igraph Tutorial

Suparna

01/04/2020

A tutorial to learn the basics of “igraph” package in R

install.packages(“igraph”) install.packages(“igraphdata”)

```
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")
```

Directed Graph

Create a directed graph with nodes that connect successive pairs. N1->N2, N2->N3, N3->N4, N4->N1, N1->5, N2->N5 with 2 isolates N6, N7

```
gr1_dir = graph(c("N1","N2","N2","N3","N3","N4","N4","N1","N1","N
5","N2","N5"), isolates=c("N6","N7"))
```

Add attributes to vertices by calling V() function. Nodes 1 to 5 are proteins, 7,8 are small molecules

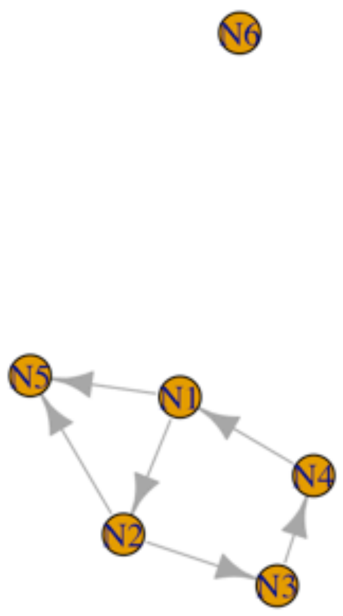
```
V(gr1_dir)$type <- c(rep("proteins",5),rep("small molecues",2))
```

Add edge weights, if you want a **weighted graph**, using E() function

```
E(gr1_dir)$weight <- c(1.5, 2.0, 4.0, 2.5, 1.9, 1.0, 1.0)
```

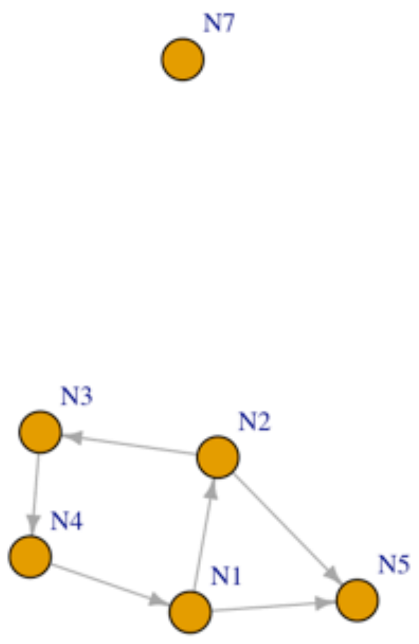
```
## Warning in eattrs[[name]][index] <- value: number of items to
replace is not a
## multiple of replacement length
```

```
plot(gr1_dir)
```

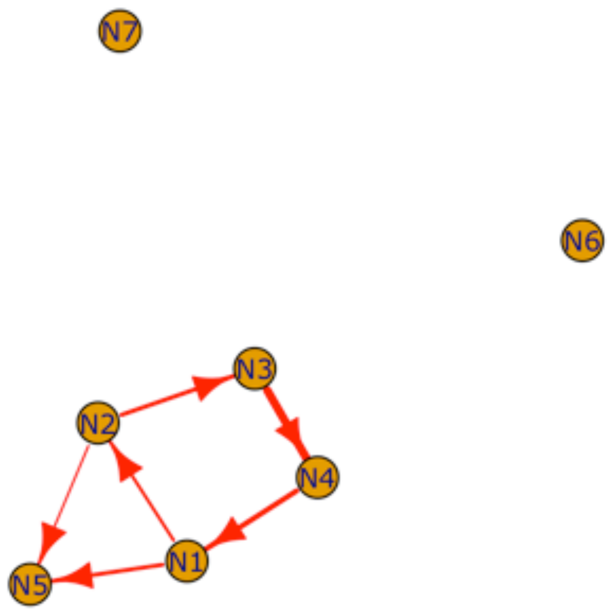


```
# We can also tamper with plot parameters

plot(gr1_dir, vertex.label.cex=0.8,vertex.label.dist=2.6, edge.ar
row.size=0.5)
```



```
plot (gr1_dir,
vertex.label.cex = 0.8,
vertex.label.family="Verdana",
edge.width=E(gr1_dir)$weight*1.0,
edge.color="red")
```



