# DBA5104 igraph Tutorial in R

This markdown is a short tutorial on how to use igraph in R. When using igraph, R is preferred over python as R is more stable and has more comprehensive functions.

You can check out the documentation: https://igraph.org/r/doc/

## Initialization

```
#install.packages("igraph")

library(igraph)
```

```
##
## Attaching package: 'igraph'

## The following objects are masked from 'package:stats':
##
##     decompose, spectrum

## The following object is masked from 'package:base':
##
##     union
```

```
library(igraphdata)
library(sand)
```

```
## Warning: package 'sand' was built under R version 4.3.2

##
## Statistical Analysis of Network Data with R, 2nd Edition
## Type in C2 (+ENTER) to start with Chapter 2.
```

### Creating Network Graphs

```
# For small, toy graphs, the function graph_from_literal can be used,
# specifying the edges in a symbolically literal manner

g <- graph_from_literal(1-2, 1-3, 2-3, 2-4, 3-5, 4-5, 4-6, 4-7, 5-6, 6-7)
```

Number of vertices

```
V(g)
```

```
## + 7/7 vertices, named, from ede8fcc:
## [1] 1 2 3 4 5 6 7
```

Number of edges

```
E(g)
```

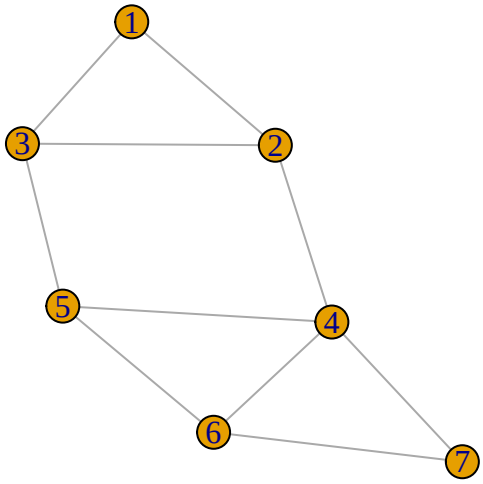
```
## + 10/10 edges from ede8fcc (vertex names):
##  [1] 1--2 1--3 2--3 2--4 3--5 4--5 4--6 4--7 5--6 6--7
```

```
#  same information, combined and in a slightly different format, is recovered
print_all(g)
```

```
## IGRAPH ede8fcc UN-- 7 10 --
## + attr: name (v/c)
## + edges (vertex names):
## 1 -- 2, 3
## 2 -- 1, 3, 4
## 3 -- 1, 2, 5
## 4 -- 2, 5, 6, 7
## 5 -- 3, 4, 6
## 6 -- 4, 5, 7
## 7 -- 4, 6
```

Visual representation

```
plot(g)
```




Adjacency matrix

```
as_adjacency_matrix(g)
```

```
## 7 x 7 sparse Matrix of class "dgCMatrix"
##   1 2 3 4 5 6 7
## 1 . 1 1 . . . .
## 2 1 . 1 1 . . .
## 3 1 1 . . 1 . .
## 4 . 1 . . 1 1 1
## 5 . . 1 1 . 1 .
## 6 . . . 1 1 . 1
## 7 . . . 1 . 1 .
```

### Graph Concepts

Check for weighted graph

```
is_weighted(g)
```

```
## [1] FALSE
```

Check for simple graph (i.e. no self-loop or multi-edges)

```r
is_simple(g)
```

```
## [1] TRUE
```

Check for neighbour of a certain node

```r
# Neighbours of node 5 in the graph g

neighbors(g,5)
```

```
## + 3/7 vertices, named, from ede8fcc:
## [1] 3 4 6
```

Check for the degrees of the nodes in the graph

```r
degree(g)
```

```
## 1 2 3 4 5 6 7
## 2 3 3 4 3 3 2
```

Check if the graph is connected

```r
is_connected(g)
```

```
## [1] TRUE
```

Check for clusters in graph (if the graph is connected - 1 component)

```r
clusters(g)
```

```
## $membership
## 1 2 3 4 5 6 7
## 1 1 1 1 1 1 1
##
## $csize
## [1] 7
##
## $no
## [1] 1
```

