```r
library(igraph)
```

```
## Warning: package 'igraph' was built under R version 4.0.3

##
## Attaching package: 'igraph'

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

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

```r
library(data.table)
```

```
## Warning: package 'data.table' was built under R version 4.0.3
```

```r
library(readr)
library(tidyr)
```

```
##
## Attaching package: 'tidyr'

## The following object is masked from 'package:igraph':
##
##     crossing
```

```r
library(igraph)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:data.table':
##
##     between, first, last

## The following objects are masked from 'package:igraph':
##
##     as_data_frame, groups, union

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(matrixcalc)
```

```
## Warning: package 'matrixcalc' was built under R version 4.0.3

##
## Attaching package: 'matrixcalc'

## The following object is masked from 'package:igraph':
##
##     %s%
```

```r
library(rlist)
```

```
## Warning: package 'rlist' was built under R version 4.0.3
```

```r
library(sets)
```

```
## Warning: package 'sets' was built under R version 4.0.3

##
## Attaching package: 'sets'

## The following object is masked from 'package:dplyr':
##
##     %>%

## The following object is masked from 'package:tidyr':
##
##     %>%

## The following object is masked from 'package:data.table':
##
##     set

## The following object is masked from 'package:igraph':
##
##     %>%
```

```r
df <- read.csv("C:/Users/10331/OneDrive/Desktop/social_and_task_network.csv")
```

```r
social <- df[,-4]
task <- df[-3 ]
social
```

```
##     ego alter social_tie
## 1     1     1      0.000
## 2     1     2      0.000
## 3     1     3      0.000
## 4     1     4      0.000
```

```
## 5      1     5       5.625
## 6      1     6       1.500
## 7      1     7       0.000
## 8      1     8       0.000
## 9      1     9       0.000
## 10     1    10       0.000
## 11     1    11       0.000
## 12     1    12       0.000
## 13     1    13       0.000
## 14     1    14       0.000
## 15     1    15       0.000
## 16     1    16       0.000
## 17     1    17       0.000
## 18     1    18       0.000
## 19     1    19       0.000
## 20     1    20       0.000
## 21     1    21       0.000
## 22     1    22       1.875
## 23     2     1       0.000
## 24     2     2       0.000
## 25     2     3       0.000
## 26     2     4       0.000
## 27     2     5       0.000
## 28     2     6       0.000
## 29     2     7       0.000
## 30     2     8       0.000
## 31     2     9       0.000
## 32     2    10       0.000
## 33     2    11       0.000
## 34     2    12       0.000
## 35     2    13       0.000
## 36     2    14       0.000
## 37     2    15       0.000
## 38     2    16       0.000
## 39     2    17       0.000
## 40     2    18       0.000
## 41     2    19       0.000
## 42     2    20       0.000
## 43     2    21       0.000
## 44     2    22       0.375
## 45     3     1       0.000
## 46     3     2       0.000
## 47     3     3       0.000
## 48     3     4       0.000
## 49     3     5       0.000
## 50     3     6       0.000
## 51     3     7       0.000
## 52     3     8       0.000
## 53     3     9       0.000
## 54     3    10       0.000
## 55     3    11       0.000
## 56     3    12       0.000
## 57     3    13       0.000
## 58     3    14       0.000
```

```
## 59    3   15      0.000
## 60    3   16      0.000
## 61    3   17      0.000
## 62    3   18      0.000
## 63    3   19      0.000
## 64    3   20      0.000
## 65    3   21      0.000
## 66    3   22      0.000
## 67    4    1      0.000
## 68    4    2      0.000
## 69    4    3      0.000
## 70    4    4      0.000
## 71    4    5      0.000
## 72    4    6      0.000
## 73    4    7      0.000
## 74    4    8      1.875
## 75    4    9      0.000
## 76    4   10      0.000
## 77    4   11      0.000
## 78    4   12      0.000
## 79    4   13      0.000
## 80    4   14      0.000
## 81    4   15      0.000
## 82    4   16      0.000
## 83    4   17      0.000
## 84    4   18      0.000
## 85    4   19      0.000
## 86    4   20      0.000
## 87    4   21      0.000
## 88    4   22      0.000
## 89    5    1      5.250
## 90    5    2      0.000
## 91    5    3      0.000
## 92    5    4      0.000
## 93    5    5      0.000
## 94    5    6      1.500
## 95    5    7      0.000
## 96    5    8      0.000
## 97    5    9      0.000
## 98    5   10      0.000
## 99    5   11      0.000
## 100   5   12      0.000
## 101   5   13      0.000
## 102   5   14      0.000
## 103   5   15      0.000
## 104   5   16      0.000
## 105   5   17      0.000
## 106   5   18      0.000
## 107   5   19      0.000
## 108   5   20      0.000
## 109   5   21      0.000
## 110   5   22      0.750
## 111   6    1      1.125
## 112   6    2      0.000
```

```
## 113   6     3       0.000
## 114   6     4       0.000
## 115   6     5       1.500
## 116   6     6       0.000
## 117   6     7       0.000
## 118   6     8       0.000
## 119   6     9       0.375
## 120   6    10       0.000
## 121   6    11       0.000
## 122   6    12       0.000
## 123   6    13       0.000
## 124   6    14       0.000
## 125   6    15       0.000
## 126   6    16       0.000
## 127   6    17       0.000
## 128   6    18       0.000
## 129   6    19       0.000
## 130   6    20       0.000
## 131   6    21       0.000
## 132   6    22       0.000
## 133   7     1       0.000
## 134   7     2       0.000
## 135   7     3       0.000
## 136   7     4       0.000
## 137   7     5       0.000
## 138   7     6       0.000
## 139   7     7       0.000
## 140   7     8       0.000
## 141   7     9       0.000
## 142   7    10       1.875
## 143   7    11       0.000
## 144   7    12       0.000
## 145   7    13       0.000
## 146   7    14       0.000
## 147   7    15       0.000
## 148   7    16       0.000
## 149   7    17       0.000
## 150   7    18       0.000
## 151   7    19       0.000
## 152   7    20       0.000
## 153   7    21       0.000
## 154   7    22       0.000
## 155   8     1       0.000
## 156   8     2       0.000
## 157   8     3       0.000
## 158   8     4       1.875
## 159   8     5       0.000
## 160   8     6       0.000
## 161   8     7       0.000
## 162   8     8       0.000
## 163   8     9       0.000
## 164   8    10       0.000
## 165   8    11       0.000
## 166   8    12       0.000
```

```
## 167   8    13       0.000
## 168   8    14       0.000
## 169   8    15       0.000
## 170   8    16       0.000
## 171   8    17       0.000
## 172   8    18       0.000
## 173   8    19       0.000
## 174   8    20       0.000
## 175   8    21       0.000
## 176   8    22       0.000
## 177   9     1       0.000
## 178   9     2       0.000
## 179   9     3       0.000
## 180   9     4       0.000
## 181   9     5       0.000
## 182   9     6       0.375
## 183   9     7       0.000
## 184   9     8       0.000
## 185   9     9       0.000
## 186   9    10       0.000
## 187   9    11       0.000
## 188   9    12       0.000
## 189   9    13       0.000
## 190   9    14       0.000
## 191   9    15       0.000
## 192   9    16       0.000
## 193   9    17       0.000
## 194   9    18       0.000
## 195   9    19       0.000
## 196   9    20       0.000
## 197   9    21       0.000
## 198   9    22       0.000
## 199  10     1       0.000
## 200  10     2       0.000
## 201  10     3       0.000
## 202  10     4       0.000
## 203  10     5       0.000
## 204  10     6       0.000
## 205  10     7       2.250
## 206  10     8       0.000
## 207  10     9       0.000
## 208  10    10       0.000
## 209  10    11       0.000
## 210  10    12       0.750
## 211  10    13       0.000
## 212  10    14       0.000
## 213  10    15       0.000
## 214  10    16       0.000
## 215  10    17       0.000
## 216  10    18       0.000
## 217  10    19       0.000
## 218  10    20       0.000
## 219  10    21       0.000
## 220  10    22       0.000
```

```
## 221  11     1       0.000
## 222  11     2       0.000
## 223  11     3       0.000
## 224  11     4       0.000
## 225  11     5       0.000
## 226  11     6       0.000
## 227  11     7       0.000
## 228  11     8       0.000
## 229  11     9       0.000
## 230  11    10       0.000
## 231  11    11       0.000
## 232  11    12       0.000
## 233  11    13       0.000
## 234  11    14       0.000
## 235  11    15       3.000
## 236  11    16       0.000
## 237  11    17       0.000
## 238  11    18       0.000
## 239  11    19       0.000
## 240  11    20       0.000
## 241  11    21       0.000
## 242  11    22       0.000
## 243  12     1       0.000
## 244  12     2       0.000
## 245  12     3       0.000
## 246  12     4       0.000
## 247  12     5       0.000
## 248  12     6       0.000
## 249  12     7       0.000
## 250  12     8       0.000
## 251  12     9       0.000
## 252  12    10       0.750
## 253  12    11       0.000
## 254  12    12       0.000
## 255  12    13       0.000
## 256  12    14       0.000
## 257  12    15       0.000
## 258  12    16       0.375
## 259  12    17       0.000
## 260  12    18       0.375
## 261  12    19       0.000
## 262  12    20       0.000
## 263  12    21       0.000
## 264  12    22       0.000
## 265  13     1       0.000
## 266  13     2       0.000
## 267  13     3       0.000
## 268  13     4       0.000
## 269  13     5       0.000
## 270  13     6       0.000
## 271  13     7       0.000
## 272  13     8       0.000
## 273  13     9       0.000
## 274  13    10       0.000
```

