社会网络分析可视化

陈华珊 (中国社科院社会发展战略研究院)

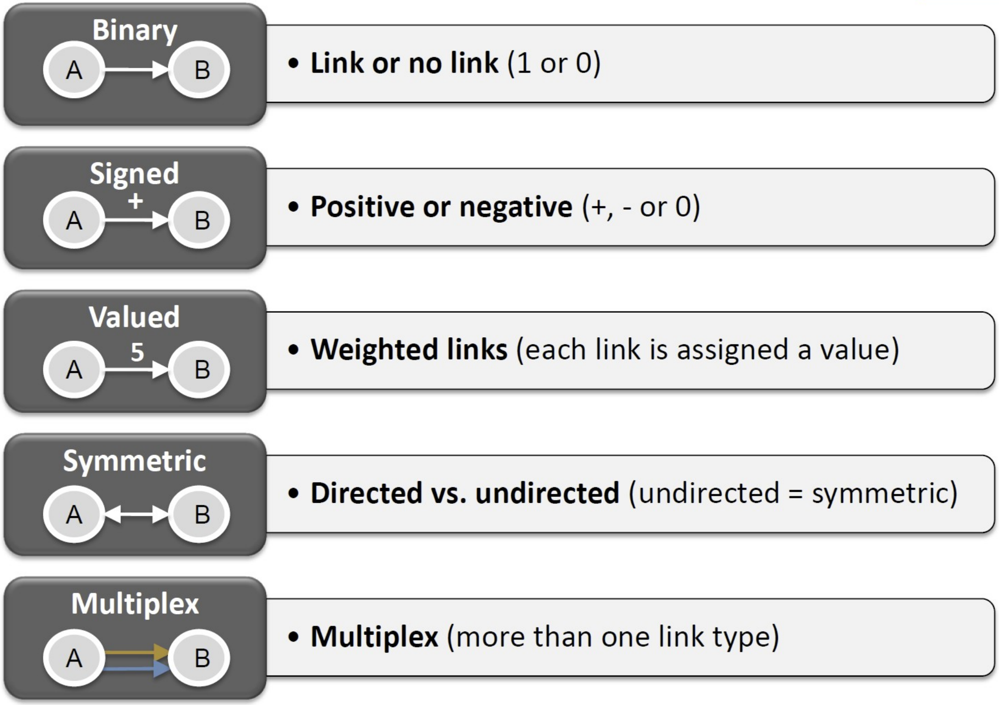
基本概念

网络数据的获取方式

Ego Networks	• Can use standard sampling techniques (e.g. random sample) • Each respondent describes their own relationships (name generators).
Complete Networks	• Boundary specification? • Each respondent reports their own relationships within the network. • Could use a roster that people use to identify contacts.
Cognitive Social Structures	• Ask not only for a person’s own relationships, but also for perceived relationships between other people in your population.
Snowball Sampling	• Individuals included in the sample identify contacts (friends, sexual partners, etc.) who are added to the study at the next step. • Often used in preventive medicine.
Secondary Data	• Digital traces, social media, hyperlink networks and many more.

网络数据的结构

关系类型



Binary
- **Link or no link** (1 or 0)

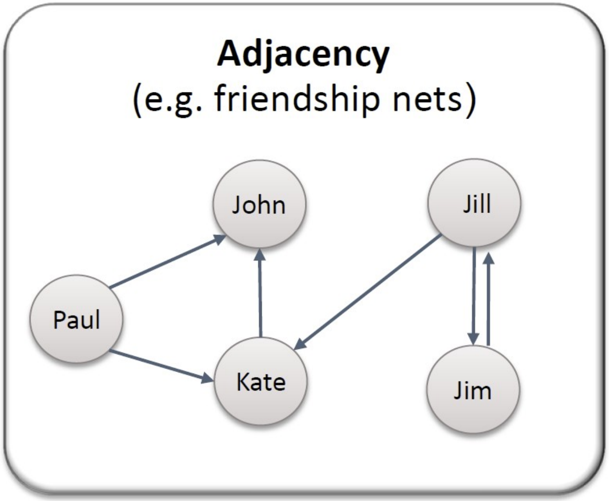
Signed
- **Positive or negative** (+, - or 0)

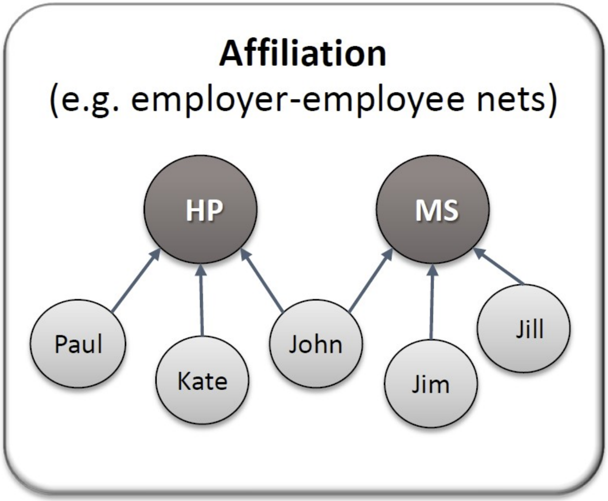
Valued
- **Weighted links** (each link is assigned a value)

Symmetric
- **Directed vs. undirected** (undirected = symmetric)

Multiplex
- **Multiplex** (more than one link type)