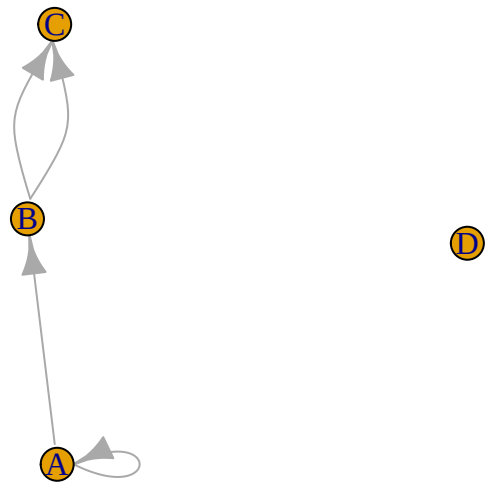
Check for diameter (Longest shortest path)

```r
diameter(g, weights=NA)
```

```
## [1] 3
```

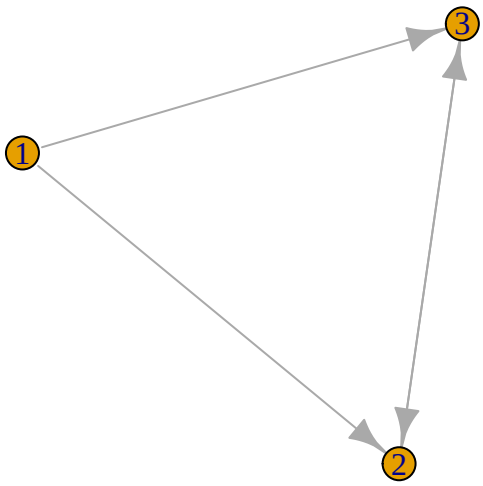
Directed graph in graph with pairs

```
g2 <- graph( c("A", "B", "B", "C", "B", "C", "A", "A"),
isolates=c("D") )
plot(g2)
```




Directed edges in graph_from_literal are indicated using a minus/plus convention

```
dg <- graph_from_literal(1-+2, 1-+3, 2++3)
plot(dg)
```




**Descriptive Analysis of Network Graph Characteristics**

Initialize toy dataset

```
head(elist.lazega)
```

```
##   V1  V2
## 1 V1 V17
## 2 V2  V7
## 3 V2 V16
## 4 V2 V17
## 5 V2 V22
## 6 V2 V26
```

```
head(v.attr.lazega)
```

```
##   Name Seniority Status Gender Office Years Age Practice School
## 1   V1         1      1      1      1    31  64        1      1
## 2   V2         2      1      1      1    32  62        2      1
## 3   V3         3      1      1      2    13  67        1      1
## 4   V4         4      1      1      1    31  59        2      3
## 5   V5         5      1      1      2    31  59        1      2
## 6   V6         6      1      1      2    29  55        1      1
```

```
g.lazega=graph_from_data_frame(elist.lazega,directed="FALSE",vertices = v.attr.lazega)
```

Betweenness centrality of every node in the graph

```
betweenness(g.lazega)
```

```
##          V1          V2          V3          V4          V5          V6
##   0.0000000  18.4428571   0.6944444  21.3070430   3.5448773   2.0849068
##          V7          V8          V9         V10         V11         V12
##   1.8547619   0.0000000   0.0000000   8.0973957   0.0000000   8.4302572
##         V13         V14         V15         V16         V17         V18
##   0.0000000  13.2791681  22.0448052  70.6494464 101.8843060  21.2662088
##         V19         V20         V21         V22         V23         V24
##  14.0866744   0.3583333   0.0000000  18.4319805   0.0000000  28.6060107
##         V25         V26         V27         V28         V29         V30
##   6.9124542  48.7275072  32.0000000  54.4436758  13.2436480   1.0075758
##         V31         V32         V33         V34         V35         V36
##  62.6624181  52.1727578   3.9833333   2.4651515   8.9770924   0.3409091
```