Undirected Graph

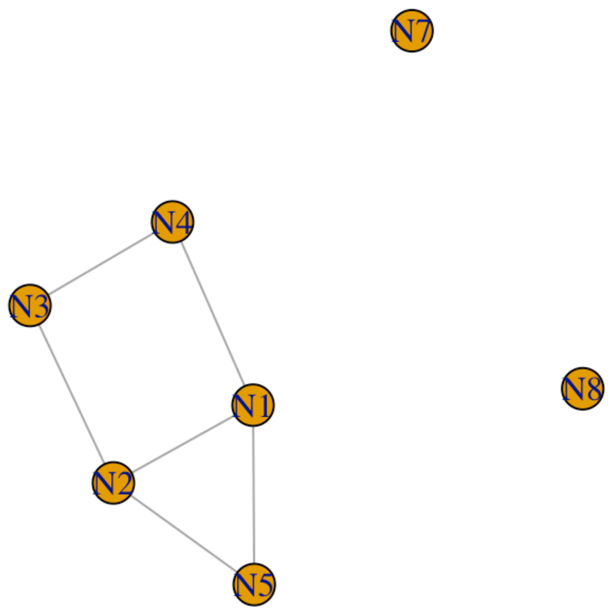
Create an undirected graph for same data.

```
gr1_undir = graph(c("N1","N2","N2","N3","N3","N4","N4","N1","N1",
"N5","N2","N5"), isolates=c("N7","N8"), directed=F)
graph(c("N1","N2","N2","N3","N3","N4","N4","N1","N1","N5","N2","N
5"), isolates=c("N7","N8"))
```

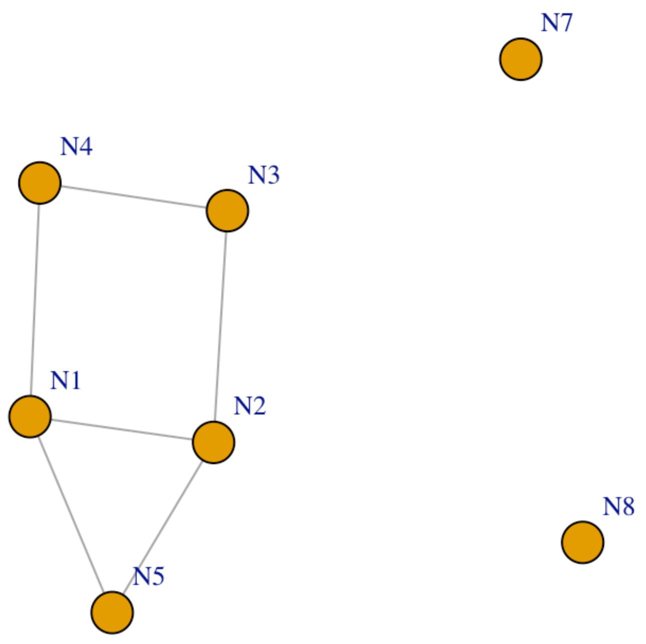
```
## IGRAPH f2a972e DN-- 7 6 --
## + attr: name (v/c)
## + edges from f2a972e (vertex names):
## [1] N1->N2 N2->N3 N3->N4 N4->N1 N1->N5 N2->N5
```

```
V(gr1_dir)$type <- c(rep("proteins",5),rep("small molecues",2))

# Plot the graph
plot(gr1_undir)
```



```
plot(gr1_undir, vertex.label.cex=0.8,vertex.label.dist=2.6, edge.
arrow.size=0.5)
```



```
# Print the igraph objects gr1_dir and gr1_undir
print(gr1_dir)
```

```
## IGRAPH 6e168ca DNWB 7 6 --
## + attr: name (v/c), type (v/c), weight (e/n)
## + edges from 6e168ca (vertex names):
## [1] N1->N2 N2->N3 N3->N4 N4->N1 N1->N5 N2->N5
```

```
print(gr1_undir)
```

```
## IGRAPH aa2f04e UN-- 7 6 --
## + attr: name (v/c)
## + edges from aa2f04e (vertex names):
## [1] N1--N2 N2--N3 N3--N4 N1--N4 N1--N5 N2--N5
```

