BCB570 Assignment 2

Ashish Jain

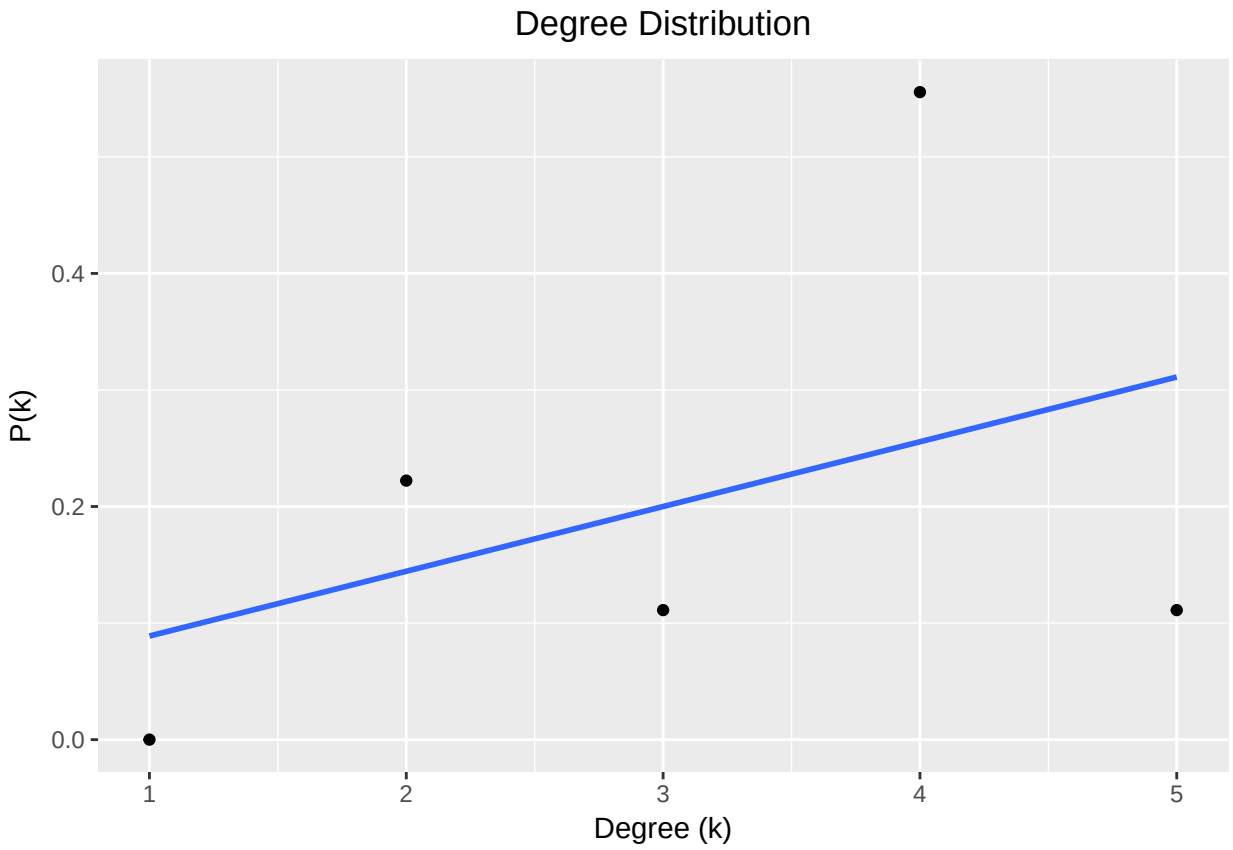
Question 1:

```
library("tidyverse")
nodesNames <- c("u", "w", "v", "y", "x", "z", "r", "t", "s")
degreeDistribution <- c(2, 4, 4, 4, 4, 5, 4, 3, 2)
degreeProb <- c(0, 2/9, 1/9, 5/9, 1/9)
degrees <- c(1:5)
ccDistribution <- c(1, 0.5, 0.5, 0.5, 0.5, 2/5, 0.5, 2/3, 1)
data <- data.frame(degree = degrees, degreeProbabilty = degreeProb,
    cc = (c(0, 1, 2/3, 0.5, 2/5)), ccProb = c(0, 2/9, 1/9, 5/9,
        1/9))
table <- data.frame(Node = nodesNames, Degree = degreeDistribution,
    CC = ccDistribution)
table %>% knitr::kable(caption = "Graph Details")
```