Eigenvector centrality of every node in the graph

```
eigen_centrality(g.lazega)$vector
```

```
##           V1           V2           V3           V4           V5           V6
## 1.120344e-01 4.780219e-01 1.754210e-01 7.565355e-01 4.163271e-01 3.656392e-01
##           V7           V8           V9          V10          V11          V12
## 1.021647e-01 1.161440e-18 2.538825e-01 3.825620e-01 1.120344e-01 7.376819e-01
##          V13          V14          V15          V16          V17          V18
## 1.141026e-01 4.592049e-01 8.243409e-01 8.772495e-01 1.000000e+00 4.338823e-01
##          V19          V20          V21          V22          V23          V24
## 8.325400e-01 3.604173e-01 2.242129e-02 7.495043e-01 1.161440e-18 6.392531e-01
##          V25          V26          V27          V28          V29          V30
## 3.327115e-01 8.866422e-01 2.001285e-01 7.991835e-01 6.511804e-01 2.643546e-01
##          V31          V32          V33          V34          V35          V36
## 7.355564e-01 8.252849e-01 2.829041e-01 5.020502e-01 5.359166e-01 2.622551e-01
```

Transitivity (i.e. clustering coefficient) of the graph

```
transitivity(g.lazega)
```

```
## [1] 0.3887689
```

### Star Wars dataset

Obtained from https://github.com/pablobarbera/data-science-workshop/blob/master/sna/data/

Description of dataset

```
edges <- read.csv("C:/Users/Eugene/Downloads/star-wars-network-edges.csv")
nodes <- read.csv("C:/Users/Eugene/Downloads/star-wars-network-nodes.csv")
head(edges)
```

```
##      source target weight
## 1     C-3PO  R2-D2     17
## 2      LUKE  R2-D2     13
## 3   OBI-WAN  R2-D2      6
## 4      LEIA  R2-D2      5
## 5       HAN  R2-D2      5
## 6 CHEWBACCA  R2-D2      3
```

```
head(nodes)
```

```
##          name id
## 1       R2-D2  0
## 2   CHEWBACCA  1
## 3       C-3PO  2
## 4        LUKE  3
## 5 DARTH VADER  4
## 6       CAMIE  5
```

Create an igraph object from the above raw data.

```
g <- graph_from_data_frame(d=edges, vertices=nodes, directed=FALSE)
#d describes the edges of the network
g
```

```
## IGRAPH efa65e7 UNW- 22 60 --
## + attr: name (v/c), id (v/n), weight (e/n)
## + edges from efa65e7 (vertex names):
##  [1] R2-D2      --C-3PO       R2-D2      --LUKE        R2-D2      --OBI-WAN
##  [4] R2-D2      --LEIA        R2-D2      --HAN         R2-D2      --CHEWBACCA
##  [7] R2-D2      --DODONNA     CHEWBACCA  --OBI-WAN     CHEWBACCA  --C-3PO
## [10] CHEWBACCA  --LUKE        CHEWBACCA  --HAN         CHEWBACCA  --LEIA
## [13] CHEWBACCA  --DARTH VADER CHEWBACCA  --DODONNA     LUKE       --CAMIE
## [16] CAMIE      --BIGGS       LUKE       --BIGGS       DARTH VADER--LEIA
## [19] LUKE       --BERU        BERU       --OWEN        C-3PO      --BERU
## [22] LUKE       --OWEN        C-3PO      --LUKE        C-3PO      --OWEN
## + ... omitted several edges
```

```
V(g)
```

```
## + 22/22 vertices, named, from efa65e7:
##  [1] R2-D2       CHEWBACCA   C-3PO       LUKE        DARTH VADER CAMIE
##  [7] BIGGS       LEIA        BERU        OWEN        OBI-WAN     MOTTI
## [13] TARKIN      HAN         GREEDO      JABBA       DODONNA     GOLD LEADER
## [19] WEDGE       RED LEADER  RED TEN     GOLD FIVE
```

```
E(g)
```

```
## + 60/60 edges from efa65e7 (vertex names):
##  [1] R2-D2      --C-3PO       R2-D2      --LUKE        R2-D2      --OBI-WAN
##  [4] R2-D2      --LEIA        R2-D2      --HAN         R2-D2      --CHEWBACCA
##  [7] R2-D2      --DODONNA     CHEWBACCA  --OBI-WAN     CHEWBACCA  --C-3PO
```

```
## [10] CHEWBACCA  --LUKE        CHEWBACCA  --HAN         CHEWBACCA  --LEIA
## [13] CHEWBACCA  --DARTH VADER CHEWBACCA  --DODONNA     LUKE       --CAMIE
## [16] CAMIE      --BIGGS       LUKE       --BIGGS       DARTH VADER--LEIA
## [19] LUKE       --BERU        BERU       --OWEN        C-3PO      --BERU
## [22] LUKE       --OWEN        C-3PO      --LUKE        C-3PO      --OWEN
## [25] C-3PO      --LEIA        LUKE       --LEIA        LEIA       --BERU
## [28] LUKE       --OBI-WAN     C-3PO      --OBI-WAN     LEIA       --OBI-WAN
## + ... omitted several edges
```

