MAS714-Homework 1

Pham Tien Hung, Zhang Xu

Excercise 1

The sequence is:

1. f3
2. f2
3. f7
4. f5
5. f1
6. f4
7. f6

Excercise 2

a)

Answer: *True*. Given $f(n) = O(g(n))$, we will prove that $\log_2 f(n) = O(\log_2 f(n))$.
Proof

First, it is noted that we assume $f(n), g(n) \geq 1$ for all $n > 0$. This assumption ensures that the log-versions of f and g are non-negative, which is a condition for the usage of the big O notation.

We need to show that there exist $c > 0, n_0 \geq 0$ such that for all $n \geq n_0$, we have

$$\log_2 f(n) \leq c \log_2 g(n). \tag{1}$$

Since $f(n) = O(g(n))$, there exist $c' > 0, n_1 \geq 0$ such that for all $n \geq n_1$, we have

$$f(n) \leq c' g(n).$$

Clearly we can select $c' > 1$. Applying $\log_2$ to both sides, we have,

$$\log_2 f(n) \leq \log_2(c' g(n)) = \log_2 c' + \log_2 g(n).$$

Since $g(n)$ is increasing, for $n \geq n_1$,

$$\log_2 g(n_1) \leq \log_2 g(n),$$

$$\log_2 c' \leq \frac{\log_2 c'}{\log_2 g(n_1)} \log_2 g(n)$$

from which implies

$$\log_2 f(n) \leq (1 + \frac{\log_2 c'}{\log_2 g(n_1)}) \log_2 g(n).$$

We have $\log_2 f(n) = O(\log_2 f(n))$ follows.

b)

Answer: *False.* Here is one counterexample

Let $f(n) = 2n, g(n) = n$, clearly $2n = O(n)$. However

$$2^{2n} = 4^n \notin O(2^n)$$

c)

Answer: *True.*

Proof 1 *Since $f(n) = O(g(n))$, there exist $c > 0, n_0 \geq 0$ such that for all $n \geq n_0$, we have*

$$f(n) \leq cg(n),$$

which is equivalent to

$$f(n)^2 \leq c^2 g(n)^2.$$

This clearly implies $f(n)^2 = O(g(n)^2)$.

Excercise 3

a) Counting the number of addition, we have

$$f(n) = \sum_{i=1}^{N-1} \sum_{j=1}^{i} j = \sum_{i=1}^{N-1} \frac{i^2 + i}{2} = \frac{1}{2} \left[\frac{(N-1)N(2N-1)}{6} + \frac{N(N-1)}{2} \right]$$

which gives us

$$f(n) = \frac{N^3}{3} - \frac{N}{3} = \Theta(N^3)$$

b)

Algorithm 1: New algorithm

Input : $A[:]$
Output: $B[:,:]$

for $i \in [1, \ldots, n-1]$ **do**
  for $j \in [i+1, \ldots, n]$ **do**
    if $j == i + 1$ **then**
      $B[i, j] = A[i] + A[j]$
    else
      $B[i, j] = B[i, j-1] + A[j]$

This algorithm has in total

$$g(n) = \sum_{i=1}^{N-1} i = \frac{N(N-1)}{2} = \Theta(N^2)$$

Excercise 4

Proof

Consider a tree $T(V, E)$, let V_b be the set of binary nodes and V_l be the set of leaf node. We will prove by induction on the number of nodes $|V|$.

For $|V| = 2$, there can be no binary node while the number of leaf node is always 1. Therefore, this is true for this case.

Assume that for all graphs $|V| = n$, this property holds. We now prove that for any graph with $|V| = (n + 1)$, this property also holds.

Consider a tree with $n+1$ nodes, we select a leaf node in V_l. Its direct parent can be either 1) a binary node or 2) not a binary node. We now remove this leaf node from the original tree $T(V, E)$ to create a new tree $T'(V', E')$. Since T' has n nodes, we know that $|V'_b| = |V'_l| - 1$.

In case 1), we have $|V'_b| = |V_b| - 1$ and $|V'_l| = |V_l|$ -1 which implies

$$|V_b| = |V_l| - 1.$$

In case 2), the number of binary nodes and leaf nodes remain unchanged. Thus, the property holds for any tree with $n+1$ nodes. With this our induction finishes.

Excercise 5

Proof

We will show that

1. There is no cross-edges and forward-edges in a DFS tree of an undirected graph.
2. DFS(G, v) = BFS(G, v) = T implies there is no back-edges in the graph returned from DFS.

These are sufficient to deduce G=T.

We will now prove 1a): there is no forward-edges in a DFS tree of an undirected graph. Assume the contradiction, there exist a unlabeled edge (u, v) at sometime t,

$$pre(u) < pre(v) < t.$$

Now, since t is when we are have already finish the recursions of the childs of u from which one is ancestor of (or is) v. We have

$$pre(u) < pre(v) < post(v) < t$$

It is clear that for a given vertex v, at a time t larger than $post(v)$, all undirected edges connecting to v are all labeled. This gives a contradiction since (u, v) is unlabeled at time $t > post(v)$.

The non-existence of cross-edges (1b) can be proved similarly.

1a. We will prove by contradition. Assume that there is a back-edge in the graph G. This implies there is an unlabeled edge (u, v) such as

$$pre(v) < pre(u)$$

Now since T is also a BFS tree, this implies that when v is visited, u is already visited or otherwise the edge (u, v) would be in the tree and not a forward-edge. This leads to a contradiction because v is an ancestor of u.