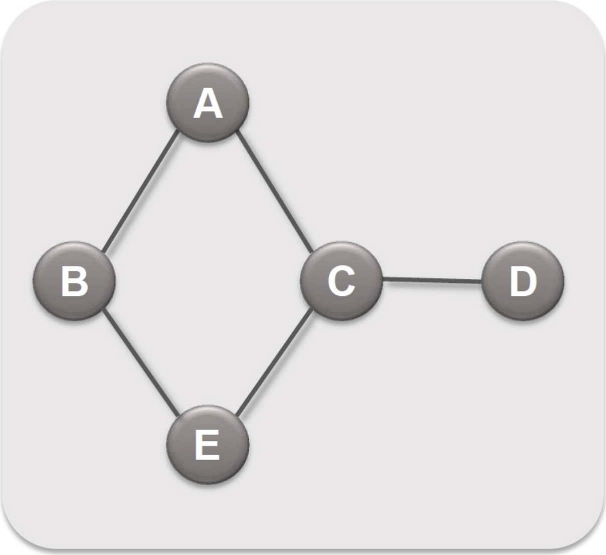
单模网与双模网


Adjacency
(e.g. friendship nets)
John
Jill
Paul
Kate
Jim


Affiliation
(e.g. employer-employee nets)
HP
MS
Paul
Kate
John
Jim
Jill

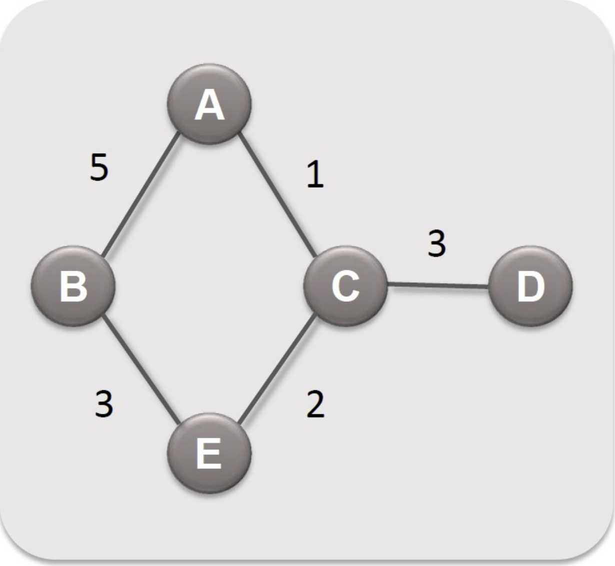
网络的矩阵表示：有向网

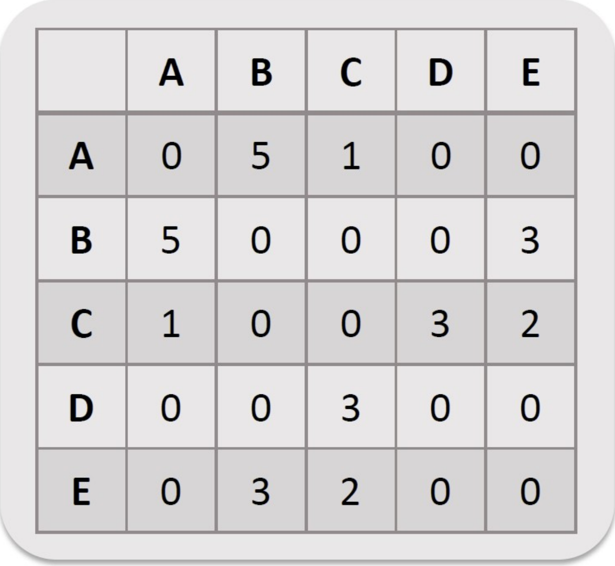
网络的矩阵表示：对称网



	A	B	C	D	E
A	0	1	1	0	0
B	1	0	0	0	1
C	1	0	0	1	1
D	0	0	1	0	0
E	0	1	1	0	0

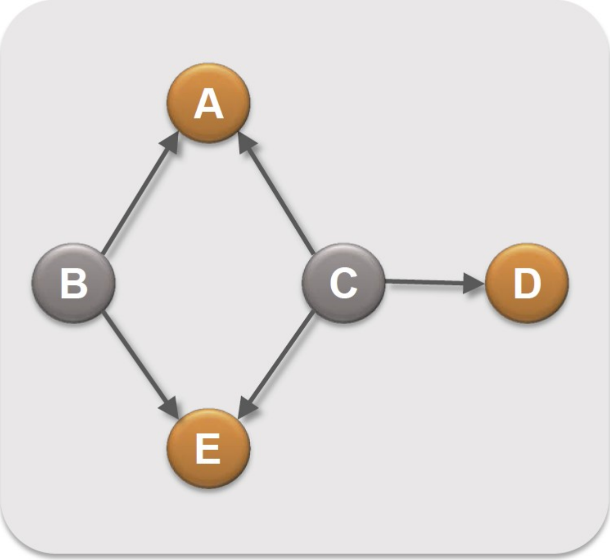
网络的矩阵表示：有价网



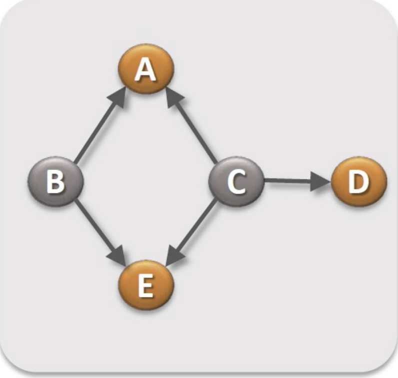


	A	B	C	D	E
A	0	5	1	0	0
B	5	0	0	0	3
C	1	0	0	3	2
D	0	0	3	0	0
E	0	3	2	0	0

网络的矩阵表示：附属网



	A	E	D
B	1	1	0
C	1	1	1



M

	A	E	D
B	1	1	0
C	1	1	1

M^T

	B	C
A	1	1
E	1	1
D	0	1

$M*M^T =$

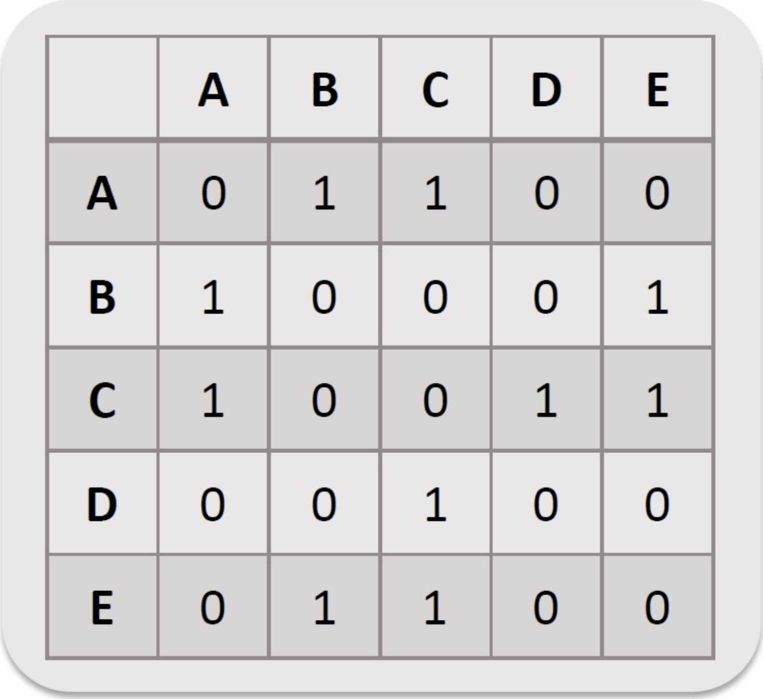
	B	C
B	2	2
C	2	3

$M^T *M=$

	A	E	D
A	2	2	1
E	2	2	1
D	1	1	1

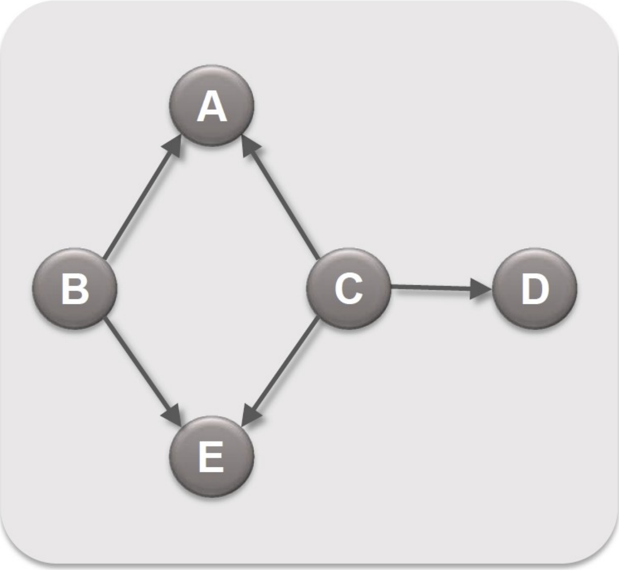
数据结构：邻接矩阵

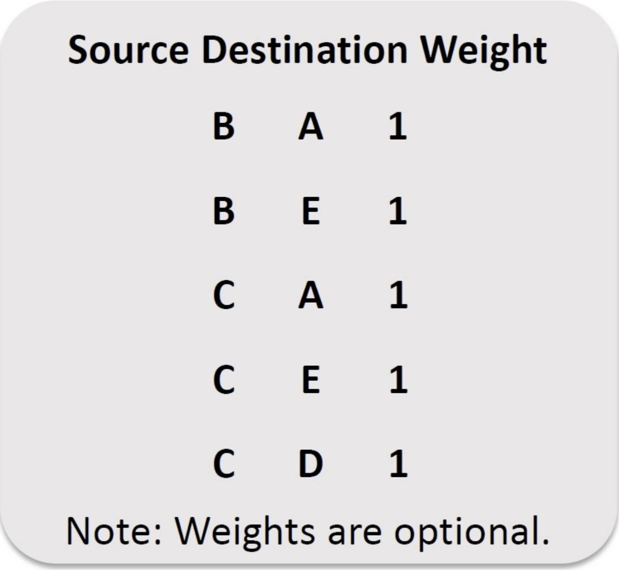
所有的社会网络内部结构都可用邻接矩阵表示：



	A	B	C	D	E
A	0	1	1	0	0
B	1	0	0	0	1
C	1	0	0	1	1
D	0	0	1	0	0
E	0	1	1	0	0

数据结构：**edgelist**

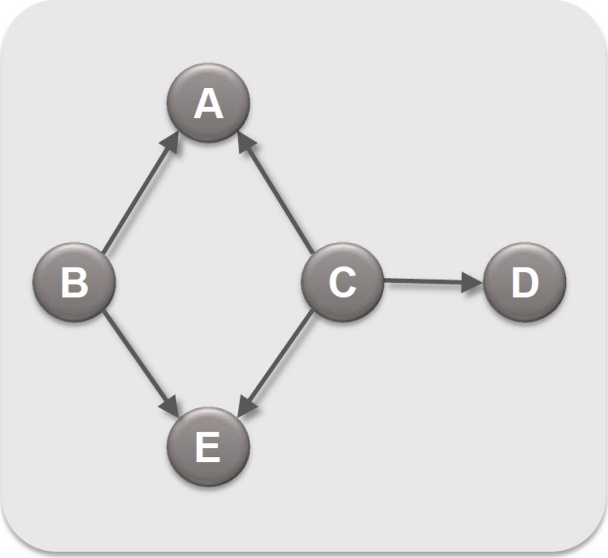




Source	Destination	Weight
B	A	1
B	E	1
C	A	1
C	E	1
C	D	1

Note: Weights are optional.

数据结构：**nodelist**

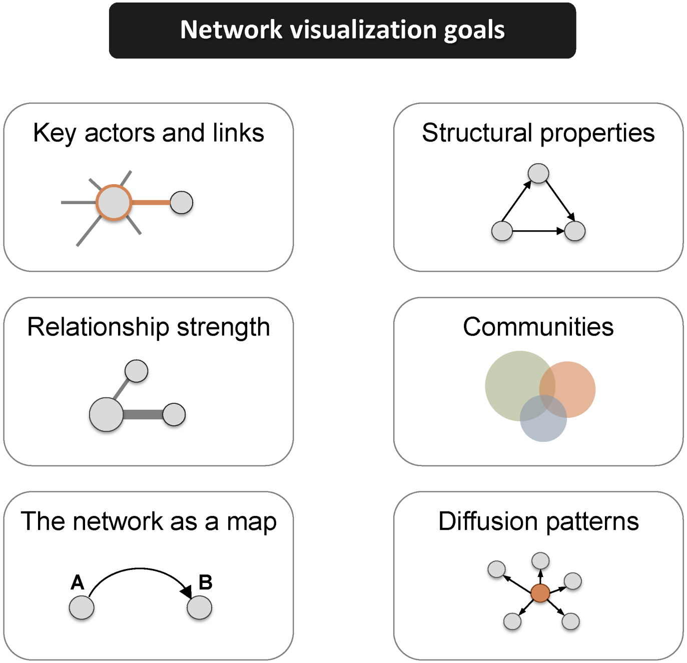


Source Destinations

B A E

C A D E

可视化的目标



Network visualization goals

Key actors and links

Structural properties

Relationship strength

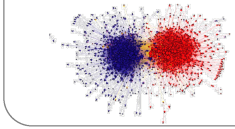
Communities

The network as a map

Diffusion patterns

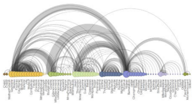
Some network visualization types

Network Maps

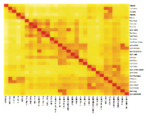


Statistical charts

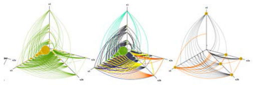
Arc diagrams



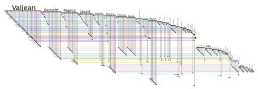
Heat maps



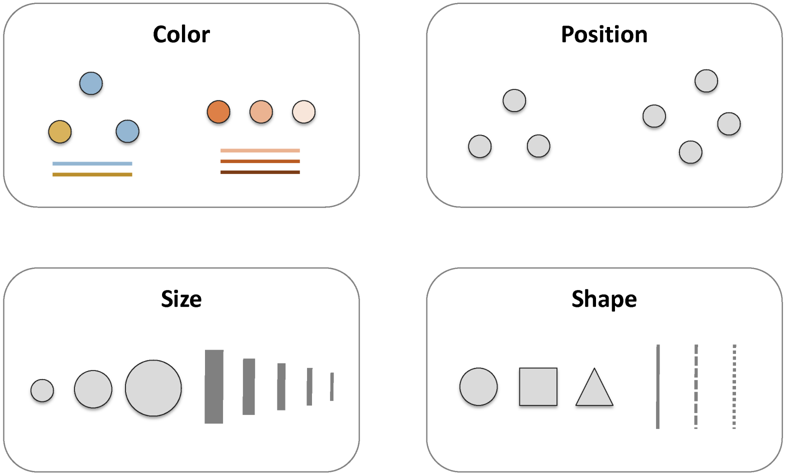
Hive plots



Biofabric



Network visualization controls



Color

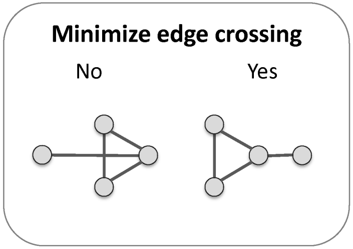
Position

Size

Shape

Honorable mention: arrows (direction) and labels (identification)