```
## 275  13    11       0.000
## 276  13    12       0.000
## 277  13    13       0.000
## 278  13    14       0.000
## 279  13    15       0.000
## 280  13    16       0.000
## 281  13    17       0.000
## 282  13    18       0.000
## 283  13    19       0.000
## 284  13    20       0.000
## 285  13    21       0.000
## 286  13    22       0.000
## 287  14     1       0.000
## 288  14     2       0.000
## 289  14     3       0.000
## 290  14     4       0.000
## 291  14     5       0.000
## 292  14     6       0.000
## 293  14     7       0.000
## 294  14     8       0.000
## 295  14     9       0.000
## 296  14    10       0.000
## 297  14    11       0.000
## 298  14    12       0.000
## 299  14    13       0.000
## 300  14    14       0.000
## 301  14    15       0.000
## 302  14    16       0.000
## 303  14    17       0.000
## 304  14    18       0.000
## 305  14    19       0.000
## 306  14    20       0.000
## 307  14    21       0.000
## 308  14    22       0.000
## 309  15     1       0.000
## 310  15     2       0.000
## 311  15     3       0.000
## 312  15     4       0.000
## 313  15     5       0.000
## 314  15     6       0.000
## 315  15     7       0.000
## 316  15     8       0.000
## 317  15     9       0.000
## 318  15    10       0.000
## 319  15    11       3.000
## 320  15    12       0.000
## 321  15    13       0.000
## 322  15    14       0.000
## 323  15    15       0.000
## 324  15    16       0.000
## 325  15    17       0.000
## 326  15    18       0.000
## 327  15    19       0.000
## 328  15    20       0.000
```

```
## 329  15    21       0.000
## 330  15    22       0.000
## 331  16     1       0.000
## 332  16     2       0.000
## 333  16     3       0.000
## 334  16     4       0.000
## 335  16     5       0.000
## 336  16     6       0.000
## 337  16     7       0.000
## 338  16     8       0.000
## 339  16     9       0.000
## 340  16    10       0.000
## 341  16    11       0.000
## 342  16    12       0.375
## 343  16    13       0.000
## 344  16    14       0.000
## 345  16    15       0.000
## 346  16    16       0.000
## 347  16    17       0.750
## 348  16    18       0.375
## 349  16    19       5.250
## 350  16    20       0.000
## 351  16    21       0.000
## 352  16    22       0.750
## 353  17     1       0.000
## 354  17     2       0.000
## 355  17     3       0.000
## 356  17     4       0.000
## 357  17     5       0.000
## 358  17     6       0.000
## 359  17     7       0.000
## 360  17     8       0.000
## 361  17     9       0.000
## 362  17    10       0.000
## 363  17    11       0.000
## 364  17    12       0.000
## 365  17    13       0.000
## 366  17    14       0.000
## 367  17    15       0.000
## 368  17    16       0.750
## 369  17    17       0.000
## 370  17    18       1.125
## 371  17    19       1.125
## 372  17    20       0.000
## 373  17    21       0.375
## 374  17    22       0.000
## 375  18     1       0.000
## 376  18     2       0.000
## 377  18     3       0.000
## 378  18     4       0.000
## 379  18     5       0.000
## 380  18     6       0.000
## 381  18     7       0.000
## 382  18     8       0.000
```

```
## 383  18    9      0.000
## 384  18   10      0.000
## 385  18   11      0.000
## 386  18   12      0.375
## 387  18   13      0.000
## 388  18   14      0.000
## 389  18   15      0.000
## 390  18   16      0.375
## 391  18   17      1.125
## 392  18   18      0.000
## 393  18   19      2.250
## 394  18   20      1.125
## 395  18   21     14.625
## 396  18   22      0.000
## 397  19    1      0.000
## 398  19    2      0.000
## 399  19    3      0.000
## 400  19    4      0.000
## 401  19    5      0.000
## 402  19    6      0.000
## 403  19    7      0.000
## 404  19    8      0.000
## 405  19    9      0.000
## 406  19   10      0.000
## 407  19   11      0.000
## 408  19   12      0.000
## 409  19   13      0.000
## 410  19   14      0.000
## 411  19   15      0.000
## 412  19   16      5.250
## 413  19   17      1.125
## 414  19   18      2.250
## 415  19   19      0.000
## 416  19   20      1.875
## 417  19   21      0.375
## 418  19   22      0.375
## 419  20    1      0.000
## 420  20    2      0.000
## 421  20    3      0.000
## 422  20    4      0.000
## 423  20    5      0.000
## 424  20    6      0.000
## 425  20    7      0.000
## 426  20    8      0.000
## 427  20    9      0.000
## 428  20   10      0.000
## 429  20   11      0.000
## 430  20   12      0.000
## 431  20   13      0.000
## 432  20   14      0.000
## 433  20   15      0.000
## 434  20   16      0.000
## 435  20   17      0.000
## 436  20   18      1.125
```

```
## 437  20    19       1.875
## 438  20    20       0.000
## 439  20    21       0.750
## 440  20    22       0.000
## 441  21     1       0.000
## 442  21     2       0.000
## 443  21     3       0.000
## 444  21     4       0.000
## 445  21     5       0.000
## 446  21     6       0.000
## 447  21     7       0.000
## 448  21     8       0.000
## 449  21     9       0.000
## 450  21    10       0.000
## 451  21    11       0.000
## 452  21    12       0.000
## 453  21    13       0.000
## 454  21    14       0.000
## 455  21    15       0.000
## 456  21    16       0.000
## 457  21    17       0.375
## 458  21    18      14.625
## 459  21    19       0.000
## 460  21    20       0.750
## 461  21    21       0.000
## 462  21    22       1.500
## 463  22     1       0.750
## 464  22     2       0.375
## 465  22     3       0.000
## 466  22     4       0.000
## 467  22     5       0.750
## 468  22     6       0.000
## 469  22     7       0.000
## 470  22     8       0.000
## 471  22     9       0.000
## 472  22    10       0.000
## 473  22    11       0.375
## 474  22    12       0.000
## 475  22    13       0.000
## 476  22    14       0.000
## 477  22    15       0.000
## 478  22    16       0.750
## 479  22    17       0.000
## 480  22    18       0.375
## 481  22    19       1.125
## 482  22    20       0.000
## 483  22    21       0.375
## 484  22    22       0.000
```

```r
#question1
#(a:social)
print("Social")
```

```
## [1] "Social"
```

```r
social <- social[social$social_tie != 0,]
row.names(social)=NULL
gs <- graph.data.frame(social,directed = TRUE)
gs
```

```
## IGRAPH 396c138 DN-- 19 57 --
## + attr: name (v/c), social_tie (e/n)
## + edges from 396c138 (vertex names):
##  [1] 1 ->5  1 ->6  1 ->22 2 ->22 4 ->8  5 ->1  5 ->6  5 ->22 6 ->1  6 ->5
## [11] 6 ->9  7 ->10 8 ->4  9 ->6  10->7  10->12 11->15 12->10 12->16 12->18
## [21] 15->11 16->12 16->17 16->18 16->19 16->22 17->16 17->18 17->19 17->21
## [31] 18->12 18->16 18->17 18->19 18->20 18->21 19->16 19->17 19->18 19->20
## [41] 19->21 19->22 20->18 20->19 20->21 21->17 21->18 21->20 21->22 22->1
## [51] 22->2  22->5  22->11 22->16 22->18 22->19 22->21
```

```r
print("Social")
```

```
## [1] "Social"
```

```r
dins <- degree(gs,v = V(gs),mode = "in")
douts <- degree(gs,mode = "out")
print("in")
```

```
## [1] "in"
```

```r
dins
```

```
##  1  2  4  5  6  7  8  9 10 11 12 15 16 17 18 19 20 21 22
##  3  1  1  3  3  1  1  1  2  2  3  1  5  4  7  5  3  5  6
```

```r
print("out")
```

```
## [1] "out"
```

```r
douts
```

```
##  1  2  4  5  6  7  8  9 10 11 12 15 16 17 18 19 20 21 22
##  3  1  1  3  3  1  1  1  2  1  3  1  5  4  6  6  3  4  8
```

```r
closes <- closeness(gs,mode = "all")
```

```
## Warning in closeness(gs, mode = "all"): At centrality.c:2784 :closeness
## centrality is not well-defined for disconnected graphs
```

```r
print("Social")
```

```
## [1] "Social"
```

```r
closes
```

```
##           1           2           4           5           6           7 
## 0.013157895 0.012345679 0.003086420 0.013157895 0.011363636 0.009433962 
##           8           9          10          11          12          15 
## 0.003086420 0.009708738 0.010989011 0.012658228 0.012820513 0.010638298 
##          16          17          18          19          20          21 
## 0.014285714 0.012500000 0.014705882 0.014084507 0.012345679 0.013888889 
##          22 
## 0.015151515
```

```r
print("Social")
```

```
## [1] "Social"
```

```r
betws <- betweenness(gs)
betws
```

```
##           1           2           4           5           6           7 
##  24.0000000   0.0000000   0.0000000  24.0000000  28.0000000   0.0000000 
##           8           9          10          11          12          15 
##   0.0000000   0.0000000  28.0000000  15.0000000  52.0000000   0.0000000 
##          16          17          18          19          20          21 
##  45.8333333   0.8333333  33.0000000  14.7500000   0.2500000  13.5000000 
##          22 
## 126.8333333
```

```r
print("Social")
```

```
## [1] "Social"
```

```r
prs <- page.rank(gs)
prs$vector
```

```
##          1          2          4          5          6          7          8 
## 0.04276027 0.01615060 0.05263158 0.04276027 0.05115620 0.02693356 0.05263158 
##          9         10         11         12         15         16         17 
## 0.02238899 0.04479722 0.08238245 0.04944339 0.07791982 0.06184064 0.05174036 
##         18         19         20         21         22 
## 0.08468420 0.06133638 0.04122745 0.05951279 0.07770226
```

