Solutions – Quiz-01

Q1 **(a)** $(\Rightarrow)$ $f(n) = O(n^k) \Rightarrow \exists\, c_1 > 0,\ n_0 \in \mathbb{N}$ s.t

$$f(n) \le c_1 \cdot n^k \quad \forall \quad n \ge n_0$$

$$\Rightarrow \log(f(n)) \le \log c_1 + k \log n = k\left\{\frac{\log c_1}{k} + \log n\right\} \quad \forall n \ge n_0$$

Now, $\exists\ n_1$ s.t $\frac{\log c_1}{k} \le \log n_1$ $\quad (\because$ LHS = constant, RHS = increasing $f^n)$

$$\exp\left(\frac{\log c_1}{k}\right) \le n_1$$

$$\Rightarrow \log(f(n)) \le 2k \log n \quad \forall\ n \ge \max(n_0, n_1).$$
$$= O(\log n)$$

$(\Leftarrow)$ $\log(f(n)) = O(\log n)$

$\exists\ c_1 > 0,\ n_0 \in \mathbb{N}$ s.t $\log(f(n)) \le c_1 \log n \quad \forall\ n \ge n_0.$

$$\Rightarrow f(n) \le \exp(\log n^{c_1}) \quad \forall\ n \ge n_0$$
$$\le n^{c_1} \quad \forall\ n \ge n_0$$
$$= O(n^{c_1})$$

Hence (a) is true.

(b) Consider $f(n) = n$ and $g(n) = n^2$

$$\min(f(n), g(n)) = \begin{cases} n^2 & n < 1 \\ n & n \ge 1 \end{cases}$$

$$f(n) + g(n) = n^2 + n = \Theta(n^2) \ne \Theta(\min(f(n), g(n))) = \Theta(n)$$

Hence (b) is false.

© $e^n = f(n)$ $\quad f(n/2) = e^{n/2}$

If $f(n) = \Theta(f(n/2))$, then $\exists\ c_1, c_2 > 0$ and $n_0 \in \mathbb{N}$ s.t

$$c_1 \cdot e^{n/2} \le e^n \le c_2 \cdot e^{n/2} \quad \forall \quad n \ge n_0$$

$$\Rightarrow \log c_1 + n/2 \le n \le \log c_2 + n/2 \quad \forall\ n \ge n_0.$$

$$\Rightarrow 2\log c_1 \le n \le 2\log c_2 \quad \forall\ n \ge n_0$$

but $\mathbb{N}$ is unbounded. Contradiction.

Hence © is <u>false</u>.

Q2 directed graph $G = (V, E)$ $\quad |V| = n \quad |E| = m$

$(u, v) \in E$ if u forwards rumor to v.

$p \in V$ is influential if $\forall\ a \in V$ $\exists$ path from p to a in G.

(a) (i) Claim: All influential people in G are in the same strongly connected component.

Proof: Let P_1, P_2 be influential persons. Then by def^n of influential, $\exists$ a path from P_1 to P_2 and from P_2 to P_1. Since P_1, P_2 are arbitrary, all influential persons are connected and hence belong to same strongly connected component.

(ii) Claim: Everyone in this SCC is influential.

Proof: Let S be the SCC with all influential persons. Suppose to the contrary that $u \in S$ is not influential. $\Rightarrow \exists\ v \in V$ s.t there is no path from u to v. $u \in S \Rightarrow \exists$ path from u to influential person, say p. By def. $\exists$ path from p to v. Contradiction!

(b) Algorithm:

1. Given $G=(V, E)$, run dfs on G starting from any arbitrary vertex $v \in V$ and store finish times of vertices. — $O(n+m)$
2. Pick the vertex with largest finish time, say s. — $O(n)$
3. Start a dfs from s and store the visited vertices. Terminate the dfs when s exits. — $O(n+m)$
4. If visited vertices $= V$: — $O(n)$

   RETURN s

   Else:

   RETURN "no influential person".