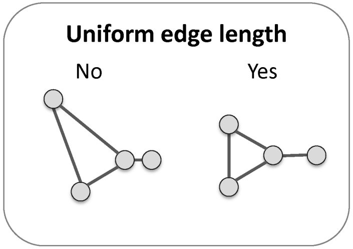
Layout aesthetics



Minimize edge crossing

No

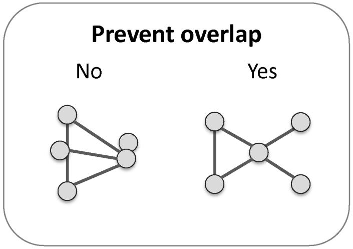
Yes



Uniform edge length

No

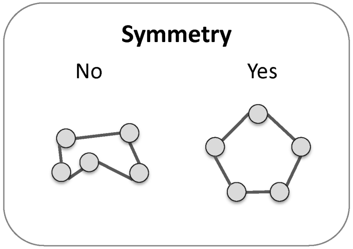
Yes



Prevent overlap

No

Yes



Symmetry

No

Yes

iGraph 制图

iGraph数据准备

示例1：**edgelist**

```
nodes <- read.csv("Dataset1-Media-Example-NODES.csv", header=T,
as.is=T)
links <- read.csv("Dataset1-Media-Example-EDGES.csv", header=T,
as.is=T)

# 检查数据
head(nodes)
##    id               media media.type type.label audience.size
## 1 s01            NY Times          1  Newspaper            20
## 2 s02     Washington Post          1  Newspaper            25
## 3 s03 Wall Street Journal          1  Newspaper            30
## 4 s04           USA Today          1  Newspaper            32
## 5 s05            LA Times          1  Newspaper            20
```

```
14      ## 6 s06        New York Post            1  Newspaper            50
        head(links)
        ##   from  to weight      type
        ## 1  s01 s02     10 hyperlink
        ## 2  s01 s02     12 hyperlink
        ## 3  s01 s03     22 hyperlink
        ## 4  s01 s04     21 hyperlink
        ## 5  s04 s11     22   mention
        ## 6  s05 s15     21   mention
```

```
nrow(nodes);
## [1] 17
length(unique(nodes$id))
## [1] 17
nrow(links);
## [1] 52
nrow(unique(links[,c("from", "to")]))
## [1] 49

# 聚合数据
links <- aggregate(links[,3], links[,-3], sum)
links <- links[order(links$from, links$to),]
colnames(links)[4] <- "weight"
rownames(links) <- NULL
```

示例2：matrix

```
nodes2 <- read.csv("Dataset2-Media-User-Example-NODES.csv", header=T,
as.is=T)
links2 <- read.csv("Dataset2-Media-User-Example-EDGES.csv", header=T,
row.names=1)

# 检查数据
head(nodes2)
##    id   media media.type media.name audience.size
## 1 s01     NYT          1  Newspaper            20
## 2 s02    WaPo          1  Newspaper            25
## 3 s03     WSJ          1  Newspaper            30
## 4 s04    USAT          1  Newspaper            32
## 5 s05 LATimes          1  Newspaper            20
## 6 s06     CNN          2         TV            56
head(links2)
##     U01 U02 U03 U04 U05 U06 U07 U08 U09 U10 U11 U12 U13 U14
U15 U16 U17
```

```
18  ## s01    1    1    1    0    0    0    0    0    0    0    0    0    0    0
    0    0    0
    ## s02    0    0    0    1    1    0    0    0    0    0    0    0    0    0
    0    0    0
    ## s03    0    0    0    0    0    1    1    1    1    0    0    0    0    0
    0    0    0
    ## s04    0    0    0    0    0    0    0    0    1    1    1    0    0    0
    0    0    0
    ## s05    0    0    0    0    0    0    0    0    0    0    1    1    1    0
    0    0    0
    ## s06    0    0    0    0    0    0    0    0    0    0    0    0    1    1
    0    0    1
    ##      U18 U19 U20
    ## s01    0    0    0
    ## s02    0    0    1
    ## s03    0    0    0
    ## s04    0    0    0
    ## s05    0    0    0
## s06     0     0     0
```

```
# links2 为双模邻接矩阵
links2 <- as.matrix(links2)
dim(links2)
## [1] 10 20
dim(nodes2)
## [1] 30  5
```

数据->iGraph

将数据转换为 `iGraph` 对象，针对多种数据格式:

- `graph.data.frame()`, `graph_from_data_frame()`, `from_data_frame()`
- `graph.edgelist()`, `graph_from_edgelist()`, `from_edgelist()`
- `graph.adjacency()`, `graph_from_adjacency_matrix()`, `from_adjacency()`

```
library(igraph)
(net <- graph.data.frame(links, nodes, directed=T))
## IGRAPH DNW- 17 49 --
## + attr: name (v/c), media (v/c), media.type (v/n), type.label
##   (v/c), audience.size (v/n), type (e/c), weight (e/n)
```

igraph 的基本属性:

- D, U: 有向网（directed）, 无向网（undirected）
- N : 网络是否命名 (节点具有属性 name）
- W: 是否有权网（网络边具有权重属性 weight）
- B: 是否双模网（bipartite /two-mode）, 节点具有属性 type

```
## IGRAPH DNW- 17 49 --
## + attr: name (v/c), media (v/c), media.type (v/n), type.label
##   (v/c), audience.size (v/n), type (e/c), weight (e/n)
```

- `(g/c)` - **g**raph-level **c**haracter attribute
- `(v/c)` - **v**ertex-level **c**haracter attribute
- `(e/n)` - **e**dge-level **n**umeric attribute

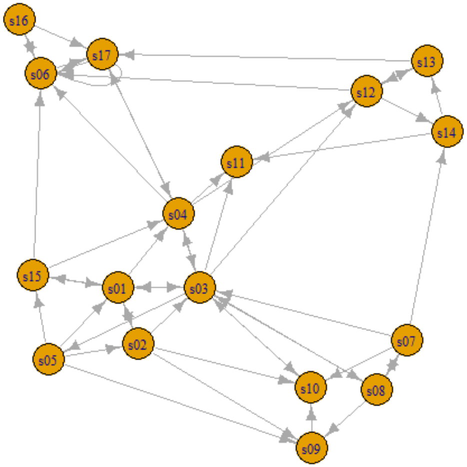
节点属性、边属性

```
E(net)       # The edges of the "net" object
V(net)       # The vertices of the "net" object
E(net)$type  # Edge attribute "type"
V(net)$media # Vertex attribute "media"

# 对网络数值直接操作
net[1,]
net[5,7]
```

画图-默认

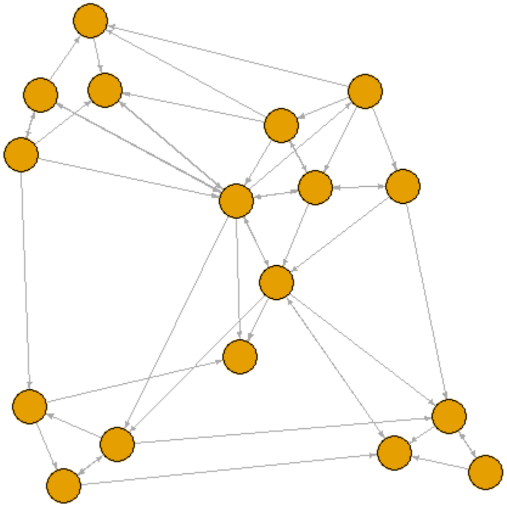
按照默认设置输出图形

```
plot(net)
```



合并复合边

```
net <- simplify(net, remove.multiple = F, remove.loops = T)
plot(net, edge.arrow.size=.4,vertex.label=NA)
```



iGraph 画图参数

Nodes

lor	节点颜色
lor	节点边框颜色
ape	节点形状: “none” , “circle” , “square” , “csquare” , “rectangle” , “vrectangle” , “pie” , “raster’
ize	节点大小 (default is 15)
ze2	节点大小 (有些形状需要两个参数控制大小，e.g. rectangle)
bel	节点标签
nily	节点标签字形 (e.g. “Times” , “Helvetica”)
ont	节点标签字体：1 plain, 2 bold, 3, italic, 4 bold italic, 5 symbol
cex	节点标签字号 (乘数常量, 随图形设备而不同)
dist	节点标签与节点形状之间的距离
ree	标签输出角度：0 right, “pi” is left, “pi/2” is below, and “-pi/” is above

Edges

边颜色

边宽度, defaults to 1

箭头大小, defaults to 1

箭头宽度, defaults to 1

线条类型, 0 or “blank” , 1 or “solid” , 2 or “dashed” , 3 or “dotted” , 4 or “dotdash” , 5 or “longdash

边标签

标签字形 (e.g. “Times” , “Helvetica”)

标签字体: 1 plain, 2 bold, 3, italic, 4 bold italic, 5 symbol

标签字号

边曲度, range 0-1 (FALSE sets it to 0, TRUE to 0.5)

指定哪些边使用箭头，向量: 0 no arrow, 1 back, 2 forward, 3 both

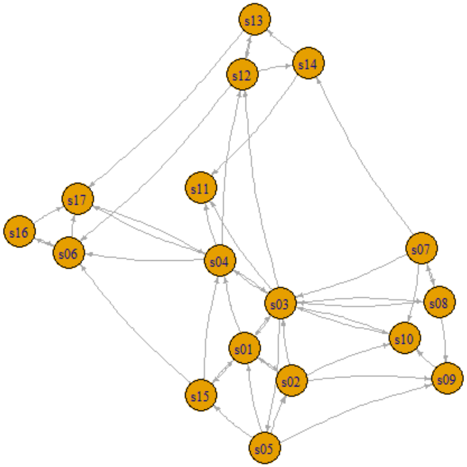
Other

margin	图形边距，长度为4的向量
frame	是否显示边框
main	图形标题
sub	图形副标题

设置参数的两种方式

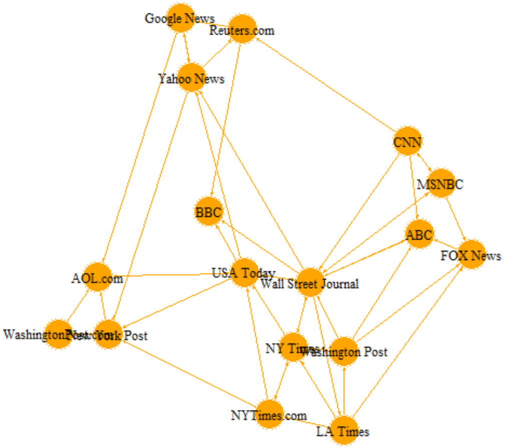
使用 **`plot()`** 函数

```
plot(net, edge.arrow.size=.4, edge.curved=.1)
```



```
##  [1] 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1
0.1 0.1 0.1
## [18] 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1
0.1 0.1 0.1
## [35] 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1
```

```
# Set edge color to light gray, the node & border color to orange
# Replace the vertex label with the node names stored in "media"
plot(net, edge.arrow.size=.2, edge.color="orange",
     vertex.color="orange", vertex.frame.color="#ffffff",
     vertex.label=V(net)$media, vertex.label.color="black")
```