```r
#(b:social)
print("Social")
```

```
## [1] "Social"
```

```r
print("cor:indegree-outdegree")
```

```
## [1] "cor:indegree-outdegree"
```

```r
cor(dins,douts)
```

```
## [1] 0.948282
```

```r
print("cor:indegree-closeness")
```

```
## [1] "cor:indegree-closeness"
```

```r
cor(dins,closes)
```

```
## [1] 0.7079371
```

```r
print("cor:indegree-betweenness")
```

```
## [1] "cor:indegree-betweenness"
```

```r
cor(dins,betws)
```

```
## [1] 0.5991529
```

```r
print("cor:indegree-page.rank")
```

```
## [1] "cor:indegree-page.rank"
```

```r
cor(dins,prs$vector)
```

```
## [1] 0.5568503
```

```r
print("cor:outdegree-closeness")
```

```
## [1] "cor:outdegree-closeness"
```

```r
cor(douts,closes)
```

```
## [1] 0.6660651
```

```r
print("cor:outdegree-betweenness")
```

```
## [1] "cor:outdegree-betweenness"
```

```r
cor(douts,betws)
```

```
## [1] 0.7242923
```

```r
print("cor:outdegree-page.rank")
```

```
## [1] "cor:outdegree-page.rank"
```

```r
cor(douts,prs$vector)
```

```
## [1] 0.4907911
```

```r
print("cor:closeness-betweenness")
```

```
## [1] "cor:closeness-betweenness"
```

```r
cor(closes,betws)
```

```
## [1] 0.4852587
```

```r
print("cor:closeness-page.rank")
```

```
## [1] "cor:closeness-page.rank"
```

```r
cor(closes,prs$vector)
```

```
## [1] 0.2602387
```

```r
print("cor:betweenness-page.rank")
```

```
## [1] "cor:betweenness-page.rank"
```

```r
cor(betws,prs$vector)
```

```
## [1] 0.4189853
```

From the above, we can see that the most closely correlated measures are indegree and outdegree. Which means for most nodes, the social relationships are reciprocated.

```r
#(a:task)
print("Task")
```

```
## [1] "Task"
```

```r
task <- task[task$task_tie != 0,]
row.names(task)=NULL
gt <- graph.data.frame(task,directed = TRUE)
gt
```

```
## IGRAPH 3982342 DN-- 20 48 --
## + attr: name (v/c), task_tie (e/n)
## + edges from 3982342 (vertex names):
##  [1] 1 ->22 2 ->22 4 ->8  5 ->22 6 ->22 7 ->22 8 ->4  9 ->22 10->22 11->22
## [11] 13->18 13->22 14->22 15->22 16->19 16->22 17->18 17->21 17->22 18->13
## [21] 18->17 18->21 18->22 19->16 19->20 19->22 20->19 20->22 21->17 21->18
## [31] 21->22 22->1  22->2  22->5  22->6  22->7  22->9  22->10 22->11 22->13
## [41] 22->14 22->15 22->16 22->17 22->18 22->19 22->20 22->21
```

```
print("Task")
```

```
## [1] "Task"
```

```
dint <- degree(gt,v = V(gt),mode = "in")
doutt <- degree(gt,mode = "out")
print("in")
```

```
## [1] "in"
```

```
dint
```

```
##  1  2  4  5  6  7  8  9 10 11 13 14 15 16 17 18 19 20 21 22
##  1  1  1  1  1  1  1  1  1  1  2  1  1  2  3  4  3  2  3 17
```

```
print("out")
```

```
## [1] "out"
```

```
doutt
```

```
##  1  2  4  5  6  7  8  9 10 11 13 14 15 16 17 18 19 20 21 22
##  1  1  1  1  1  1  1  1  1  1  2  1  1  2  3  4  3  2  3 17
```

```
closet <- closeness(gt,mode = "all")
```

```
## Warning in closeness(gt, mode = "all"): At centrality.c:2784 :closeness
## centrality is not well-defined for disconnected graphs
```

```
print("Task")
```

```
## [1] "Task"
```

```
closet
```

```
##           1           2           4           5           6           7
## 0.013698630 0.013698630 0.002770083 0.013698630 0.013698630 0.013698630
##           8           9          10          11          13          14
## 0.002770083 0.013698630 0.013698630 0.013698630 0.013888889 0.013698630
##          15          16          17          18          19          20
## 0.013698630 0.013888889 0.014084507 0.014285714 0.014084507 0.013888889
##          21          22
## 0.014084507 0.017543860
```

```r
print("Task")
```

```
## [1] "Task"
```

```r
betwt <- betweenness(gt)
betwt
```

```
##   1   2   4   5   6   7   8   9  10  11  13  14  15  16  17  18  19  20  21  22
##   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   2   1   0   0 257
```

```r
print("Task")
```

```
## [1] "Task"
```

```r
prt <- page.rank(gt)
prt$vector
```

```
##          1          2          4          5          6          7          8
## 0.02333208 0.02333208 0.05000000 0.02333208 0.02333208 0.02333208 0.05000000
##          9         10         11         13         14         15         16
## 0.02333208 0.02333208 0.02333208 0.03814414 0.02333208 0.02333208 0.03944171
##         17         18         19         20         21         22
## 0.05322438 0.06970382 0.05685753 0.03944171 0.05322438 0.31664155
```

```r
#(b:Task)
print("Task")
```

```
## [1] "Task"
```

```r
print("cor:indegree-outdegree")
```

```
## [1] "cor:indegree-outdegree"
```

```r
cor(dint,doutt)
```

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

```r
print("cor:indegree-closeness")
```

```
## [1] "cor:indegree-closeness"
```

```r
cor(dint,closet)
```

```
## [1] 0.3649698
```

```r
print("cor:indegree-betweenness")
```

```
## [1] "cor:indegree-betweenness"
```

```r
cor(dint,betwt)
```

```
## [1] 0.9668378
```

```r
print("cor:indegree-page.rank")
```

```
## [1] "cor:indegree-page.rank"
```

```r
cor(dint,prt$vector)
```

```
## [1] 0.9900638
```

```r
print("cor:outdegree-closeness")
```

```
## [1] "cor:outdegree-closeness"
```

```r
cor(doutt,closet)
```

```
## [1] 0.3649698
```

```r
print("cor:outdegree-betweenness")
```

```
## [1] "cor:outdegree-betweenness"
```

```r
cor(doutt,betwt)
```

```
## [1] 0.9668378
```

```r
print("cor:outdegree-page.rank")
```

```
## [1] "cor:outdegree-page.rank"
```

```r
cor(doutt,prt$vector)
```

```
## [1] 0.9900638
```

```r
print("cor:closeness-betweenness")
```

```
## [1] "cor:closeness-betweenness"
```

```r
cor(closet,betwt)
```

```
## [1] 0.3063969
```

```r
print("cor:closeness-page.rank")
```

```
## [1] "cor:closeness-page.rank"
```

```r
cor(closet,prt$vector)
```

```
## [1] 0.23672
```

```r
print("cor:betweenness-page.rank")
```

```
## [1] "cor:betweenness-page.rank"
```

```r
cor(betwt,prt$vector)
```

```
## [1] 0.9743543
```

From the above, we can see that indegree and outdegree are perfectly correlated. Which means for all nodes, the task relationships are reciprocated.

```r
#Correlation Comparision of two table
print("Social and Task")
```

```
## [1] "Social and Task"
```

```r
print("cor:indegree")
```

```
## [1] "cor:indegree"
```

```r
#cor(dins,dint)
print("0.5578869")
```

```
## [1] "0.5578869"
```

```r
print("cor:outdegree")
```

```
## [1] "cor:outdegree"
```

```r
#cor(douts,doutt)
print("0.6996636")
```

```
## [1] "0.6996636"
```

```r
print("cor:closeness")
```

```
## [1] "cor:closeness"
```

```r
#cor(closes,closet)
print("0.4132661")
```

```
## [1] "0.4132661"
```

```r
print("cor:betweennesspage.rank")
```

```
## [1] "cor:betweennesspage.rank"
```

```r
#cor(betws,betwt)
print("0.7516857")
```

```
## [1] "0.7516857"
```

```r
print("cor:page.rank")
```

```
## [1] "cor:page.rank"
```

```r
#cor(prs$vector,prt$vector)
print("0.1730575")
```

```
## [1] "0.1730575"
```

```r
#THose code works on my local machine, but when i create HTML with KNIT, it fails so I print out the ou
```

The highest correlation existing between two betweenness scores, Which makes sense too. The people that are information bridge could be bridge in both social and task relationships. And these people performs as coordinator or gateholder.