Character names

```
V(g)$name
```

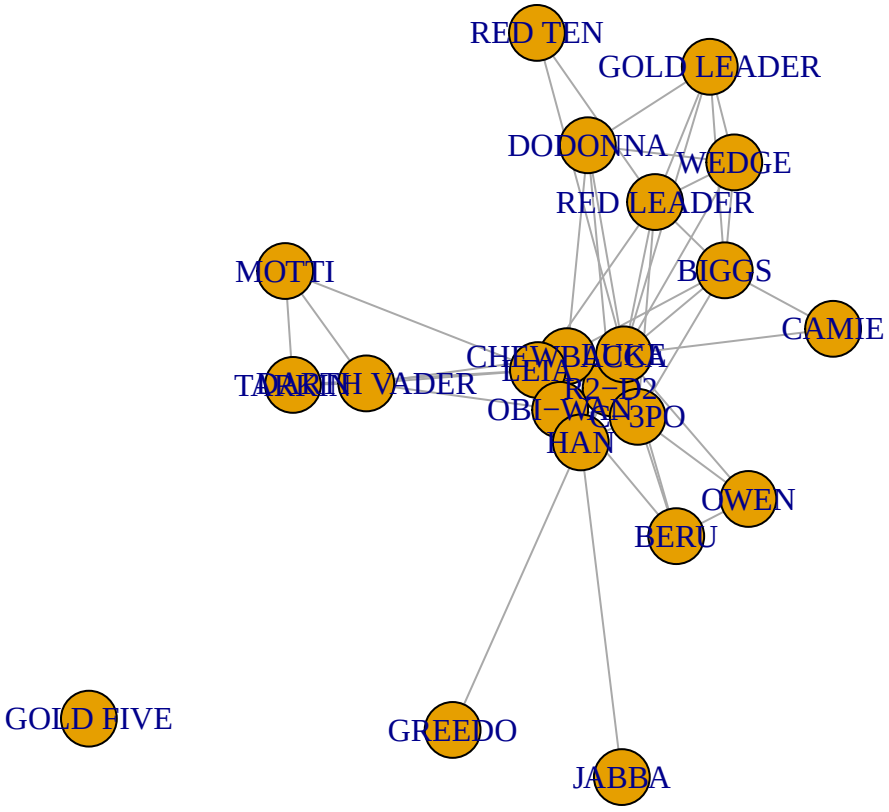
```
##  [1] "R2-D2"       "CHEWBACCA"   "C-3PO"       "LUKE"        "DARTH VADER"
##  [6] "CAMIE"       "BIGGS"       "LEIA"        "BERU"        "OWEN"
## [11] "OBI-WAN"     "MOTTI"       "TARKIN"      "HAN"         "GREEDO"
## [16] "JABBA"       "DODONNA"     "GOLD LEADER" "WEDGE"       "RED LEADER"
## [21] "RED TEN"     "GOLD FIVE"
```