NOTE : In the above printout, “U”->undirected, D–>“Directed”, “N”–>named graph (edges named), “W”-> weighted graph“,”B"-> bipartite, where nodes have names.

Above graph is “UN”, Undirected and Named. The two numbers 7,6 refer to 7 nodes, 6 edges

```
# Print the vertices (nodes) and edges
print(V(gr1_undir))
```

```
## + 7/7 vertices, named, from aa2f04e:
## [1] N1 N2 N3 N4 N5 N7 N8
```

```
print(E(gr1_undir))
```

```
## + 6/6 edges from aa2f04e (vertex names):
## [1] N1--N2 N2--N3 N3--N4 N1--N4 N1--N5 N2--N5
```

```
print(V(gr1_dir))
```

```
## + 7/7 vertices, named, from 6e168ca:
## [1] N1 N2 N3 N4 N5 N6 N7
```

```
print(E(gr1_dir))
```

```
## + 6/6 edges from 6e168ca (vertex names):
## [1] N1->N2 N2->N3 N3->N4 N4->N1 N1->N5 N2->N5
```

```
# Print the sparse network matrix
print(gr1_dir[])
```

```
## 7 x 7 sparse Matrix of class "dgCMatrix"
##     N1  N2 N3 N4  N5 N6 N7
## N1 .   1.5  .  . 1.9  .  .
## N2 .   .    2  . 1.0  .  .
## N3 .   .    .  4 .    .  .
## N4 2.5 .    .  . .    .  .
## N5 .   .    .  . .    .  .
## N6 .   .    .  . .    .  .
## N7 .   .    .  . .    .  .
```

```
print(gr1_undir[])
```

```
## 7 x 7 sparse Matrix of class "dgCMatrix"
##    N1 N2 N3 N4 N5 N7 N8
## N1  .  1  .  1  1  .  .
## N2  1  .  1  .  1  .  .
## N3  .  1  .  1  .  .  .
## N4  1  .  1  .  .  .  .
## N5  1  1  .  .  .  .  .
## N7  .  .  .  .  .  .  .
## N8  .  .  .  .  .  .  .
```

We can also access rows and columns of this matrix, as well as the attributes

```
# Rows, columns
print(gr1_dir[1,])
```

```
##  N1  N2  N3  N4  N5  N6  N7
## 0.0 1.5 0.0 0.0 1.9 0.0 0.0
```

```
# Attributes
print(  V(gr1_dir)$name )
```

```
## [1] "N1" "N2" "N3" "N4" "N5" "N6" "N7"
```

```
print(  V(gr1_dir)$type )
```

```
## [1] "proteins"       "proteins"       "proteins"       "protei
ns"
## [5] "proteins"       "small molecues" "small molecues"
```

```
print( E(gr1_dir)$weight)
```

```
## [1] 1.5 2.0 4.0 2.5 1.9 1.0
```

Adding new vertices (nodes) and edges to the graph:

```
#Add 2 new nodes "N8", "N9" to gr1_undir
gr1_dir <- gr1_dir + vertices(c("N8","N9"), type=c("gene","gene"
) )
print(V(gr1_dir))
```

```
## + 9/9 vertices, named, from 1406783:
## [1] N1 N2 N3 N4 N5 N6 N7 N8 N9
```

```
print(vertex_attr(gr1_dir))
```