Table 1: Graph Details

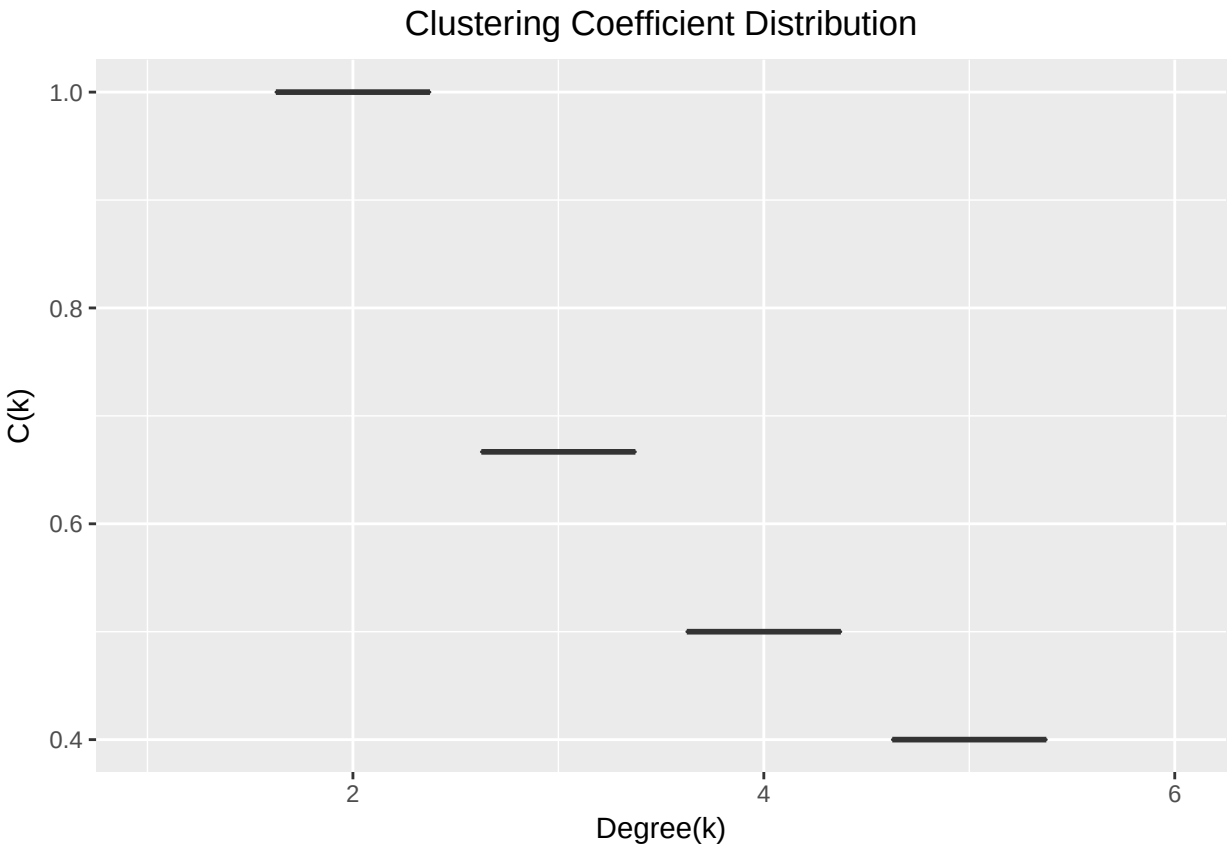
Node	Degree	CC
u	2	1.0000000
w	4	0.5000000
v	4	0.5000000
y	4	0.5000000
x	4	0.5000000
z	5	0.4000000
r	4	0.5000000
t	3	0.6666667
s	2	1.0000000

```
data %>% ggplot(aes(x = degrees, y = degreeProbabilty)) + geom_point() +
    labs(y = "P(k)", x = "Degree (k)", title = "Degree Distribution") +
    theme(plot.title = element_text(hjust = 0.5)) + geom_smooth(method = "lm",
    se = FALSE)
```