Edge weights

```
E(g)$weight
```

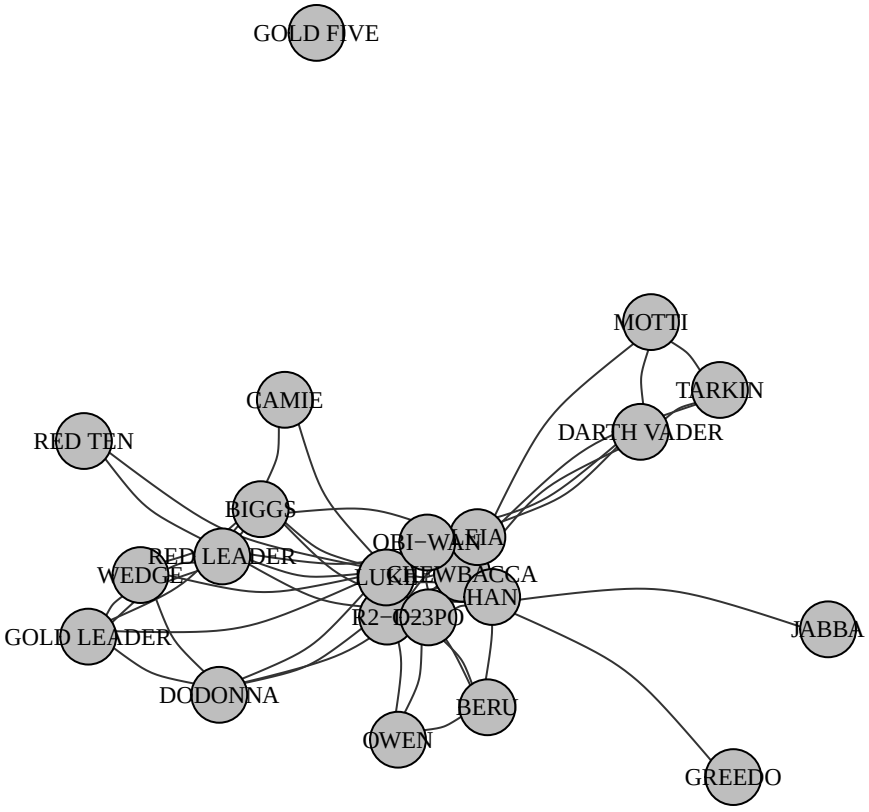
```
##  [1] 17 13  6  5  5  3  1  7  5 16 19 11  1  1  2  2  4  1  3  3  2  3 18  2  6
## [26] 17  1 19  6  1  2  1  7  9 26  1  1  6  1  1 13  1  1  1  1  1  1  2  1  1
## [51]  3  3  1  1  3  1  2  1  1  1
```

```
par(mar=c(0,0,0,0))
plot(g)
```




Visualise it better? To see all the available plotting options, you can check ?igraph.plotting.

```
par(mar=c(0,0,0,0))
plot(g,
vertex.color = "grey", # change color of nodes
vertex.label.color = "black", # change color of labels
vertex.label.cex = .75, # change size of labels to 75% of original size
edge.curved=.25, # add a 25% curve to the edges
edge.color="grey20") # change edge color to grey
```