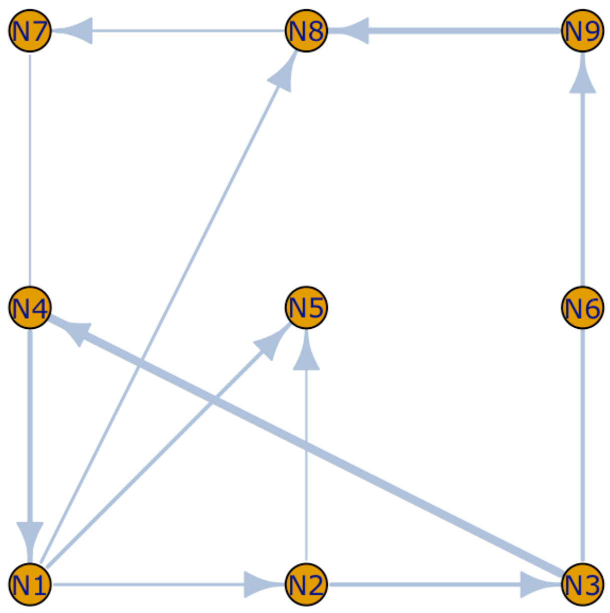
```
## $name
## [1] "N1" "N2" "N3" "N4" "N5" "N6" "N7" "N8" "N9"
##
## $type
## [1] "proteins"       "proteins"       "proteins"       "protei
ns"
## [5] "proteins"       "small molecues" "small molecues" "gene"
## [9] "gene"
```

```
#Adding new edges that covers newly added vertices
gr1_dir <- gr1_dir + edges(c("N1","N8","N3","N9","N9","N8","N7","
N1","N8","N7","N9","N7"),weight=c(1.5,2.0,3.0,1.0,1.0,1.0))
print(E(gr1_dir))
```

```
## + 12/12 edges from 9ce51c9 (vertex names):
##  [1] N1->N2 N2->N3 N3->N4 N4->N1 N1->N5 N2->N5 N1->N8 N3->N9 N
9->N8 N7->N1
## [11] N8->N7 N9->N7
```

```
# Plot the new network

# gr1_dir <- delete_graph_attr(gr1_dir,"gene")
l = layout_on_grid(gr1_dir)
plot (gr1_dir,
vertex.label.cex = 0.8,
vertex.label.family="Verdana",
layout=l,
edge.width=E(gr1_dir)$weight*1.0,
edge.color="lightsteelblue")
```



Computing the properties of the network

Degree Measures

1. Edge density = ratio between edges in the graph to all pssoble edges between nodes

```
print(  edge_density(gr1_dir) )
```

```
## [1] 0.1666667
```