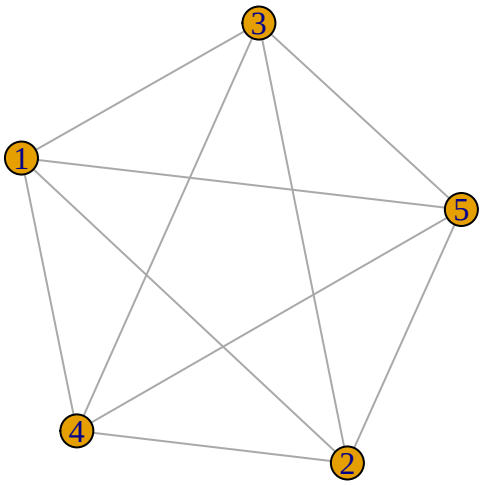

```
# boxplot(ccDistribution~degreeDistribution,xlab='Degree(k)',ylab='Clustering
# Coefficient (C(k))',main='Clustering Coefficient
# Distribution')

table %>% ggplot(aes(x = Degree, y = CC)) + geom_boxplot(aes(group = Degree)) +
    xlim(c(1, 6)) + labs(y = "C(k)", x = "Degree(k)", title = "Clustering Coefficient Distribution") +
    theme(plot.title = element_text(hjust = 0.5))
```



Question 4:

```
library("igraph")
completegraph <- graph(combn(1:5, 2), directed = FALSE)
plot.igraph(completegraph)
```



Question 5:

```
graphDetails <- function(edgeFile) {
    graph <- as.matrix(read.table(edgeFile))
    graphGraph <- graph.data.frame(graph, directed = FALSE)
    graphDetails <- c(vcount(graphGraph), ecount(graphGraph),
        graph.density(graphGraph), diameter(graphGraph), radius(graphGraph),
        transitivity(graphGraph, type = "average"), transitivity(graphGraph,
            type = "global"), mean_distance(graphGraph), toString(head(order(igraph::degree(graphGraph)
            decreasing = TRUE), 5)), toString(head(order(igraph::betweenness(graphGraph),
            decreasing = TRUE), 5)), toString(intersect(head(order(igraph::degree(graphGraph),
            decreasing = TRUE), 5), head(order(igraph::betweenness(graphGraph),
            decreasing = TRUE), 5))))
    return(list(graphDetails, graphGraph))
}

details <- c("Number of Nodes", "Number of Edges", "Graph Density",
    "Diameter", "Radius", "Average Clustering Coefficent", "Global Clustering Coefficent",
    "Average Shortest Path", "Top 5 Central Nodes(Hightest Degree)",
    "Top 5 Central Nodes (Betweeness)", "Common Nodes")
path <- "https://raw.githubusercontent.com/ashishjain1988/BCB570/master/HW2/"
outputA <- graphDetails(paste0(path, "graphA.txt"))
outputB <- graphDetails(paste0(path, "graphB.txt"))
outputC <- graphDetails(paste0(path, "graphC.txt"))

table <- data.frame(Headings = details, Graph_A = outputA[[1]],
    Graph_B = outputB[[1]], Graph_C = outputC[[1]])
table %>% knitr::kable(caption = "Graph Details")
```