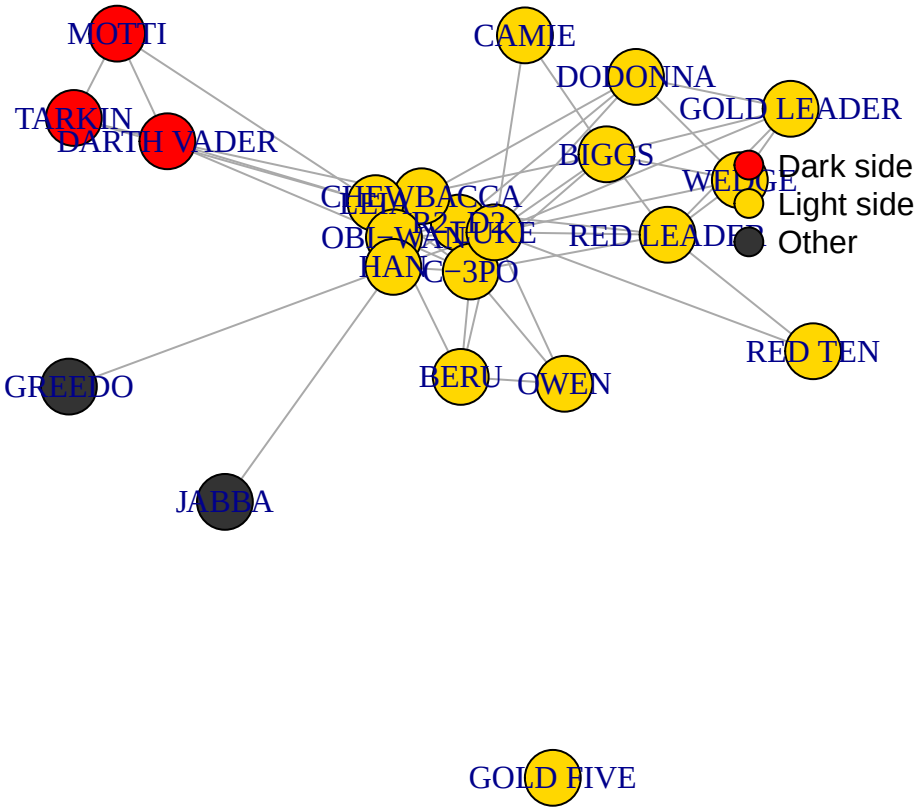

Change based on which side they belong to:

```
# create vectors with characters in each side
dark_side <- c("DARTH VADER", "MOTTI", "TARKIN")
light_side <- c("R2-D2", "CHEWBACCA", "C-3PO", "LUKE", "CAMIE", "BIGGS",
"LEIA", "BERU", "OWEN", "OBI-WAN", "HAN", "DODONNA",
"GOLD LEADER", "WEDGE", "RED LEADER", "RED TEN", "GOLD FIVE")
other <- c("GREEDO", "JABBA")
# node we'll create a new color variable as a node property
V(g)$color <- NA
V(g)$color[V(g)$name %in% dark_side] <- "red"
V(g)$color[V(g)$name %in% light_side] <- "gold"
V(g)$color[V(g)$name %in% other] <- "grey20"
vertex_attr(g)
```

```
## $name
##  [1] "R2-D2"       "CHEWBACCA"   "C-3PO"       "LUKE"        "DARTH VADER"
##  [6] "CAMIE"       "BIGGS"       "LEIA"        "BERU"        "OWEN"
## [11] "OBI-WAN"     "MOTTI"       "TARKIN"      "HAN"         "GREEDO"
## [16] "JABBA"       "DODONNA"     "GOLD LEADER" "WEDGE"       "RED LEADER"
## [21] "RED TEN"     "GOLD FIVE"
##
## $id
##  [1]  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21
##
## $color
```

