

HW04 for ECE 9343

Tongda XU, N18100977

November 13, 2018

1 Question 1: CLRS Exercise 22.1-3

```
TRANSPOSE(Adjlist)
1  new AdjlistPrime
2  for each node in Adjlist
3      for each subnode in Adjlist(node)
4          AdjlistPrime(subnode).insert(node)
5      Adjlist = AdjlistPrime
```

For adjacent list: just traverse every node and rebuild one
 $\Theta(E + V)$ for time and space complexity, hard to do it inplace

```
TRANSPOSE(Adjmatrix)
1  for each pair(i, j) in upper left Adjmatrix
2      SWAP(Adjmatrix[i, j], Adjmatrix[j, i])
```

For adjacent matrix: just transpose the matrix
 $\Theta(V^2)$ for time and $\Theta(1)$ for space

2 Question 2: CLRS Exercise 22.1-5

For adjacent list, it is hard. We should regard it as a BREADTH-FIRST-SEARCH(G)
end at $d = 2$:

```
SQUARE(G)
1  for each u in G.vertices
2      G.reset()
3      list =  $\emptyset$ 
4      u.adjlist' = BFS-AID(G, u, list, 0)
```

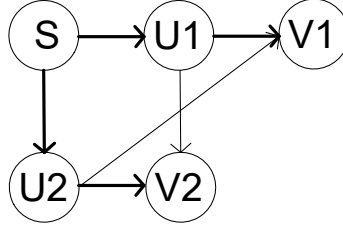


Figure 1: 22.2-6

```

BFS-AID( $G, u, list, dist$ )
1  for each  $v$  in  $u.adjlist$ 
2      if  $v.color = white$  and  $dist \leq 2$ 
3           $list.insert(u)$ 
4          BFS-AID( $G, v, list, dist + 1$ ) =
5  return  $list$ 

```

This could cost $\Theta(V^2 + VE)$ time and $\Theta(V + E)$ space (if optimized).

For adjacent matrix, the square process would be simple. for each index m of matrix row, if $matrix[m][n]$ exist, calculate bool union of $matrix[m]$ and $matrix[n]$:

```

SQUARE( $G$ )
1  for each  $m$  in  $G.adjMatrix$ 
2      for each  $n$   $G.adjMatrix[m]$ 
3          if  $G.adjMatrix[m][n] == 1$ 
4               $G'.adjMatrix[m] = AND(G.adjMatrix[m], G.adjMatrix[n])$ 
5  return  $G'$ 

```

The SQUARE(G) cost $\Theta(V^3)$ time and $\Theta(V)$ space (if optimize)

3 Question 3: CLRS Exercise 22.2-6

Consider the following condition in Figure 1:

$E_\pi = \langle s, u1 \rangle, \langle u1, v1 \rangle, \langle s, u2 \rangle, \langle u2, v2 \rangle$

In BFS Tree, $\delta(s, v1), \delta(s, v2)$ is either $\langle s, u1, v1 \rangle, \langle s, u1, v2 \rangle$ or $\langle s, u2, v1 \rangle, \langle s, u2, v2 \rangle$

4 Question 4: Traverse Edge of undirected graph

According to *Theorem 22.10*, all edges are either tree edge or back edge. Modify the $\text{DFS-VISIT}(G, u)$, add a $\text{PRINT-PATH}(G, u)$ would do it. Assume a $\text{root} = u$ is selected:

```
DFS-VISIT( $G, u$ )
1   $u.\text{color} = \text{grey}$ 
2   $\text{dict}[(\text{Vertex}, \text{Vertex}), \text{edgeType}] = \emptyset$ 
3  for each  $v$  in  $u.\text{adjList}$ 
4      if  $v.\text{color} == \text{white}$ 
5           $\text{dict}(u, v) = \text{treeEdge}$ 
6          DFS-VISIT( $G, v$ )
7      else  $\text{dict}(u, v) = \text{backEdge}$ 
8  PRINT-PATH( $G, u$ )
```

```
PRINT-PATH( $G, u$ )
1  PRINT (" $u$ ")
2  for each  $v$  in  $u.\text{adjList}$ 
3      if  $(u, v) == \text{treeEdge}$ 
4          PRINT (" $\rightarrow$ ")
5          PRINT-PATH( $G, v$ )
6      else PRINT (" $\rightarrow v$ ")
```

line 4,6 cost same level of time as the comparison in line 3, would not change the $\Theta(V + E)$ time complexity of $\text{DFS}(G)$

the print path function as:

This procedure cost $\Theta(V + E)$ as well

5 Question 5: CLRS Exercise 22.3-12

Tweak the $\text{DFS-VISIT}(G, u)$ and $\text{DFS}(G)$ would be enough:

```
DFS( $G$ )
1  for each  $u$  in  $G.V$ 
2       $u.\text{color} = \text{white}$ 
3   $c = 1$ 
4  for each  $u$  in  $G.V$ 
5      if  $u.\text{color} = \text{white}$ 
6          DFS-VISIT( $G, u, c$ )
7       $c++$ 
```