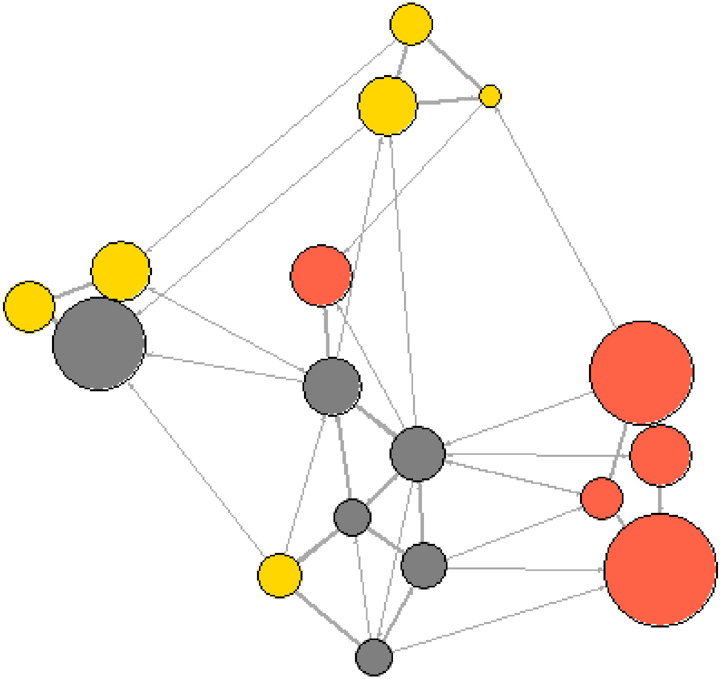


将参数设置到**igraph**对象上

```
# Generate colors base on media type:
colrs <- c("gray50", "tomato", "gold")
V(net)$color <- colrs[V(net)$media.type]

# Compute node degrees (#links) and use that to set node size:
deg <- igraph::degree(net, mode="all")
V(net)$size <- deg*3
# We could also use the audience size value:
V(net)$size <- V(net)$audience.size*0.6

# The labels are currently node IDs.
# Setting them to NA will render no labels:
V(net)$label <- NA
```

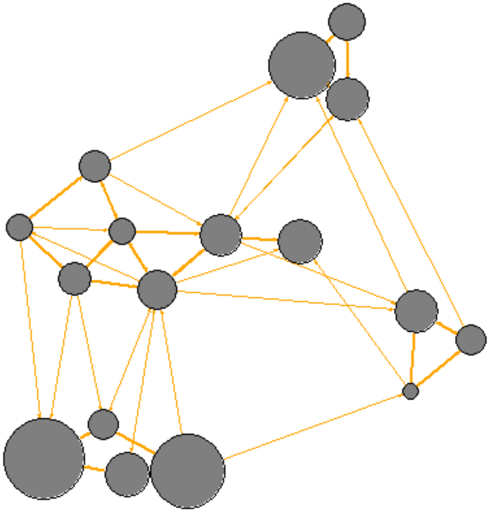
```
# Set edge width based on weight:
E(net)$width <- E(net)$weight/6

###change arrow size and edge color:
```

```
E(net)$arrow.size <- .2
E(net)$edge.color <- "gray80"
E(net)$width <- 1+E(net)$weight/12
plot(net)
```



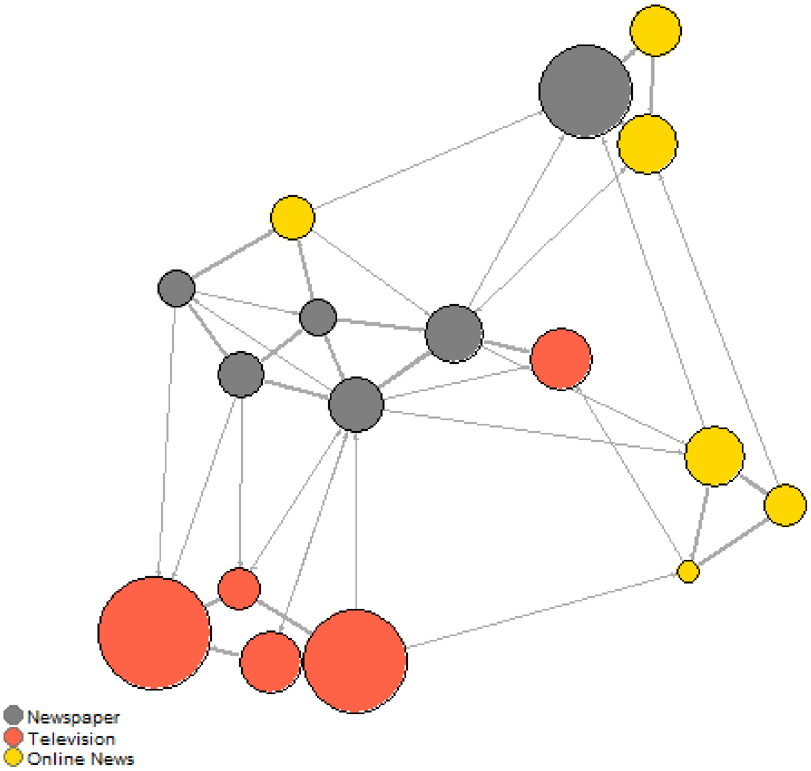
`plot()` 可用于重置前述设置

```
plot(net, edge.color="orange", vertex.color="gray50")
```



添加图例

```
plot(net)
legend(x=-1.5, y=-1.1, c("Newspaper","Television", "Online News"),
pch=21,
       col="#777777", pt.bg=colrs, pt.cex=2, cex=.8, bty="n",
ncol=1)
```

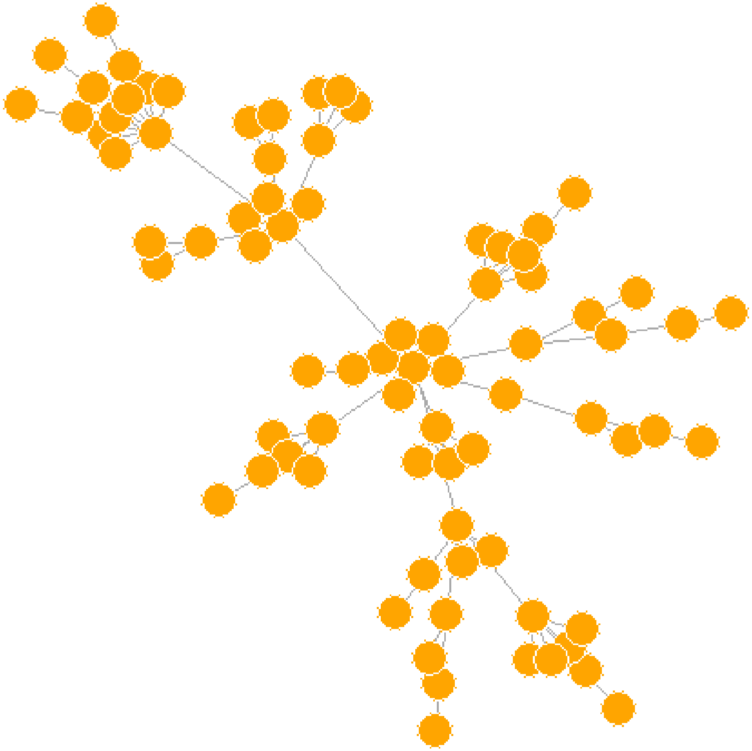

● Newspaper
● Television
● Online News

布局（Layout）

使用布局

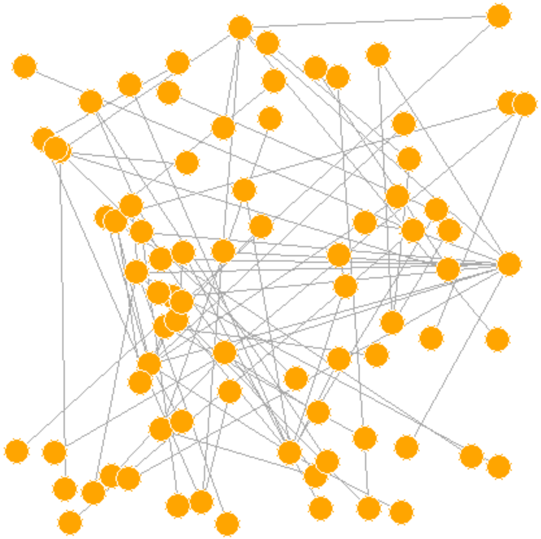
默认布局

```
net.bg <- barabasi.game(80)
V(net.bg)$frame.color <- "white"
V(net.bg)$color <- "orange"
V(net.bg)$label <- ""
V(net.bg)$size <- 10
E(net.bg)$arrow.mode <- 0
plot(net.bg)
```



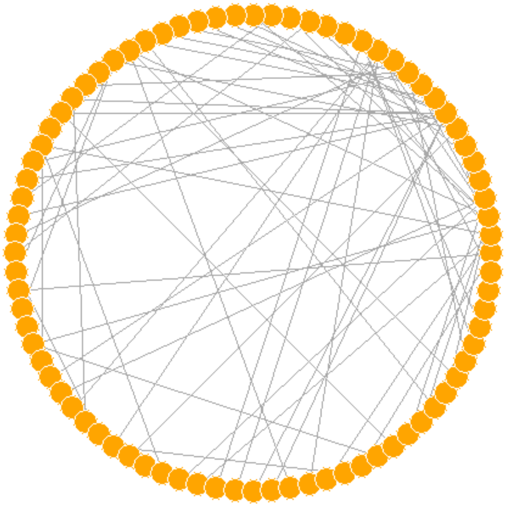
随机布局

```
plot(net.bg, layout=layout.random)
```