```r
#Question2
#(a)
social$type <- "Social"
colnames(social)[3] = "Tie"
means = mean(social$Tie)
for (i in 1:nrow(social)){
  if (social[i,"Tie"]>means){
    social[i,"Strength"] = "Strong"
  } else{
    social[i,"Strength"] = "Weak"
  }
}
task$type <- "Task"
colnames(task)[3] = "Tie"
meant = mean(task$Tie)
```

```r
for (i in 1:nrow(task)){
  if (task[i,"Tie"]>meant){
    task[i,"Strength"] = "Strong"
  } else{
    task[i,"Strength"] = "Weak"
  }
}
comb <- rbind(social,task)
row.names(comb)=NULL
g <- graph.data.frame(comb)
comb
```

```
##     ego alter    Tie   type Strength
## 1     1     5  5.625 Social   Strong
## 2     1     6  1.500 Social     Weak
## 3     1    22  1.875 Social   Strong
## 4     2    22  0.375 Social     Weak
## 5     4     8  1.875 Social   Strong
## 6     5     1  5.250 Social   Strong
## 7     5     6  1.500 Social     Weak
## 8     5    22  0.750 Social     Weak
## 9     6     1  1.125 Social     Weak
## 10    6     5  1.500 Social     Weak
## 11    6     9  0.375 Social     Weak
## 12    7    10  1.875 Social   Strong
## 13    8     4  1.875 Social   Strong
## 14    9     6  0.375 Social     Weak
## 15   10     7  2.250 Social   Strong
## 16   10    12  0.750 Social     Weak
## 17   11    15  3.000 Social   Strong
## 18   12    10  0.750 Social     Weak
## 19   12    16  0.375 Social     Weak
## 20   12    18  0.375 Social     Weak
## 21   15    11  3.000 Social   Strong
## 22   16    12  0.375 Social     Weak
## 23   16    17  0.750 Social     Weak
## 24   16    18  0.375 Social     Weak
## 25   16    19  5.250 Social   Strong
## 26   16    22  0.750 Social     Weak
## 27   17    16  0.750 Social     Weak
## 28   17    18  1.125 Social     Weak
## 29   17    19  1.125 Social     Weak
## 30   17    21  0.375 Social     Weak
## 31   18    12  0.375 Social     Weak
## 32   18    16  0.375 Social     Weak
## 33   18    17  1.125 Social     Weak
## 34   18    19  2.250 Social   Strong
## 35   18    20  1.125 Social     Weak
## 36   18    21 14.625 Social   Strong
## 37   19    16  5.250 Social   Strong
## 38   19    17  1.125 Social     Weak
## 39   19    18  2.250 Social   Strong
## 40   19    20  1.875 Social   Strong
```

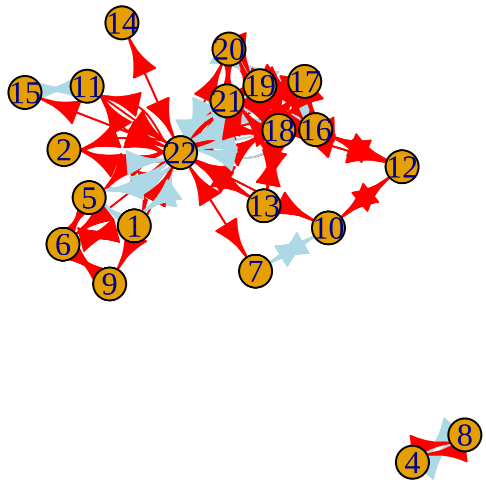
```
## 41   19    21  0.375 Social     Weak
## 42   19    22  0.375 Social     Weak
## 43   20    18  1.125 Social     Weak
## 44   20    19  1.875 Social   Strong
## 45   20    21  0.750 Social     Weak
## 46   21    17  0.375 Social     Weak
## 47   21    18 14.625 Social   Strong
## 48   21    20  0.750 Social     Weak
## 49   21    22  1.500 Social     Weak
## 50   22     1  0.750 Social     Weak
## 51   22     2  0.375 Social     Weak
## 52   22     5  0.750 Social     Weak
## 53   22    11  0.375 Social     Weak
## 54   22    16  0.750 Social     Weak
## 55   22    18  0.375 Social     Weak
## 56   22    19  1.125 Social     Weak
## 57   22    21  0.375 Social     Weak
## 58    1    22 11.250   Task   Strong
## 59    2    22  2.250   Task     Weak
## 60    4     8  0.750   Task     Weak
## 61    5    22  7.125   Task   Strong
## 62    6    22  5.250   Task   Strong
## 63    7    22  1.125   Task     Weak
## 64    8     4  0.750   Task     Weak
## 65    9    22  2.250   Task     Weak
## 66   10    22  1.125   Task     Weak
## 67   11    22  2.625   Task     Weak
## 68   13    18  0.750   Task     Weak
## 69   13    22  0.750   Task     Weak
## 70   14    22  0.750   Task     Weak
## 71   15    22  2.250   Task     Weak
## 72   16    19  1.125   Task     Weak
## 73   16    22 10.500   Task   Strong
## 74   17    18  0.375   Task     Weak
## 75   17    21  0.375   Task     Weak
## 76   17    22  1.125   Task     Weak
## 77   18    13  0.750   Task     Weak
## 78   18    17  0.375   Task     Weak
## 79   18    21  1.125   Task     Weak
## 80   18    22  3.375   Task   Strong
## 81   19    16  1.125   Task     Weak
## 82   19    20  0.375   Task     Weak
## 83   19    22 10.125   Task   Strong
## 84   20    19  0.375   Task     Weak
## 85   20    22  3.375   Task   Strong
## 86   21    17  0.375   Task     Weak
## 87   21    18  1.125   Task     Weak
## 88   21    22 11.625   Task   Strong
## 89   22     1  7.500   Task   Strong
## 90   22     2  1.125   Task     Weak
## 91   22     5  6.375   Task   Strong
## 92   22     6  2.250   Task     Weak
## 93   22     7  1.500   Task     Weak
## 94   22     9  0.750   Task     Weak
```

```
## 95   22    10  1.125   Task     Weak
## 96   22    11  0.750   Task     Weak
## 97   22    13  1.500   Task     Weak
## 98   22    14  0.375   Task     Weak
## 99   22    15  1.500   Task     Weak
## 100  22    16  2.625   Task     Weak
## 101  22    17  1.125   Task     Weak
## 102  22    18  1.875   Task     Weak
## 103  22    19  3.000   Task   Strong
## 104  22    20  0.750   Task     Weak
## 105  22    21  5.625   Task   Strong
```

```
gc <- graph.data.frame(comb,directed = TRUE)
gc
```

```
## IGRAPH 399fa59 DN-- 21 105 --
## + attr: name (v/c), Tie (e/n), type (e/c), Strength (e/c)
## + edges from 399fa59 (vertex names):
##  [1] 1 ->5  1 ->6  1 ->22 2 ->22 4 ->8  5 ->1  5 ->6  5 ->22 6 ->1  6 ->5
## [11] 6 ->9  7 ->10 8 ->4  9 ->6  10->7  10->12 11->15 12->10 12->16 12->18
## [21] 15->11 16->12 16->17 16->18 16->19 16->22 17->16 17->18 17->19 17->21
## [31] 18->12 18->16 18->17 18->19 18->20 18->21 19->16 19->17 19->18 19->20
## [41] 19->21 19->22 20->18 20->19 20->21 21->17 21->18 21->20 21->22 22->1
## [51] 22->2  22->5  22->11 22->16 22->18 22->19 22->21 1 ->22 2 ->22 4 ->8
## [61] 5 ->22 6 ->22 7 ->22 8 ->4  9 ->22 10->22 11->22 13->18 13->22 14->22
## [71] 15->22 16->19 16->22 17->18 17->21 17->22 18->13 18->17 18->21 18->22
## + ... omitted several edges
```

```
plot(gc, edge.color = c("light blue","red")[as.factor(E(gc)$Strength)])
```




```
ties <-matrix(ncol = 2)
strong = comb[comb$Strength=='Strong',]
unique_ego = unique(strong$ego)
unique_alter = unique(strong$alter)
strong_list = unique(do.call(c, list(unique_ego, unique_alter)))
```

```
for (i in 1:length(strong_list)){
  tmp = strong[strong$ego==strong_list[i], ]
  if (length(unique(tmp$alter))>1){
    tmp1 <-combn(unique(tmp$alter),m=2)
    for (j in 1:ncol(tmp1) ){
      v1=tmp1[1,j]
      v2 = tmp1[2,j]
      ties <-rbind(c(v1,v2), ties)
      ties <-rbind(c(v2,v1), ties)
    }
  }
}
```

```
nodes = unique(df$ego)
ties=na.omit(ties)
ties= ties[!duplicated(ties),]
final <- graph.data.frame(ties, vertices = nodes, directed = TRUE)

E(difference(final,g, byname = TRUE))
```

```
## + 11/11 edges from 39baa7a (vertex names):
##  [1] 1 ->21 1 ->19 5 ->21 5 ->19 16->20 19->5  19->1  20->16 21->19 21->5
## [11] 21->1
```

There are 15 nodes involved holds strong ties and could form Triadic Closure. However, there's 11 violations of Strong Triadic Closure, under the definition of mean.

```r
#(b)
social1 = social[,]
medians = median(social1$Tie)
for (i in 1:nrow(social1)){
  if (social1[i,"Tie"]>medians){
    social1[i,"Strength"] = "Strong"
  } else{
    social1[i,"Strength"] = "Weak"
  }
}
task1 <- task[,]
mediant = median(task1$Tie)
for (i in 1:nrow(task1)){
  if (task1[i,"Tie"]>mediant){
    task1[i,"Strength"] = "Strong"
  } else{
    task1[i,"Strength"] = "Weak"
  }
}
comb2 <- rbind(social1,task1)
row.names(comb2)=NULL
comb2
```

```
##     ego alter   Tie   type Strength
## 1     1     5 5.625 Social   Strong
## 2     1     6 1.500 Social   Strong
## 3     1    22 1.875 Social   Strong
## 4     2    22 0.375 Social     Weak
## 5     4     8 1.875 Social   Strong
## 6     5     1 5.250 Social   Strong
## 7     5     6 1.500 Social   Strong
## 8     5    22 0.750 Social     Weak
## 9     6     1 1.125 Social     Weak
## 10    6     5 1.500 Social   Strong
## 11    6     9 0.375 Social     Weak
## 12    7    10 1.875 Social   Strong
## 13    8     4 1.875 Social   Strong
## 14    9     6 0.375 Social     Weak
## 15   10     7 2.250 Social   Strong
## 16   10    12 0.750 Social     Weak
## 17   11    15 3.000 Social   Strong
## 18   12    10 0.750 Social     Weak
## 19   12    16 0.375 Social     Weak
## 20   12    18 0.375 Social     Weak
## 21   15    11 3.000 Social   Strong
## 22   16    12 0.375 Social     Weak
## 23   16    17 0.750 Social     Weak
```