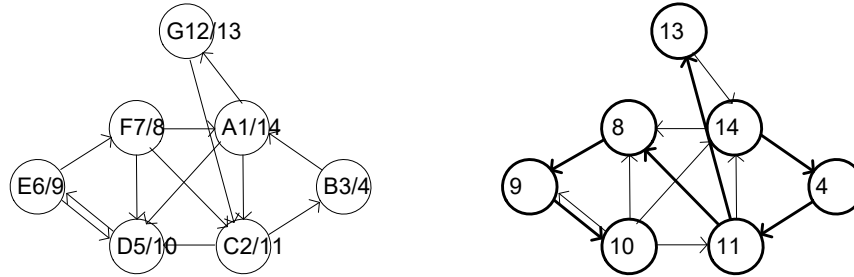


Figure 2: 22.2-6

DFS-VISIT(G, u, c)

```

1   $u.color = grey$ 
2   $u.cc = c$ 
3  for each  $v$  in  $u.adjList$ 
4      if  $v.color == white$ 
5          DFS-VISIT( $G, v$ )

```

DFS(G) could be tweaked to do it as well

6 Question 6: CLRS Exercise 22.4-1

$p[27 : 28] \rightarrow n[21 : 26] \rightarrow o[22 : 25] \rightarrow s[23 : 24] \rightarrow$
 $m[1 : 20] \rightarrow r[6 : 19] \rightarrow y[9 : 18] \rightarrow v[10 : 17] \rightarrow x[15 : 16] \rightarrow$
 $w[11 : 14] \rightarrow z[12 : 13] \rightarrow u[7 : 8] \rightarrow q[2 : 5] \rightarrow t[3 : 4]$

7 Question 7: Show process of SCC

See Figure 2

8 Question 8: CLRS Problem Set 22.1

8.1 a-1

Suppose (v, u) is a backedge. u is ancestor elder than parent of v . This means $(s, u) + \text{forwardEdge}$ is shorter than (s, v) produced by BFS which is $\delta(s, v)$ by **Theorem 22.5**. Same reason for forward edge.

8.2 a-2

By **Theorem 22.5** $\delta(s, v) = \delta(s, v.parent) + (v.parent, v) = \delta(s, u) + (u, v) \rightarrow v.d = u.d + 1$

8.3 a-3

$v.d \leq u.d + 1$: Same as a-1, if $v.d > u.d + 1$, $\delta(s, v) = (s, u) + cross$ instead of (s, v) .

$v.d \geq u.d$: If $v.d < u.d$, (v, u) should be find out first, since this is undirected graph.

8.4 b-1

Same as a-1, the $(s, u) + backEdge$ would be shorter than (s, v)

8.5 b-2

Same as a-2, By **Theorem 22.5** $\delta(s, v) = \delta(s, v.parent) + (v.parent, v) = \delta(s, u) + (u, v) \rightarrow v.d = u.d + 1$

8.6 b-3

Only the first half of a-3. if $v.d > u.d + 1$, $\delta(s, v) = (s, u) + cross$ instead of (s, v) .

8.7 b-4

By **Corollary 22.4** and By **Theorem 22.5**, we know that if v is an ancestor of u $\delta(s, u) = \delta(s, v) + k \rightarrow \delta(s, u) > \delta(s, v) \rightarrow u.d > v.d$, I did not see how $u.d = v.d$ but the statement is correct.

9 Question 9: CLRS Problem Set 22.3

9.1 1. proof

Euler tour exist \rightarrow in-degree == out-degree: Suppose the cycle through i vertex n times would be $E - cycle = \{v_i, v_j, v_k, \dots, v_i\}$. The in-degree of v_j would be the time of v_j appears with element in front, and out-degree of v_j would be the time of v_j appears with element in the back. If v_j is not head or tail, this is obvious that every time v_j appear, there is element in front and tail. If v_j is head, it must also be tail, which balance the in-degree and out-degree again.

in-degree == out-degree \rightarrow Euler tour exist

9.2 2. implement

This is very similar to SCC, we find closed cycle first then join them with other edge set. This procedure would return a cycle, which is a list of vertex. closed cycle has $cycle.begin() = cycle.end()$, open cycle(path, not a cycle) do not has it. But if Euler tour exist, open cycle would join close cycle into a big cycle.

```

CIRCLEFIND(cycle, u, v)
1  ClosedCycleSet, OpenCycleSet =  $\emptyset$ 
2  while alladjList! =  $\emptyset$ 
3      for v in Vertex with adjList! =  $\emptyset$ 
4          if v.adjList! =  $\emptyset$ 
5              new cycle =  $\emptyset$ 
6              CIRCLEFINDAID(cycle, u, NIL)
7              if cycle.type == closed
8                  ClosedCycleSet.push(cycle)
9              else OpenCycleSet.push(cycle)

