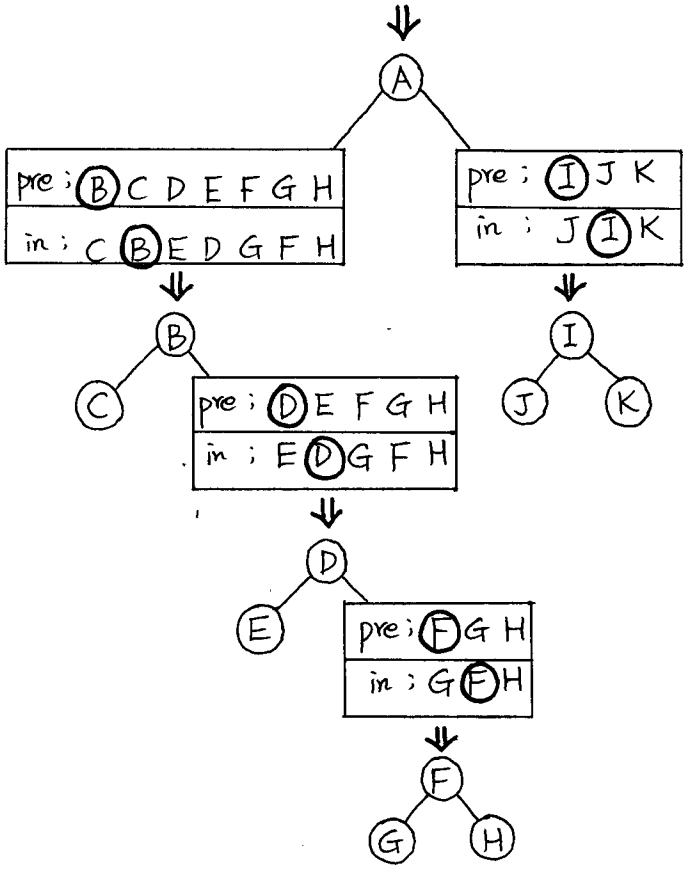
[CS251] Homework 2 Solution

Quick Solution Key: 2/3/3/1/1/2/4/3/1/4.

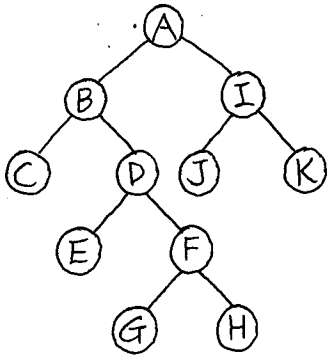
#1~2. preorder : (A) B C D E F G H I J K (root → left → right)

inorder : C B E D G F H (A) J I K (left → root → right)

left children　　right children



∴ The binary tree T looks like :



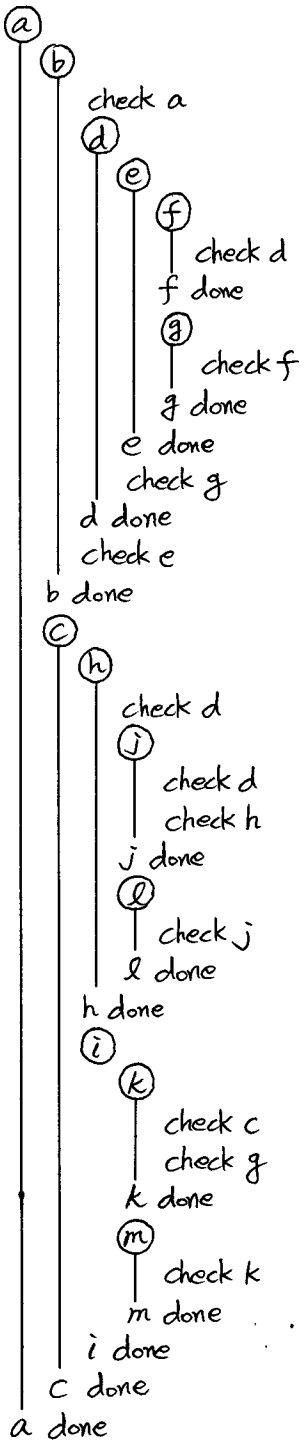
⇒ postorder : C E G H F D B J K I A　　∴ 2

(left → right → root)

level-order : A B I C D J K E F G H　　∴ 3

■

#3. Running a dfs on G^{Rev} starting at a vertex a:



∴ The order in which the vertices are visited is

a b d e f g c h j l i k m ∴ 3

#4. The algorithm is as follows:

1) Run dfs on G^{Rev} to compute reverse postorder

2) Run dfs on G, considering vertices in order given by the previous dfs

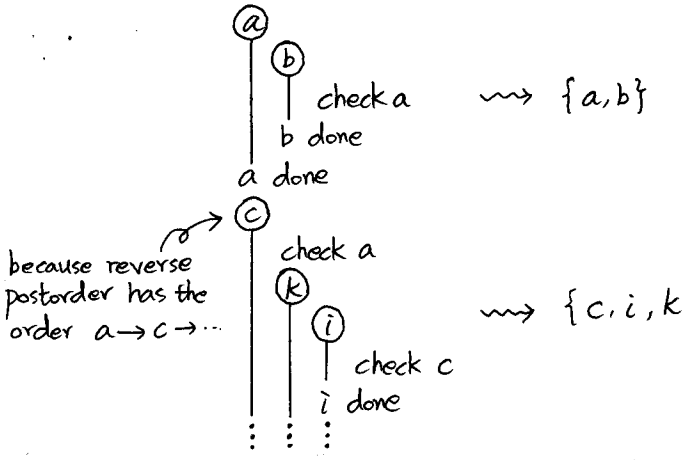
⇝ reverse postorder: a c i m k h l j b d e g f

(read the vertices which are done in a reverse order)

Now, to run dfs on G, we need to make adjacency lists representation for G:

a → b
b → a
c → a k
d → b f h j
e → b d
f → e g
g → d e k
h → c j
i → c
j → h l
k → i m
l → h
m → i

Based on the lists above, run dfs on G in order given by the reverse postorder:



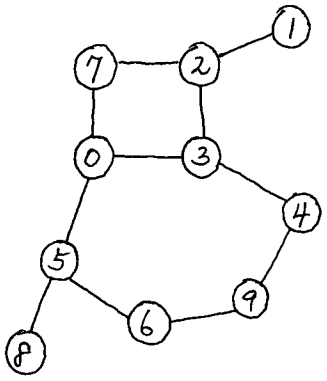
⇝ {a, b}

⇝ {c, i, k, m}

∴ 1

(lack of page ⋯ you can finish it!)

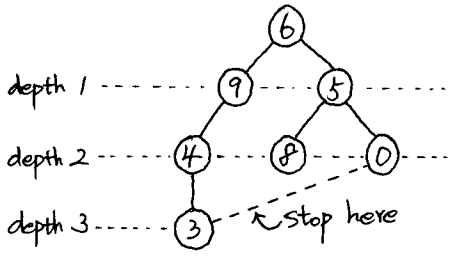
#5. Consider the following example :



[adjacency lists]

0 → 3 5 7
1 → 2
2 → 1 3 7
3 → 0 2 4
4 → 3 9
5 → 6 8 0
6 → 9 5
7 → 0 2
8 → 5
9 → 4 6

To compute $f(6)$, do a bfs starting at 6 ⇒ the breadth-first tree is



$\therefore f(6) = \text{depth}(0) + \text{depth}(3) + 1$

$= 2 + 3 + 1 = 6.$

↑ such 1 indicates the non-tree edge connecting ⓪ and ③.

$f(6)$ is the length of one cycle in G.

(∵ the root ⑥ is included in the cycle)

Similarly, we have that $\begin{cases} f(0) = f(2) = f(3) = f(7) = 4. \\ f(1) = f(4) = f(5) = f(6) = f(9) = 6 \\ f(8) = 8. \end{cases}$

⇒ $\Delta = \min_v f(v) = 4$, the length of a shortest cycle in G. ∴ 1

#6. In the worst case, we require n bfs (bfs at each vertex).

Think about when do we get a cycle while exploring.

⇒ In the worst case, it occurs after $(n-1)$ edges are explored

(∵ the tree will have only $(n-1)$ edges)

∴ $n(n-1) = O(n^2)$. ∴ 2

▨

#7. 1. If $n = 1 + z$, then $X \subset Y \subset Z \Rightarrow \ell(X, Y) = x$. (true).

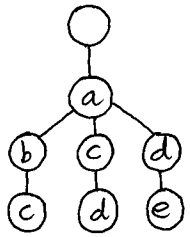
2. (counterexample) $\left.\begin{array}{l} X = \text{sky} \\ Y = \text{skype} \\ Z = \text{skypechat} \end{array}\right\} \Rightarrow X \neq Y$, but $n = 1 + z$.

$\therefore$ (false)

3. If $n = 1 + x + y + z$, there is no common prefix between any two strings.

$\Rightarrow \ell(X, Y) = \ell(X, Z) = \ell(Y, Z) = 0$. (true)

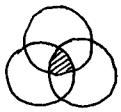
4. (counterexample) $\left.\begin{array}{l} X = \text{abc} \\ Y = \text{acd} \\ Z = \text{ade} \end{array}\right\} \Rightarrow$ the resulting R-way trie is



$\therefore n = 8$, but $1 + x + y + z - \ell(X, Y) - \ell(X, Z) - \ell(Y, Z)$

$= 1 + 3 + 3 + 3 - 1 - 1 - 1 = 7$. $\therefore$ (false)

$\because n = 1 + x + y + z - \ell(X, Y) - \ell(X, Z) - \ell(Y, Z) + \ell(X, Y, Z)$



$\ell(X, Y, Z)$: the length of the longest common prefix to X, Y, Z.