2. Degree of a node

```
# Directed
deg <- degree(gr1_dir)
print(deg)
```

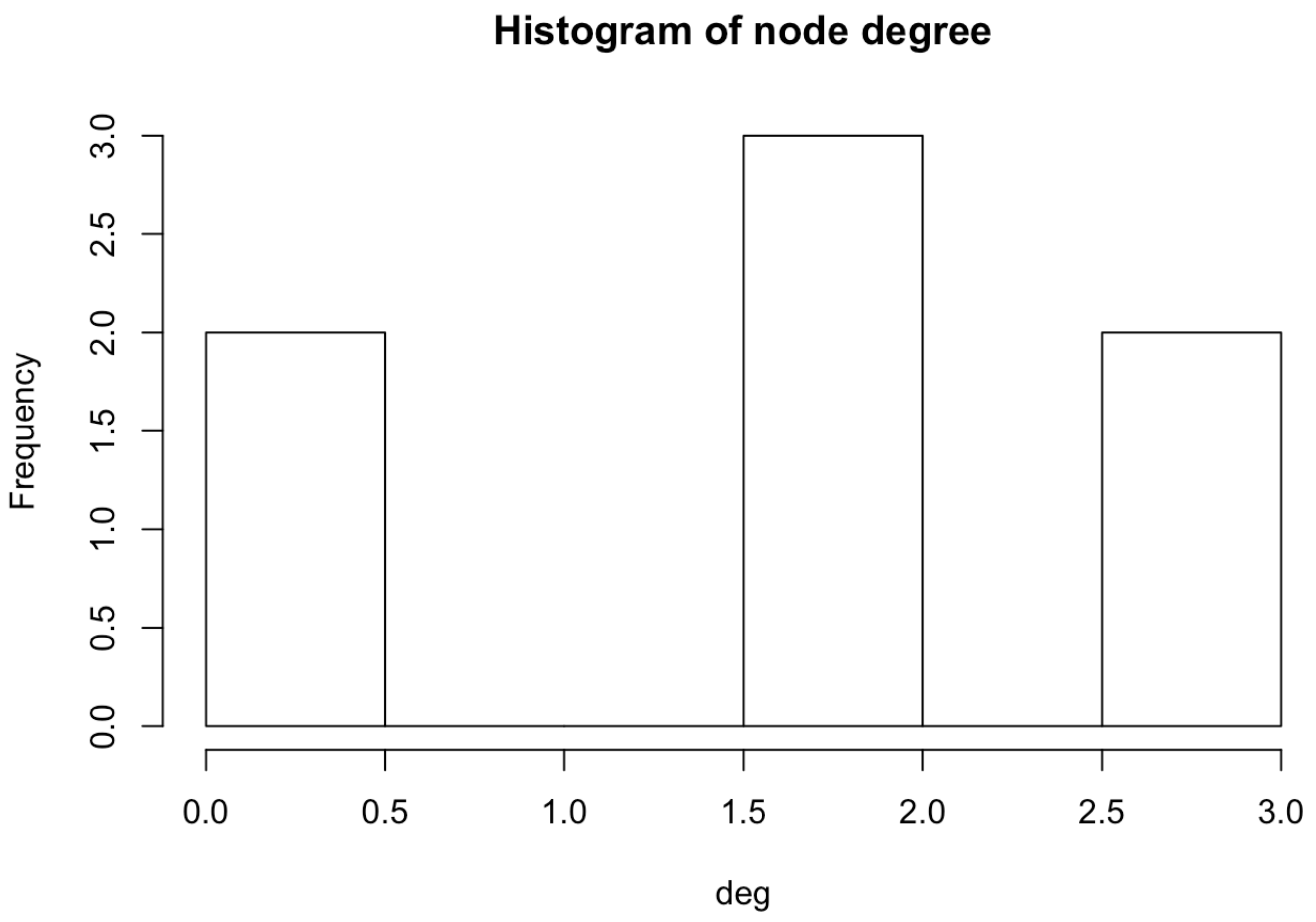
```
## N1 N2 N3 N4 N5 N6 N7 N8 N9
##  5  3  3  2  2  0  3  3  3
```

```
# Undirected
deg <- degree(gr1_undir)
print(deg)
```

```
## N1 N2 N3 N4 N5 N7 N8
##  3  3  2  2  2  0  0
```

