lowerbound the running time of SUMMING-UP.

Problem 8

[Optional] Give an $O(n)$ time algorithm to detect whether a given undirected graph contains a cycle. If the answer is yes, the algorithm should produce a cycle. (Assume adjacency list representation.)

Problem 9

[COMP3927 only] Prove that the analysis of the number of iterations of Gale-Shapley algorithm in the lecture is tight by designing an instance where the number of proposals is $\Omega(n^2)$, where n is the number of men/women.