```
## 24   16    18  0.375 Social     Weak
## 25   16    19  5.250 Social   Strong
## 26   16    22  0.750 Social     Weak
## 27   17    16  0.750 Social     Weak
## 28   17    18  1.125 Social     Weak
## 29   17    19  1.125 Social     Weak
## 30   17    21  0.375 Social     Weak
## 31   18    12  0.375 Social     Weak
## 32   18    16  0.375 Social     Weak
## 33   18    17  1.125 Social     Weak
## 34   18    19  2.250 Social   Strong
## 35   18    20  1.125 Social     Weak
## 36   18    21 14.625 Social   Strong
## 37   19    16  5.250 Social   Strong
## 38   19    17  1.125 Social     Weak
## 39   19    18  2.250 Social   Strong
## 40   19    20  1.875 Social   Strong
## 41   19    21  0.375 Social     Weak
## 42   19    22  0.375 Social     Weak
## 43   20    18  1.125 Social     Weak
## 44   20    19  1.875 Social   Strong
## 45   20    21  0.750 Social     Weak
## 46   21    17  0.375 Social     Weak
## 47   21    18 14.625 Social   Strong
## 48   21    20  0.750 Social     Weak
## 49   21    22  1.500 Social   Strong
## 50   22     1  0.750 Social     Weak
## 51   22     2  0.375 Social     Weak
## 52   22     5  0.750 Social     Weak
## 53   22    11  0.375 Social     Weak
## 54   22    16  0.750 Social     Weak
## 55   22    18  0.375 Social     Weak
## 56   22    19  1.125 Social     Weak
## 57   22    21  0.375 Social     Weak
## 58    1    22 11.250   Task   Strong
## 59    2    22  2.250   Task   Strong
## 60    4     8  0.750   Task     Weak
## 61    5    22  7.125   Task   Strong
## 62    6    22  5.250   Task   Strong
## 63    7    22  1.125   Task     Weak
## 64    8     4  0.750   Task     Weak
## 65    9    22  2.250   Task   Strong
## 66   10    22  1.125   Task     Weak
## 67   11    22  2.625   Task   Strong
## 68   13    18  0.750   Task     Weak
## 69   13    22  0.750   Task     Weak
## 70   14    22  0.750   Task     Weak
## 71   15    22  2.250   Task   Strong
## 72   16    19  1.125   Task     Weak
## 73   16    22 10.500   Task   Strong
## 74   17    18  0.375   Task     Weak
## 75   17    21  0.375   Task     Weak
## 76   17    22  1.125   Task     Weak
## 77   18    13  0.750   Task     Weak
```

```
## 78    18     17  0.375    Task      Weak
## 79    18     21  1.125    Task      Weak
## 80    18     22  3.375    Task    Strong
## 81    19     16  1.125    Task      Weak
## 82    19     20  0.375    Task      Weak
## 83    19     22 10.125    Task    Strong
## 84    20     19  0.375    Task      Weak
## 85    20     22  3.375    Task    Strong
## 86    21     17  0.375    Task      Weak
## 87    21     18  1.125    Task      Weak
## 88    21     22 11.625    Task    Strong
## 89    22      1  7.500    Task    Strong
## 90    22      2  1.125    Task      Weak
## 91    22      5  6.375    Task    Strong
## 92    22      6  2.250    Task    Strong
## 93    22      7  1.500    Task    Strong
## 94    22      9  0.750    Task      Weak
## 95    22     10  1.125    Task      Weak
## 96    22     11  0.750    Task      Weak
## 97    22     13  1.500    Task    Strong
## 98    22     14  0.375    Task      Weak
## 99    22     15  1.500    Task    Strong
## 100   22     16  2.625    Task    Strong
## 101   22     17  1.125    Task      Weak
## 102   22     18  1.875    Task    Strong
## 103   22     19  3.000    Task    Strong
## 104   22     20  0.750    Task      Weak
## 105   22     21  5.625    Task    Strong
```

```
ties <-matrix(ncol = 2)
strong = comb2[comb2$Strength=='Strong',]
unique_ego = unique(strong$ego)
unique_alter = unique(strong$alter)
strong_list = unique(do.call(c, list(unique_ego, unique_alter)))
```

```
for (i in 1:length(strong_list)){
  tmp = strong[strong$ego==strong_list[i], ]
  if (length(unique(tmp$alter))>1){
    tmp1 <-combn(unique(tmp$alter),m=2)
    for (j in 1:ncol(tmp1) ){
      v1=tmp1[1,j]
      v2 = tmp1[2,j]
      ties <-rbind(c(v1,v2), ties)
      ties <-rbind(c(v2,v1), ties)
    }
  }
}
```

```
ties=na.omit(ties)
ties= ties[!duplicated(ties),]
final <- graph.data.frame(ties, vertices = nodes, directed = TRUE)

E(difference(final,g, byname = TRUE))
```

```
## + 75/75 edges from 39c88a8 (vertex names):
##  [1] 1 ->13 1 ->21 1 ->19 1 ->18 1 ->16 1 ->15 1 ->7  5 ->13 5 ->21 5 ->19
## [11] 5 ->18 5 ->16 5 ->15 5 ->7  6 ->13 6 ->21 6 ->19 6 ->18 6 ->16 6 ->15
## [21] 6 ->7  7 ->13 7 ->21 7 ->19 7 ->18 7 ->16 7 ->15 7 ->6  7 ->5  7 ->1
## [31] 15->13 15->21 15->19 15->18 15->16 15->7  15->6  15->5  15->1  16->13
## [41] 16->21 16->20 16->15 16->7  16->6  16->5  16->1  18->15 18->7  18->6
## [51] 18->5  18->1  19->13 19->15 19->7  19->6  19->5  19->1  20->16 21->13
## [61] 21->19 21->16 21->15 21->7  21->6  21->5  21->1  13->21 13->19 13->16
## [71] 13->15 13->7  13->6  13->5  13->1
```

There are 75 violations of Strong Triadic Closure, under the defination of median.

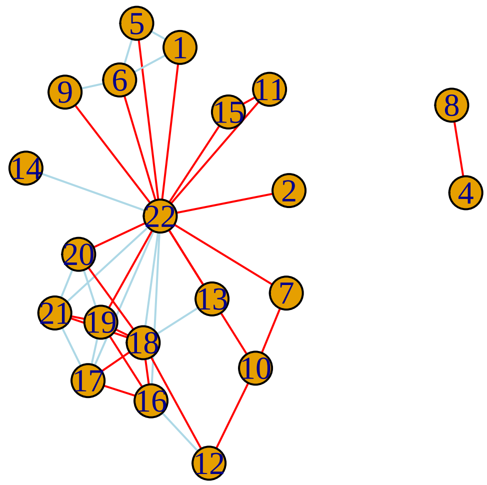
```r
comb1 <- comb[,-c(3,4)]
comb1 <- rbind(comb1[,c("ego", "alter","Strength")],comb1[,c("alter", "ego","Strength")])
comb1[comb1$Strength == "Strong","Strength"] = 1
comb1[comb1$Strength == "Weak","Strength"] = 0
comb1$Strength <- as.numeric(comb1$Strength)
comb1 <- aggregate(comb1[,3],comb1[,-3],sum)
colnames(comb1)[3] = "Strength"
for (i in nrow(comb1)){
  for (j in nrow(comb1)){
    if ((comb1[i,"ego"] == comb1[j,"alter"]) & (comb1[j,"ego"] == comb1[i,"alter"])){
      if (comb1[i,"Strength"] == "Strong"){
        comb1[i,"Strength"] = "Strong"
        comb1[j,"Strength"] = "Strong"
      }else if(comb1[j,"Strength"] == "Strong"){
        comb1[i,"Strength"] = "Strong"
        comb1[j,"Strength"] = "Strong"
      }
    }
  }
}
comb1[comb1$Strength == 2,"Strength"] = "Strong"
comb1[comb1$Strength == 1,"Strength"] = "Strong"
comb1[comb1$Strength == 0,"Strength"] = "Weak"
comb1
```

```
##    ego alter Strength
## 1    5     1   Strong
## 2    6     1     Weak
## 3   22     1   Strong
## 4   22     2     Weak
## 5    8     4   Strong
## 6    1     5   Strong
## 7    6     5     Weak
## 8   22     5   Strong
## 9    1     6     Weak
## 10   5     6     Weak
## 11   9     6     Weak
## 12  22     6     Weak
## 13  10     7   Strong
## 14  22     7     Weak
## 15   4     8   Strong
## 16   6     9     Weak
```

```
## 17  22    9    Weak
## 18   7   10  Strong
## 19  12   10    Weak
## 20  22   10    Weak
## 21  15   11  Strong
## 22  22   11    Weak
## 23  10   12    Weak
## 24  16   12    Weak
## 25  18   12    Weak
## 26  18   13    Weak
## 27  22   13    Weak
## 28  22   14    Weak
## 29  11   15  Strong
## 30  22   15    Weak
## 31  12   16    Weak
## 32  17   16    Weak
## 33  18   16    Weak
## 34  19   16  Strong
## 35  22   16    Weak
## 36  16   17    Weak
## 37  18   17    Weak
## 38  19   17    Weak
## 39  21   17    Weak
## 40  22   17    Weak
## 41  12   18    Weak
## 42  13   18    Weak
## 43  16   18    Weak
## 44  17   18    Weak
## 45  19   18  Strong
## 46  20   18    Weak
## 47  21   18  Strong
## 48  22   18    Weak
## 49  16   19  Strong
## 50  17   19    Weak
## 51  18   19  Strong
## 52  20   19  Strong
## 53  22   19  Strong
## 54  18   20    Weak
## 55  19   20  Strong
## 56  21   20    Weak
## 57  22   20    Weak
## 58  17   21    Weak
## 59  18   21  Strong
## 60  19   21    Weak
## 61  20   21    Weak
## 62  22   21  Strong
## 63   1   22       4
## 64   2   22    Weak
## 65   5   22  Strong
## 66   6   22  Strong
## 67   7   22    Weak
## 68   9   22    Weak
## 69  10   22    Weak
## 70  11   22    Weak
```