3. Histogram of node degree

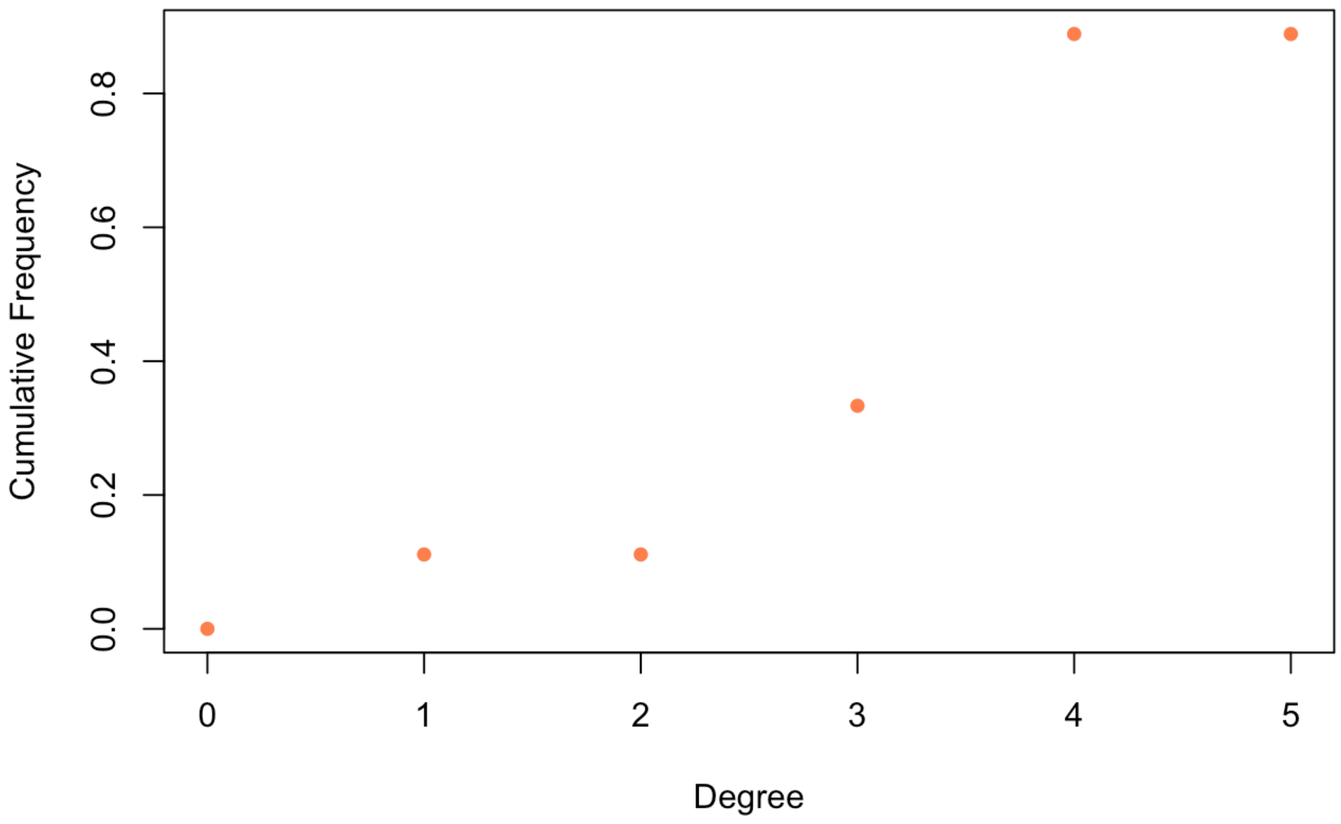
```
hist(deg, breaks=10, main="Histogram of node degree")
```



4. Degree distribution and cumulative degree distribution

```
deg <- degree(gr1_dir)
deg.dist = degree_distribution(gr1_dir,cumulative=T)

plot( x=0:max(deg), y=1-deg.dist, pch=19, cex=0.8, col="coral", x
lab="Degree", ylab="Cumulative Frequency")
```



Centrality Measures

1. Degree and Degree Centrality

```
print (degree(gr1_dir))
```

```
## N1 N2 N3 N4 N5 N6 N7 N8 N9
##  5  3  3  2  2  0  3  3  3
```

```
print(centr_degree(gr1_dir))
```

```
## $res
## [1] 5 3 3 2 2 0 3 3 3
##
## $centralization
## [1] 0.1640625
##
## $theoretical_max
## [1] 128
```