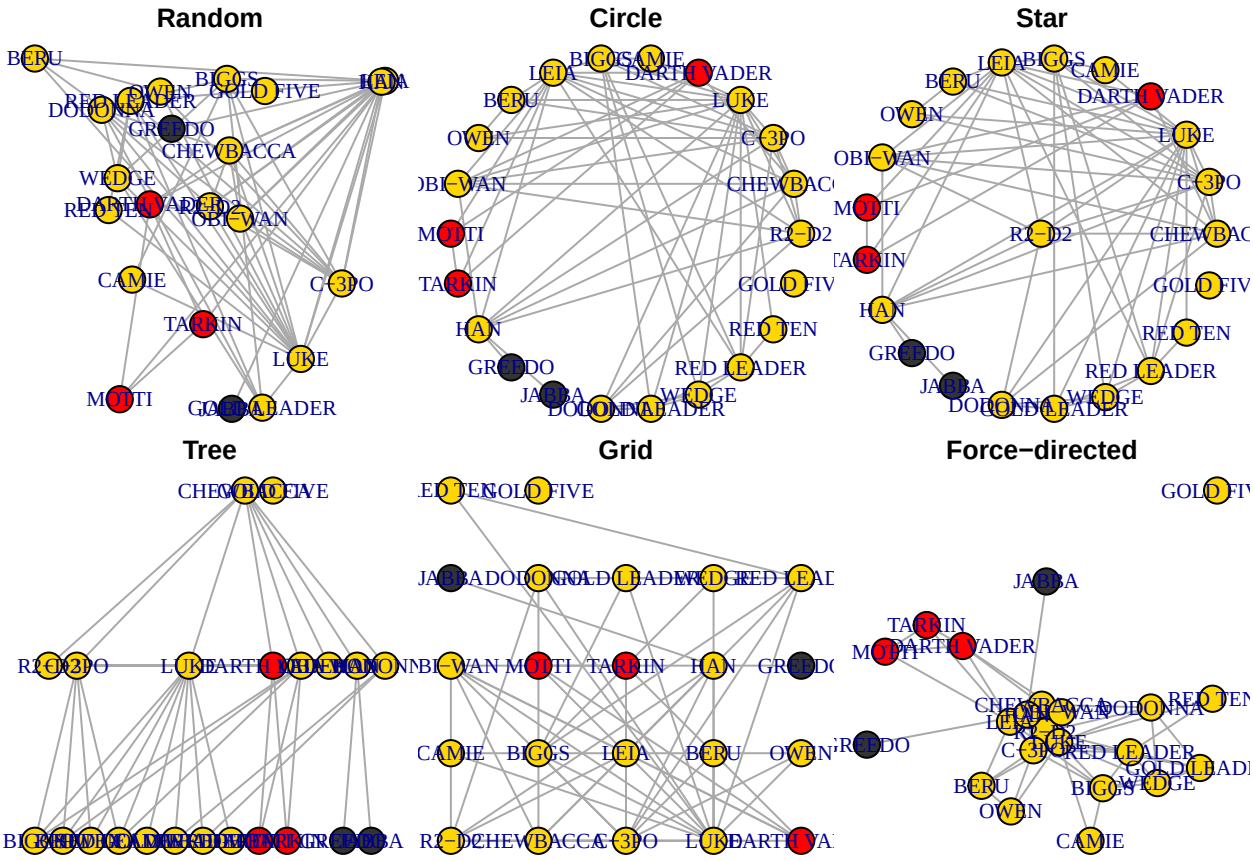
```
##  [1] "gold"   "gold"   "gold"   "gold"   "red"    "gold"   "gold"   "gold"
##  [9] "gold"   "gold"   "gold"   "red"    "red"    "gold"   "grey20" "grey20"
## [17] "gold"   "gold"   "gold"   "gold"   "gold"   "gold"
```

```
par(mar=c(0,0,0,0)); plot(g)
legend(x=.75, y=.75, legend=c("Dark side", "Light side", "Other"),
pch=21, pt.bg=c("red", "gold", "grey20"), pt.cex=2, bty="n")
```




Up to now, each time we run the plot function, the nodes appear to be in a different location. Why? Because it's running a probabilistic function trying to locate them in the optimal way possible. However, we can also specify the layout for the plot; that is, the (x,y) coordinates where each node will be placed. igraph has a few different layouts built-in, that will use different algorithms to find an optimal distribution of nodes. The following code illustrates some of these:

```
par(mfrow=c(2, 3), mar=c(0,0,1,0))
plot(g, layout=layout_randomly, main="Random")
plot(g, layout=layout_in_circle, main="Circle")
plot(g, layout=layout_as_star, main="Star")
plot(g, layout=layout_as_tree, main="Tree")
plot(g, layout=layout_on_grid, main="Grid")
plot(g, layout=layout_with_fr, main="Force-directed")
```




## References

Statistical Analysis of Network Data with R - Eric D. Kolaczyk, Gábor Csárdi

**More help:**

Use command: ?igraph or visit https://cran.r-project.org/web/packages/igraph/igraph.pdf