布局重复使用

```
l <- layout.circle(net.bg)
plot(net.bg, layout=l)
```



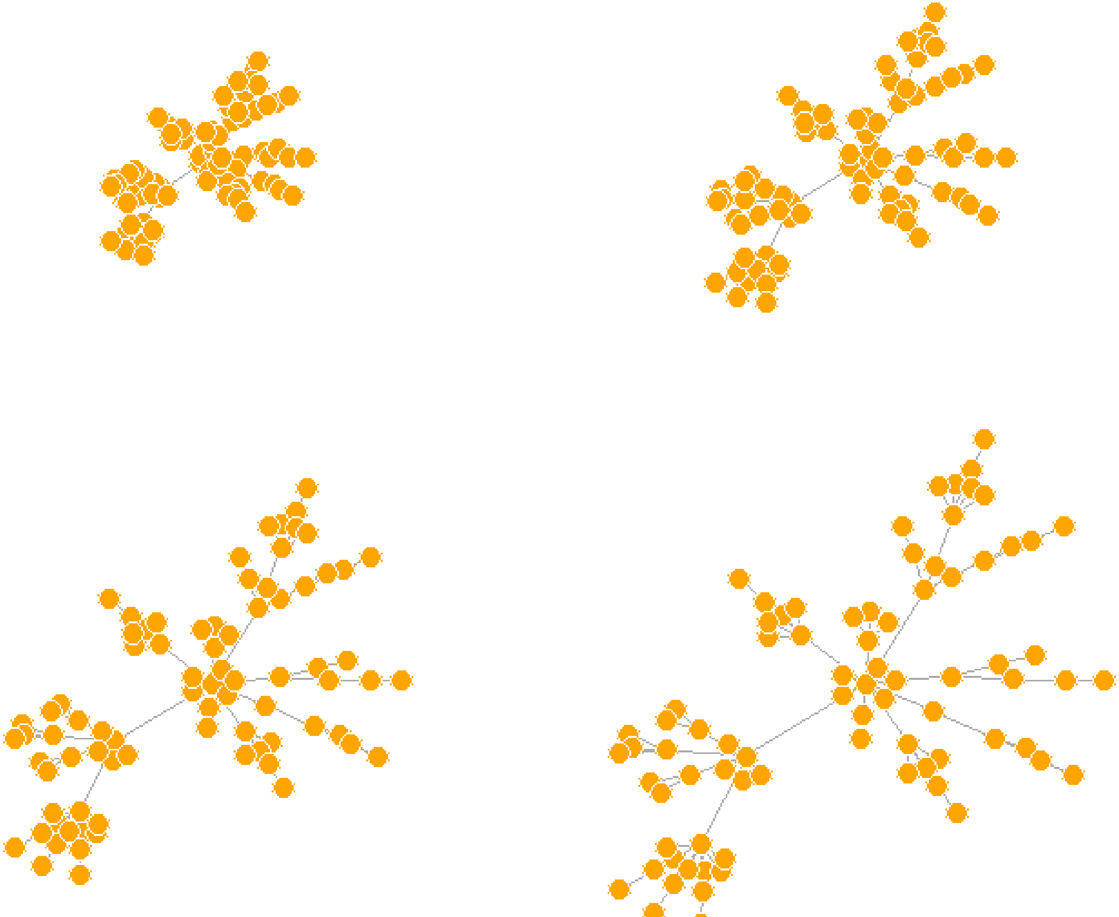
更多布局

```
plot(net.bg, layout=layout.circle)
plot(net.bg, layout=layout.sphere)
```

布局与坐标

- l 仅仅是一个两列的坐标矩阵:
  - (N x 2) for the N nodes in the graph
  - colname: x, y
- 默认坐标范围为 [-1, 1]
- 用参数 rescale=FALSE 进行坐标范围的自定义
- 用 layout.norm 对坐标范围进行归一化。

```
l <- layout.fruchterman.reingold(net.bg)
l <- layout.norm(l, ymin=-1, ymax=1, xmin=-1, xmax=1)

par(mfrow=c(2,2), mar=c(0,0,0,0))
plot(net.bg, rescale=F, layout=l*0.4)
plot(net.bg, rescale=F, layout=l*0.6)
plot(net.bg, rescale=F, layout=l*0.8)
plot(net.bg, rescale=F, layout=l*1.0)
```

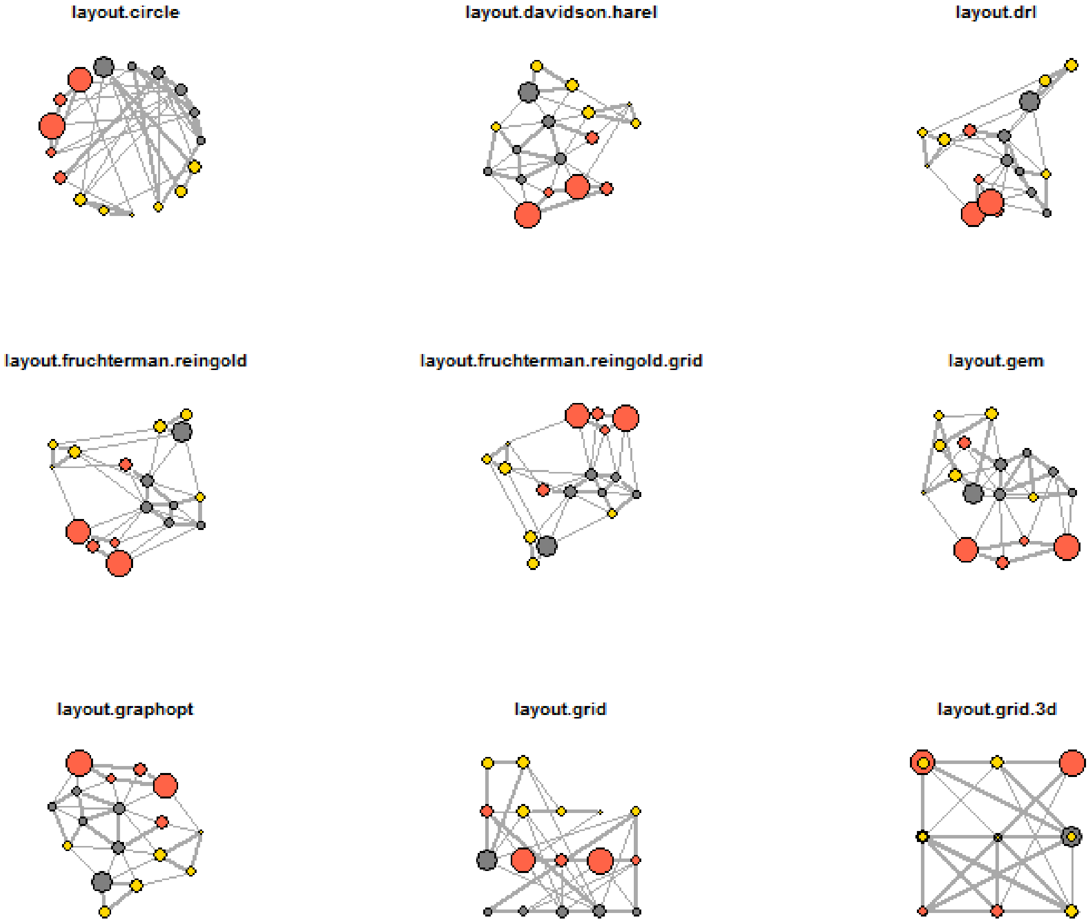


igraph的不同布局

```
layouts <- grep("^layout\\.", ls("package:igraph"), value=TRUE)
# Remove layouts that do not apply to our graph.
layouts <- layouts[   !grepl("auto|bipartite|merge|norm|sugiyama",
layouts)]

par(mfrow=c(3,3))

for (layout in layouts) {
  print(layout)
  l <- do.call(layout, list(net))
  plot(net, edge.arrow.mode=0, layout=l, main=layout) }
## [1] "layout.circle"
## [1] "layout.davidson.harel"
## [1] "layout.drl"
## [1] "layout.fruchterman.reingold"
## [1] "layout.fruchterman.reingold.grid"
## [1] "layout.gem"
```

18

```
## [1] "layout.graphopt"
## [1] "layout.grid"
## [1] "layout.grid.3d"
```



layout.circle

layout.davidson.harel

layout.drl

layout.fruchterman.reingold

layout.fruchterman.reingold.grid

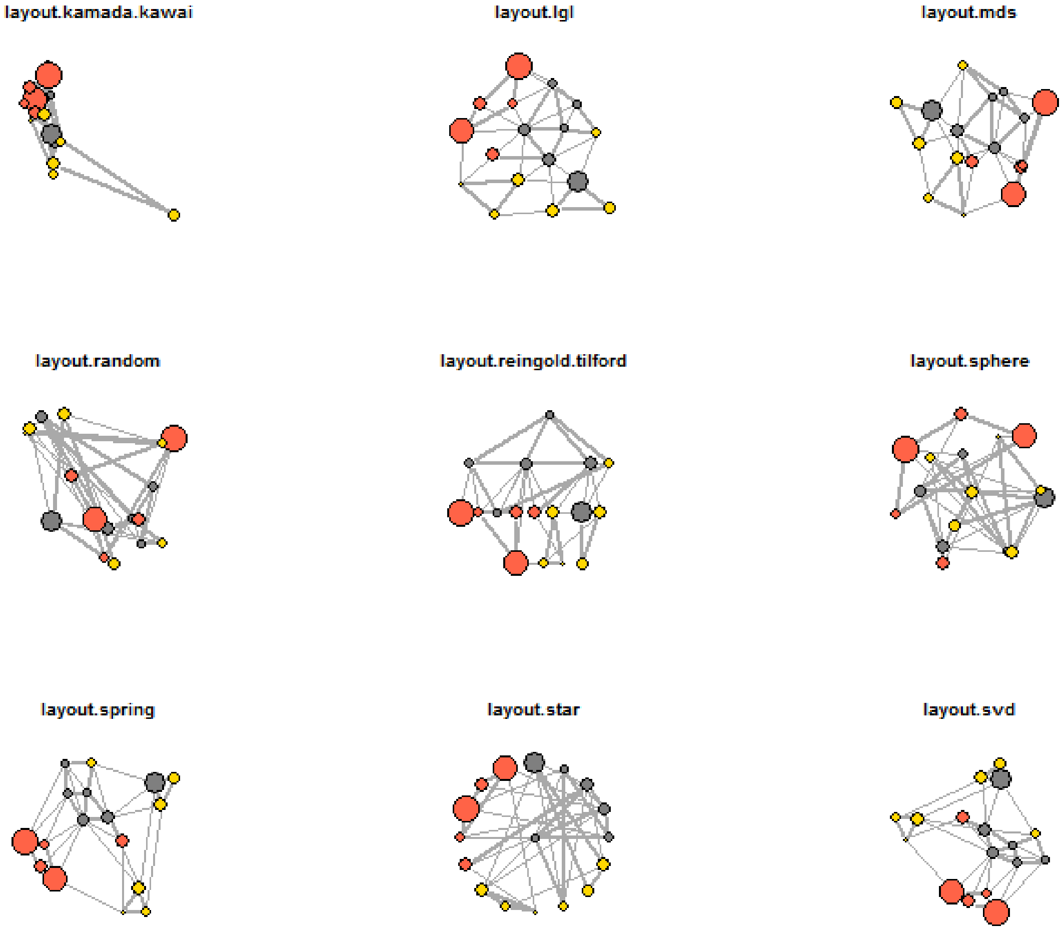
layout.gem

layout.graphopt

layout.grid

layout.grid.3d

```
## [1] "layout.kamada.kawai"
## [1] "layout.lgl"
## [1] "layout.mds"
## [1] "layout.random"
## [1] "layout.reingold.tilford"
## [1] "layout.sphere"
## [1] "layout.spring"
## [1] "layout.star"
## [1] "layout.svd"
```