2. Closeness and Closeness centrality (not well defined for disconnected graphs)

```
print( closeness(gr1_dir) )
```

```
## Warning in closeness(gr1_dir): At centrality.c:2617 :closeness
centrality is not
## well-defined for disconnected graphs
```

```
##          N1          N2          N3          N4          N5
N6          N7
## 0.03039514 0.02500000 0.02604167 0.02331002 0.01388889 0.01388
889 0.02673797
##          N8          N9
## 0.02386635 0.02673797
```

```
print( centr_clo(gr1_dir) )
```

```
## Warning in centr_clo(gr1_dir): At centrality.c:2784 :closeness
centrality is not
## well-defined for disconnected graphs
```

```
## $res
## [1] 0.3636364 0.3333333 0.3478261 0.3076923 0.1111111 0.111111
1 0.2962963
## [8] 0.2500000 0.2857143
##
## $centralization
## [1] 0.1217821
##
## $theoretical_max
## [1] 7.111111
```

3. Eigenvector Centrality

```
print( eigen_centrality(gr1_dir)$vector )
```

```
##        N1        N2        N3        N4        N5        N6        
N7        N8
## 0.8009517 0.5642208 1.0000000 0.9615461 0.3324034 0.1346352 0.
3143146 0.5442160
##        N9
## 0.6313596
```

```
print( centr_eigen(gr1_dir)$centralization )
```

```
## [1] 0.5138993
```

4. Betweenness centrality

```
print( betweenness(gr1_dir) )
```

```
## N1 N2 N3 N4 N5 N6 N7 N8 N9
## 23 13 13  0  0  0 15  2  8
```

```
print( centr_betw(gr1_dir)  )
```

```
## $res
## [1] 23 13 13  4  0  0 11  2  4
##
## $centralization
## [1] 0.3058036
##
## $theoretical_max
## [1] 448
```

```
print( edge_betweenness(gr1_dir) )
```

```
##  [1] 19 19  6  7  6  1  5 14  3 22  9 12
```

Distance Measures

Get all the shortest paths in the graph:

```
# Gives a matrix of distances between two nodes or two sets of no
des
print( distances(gr1_dir) )
```

```
##     N1  N2  N3  N4  N5  N6  N7  N8  N9
## N1 0.0 1.5 3.5 2.5 1.9 Inf 1.0 1.5 2.0
## N2 1.5 0.0 2.0 4.0 1.0 Inf 2.5 3.0 3.5
## N3 3.5 2.0 0.0 4.0 3.0 Inf 3.0 4.0 2.0
## N4 2.5 4.0 4.0 0.0 4.4 Inf 3.5 4.0 4.5
## N5 1.9 1.0 3.0 4.4 0.0 Inf 2.9 3.4 3.9
## N6 Inf Inf Inf Inf Inf   0 Inf Inf Inf
## N7 1.0 2.5 3.0 3.5 2.9 Inf 0.0 1.0 1.0
## N8 1.5 3.0 4.0 4.0 3.4 Inf 1.0 0.0 2.0
## N9 2.0 3.5 2.0 4.5 3.9 Inf 1.0 2.0 0.0
```

```
print( distances(gr1_undir) )
```

```
##     N1  N2  N3  N4  N5  N7  N8
## N1   0   1   2   1   1 Inf Inf
## N2   1   0   1   2   1 Inf Inf
## N3   2   1   0   1   2 Inf Inf
## N4   1   2   1   0   2 Inf Inf
## N5   1   1   2   2   0 Inf Inf
## N7 Inf Inf Inf Inf Inf   0 Inf
## N8 Inf Inf Inf Inf Inf Inf   0
```

Get the average shortest path

```
print(mean_distance(gr1_dir) )
```

```
## [1] 2.428571
```

```
print(mean_distance(gr1_undir) )
```

```
## [1] 1.4
```