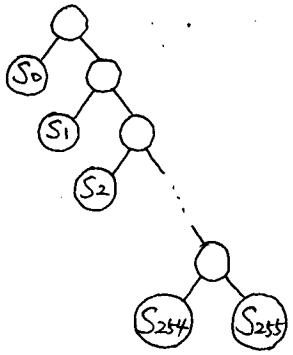
$\therefore$ 4 ▨

#8. • If $f_0 = f_1 = \cdots = f_{255}$, the Huffman tree will be a complete binary tree.

$\Rightarrow$ every symbol's code has length $\log_2 256 = 8$.

• If $f_i > f_{i+1} + \cdots + f_{255}$ for every $i = 0, 1, \cdots, 254$, the Huffman tree is

($\Rightarrow f_0 > f_1 > \cdots > f_{254} > f_{255}$)



$\Rightarrow S_i$ has length $1 + i$ for $i = 0, 1, \cdots, 254$.

$\therefore$ 3 ▨

#9. Intuitively, we can observe that every vertex has an even degree. Therefore, we have that this problem is related to Euler graph. In Euler graph, we can find an Euler cycle so that we could easily conclude that such graph is strongly connected.

Formally, we claim the following conclusion:

A digraph has an Euler cycle* if and only if it is connected and the indegree of each vertex equals its outdegree.

---

* Euler cycle: a closed path that goes through each edge exactly once

proof) Let $G(V, E)$ be a connected digraph with V as the set of vertices and E the set of edges. (we're only interested in the "if" part)

If $|V| = 1$, there is nothing to prove.

Otherwise, start with any vertex, proceed along any outgoing edge, and continue this way until a cycle has been created.

(Since indegree = outdegree, it could not end at a vertex from which there is no way out unless this is where the path has started.)

If the cycle contains all edges of E, we're done.

Otherwise, denote the edges of the cycle E' and the nodes V' and consider $G(V-V', E-E')$.

Since G is connected, the cycle could not be a connected component (∵ We assumed it's not an Euler cycle)

⇒ Every connected component of G' shares at least one vertex with the cycle.

Now, start with those vertices to find a cycle in each of the connected components of G', continue in this manner until no edges remain.

⇒ The process ends up with a set of cycles with vertices in G whose union contains every edge from E exactly once.

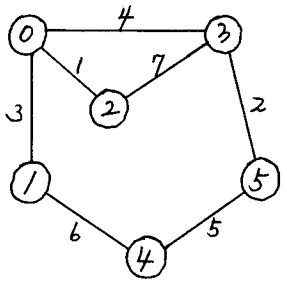
Since the union of a finite number of cycles of a connected digraph is a cycle, we're done.

⇒ Since G has an Euler cycle, it is strongly connected. ∴ /

▨

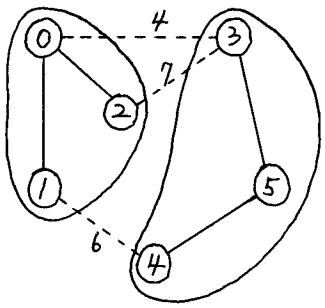
#10. Consider the following example :

G_0 :



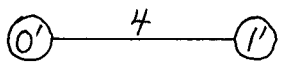
Then we have

Cheap(0) = 2
Cheap(1) = 0
Cheap(2) = 0
Cheap(3) = 5
Cheap(4) = 5
Cheap(5) = 3.



$\therefore$ H_0 has two connected components.

Collapsing those into a single vertex, and choosing the cheapest edge (which is 4) only, we get :

G_1 : 

$\Rightarrow$ G_2 : (0'') $\therefore$ $\hat{k} = 2$.

Looking at the process, every connected component in H_{k-1} contains at least two vertices of G_{k-1} by construction.

$\therefore$ If $G_{k-1} = G_{k-1}(V_{k-1}, E_{k-1})$ and $G_k = G_k(V_k, E_k)$, then

$$|V_k| \leq \frac{1}{2}|V_{k-1}|.$$

Therefore, we clearly have that $\hat{k} \leq \log_2 n$.

($\because$ $1 = |V_{\hat{k}}| \leq \frac{1}{2}|V_{\hat{k}-1}| \leq \left(\frac{1}{2}\right)^2 |V_{\hat{k}-2}| \leq \cdots \leq \left(\frac{1}{2}\right)^{\hat{k}} |V_0| = \left(\frac{1}{2}\right)^{\hat{k}} n$

$\Rightarrow$ $2^{\hat{k}} \leq n$ $\Rightarrow$ $\hat{k} \leq \log_2 n$)

Moreover, since we only choose the cheapest edges, the sum of the costs of the edges in $H_0, \cdots, H_{\hat{k}-1}$ would be equal to the cost of the minimum spanning tree of G. It's clear that it is impossible to make a cycle during the construction.

$\Rightarrow$ Statements 1 and 3 are true.

$\therefore$ 4

▨