Table 2: Graph Details

Headings	Graph_A	Graph_B	Graph_C
Number of Nodes	100	100	200
Number of Edges	140	850	222
Graph Density	0.0282828282828283	0.171717171717172	0.0111557788944724
Diameter	10	3	12
Radius	6	2	1
Average Clustering Coefficent	0.006	0.392361901365684	0.0141666666666667
Global Clustering Coefficent	0.0105263157894737	0.240327305788463	0.00802139037433155
Average Shortest Path	5.06161616161616	1.86060606060606	4.89074344023324
Top 5 Central Nodes(Hightest Degree)	26, 14, 29, 70, 73	29, 33, 50, 69, 68	62, 44, 45, 126, 93
Top 5 Central Nodes (Betweeness)	26, 82, 9, 73, 25	50, 33, 29, 69, 13	119, 62, 45, 126, 93
Common Nodes	26, 73	29, 33, 50, 69	62, 45, 126, 93

i). As from the table, the number of nodes for graph A and C is close to the number of edges. Apart from that, the graph density which is the ratio of the edges and number of possible edges is very low. From these two parameters we can say that Graph A and C are sparse. In the case of graph B, the number of edges is not close to either the number of nodes or $nodes^2$. So, from that we can say that this graph is somewhere in-between.

ii). The diameter and the radius of the graphs is shown in table 2.

iii). The global clustering coefficient is the ratio of the number of the traingles or closed triplets and the number of connected triples of vertices in the graph. On the other hand, the average clustering coefficient is the average of the local clustering coefficient of all the vertices in the graph. The local clustering coefficient is the ratio of the triangles connected to the vertex and the triples centered on the vertex. From their definations, the difference between them is that global clustering coefficient puts more weight on the high degree nodes, while the local clustering coefficient places more weight on the low degree nodes.

iv).The average shortest path length of the graphs is shown in table 2.

v). Degree centrality and Betweenness centrality have been used to find central nodes. Degree centrality is defined as the number of edges incident upon a link. Betweeness centrality is another centrality measure in a graph. It quantifies the number of times a node acts as a bridge along the shortest path between two other nodes.

```
# graphA<-outputA[[2]] graphB<-outputB[[2]]
# graphC<-outputC[[2]] plot(graphA,
# vertex.size=igraph::betweenness(graphA)/100) plot(graphA,
# vertex.size=igraph::degree(graphA)) plot(graphB,
# vertex.size=igraph::betweenness(graphB)/100) plot(graphB,
# vertex.size=igraph::degree(graphB)) plot(graphC,
# vertex.size=igraph::betweenness(graphC)/100) plot(graphC,
# vertex.size=igraph::degree(graphC))
```

The overlap between the central nodes measured by these centrality measures is very high. The results of the overlap between the central nodes for the three graphs is shown in table 2. In graph A, B, and C, there are 2, 4, and 4 common central nodes out of top 5 nodes. These nodes are found to be central because the connections of the those nodes is very large due to which they appears frequently in the shortest path between several nodes. These nodes are also called as hub nodes due to their easier access and strong interaction with other nodes.

Question 6:

In order to check the small-world property of the networks, we can do the following:

i). First we should compute the degree distribution of the graph. It is known that the small-world networks have over abundance of hub nodes.

ii). The next step is to compare the clustering coefficient of the network with a random network of same vertices and edges. The clustering coefficient of small-world networks is high.

$$(Clustering\ Coefficient)_{Network} \gg (Clustering\ Coefficient)_{RandomNetwork}$$

iii). We can also compare the average path lengths of the given graph with that of a random network. The average path length of a small-world network is comparatively less than that of a random network of small vertices and edges.