Examine the output of **shortest_path**

```
# Single target node
selectedPaths1 <- shortest_paths(gr1_dir,from=V(gr1_dir)[name=="N
1"],to = V(gr1_dir)[name=="N4"], output="both")
print(selectedPaths1)
```

```
## $vpath
## $vpath[[1]]
## + 4/9 vertices, named, from 9ce51c9:
## [1] N1 N2 N3 N4
##
##
## $epath
## $epath[[1]]
## + 3/12 edges from 9ce51c9 (vertex names):
## [1] N1->N2 N2->N3 N3->N4
##
##
## $predecessors
## NULL
##
## $inbound_edges
## NULL
```

```
# Multiple target nodes
selectedPaths2 <- shortest_paths(gr1_dir, from=V(gr1_dir)[name=="
N1"], to = V(gr1_dir)[type=="genes"], output="both")
print(selectedPaths2)
```

```
## $vpath
## list()
##
## $epath
## list()
##
## $predecessors
## NULL
##
## $inbound_edges
## NULL
```

All shortest paths with weight

```
all_paths1 <- all_shortest_paths(gr1_dir,V(gr1_dir)[name=="N1"],t
o=V(gr1_dir)[name=="N5"])
print(all_paths1)
```

```
## $res
## $res[[1]]
## + 2/9 vertices, named, from 9ce51c9:
## [1] N1 N5
##
##
## $nrgeo
## [1] 1 1 0 0 1 0 0 1 0
```

Neighbourhood Functions

```
# Edges of specific single node:
print (  incident(gr1_dir, V(gr1_dir)$name=="N3") )
```

```
## + 3/12 edges from 9ce51c9 (vertex names):
## [1] N3->N4 N3->N9 N2->N3
```

```
# Edges from a specific set of nodes
print( incident_edges(gr1_dir, V(gr1_dir)$type=="proteins") )
```

```
## $N1
## + 3/12 edges from 9ce51c9 (vertex names):
## [1] N1->N2 N1->N5 N1->N8
##
## $N2
## + 2/12 edges from 9ce51c9 (vertex names):
## [1] N2->N3 N2->N5
##
## $N3
## + 2/12 edges from 9ce51c9 (vertex names):
## [1] N3->N4 N3->N9
##
## $N4
## + 1/12 edge from 9ce51c9 (vertex names):
## [1] N4->N1
##
## $N5
## + 0/12 edges from 9ce51c9 (vertex names):
```

```
# Neighbouring nodes of a single node
print( neighbors(gr1_dir, V(gr1_dir)$name=="N4") )
```

```
## + 1/9 vertex, named, from 9ce51c9:
## [1] N1
```

```
# Neighbouring nodes of a set of nodes
print( adjacent_vertices(gr1_dir, V(gr1_dir)$type=="proteins") )
```

```
## $N1
## + 3/9 vertices, named, from 9ce51c9:
## [1] N2 N5 N8
##
## $N2
## + 2/9 vertices, named, from 9ce51c9:
## [1] N3 N5
##
## $N3
## + 2/9 vertices, named, from 9ce51c9:
## [1] N4 N9
##
## $N4
## + 1/9 vertex, named, from 9ce51c9:
## [1] N1
##
## $N5
## + 0/9 vertices, named, from 9ce51c9:
```

Order of the neighbour

The nodes within a distance of one edge from our node of interest are called first order neighbours. The ego function allows us to extract nodes further away from our node, within a specific order.

```
# Identify nodes within 2 orders from the node A
print(  ego(gr1_dir, order=2, nodes=V(gr1_dir)$name=="proteins")
)
```

```
## list()
```

```
## We can also select all the edges between two specific set of v
ertices.
print( E(gr1_dir)[V(gr1_dir)[type=="proteins"] %--% V(gr1_dir)[ty
pe=="gene"]] )
```

```
## + 2/12 edges from 9ce51c9 (vertex names):
## [1] N1->N8 N3->N9
```