layout.kamada.kawai

layout.lgl

layout.mds

layout.random

layout.reingold.tilford

layout.sphere

layout.spring

layout.star

layout.svd

```
dev.off()
## null device
##           1
```

用 **`tkplot`** 手工调整布局

通过 `tkplot()` 手工调整节点位置，再将 layout 信息保存起来

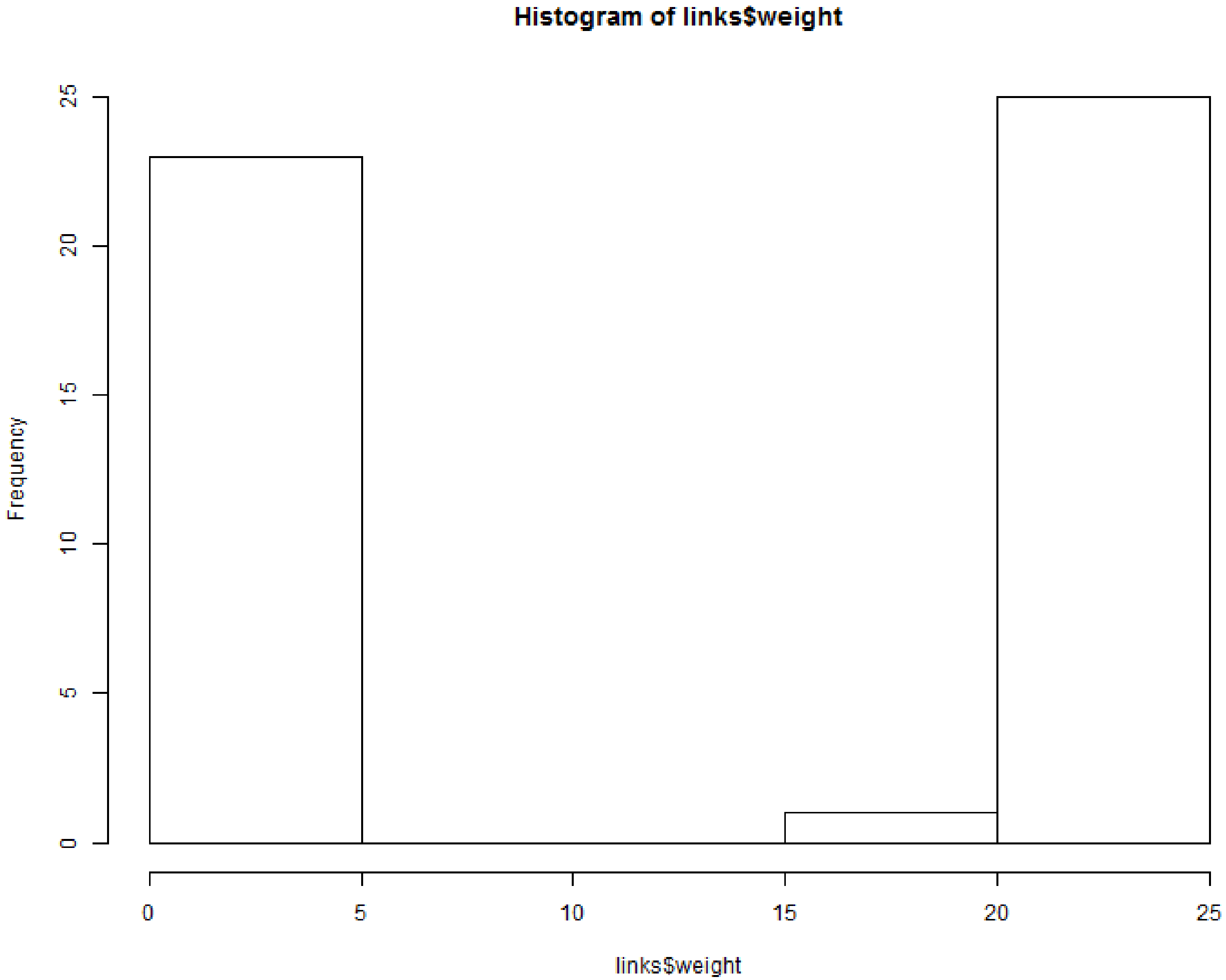
```
L = layout.fruchterman.reingold(G)
tkplot(G, layout=L)
L = tkplot.getcoords(1)
```

优化

过滤

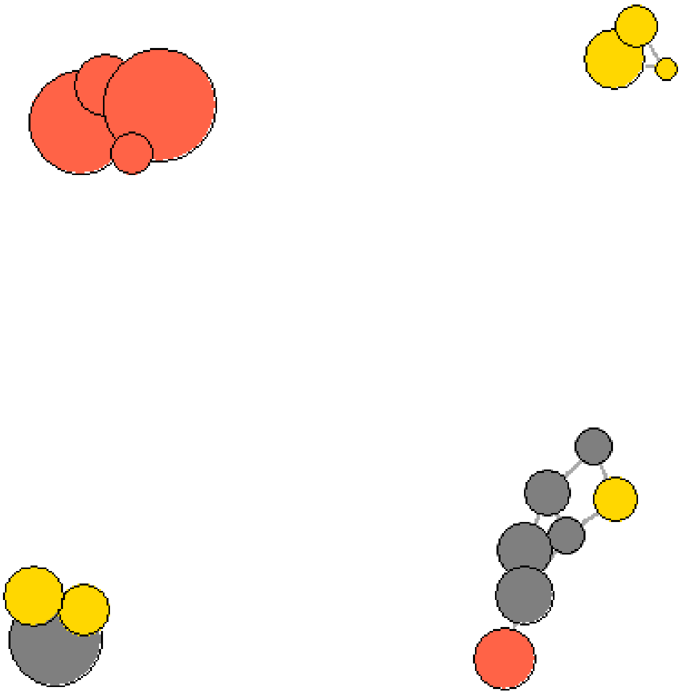
为了凸显网络结构，通常需要对过于密集的网络进行过滤

```
hist(links$weight)
```


Histogram of links$weight
Frequency
0
5
10
15
20
25
0
5
10
15
20
25
links$weight

```
mean(links$weight)
## [1] 12.40816
sd(links$weight)
## [1] 9.905635
```

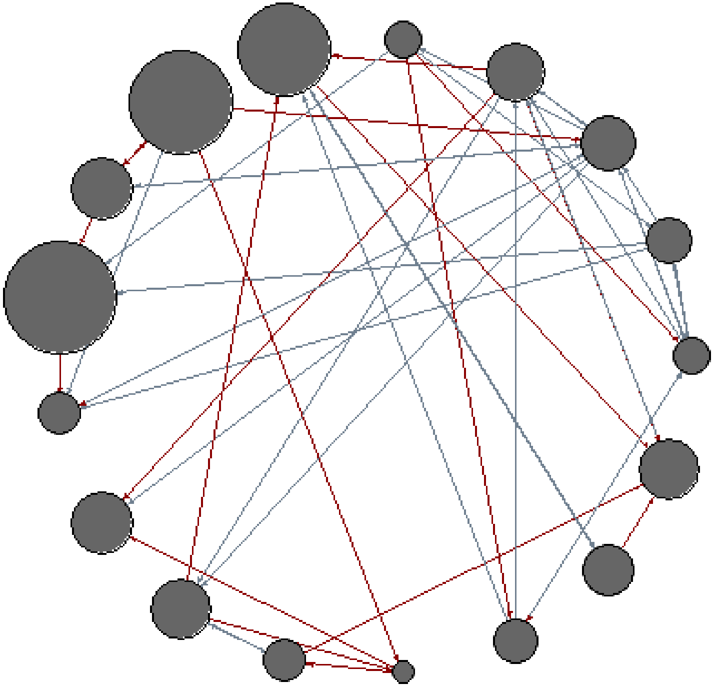
```
cut.off <- mean(links$weight)
net.sp <- igraph::delete.edges(net, E(net)[weight<cut.off])
l <- layout.fruchterman.reingold(net.sp, repulserad=vcount(net)^2.1)
plot(net.sp, layout=l)
```