$$(Average\ Path\ Length)_{Network} \geq (Average\ Path\ Length)_{RandomNetwork}$$

iv). We can also check the distribution of the average shortest path of the network. For small world networks,

$$Average\ shortest\ path \sim log(N)$$

where N is the number of nodes in the network.

```
graphA <- outputA[[2]]
graphB <- outputB[[2]]
graphC <- outputC[[2]]

getRandomNetworkDetails <- function(graph) {
    randomNetwork <- erdos.renyi.game(vcount(graph), ecount(graph),
        type = "gnm")
    output1 <- average.path.length(graph)
    output2 <- average.path.length(randomNetwork)
    output3 <- transitivity(graph, type = "global")
    output4 <- transitivity(randomNetwork, type = "global")
    return(c(output1, output2, output3, output4))
}

graphAProp <- getRandomNetworkDetails(graphA)
graphBProp <- getRandomNetworkDetails(graphB)
graphCProp <- getRandomNetworkDetails(graphC)

details <- c("Average Path Length-Graph", "Average Path Length-Random Graph",
    "Clustering Coefficient-Graph", "Clustering Coefficient-Random Graph")

table <- data.frame(Features = details, Graph_A = graphAProp,
    Graph_B = graphBProp, Graph_C = graphCProp)
table %>% knitr::kable(caption = "Small-World Property Test")
```

Table 3: Small-World Property Test

Features	Graph_A	Graph_B	Graph_C
Average Path Length-Graph	5.0616162	1.8606061	4.8907434
Average Path Length-Random Graph	4.4067189	1.8761616	5.8992662
Clustering Coefficient-Graph	0.0105263	0.2403273	0.0080214
Clustering Coefficient-Random Graph	0.0000000	0.1670621	0.0123967

From the table 3, it can be seen that none of the graphs satisfies the small-world network properties. None of them have both their clustering coefficient and average path length greater than the random network.

Question 7:

In order to check that whether a given network is scale-free or not:

i). First plot the degree distribution on a log-log scale (Probability of degree (P(k)) V/S degree (k)). After that, try to fit the power law distribution to the degree distribution. For scale-free networks, this plot will follow the power-law distribution which is a linear decreasing distribution.

$$P(k) \sim k^{-\alpha}$$

In R, we can do this by using function *power.law.fit* in igraph and if the p-value of this test is less than 0.05, the network is not scale free (Rejection of null hypothesis that this data have been drawn from the fitted power-law distribution).

ii). Check the number of hubs (high degree nodes). Scale-free networks have a small number of hubs.

iii). After that, plot clustering coefficient (C(k)) vs degree (k) on a log-log scale. For scale-free networks the clustering coefficient decreases as the node degree increases.

iv). Take out a small part of the network and check its properties including degree distribution and Clustering coefficient distribution. If it is a scale-free network the subnetwork will have statistically similar to the overall network.

```
graphA <- outputA[[2]]
graphB <- outputB[[2]]
graphC <- outputC[[2]]

# plot(degree.distribution(graphA), log='xy',type='o',xlab =
# 'log(k)',ylab = 'log(P(k))',main = 'Degree Distribution of
# Graph A') plot(degree.distribution(graphB),
# log='xy',type='o',xlab = 'log(k)',ylab = 'log(P(k))',main =
# 'Degree Distribution of Graph B')
# plot(degree.distribution(graphC), log='xy',type='o',xlab =
# 'log(k)',ylab = 'log(P(k))',main = 'Degree Distribution of
# Graph C')

fit1 <- power.law.fit(degree(graphA), 9, implementation = "plfit")
fit2 <- power.law.fit(degree(graphB), 9, implementation = "plfit")
fit3 <- power.law.fit(degree(graphC), 9, implementation = "plfit")

details <- c("Graph A", "Graph B", "Graph C")

table <- data.frame(Graph = details, Alpha = c(fit1$alpha, fit2$alpha,
    fit3$alpha), X_min = c(fit1$xmin, fit2$xmin, fit3$xmin),
    P_Value = c(fit1$KS.p, fit2$KS.p, fit3$KS.p))
table %>% knitr::kable(caption = "Power Law Fit Details")
```

Table 4: Power Law Fit Details

Graph	Alpha	X_min	P_Value
Graph A	3.000000	9	1.0000000
Graph B	2.799903	9	0.0170086
Graph C	3.070681	9	0.0536259

From table 4, it can be seen that Graph A and Graph C has P-Values greater than 0.05 which means that we are not able to reject the null hypothesis that the original data could have been drawn from the fitted power-law distribution. Although, the hubs in the graphs are not relevant as the α is greater than 3 in both the cases.