Runtime: Algorithm involves running dfs twice — $O(n+m)$ time.
Auxillary steps include inspecting vertices — $O(n)$ time.
Total Run time: $O(n+m)$

Proof of correctness:

Claim: If G has atleast one influential person, then the vertex with largest finish time (last to exit) will be influential.

Proof: Suppose G has influential person(s).
We know that if there is a path from v to u then (if $pre(v) < pre(u)$, then finish time$(u) <$ finish time(v)) in any dfs traversal of G. —— (2)

To the contrary, suppose the vertex s with largest finish time is not influential. Let I be an influential vertex with largest finish time.

$\Rightarrow$ finish time$(I) <$ finish time(s). —— (1)

Case-1: $pre(I) > pre(s)$.

$\Rightarrow$ $\exists$ path from s to I. (from ①, ②)

We know $\exists$ path from I to v for every $v \in V$.

$\Rightarrow$ $\exists$ path from s to v for every $v \in V$.

$\Rightarrow$ s is influential. Contradiction.

Case-2: $pre(I) < pre(s)$.

We know that $\exists$ path from I to s.

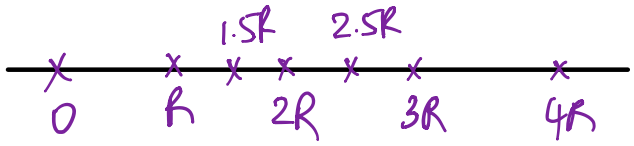
$\Rightarrow$ finish time (s) < finish time (I)

Contradiction. to ①

$\therefore$ s has to be influential.

This ensures the correctness of our algorithm. If $\exists$ an influential person in G, then the last vertex to exit G will be influential Our algorithm will return an influential person. If not, then the last person to exit the dfs will not be influential and the check fails, we return "no influential person" as required.

Q3 (a)



Consider an instance where towns be located at

0, R, 1.5R, 2R, 2.5R, 3R, 4R.

Using the algorithm: build train stations where you maximize the # towns covered, until all towns are covered.

→ We first build a station at 2R:

covers towns at R, 1.5R, 2R, 2.5R, 3R → 5 in #.

Next, we have to build two more stations to cover the towns at 0, 4R.

Hence, this algorithm builds 3 stations

Consider the following: build stations at R and 3R

covers towns 0, R, 1.5R, 2R

covers towns 2.5R, 3R, 4R.

Hence $OPT \leq 2 < 3$. ∴ Given algorithm is incorrect.

(b) We will say that station S covers town T if T is within distance R of S

Algorithm:

$T' \leftarrow T$ Initialize set of uncovered towns.

$S \leftarrow \phi$ Initialize location of stations.

While $T' \neq \phi$:

Let t be the leftmost town in T', at 'd' on the line.

$S \leftarrow S \cup \{$station s at 'd+R' on the line$\}$

$T' \leftarrow T' \setminus \{$towns covered by s$\}$

RETURN S.

Note: Algorithm can be implemented in $O(|T|)$ time.

Proof of correctness:

Claim: Algorithm returns a set of stations that covers every station.

↳ True by construction.

Let OPT be an optimal solution and ALG be the solution obtained by using the above algorithm.

We look at the leftmost station that is placed differently in OPT, ALG (If there is no such station, we are done any way s).

Let S_1 be location of the station in OPT and S_2 be the location in ALG.

Let t be the location of the town that decided S_2: $t = S_2 - R$.

Since all stations preceding are same in OPT, ALG, t isn't covered yet. S_1, S_2 must cover t. $\Rightarrow$ $S_1 < S_2 \Rightarrow$ all uncovered towns covered by S_1 are also covered by S_2

Consider a solution $OPT' = OPT \cup \{S_2\} \setminus \{S_1\}$.

Observe that OPT' also covers all towns and has same # stations as in OPT. Hence OPT' is also optimal and is matching ALG on one more station.

Hence, by repeated exchange arguments, we see that ALG is also optimal.