区分

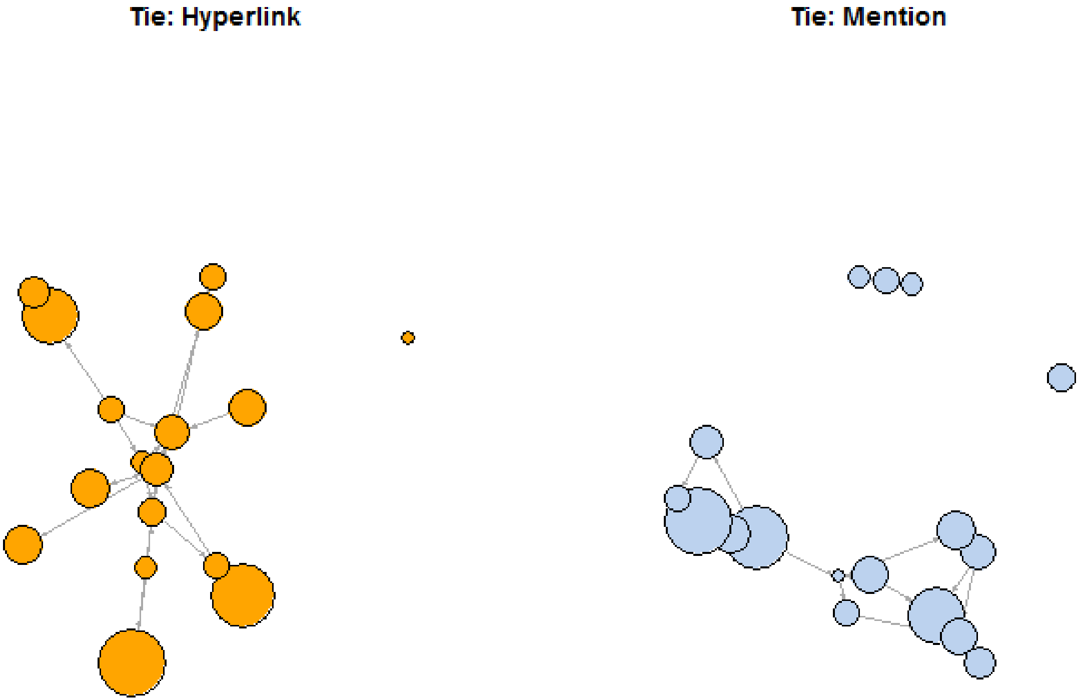
区分关系类型

```
E(net)$width <- 1.5
plot(net, edge.color=c("dark red", "slategrey")[(E(net)$type=="hyperlink")+
      vertex.color="gray40", layout=layout.circle)
```



拆分为两个网络

```
net.m <- net - E(net)  [E(net)$type=="hyperlink"] # another way to
delete edges
net.h <- net - E(net)[E(net)$type=="mention"]

par(mfrow=c(1,2))
plot(net.h, vertex.color="orange", main="Tie: Hyperlink")
plot(net.m, vertex.color="lightsteelblue2", main="Tie: Mention")
```

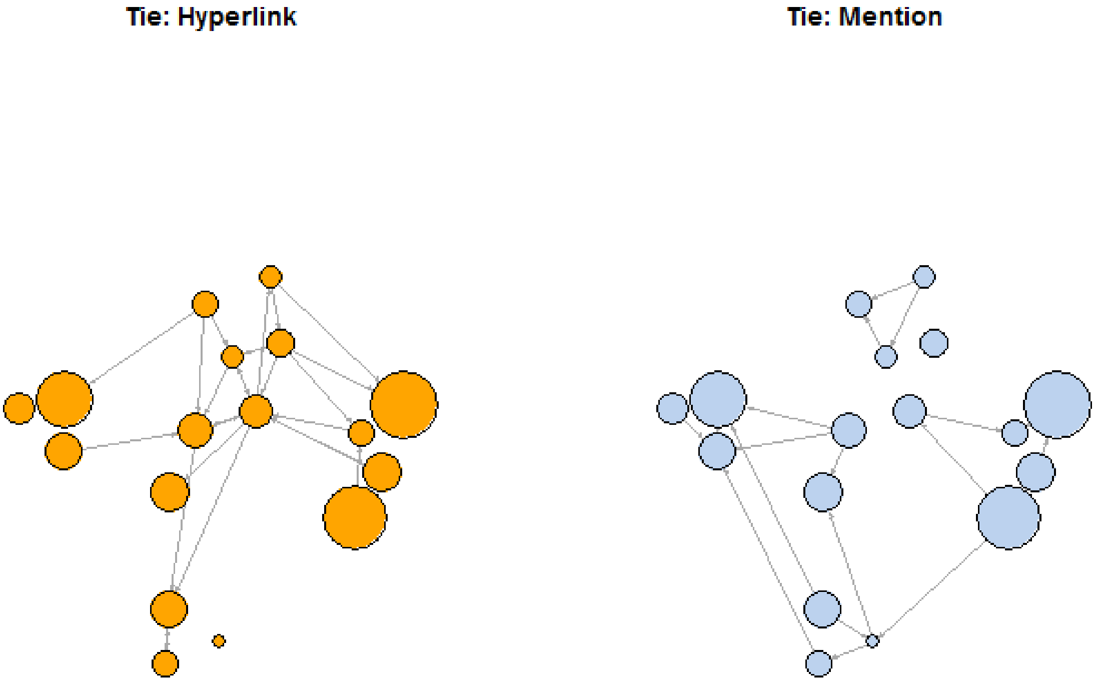


Tie: Hyperlink

Tie: Mention

```
dev.off()
## null device
##           1
```

```
par(mfrow=c(1,2))
l <- layout.fruchterman.reingold(net)
plot(net.h, vertex.color="orange", layout=l, main="Tie: Hyperlink")
plot(net.m, vertex.color="lightsteelblue2", layout=l, main="Tie:
Mention")
```



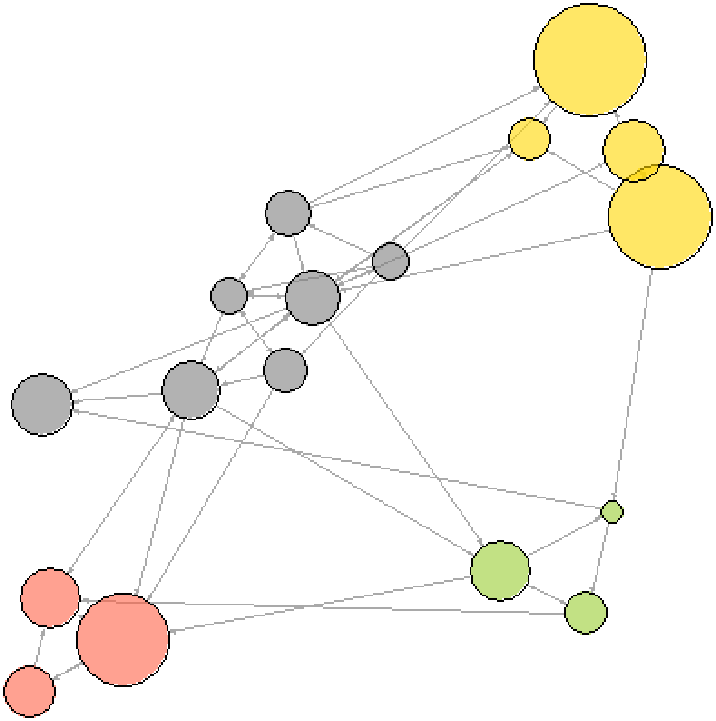
Tie: Hyperlink

Tie: Mention

```
dev.off()
## null device
##           1
```

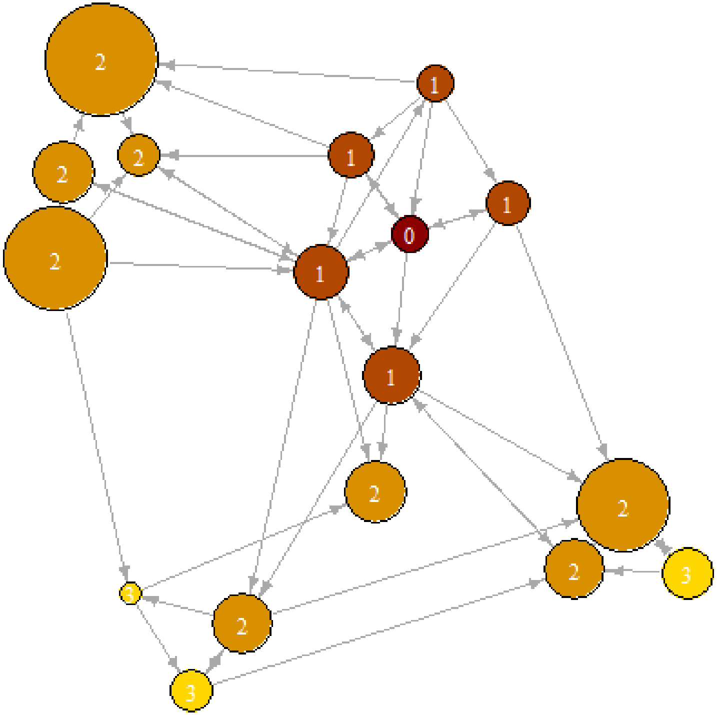
子群

```
V(net)$community <- optimal.community(net)$membership
colrs <- adjustcolor( c("gray50", "tomato", "gold", "yellowgreen"),
alpha=.6)
plot(net, vertex.color=colrs[V(net)$community])
```



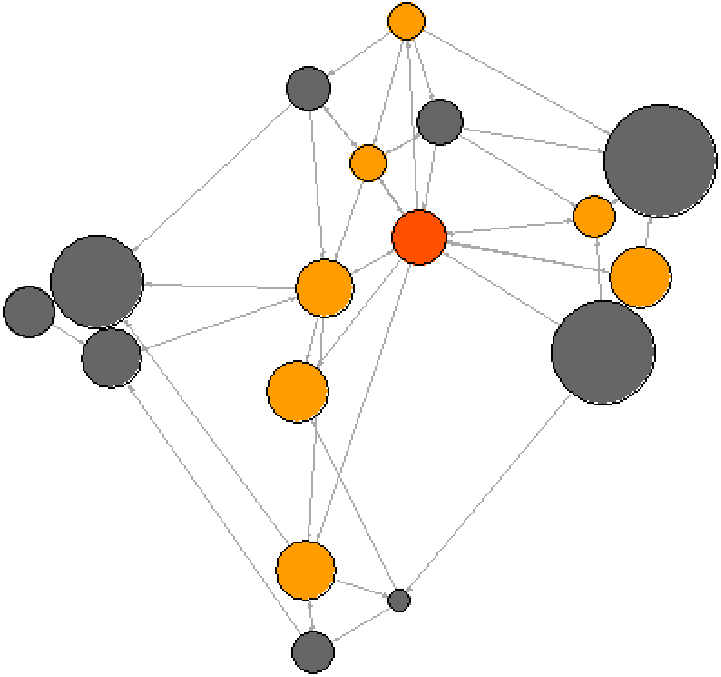
突出指定节点和边

```
dist.from.NYT <- shortest.paths(net, algorithm="unweighted")[1,]
oranges <- colorRampPalette(c("dark red", "gold"))
col <- oranges(max(dist.from.NYT)+1)[dist.from.NYT+1]

plot(net, vertex.color=col, vertex.label=dist.from.NYT, edge.arrow.size=.
     vertex.label.color="white")
```