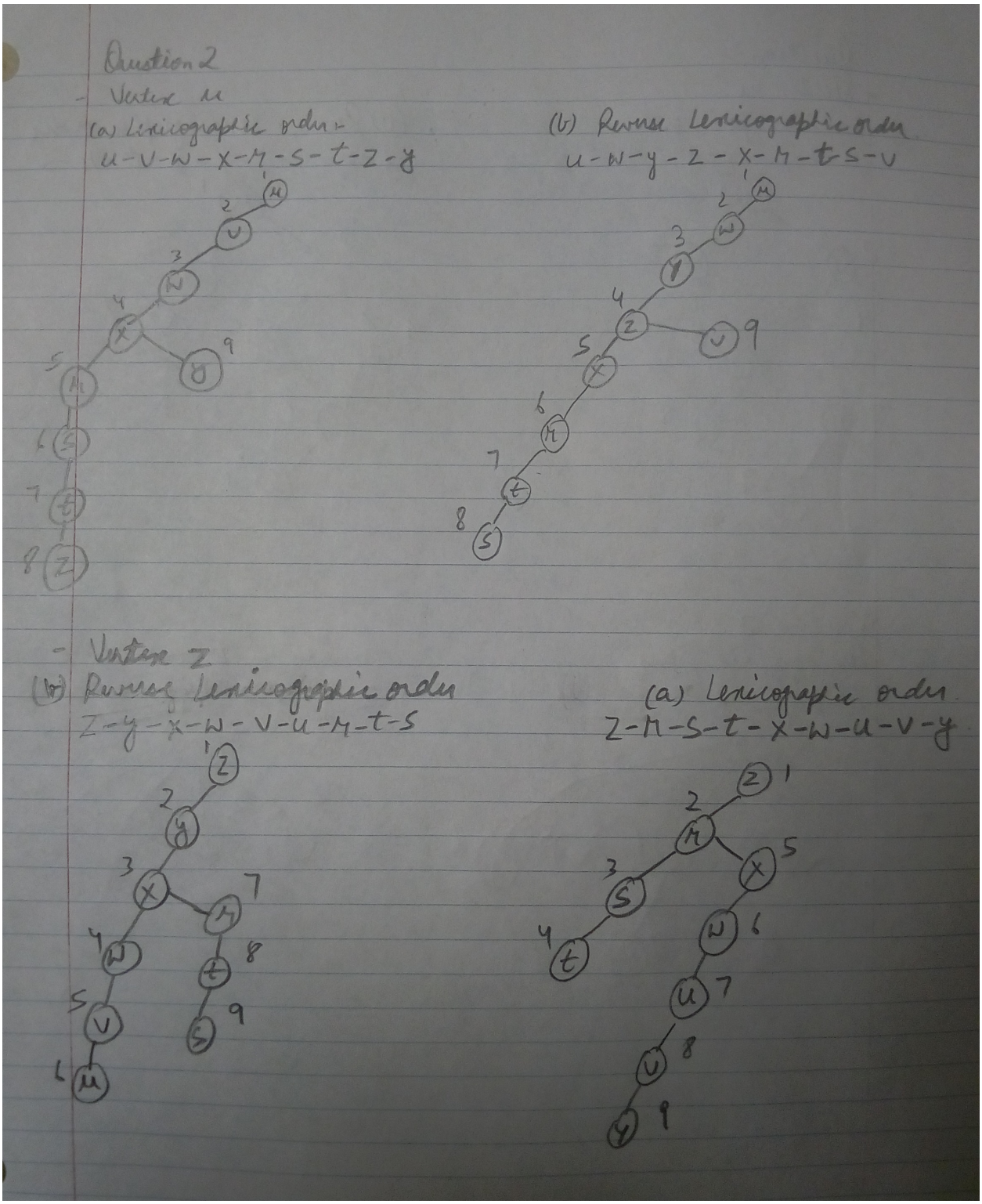


Figure 1: Depth First Search