```
## 71  13    22     Weak
## 72  14    22     Weak
## 73  15    22     Weak
## 74  16    22   Strong
## 75  17    22     Weak
## 76  18    22   Strong
## 77  19    22   Strong
## 78  20    22   Strong
## 79  21    22   Strong
```

```
gc1 <- graph.data.frame(comb1,directed = FALSE)
gc1 <- simplify(gc1,remove.multiple = TRUE,remove.loops = TRUE)
plot(gc1, edge.color = c("light blue","red")[as.factor(E(gc)$Strength)])
```




```
gc1
```

```
## IGRAPH 39d1b22 UN-- 21 40 --
## + attr: name (v/c)
## + edges from 39d1b22 (vertex names):
##  [1] 5 --6  5 --22 5 --1  6 --22 6 --1  6 --9  22--1  22--9  22--10 22--7
## [11] 22--15 22--16 22--18 22--11 22--17 22--19 22--21 22--13 22--20 22--2
## [21] 22--14 8 --4  10--7  10--12 12--16 12--18 15--11 16--18 16--17 16--19
## [31] 18--17 18--19 18--21 18--13 18--20 17--19 17--21 19--21 19--20 21--20
```

The graph above shows strong tie as blue and weak tie as red. As looking at it closer, we can identify the nodes that have strong tie with two other nodes. Finding all those combinations, we can come up with a idea of Strong Triadic Closure. If the other two nodes are linked also, then it follows Strong Triadic Closure. If the other nodes are disconnected, then it's violent Strong Triadic Closure.

```
#Question3
#(a)

#Difinition1: Mean

gs <- graph.data.frame(social,directed = TRUE)
gt <- graph.data.frame(task,directed = TRUE)

betws <- betweenness(gs)
ebetws <- edge_betweenness(gs, e = E(gs), directed = TRUE)
print("Social node:")
```

```
## [1] "Social node:"
```

```
betws
```

```
##           1           2           4           5           6           7
##  24.0000000   0.0000000   0.0000000  24.0000000  28.0000000   0.0000000
##           8           9          10          11          12          15
##   0.0000000   0.0000000  28.0000000  15.0000000  52.0000000   0.0000000
##          16          17          18          19          20          21
##  45.8333333   0.8333333  33.0000000  14.7500000   0.2500000  13.5000000
##          22
## 126.8333333
```

```
print("Social edge:")
```

```
## [1] "Social edge:"
```

```
ebetws
```

```
##  [1]  1.000000 13.000000 26.000000 16.000000  1.000000  1.000000 13.000000
##  [8] 26.000000 15.000000 15.000000 14.000000 16.000000  1.000000 16.000000
## [15] 14.000000 30.000000 16.000000 26.000000 30.000000 12.000000  1.000000
## [22] 15.000000  4.250000  1.750000  3.250000 37.583333  5.500000  2.833333
## [29]  4.250000  4.250000 21.000000  4.500000  4.333333  5.416667  6.833333
## [36]  6.916667  3.000000  2.833333  2.500000  3.833333  1.250000 17.333333
## [43]  4.833333  6.083333  5.333333  3.416667  4.583333  3.583333 17.916667
## [50] 22.000000 14.000000 22.000000 30.000000 16.833333 18.500000  9.750000
## [57]  9.750000
```

```
betwt <- betweenness(gt)
print("Task node:")
```

```
## [1] "Task node:"
```

```r
betwt
```

```
##    1    2    4    5    6    7    8    9   10   11   13   14   15   16   17   18   19   20   21   22
##    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    2    1    0    0 257
```

```r
ebetwt <- edge_betweenness(gt, e = E(gt), directed = TRUE)
print("Task edge:")
```

```
## [1] "Task edge:"
```

```r
ebetwt
```

```
##  [1] 17.0 17.0  1.0 17.0 17.0 17.0  1.0 17.0 17.0 17.0  2.0 15.0 17.0 17.0  1.5
## [16] 15.5  1.5  1.0 14.5  2.0  1.5  1.5 14.0  1.5  1.5 15.0  1.5 15.5  1.0  1.5
## [31] 14.5 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 15.0 17.0 17.0 15.5 14.5 14.0
## [46] 15.0 15.5 14.5
```

```r
#(b)

#Difinition2: Mean

datas <- data.frame(E(gs)$Strength,ebetws)
datas <- datas[order(-ebetws),]
datas
```

```
##    E.gs..Strength    ebetws
## 26           Weak 37.583333
## 16           Weak 30.000000
## 19           Weak 30.000000
## 53           Weak 30.000000
## 3          Strong 26.000000
## 8            Weak 26.000000
## 18           Weak 26.000000
## 50           Weak 22.000000
## 52           Weak 22.000000
## 31           Weak 21.000000
## 55           Weak 18.500000
## 49           Weak 17.916667
## 42           Weak 17.333333
## 54           Weak 16.833333
## 4            Weak 16.000000
## 12         Strong 16.000000
## 14           Weak 16.000000
## 17         Strong 16.000000
## 9            Weak 15.000000
## 10           Weak 15.000000
## 22           Weak 15.000000
## 11           Weak 14.000000
## 15         Strong 14.000000
## 51           Weak 14.000000
## 2            Weak 13.000000
```

```
## 7            Weak 13.000000
## 20           Weak 12.000000
## 56           Weak  9.750000
## 57           Weak  9.750000
## 36         Strong  6.916667
## 35           Weak  6.833333
## 44         Strong  6.083333
## 27           Weak  5.500000
## 34         Strong  5.416667
## 45           Weak  5.333333
## 43           Weak  4.833333
## 47         Strong  4.583333
## 32           Weak  4.500000
## 33           Weak  4.333333
## 23           Weak  4.250000
## 29           Weak  4.250000
## 30           Weak  4.250000
## 40         Strong  3.833333
## 48           Weak  3.583333
## 46           Weak  3.416667
## 25         Strong  3.250000
## 37         Strong  3.000000
## 38           Weak  2.833333
## 28           Weak  2.833333
## 39         Strong  2.500000
## 24           Weak  1.750000
## 41           Weak  1.250000
## 1          Strong  1.000000
## 5          Strong  1.000000
## 6          Strong  1.000000
## 13         Strong  1.000000
## 21         Strong  1.000000
```

```
datat <- data.frame(E(gt)$Strength,ebetwt)
datat <- datat[order(-ebetwt),]
datat
```

```
##    E.gt..Strength ebetwt
## 1          Strong   17.0
## 2            Weak   17.0
## 4          Strong   17.0
## 5          Strong   17.0
## 6            Weak   17.0
## 8            Weak   17.0
## 9            Weak   17.0
## 10           Weak   17.0
## 13           Weak   17.0
## 14           Weak   17.0
## 32         Strong   17.0
## 33           Weak   17.0
## 34         Strong   17.0
## 35           Weak   17.0
## 36           Weak   17.0
## 37           Weak   17.0
```

```
## 38           Weak   17.0
## 39           Weak   17.0
## 41           Weak   17.0
## 42           Weak   17.0
## 16         Strong   15.5
## 28         Strong   15.5
## 43           Weak   15.5
## 47           Weak   15.5
## 12           Weak   15.0
## 26         Strong   15.0
## 40           Weak   15.0
## 46         Strong   15.0
## 19           Weak   14.5
## 31         Strong   14.5
## 44           Weak   14.5
## 48         Strong   14.5
## 23         Strong   14.0
## 45           Weak   14.0
## 11           Weak    2.0
## 20           Weak    2.0
## 15           Weak    1.5
## 17           Weak    1.5
## 21           Weak    1.5
## 22           Weak    1.5
## 24           Weak    1.5
## 25           Weak    1.5
## 27           Weak    1.5
## 30           Weak    1.5
## 3            Weak    1.0
## 7            Weak    1.0
## 18           Weak    1.0
## 29           Weak    1.0
```

```r
#(a)

#Definition2: Median

gs <- graph.data.frame(social1,directed = TRUE)
gt <- graph.data.frame(task1,directed = TRUE)

betws <- betweenness(gs)
ebetws <- edge_betweenness(gs, e = E(gs), directed = TRUE)
print("Social node:")
```

```
## [1] "Social node:"
```

```r
betws
```

```
##          1          2          4          5          6          7
## 24.0000000  0.0000000  0.0000000 24.0000000 28.0000000  0.0000000
##          8          9         10         11         12         15
##  0.0000000  0.0000000 28.0000000 15.0000000 52.0000000  0.0000000
##         16         17         18         19         20         21
```

```
##  45.8333333   0.8333333  33.0000000  14.7500000   0.2500000  13.5000000
##          22
## 126.8333333
```

```r
print("Social edge:")
```

```
## [1] "Social edge:"
```

```r
ebetws
```

```
##  [1]  1.000000 13.000000 26.000000 16.000000  1.000000  1.000000 13.000000
##  [8] 26.000000 15.000000 15.000000 14.000000 16.000000  1.000000 16.000000
## [15] 14.000000 30.000000 16.000000 26.000000 30.000000 12.000000  1.000000
## [22] 15.000000  4.250000  1.750000  3.250000 37.583333  5.500000  2.833333
## [29]  4.250000  4.250000 21.000000  4.500000  4.333333  5.416667  6.833333
## [36]  6.916667  3.000000  2.833333  2.500000  3.833333  1.250000 17.333333
## [43]  4.833333  6.083333  5.333333  3.416667  4.583333  3.583333 17.916667
## [50] 22.000000 14.000000 22.000000 30.000000 16.833333 18.500000  9.750000
## [57]  9.750000
```

```r
betwt <- betweenness(gt)
print("Task node:")
```

```
## [1] "Task node:"
```

```r
betwt
```

```
##   1   2   4   5   6   7   8   9  10  11  13  14  15  16  17  18  19  20  21  22
##   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   2   1   0   0 257
```

```r
ebetwt <- edge_betweenness(gt, e = E(gt), directed = TRUE)
print("Task edge:")
```