2
1
1
2
2
1
0
2
1
1
2
2
2
3
3
2
3

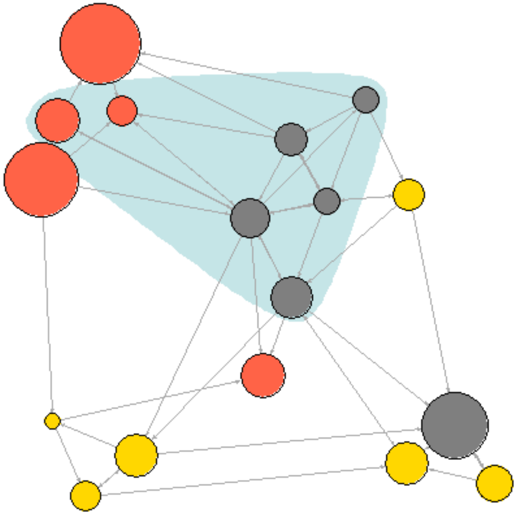
强调邻居

```
col <- rep("grey40", vcount(net))
col[V(net)$media=="Wall Street Journal"] <- "#ff5100"

neigh.nodes <- neighbors(net, V(net)[media=="Wall Street Journal"],
mode="out")

col[neigh.nodes] <- "#ff9d00"
plot(net, vertex.color=col)
```



用底色突出子群

```
plot(net, mark.groups=c(1,4,5,8), mark.col="#C5E5E7", mark.border=NA)
```



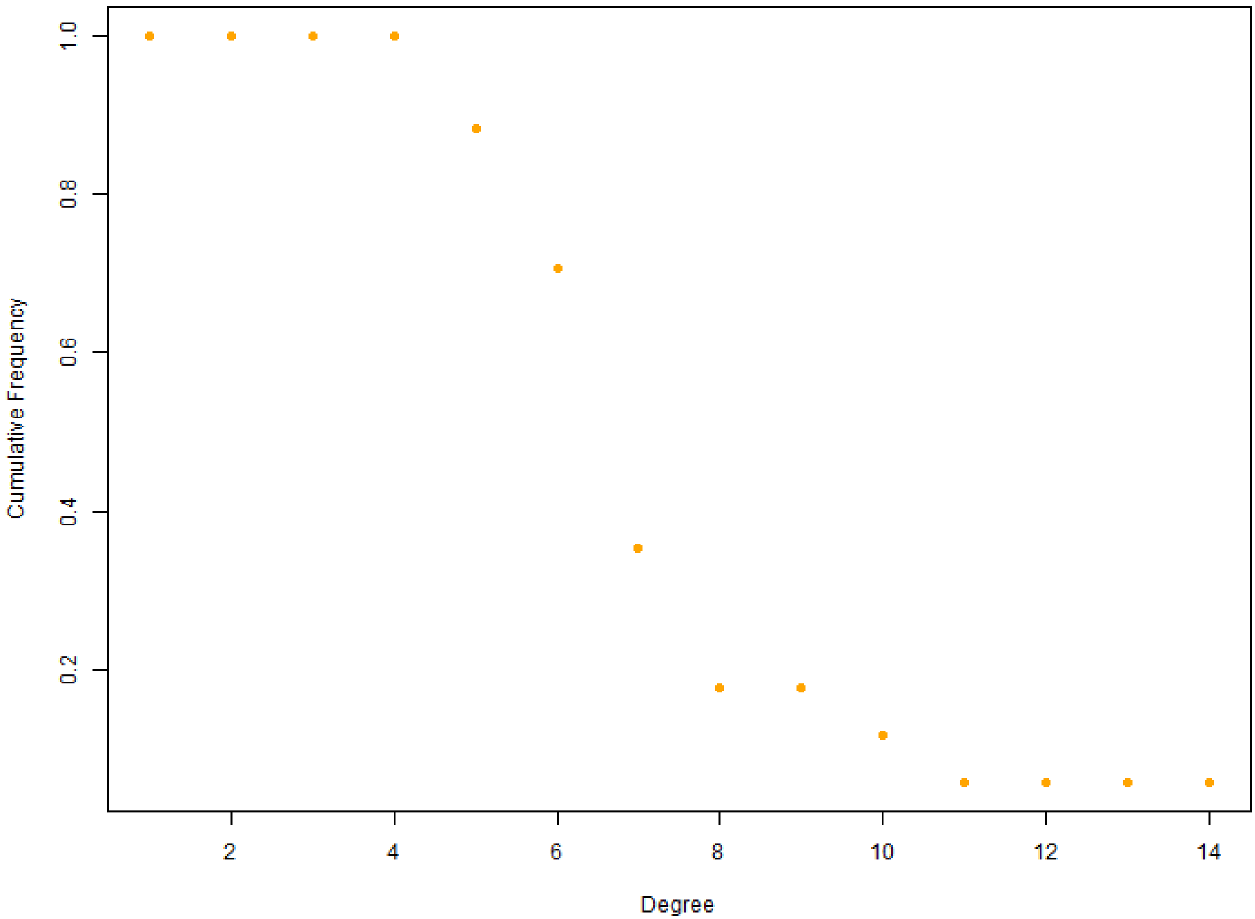
作业

- 如何高亮一条路径?
- 如何突出显示多个子群?

其它图形呈现方式

中心度累计分布图

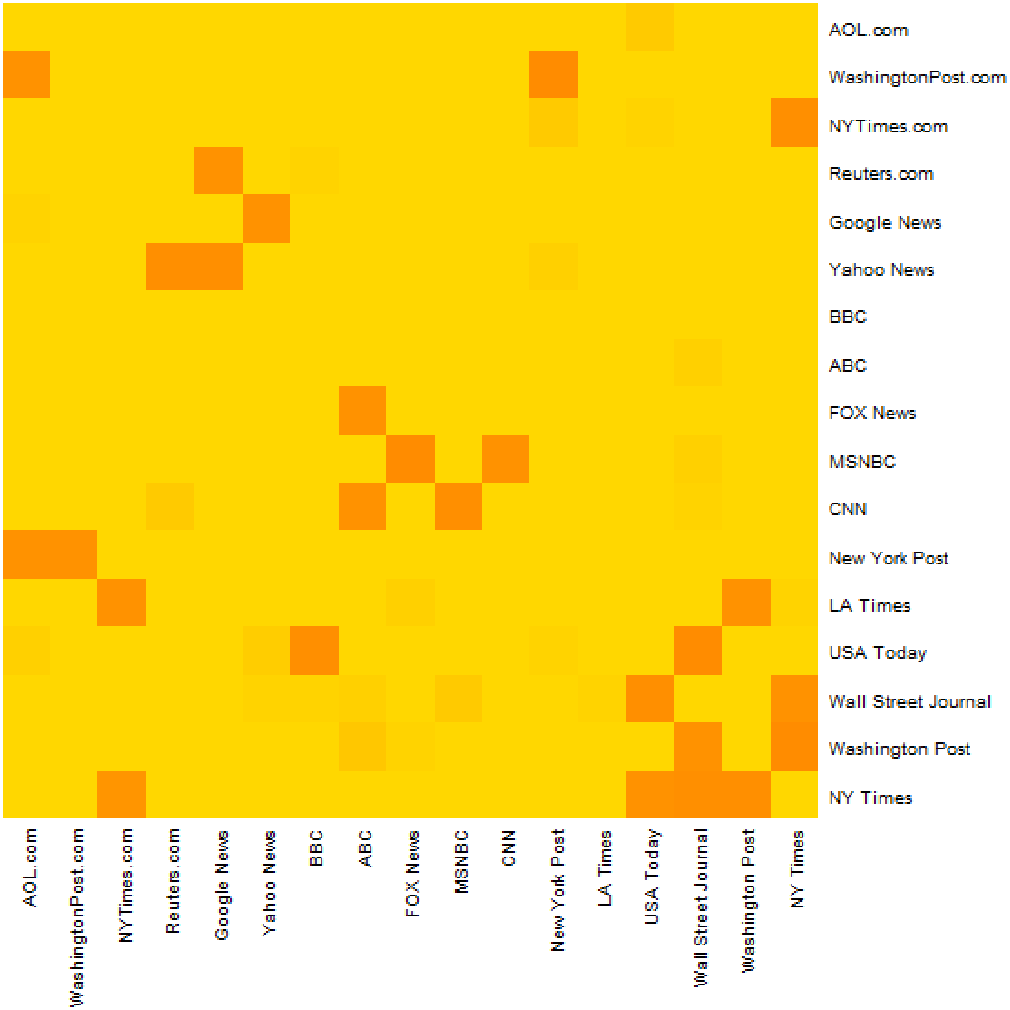
```
dd <- degree.distribution(net, cumulative=T, mode="all")
plot(dd, pch=19, cex=1, col="orange", xlab="Degree", ylab="Cumulative
Frequency")
```



Cumulative Frequency
1.0
0.8
0.6
0.4
0.2
2
4
6
8
10
12
14
Degree

热力图

```
netm <- get.adjacency(net, attr="weight", sparse=F)
colnames(netm) <- V(net)$media
rownames(netm) <- V(net)$media

palf <- colorRampPalette(c("gold", "dark orange"))
heatmap(netm[,17:1], Rowv = NA, Colv = NA, col = palf(100),
        scale="none", margins=c(10,10) )
```


AOL.com
WashingtonPost.com
NYTimes.com
Reuters.com
Google News
Yahoo News
BBC
ABC
FOX News
MSNBC
CNN
New York Post
LA Times
USA Today
Wall Street Journal
Washington Post
NY Times
AOL.com
WashingtonPost.com
NYTimes.com
Reuters.com
Google News
Yahoo News
BBC
ABC
FOX News
MSNBC
CNN
New York Post
LA Times
USA Today
Wall Street Journal
Washington Post
NY Times

高级篇：用图形表示节点

```
library(png)
img.1 <- readPNG("./data/news.png")
## Error in readPNG("./data/news.png"): unable to open ./data/news.png
img.2 <- readPNG("./data/user.png")
## Error in readPNG("./data/user.png"): unable to open ./data/user.png

V(net2)$raster <- list(img.1, img.2)[V(net2)$type+1]
## Error in eval(expr, envir, enclos): object 'img.1' not found

plot(net2, vertex.shape="raster", vertex.label=NA,
     vertex.size=16, vertex.size2=16, edge.width=2)
## Error in plot(net2, vertex.shape = "raster", vertex.label =
NA, vertex.size = 16, : object 'net2' not found
```