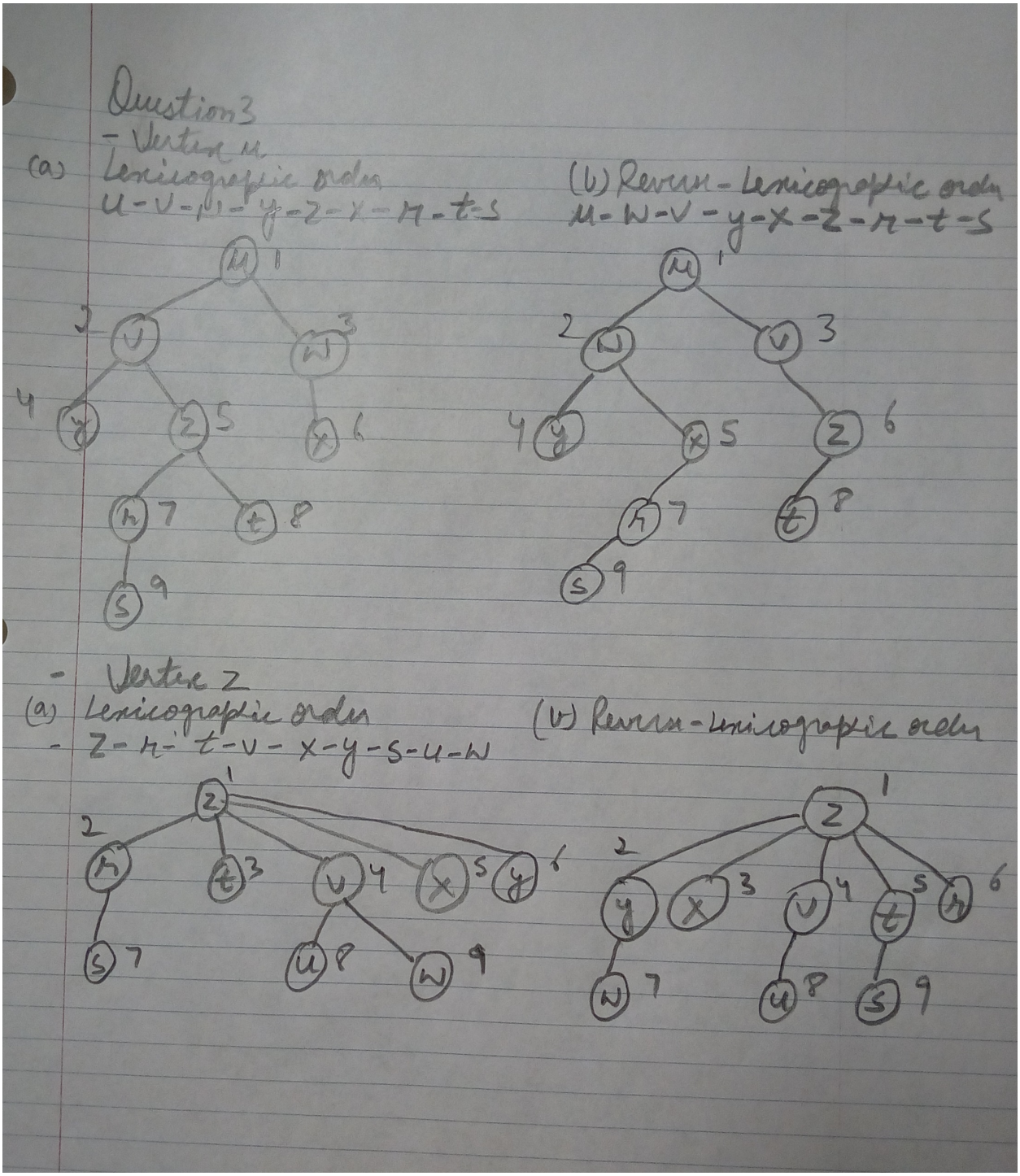


Figure 2: Breath First Search