```
## [1] "Task edge:"
```

```r
ebetwt
```

```
##  [1] 17.0 17.0  1.0 17.0 17.0 17.0  1.0 17.0 17.0 17.0  2.0 15.0 17.0 17.0  1.5
## [16] 15.5  1.5  1.0 14.5  2.0  1.5  1.5 14.0  1.5  1.5 15.0  1.5 15.5  1.0  1.5
## [31] 14.5 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 15.0 17.0 17.0 15.5 14.5 14.0
## [46] 15.0 15.5 14.5
```

```r
#(b)

#Difinition2: Median

datas <- data.frame(E(gs)$Strength,ebetws)
datas <- datas[order(-ebetws),]
datas
```

```
##    E.gs..Strength    ebetws
## 26           Weak 37.583333
## 16           Weak 30.000000
## 19           Weak 30.000000
## 53           Weak 30.000000
## 3          Strong 26.000000
## 8            Weak 26.000000
## 18           Weak 26.000000
## 50           Weak 22.000000
## 52           Weak 22.000000
## 31           Weak 21.000000
## 55           Weak 18.500000
## 49         Strong 17.916667
## 42           Weak 17.333333
## 54           Weak 16.833333
## 4            Weak 16.000000
## 12         Strong 16.000000
## 14           Weak 16.000000
## 17         Strong 16.000000
## 9            Weak 15.000000
## 10         Strong 15.000000
## 22           Weak 15.000000
## 11           Weak 14.000000
## 15         Strong 14.000000
## 51           Weak 14.000000
## 2          Strong 13.000000
## 7          Strong 13.000000
## 20           Weak 12.000000
## 56           Weak  9.750000
## 57           Weak  9.750000
## 36         Strong  6.916667
## 35           Weak  6.833333
## 44         Strong  6.083333
## 27           Weak  5.500000
## 34         Strong  5.416667
## 45           Weak  5.333333
## 43           Weak  4.833333
## 47         Strong  4.583333
## 32           Weak  4.500000
## 33           Weak  4.333333
## 23           Weak  4.250000
## 29           Weak  4.250000
## 30           Weak  4.250000
## 40         Strong  3.833333
## 48           Weak  3.583333
## 46           Weak  3.416667
## 25         Strong  3.250000
## 37         Strong  3.000000
## 38           Weak  2.833333
## 28           Weak  2.833333
## 39         Strong  2.500000
## 24           Weak  1.750000
## 41           Weak  1.250000
## 1          Strong  1.000000
```

```
## 5           Strong  1.000000
## 6           Strong  1.000000
## 13          Strong  1.000000
## 21          Strong  1.000000
```

```
datat <- data.frame(E(gt)$Strength,ebetwt)
datat <- datat[order(-ebetwt),]
datat
```

```
##    E.gt..Strength ebetwt
## 1          Strong   17.0
## 2          Strong   17.0
## 4          Strong   17.0
## 5          Strong   17.0
## 6            Weak   17.0
## 8          Strong   17.0
## 9            Weak   17.0
## 10         Strong   17.0
## 13           Weak   17.0
## 14         Strong   17.0
## 32         Strong   17.0
## 33           Weak   17.0
## 34         Strong   17.0
## 35         Strong   17.0
## 36         Strong   17.0
## 37           Weak   17.0
## 38           Weak   17.0
## 39           Weak   17.0
## 41           Weak   17.0
## 42         Strong   17.0
## 16         Strong   15.5
## 28         Strong   15.5
## 43         Strong   15.5
## 47           Weak   15.5
## 12           Weak   15.0
## 26         Strong   15.0
## 40         Strong   15.0
## 46         Strong   15.0
## 19           Weak   14.5
## 31         Strong   14.5
## 44           Weak   14.5
## 48         Strong   14.5
## 23         Strong   14.0
## 45         Strong   14.0
## 11           Weak    2.0
## 20           Weak    2.0
## 15           Weak    1.5
## 17           Weak    1.5
## 21           Weak    1.5
## 22           Weak    1.5
## 24           Weak    1.5
## 25           Weak    1.5
## 27           Weak    1.5
## 30           Weak    1.5
```

```
## 3            Weak    1.0
## 7            Weak    1.0
## 18           Weak    1.0
## 29           Weak    1.0
```

As shown in the data table above, for both definition: The high social betweenness tend to be weaker, and high social betweenness tend to be stronger. And high task betweenness tend to be stronger, low task betweenness tend to be weaker. It makes sense, because task edges have high betweenness process more information going back and forth between two people. And taks edges with low betweenness tend to be less commnicated so they are weaker. While, high betweenness among social edges, those people might be to distracted by so many friends, so the link is weaker. And low betweenness among social edges, meaning those are their friend of a only few, so they commnicate more frequently, so those are stronger.

```
#Question4

gs <- graph.data.frame(social,directed = TRUE)
gt <- graph.data.frame(task,directed = TRUE)

gsm <- as_adjacency_matrix(gs)
gtm <- as_adjacency_matrix(gt)
```

```
distance <- distances(gs)
distance
```

```
##      1   2   4   5   6   7   8   9  10  11  12  15  16  17  18  19  20  21  22
## 1    0   2 Inf   1   1   5 Inf   2   4   2   3   3   2   3   2   2   3   2   1
## 2    2   0 Inf   2   3   5 Inf   4   4   2   3   3   2   3   2   2   3   2   1
## 4  Inf Inf   0 Inf Inf Inf   1 Inf Inf Inf Inf Inf Inf Inf Inf Inf Inf Inf Inf
## 5    1   2 Inf   0   1   5 Inf   2   4   2   3   3   2   3   2   2   3   2   1
## 6    1   3 Inf   1   0   6 Inf   1   5   3   4   4   3   4   3   3   4   3   2
## 7    5   5 Inf   5   6   0 Inf   7   1   5   2   6   3   4   3   4   4   4   4
## 8  Inf Inf   1 Inf Inf Inf   0 Inf Inf Inf Inf Inf Inf Inf Inf Inf Inf Inf Inf
## 9    2   4 Inf   2   1   7 Inf   0   6   4   5   5   4   5   4   4   5   4   3
## 10   4   4 Inf   4   5   1 Inf   6   0   4   1   5   2   3   2   3   3   3   3
## 11   2   2 Inf   2   3   5 Inf   4   4   0   3   1   2   3   2   2   3   2   1
## 12   3   3 Inf   3   4   2 Inf   5   1   3   0   4   1   2   1   2   2   2   2
## 15   3   3 Inf   3   4   6 Inf   5   5   1   4   0   3   4   3   3   4   3   2
## 16   2   2 Inf   2   3   3 Inf   4   2   2   1   3   0   1   1   1   2   2   1
## 17   3   3 Inf   3   4   4 Inf   5   3   3   2   4   1   0   1   1   2   1   2
## 18   2   2 Inf   2   3   3 Inf   4   2   2   1   3   1   1   0   1   1   1   1
## 19   2   2 Inf   2   3   4 Inf   4   3   2   2   3   1   1   1   0   1   1   1
## 20   3   3 Inf   3   4   4 Inf   5   3   3   2   4   2   2   1   1   0   1   2
## 21   2   2 Inf   2   3   4 Inf   4   3   2   2   3   2   1   1   1   1   0   1
## 22   1   1 Inf   1   2   4 Inf   3   3   1   2   2   1   2   1   1   2   1   0
```

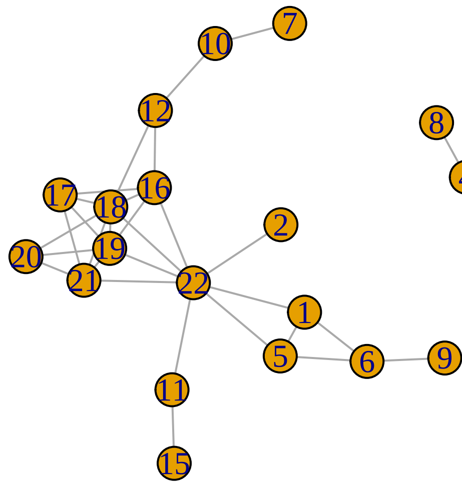
```
print("No relationship/Walk at all for Social:")
```

```
## [1] "No relationship/Walk at all for Social:"
```

```
sum(distance > 100 )/2
```

```
## [1] 34
```

```r
gsm <- graph.adjacency(gsm,mode = "undirected")
gsm <- simplify(gsm,remove.multiple = TRUE,remove.loops = TRUE)
plot(gsm)
```



And yes we can see from the graph after removing all the multiples, and prove our calculation above. Since there's only two standing alone, the number of no walk is combination of those two and everything else.