```

```

CIRCLEFINDAID(cycle, u, v)
1  cycle.insert(u)
2  if v! = NIL
3      v.adjList.erase(u)
4  if u == NIL
5      cycle.type = open
6      return
7  elseif u == cycle.start
8      cycle.type = close
9      return
10 else v = u
11     u = u.adjList.begin()
12     CIRCLEFIND(cycle, u, v)

```

It is easy to find that as we remove an edge from adjacent list once we find it, and we traverse every edge, the time complexity would be $\Theta(E)$

10 Question 10: CLRS Exercise 12.2-1

- impossible since {330, 344, 397, 363} should be in left sub tree of node 338
- possible
- impossible since {912, 245, 363} should be in left sub tree of node 911
- possible
- impossible since {299, 392, 358, 363} should be in right sub tree of node 347

11 Question 11: CLRS Exercise 12.2-5

For successor: **if node X's right child has no left child:** then the successor would be X's right child which has no left child. **If node X's right child has left child:** then the successor would be the minimum element in the left sub tree of X's right child, which could not have left child or its left child would be less than it self.

For predecessor: same as successor.

12 Question 12: CLRS Exercise 12.3-3

Best case: $\Theta(n \log n)$ Consider a sorted sequence. Each time we insert the median and slice into before median and after median, and do the same process recursively.

Worst case: $\Theta(n^2)$ Consider constantly insert element at the end of a *List*. This could happen when we insert the *in-order* traverse sequence either in an increase or decrease order.

13 Question 13: CLRS Exercise 12.3-3

13.1 a

The value would be $\{16, 23, 7, 20\}$

13.2 b

For insertion, only one rotation would be triggered

13.3 c

See Figure 3

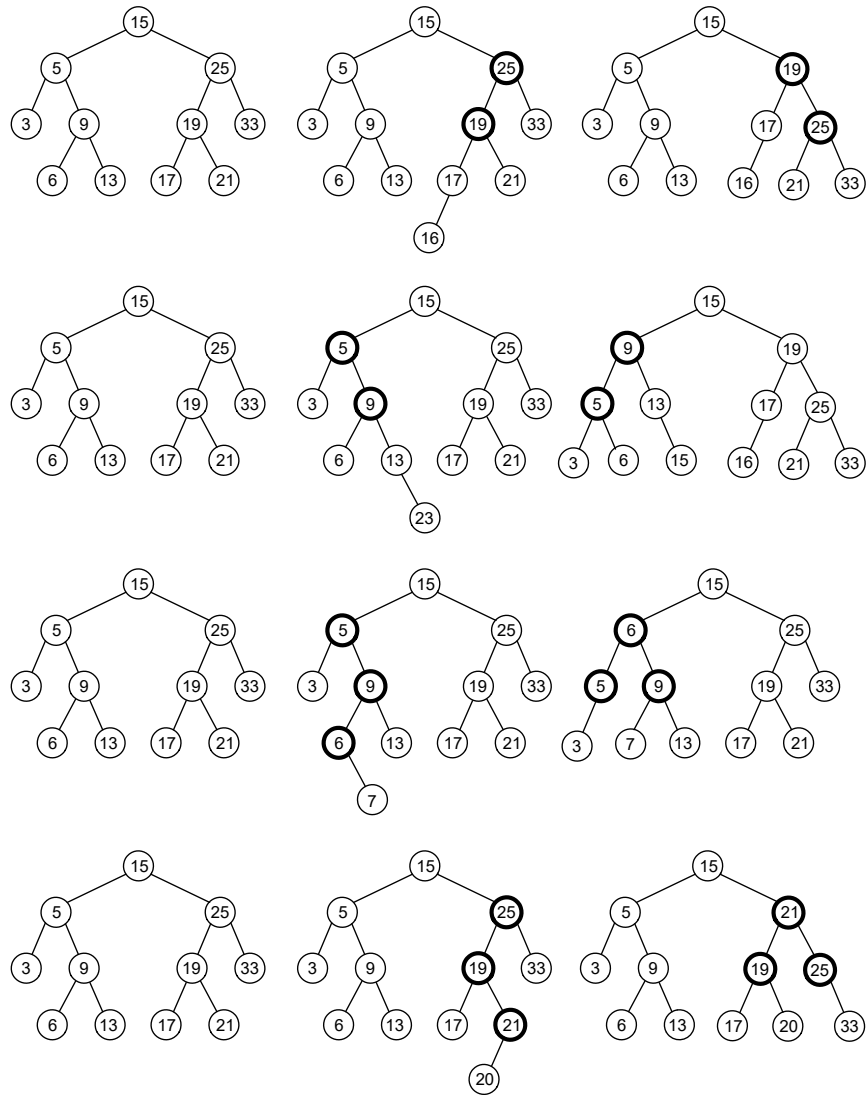


Figure 3: 22.2-6