```r
distance <- distances(gt)
distance
```

```
##       1   2   4   5   6   7   8   9  10  11  13  14  15  16  17  18  19  20  21
## 1     0   2 Inf   2   2   2 Inf   2   2   2   2   2   2   2   2   2   2   2   2
## 2     2   0 Inf   2   2   2 Inf   2   2   2   2   2   2   2   2   2   2   2   2
## 4   Inf Inf   0 Inf Inf Inf   1 Inf Inf Inf Inf Inf Inf Inf Inf Inf Inf Inf Inf
## 5     2   2 Inf   0   2   2 Inf   2   2   2   2   2   2   2   2   2   2   2   2
## 6     2   2 Inf   2   0   2 Inf   2   2   2   2   2   2   2   2   2   2   2   2
## 7     2   2 Inf   2   2   0 Inf   2   2   2   2   2   2   2   2   2   2   2   2
## 8   Inf Inf   1 Inf Inf Inf   0 Inf Inf Inf Inf Inf Inf Inf Inf Inf Inf Inf Inf
## 9     2   2 Inf   2   2   2 Inf   0   2   2   2   2   2   2   2   2   2   2   2
## 10    2   2 Inf   2   2   2 Inf   2   0   2   2   2   2   2   2   2   2   2   2
## 11    2   2 Inf   2   2   2 Inf   2   2   0   2   2   2   2   2   2   2   2   2
## 13    2   2 Inf   2   2   2 Inf   2   2   2   0   2   2   2   2   1   2   2   2
## 14    2   2 Inf   2   2   2 Inf   2   2   2   2   0   2   2   2   2   2   2   2
## 15    2   2 Inf   2   2   2 Inf   2   2   2   2   2   0   2   2   2   2   2   2
## 16    2   2 Inf   2   2   2 Inf   2   2   2   2   2   2   0   2   2   1   2   2
## 17    2   2 Inf   2   2   2 Inf   2   2   2   2   2   2   2   0   1   2   2   1
```

```
## 18   2   2 Inf   2   2   2 Inf   2   2   2   1   2   2   2   1   0   2   2   1
## 19   2   2 Inf   2   2   2 Inf   2   2   2   2   2   2   1   2   2   0   1   2
## 20   2   2 Inf   2   2   2 Inf   2   2   2   2   2   2   2   2   2   1   0   2
## 21   2   2 Inf   2   2   2 Inf   2   2   2   2   2   2   2   1   1   2   2   0
## 22   1   1 Inf   1   1   1 Inf   1   1   1   1   1   1   1   1   1   1   1   1
##     22
## 1    1
## 2    1
## 4  Inf
## 5    1
## 6    1
## 7    1
## 8  Inf
## 9    1
## 10   1
## 11   1
## 13   1
## 14   1
## 15   1
## 16   1
## 17   1
## 18   1
## 19   1
## 20   1
## 21   1
## 22   0
```

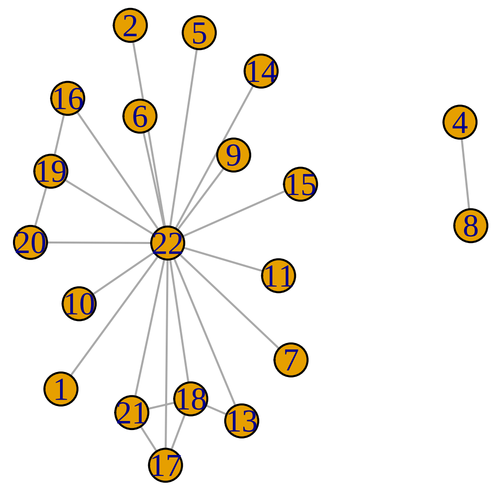
```r
print("No relationship/Walk at all for Task:")
```

```
## [1] "No relationship/Walk at all for Task:"
```

```r
sum(distance > 100 )/2
```

```
## [1] 36
```

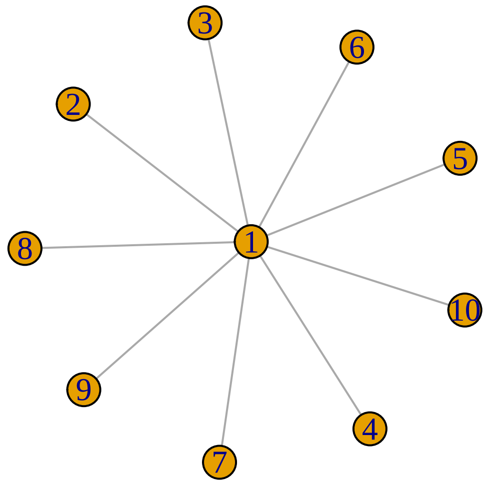
```r
gtm <- graph.adjacency(gtm,mode = "undirected")
gtm <- simplify(gtm,remove.multiple = TRUE,remove.loops = TRUE)
plot(gtm)
```



And yes we can see from the graph after removing all the multiples, and prove our calculation above. Since there's only two standing alone, the number of no walk is combination of those two and everything else.

```
#Question5
#network-level measure of degree centrality is equal to 1
#As network centrality is calculated by max minus all other degrees
#We could easily find that a star network of one level with any number of nodes would have network cent

star1 = make_star(10, "undirected")
plot(star1)
```




```r
print("Star1 centrality")
```

```
## [1] "Star1 centrality"
```

```r
dc1 <- sum((max(degree(star1)) - degree(star1)))/((length(V(star1)) -1)*(length(V(star1))-2))
dc1
```

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

```r
close <- closeness(star1,mode = "all")
max(close)
```

```
## [1] 0.1111111
```

```r
mean(close)
```

```
## [1] 0.06405229
```

```r
sd(close)
```

```
## [1] 0.01653479
```

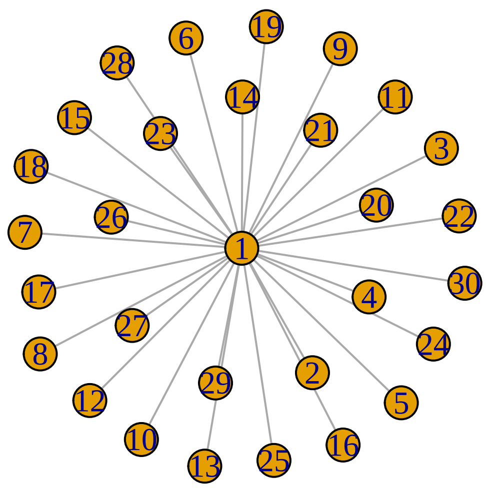
```
betw <- betweenness(star1)
mean(betw)
```

```
## [1] 3.6
```

```
sd(betw)
```

```
## [1] 11.3842
```

```
star2 = make_star(30, "undirected")
plot(star2)
```




```
print("Star2 centrality")
```

```
## [1] "Star2 centrality"
```

```
dc2 <- sum((max(degree(star2)) - degree(star2)))/((length(V(star2)) -1)* (length(V(star2))-2))
dc2
```

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

```r
close <- closeness(star2,mode = "all")
max(close)
```

```
## [1] 0.03448276
```

```r
mean(close)
```

```
## [1] 0.01810849
```

```r
sd(close)
```

```
## [1] 0.003092606
```

```r
betw <- betweenness(star2)
mean(betw)
```

```
## [1] 13.53333
```

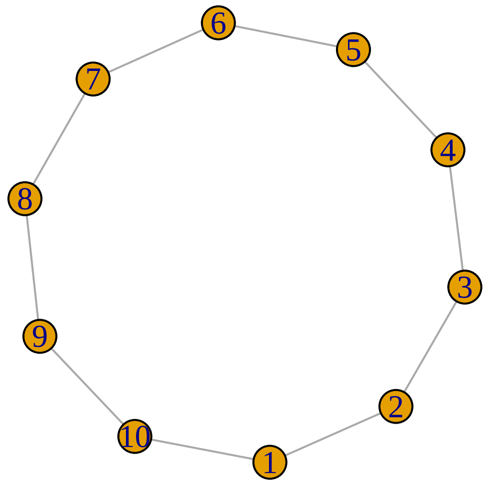
```r
sd(betw)
```

```
## [1] 74.12512
```

From the result, we can see that although both star1 and star2 have network centrality of 1, And we can see that the mean of closeness and betweenness is lower, and std of closeness and betweenness are quite big, which might indicates that the network is quite centralized one nodes, so the relationship hold true for these networks for other measures of centrality.

```r
#network-level measure of degree centrality is equal to 0
#As network centrality is calculated by max minus all other degrees
#We could easily find that a ring network with any number of nodes would have network centrality = 0.

ring1 = make_ring(10)
plot(ring1)
```




```
print("ring1 centrality")
```

```
## [1] "ring1 centrality"
```

```
dc1 <- sum((max(degree(ring1)) - degree(ring1)))/((length(V(ring1)) -1)* (length(V(ring1))-2))
dc1
```

```
## [1] 0
```

```
close <- closeness(ring1,mode = "all")
max(close)
```

```
## [1] 0.04
```

```
mean(close)
```

```
## [1] 0.04
```

```
sd(close)
```

```
## [1] 0
```

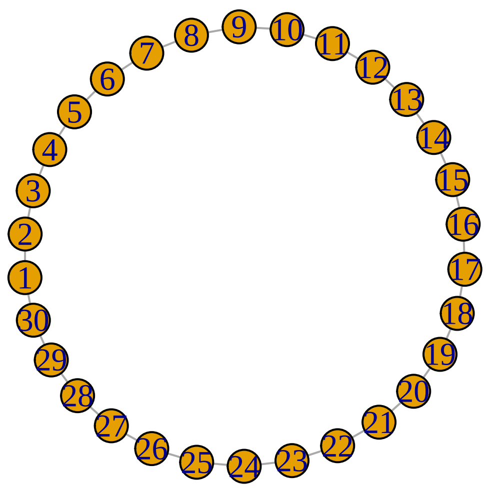
```r
betw <- betweenness(ring1)
mean(betw)
```

```
## [1] 8
```

```r
sd(betw)
```

```
## [1] 0
```

```r
ring2 = make_ring(30)
plot(ring2)
```




```r
print("ring2 centrality")
```

```
## [1] "ring2 centrality"
```

```r
dc2 <- sum((max(degree(ring2)) - degree(ring2)))/((length(V(ring2)) -1)* (length(V(ring2))-2))
dc2
```

```
## [1] 0
```

```r
close <- closeness(ring2,mode = "all")
max(close)
```

```
## [1] 0.004444444
```

```r
mean(close)
```

```
## [1] 0.004444444
```

```r
sd(close)
```

```
## [1] 0
```

```r
betw <- betweenness(ring2)
mean(betw)
```

```
## [1] 98
```

```r
sd(betw)
```

```
## [1] 0
```

From the result, we can see that although both ring1 and ring2 have network centrality of 0, And we can see that the mean of of closeness and betweenness is quite big, while std of closeness and betweenness are quite small, which might indicates that the network is quite spreaded out and not so centralized to one nodes. So the relationship hold true for these networks for other measures of centrality.