Report_2_pascal_baertschi

Pascal Bärtschi (20-713-962)

2023-03-29

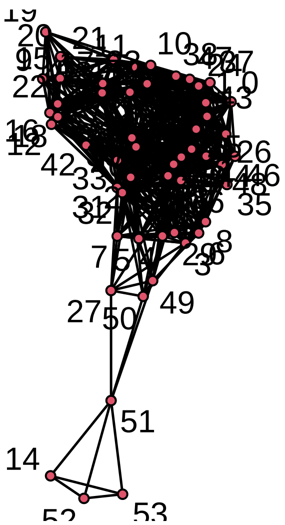
load data and libraries

1. Visualization

a) plot undirected

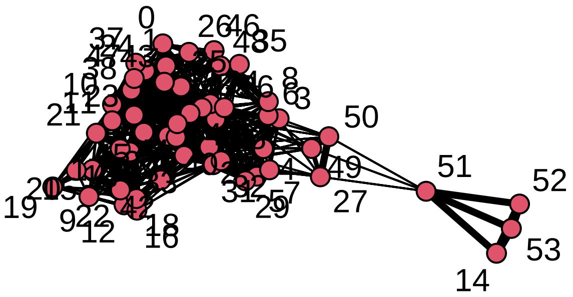
```
# network of hunter gatherer with interaction weights
hg_net <- network(edgelist[,c("ID1", "ID2")], directed = F)

plot(hg_net, gmode = "graph", displaylabels = T, vertex.cex. = 2)
```



## b) plot with weights

```r
# add weights
# igraph:set.edge.attribute(hg_net, "weights", edgelist$weights)
hg_net %e% "weights" <- edgelist$weights
# plot with weights
plot(hg_net, edge.lwd = 0.05 * (hg_net %e% "weights"),displaylabels = T, vertex.cex = 2)
```



2. Main network properties

a) - f)

```r
# convert to igraph object
ihg_net <-asIgraph(hg_net)
print(paste("a) Size:", network::network.size(hg_net)))
```

```
## [1] "a) Size: 54"
```

```r
print(paste("b) Density:", sna::gden(hg_net)))
```

```
## [1] "b) Density: 0.414395527603075"
```

```
print(paste("c) Components:", sna::components(hg_net)))
```

```
## [1] "c) Components: 1"
```

```
print(paste("d) Diameter:", igraph::diameter(ihg_net)))
```

```
## [1] "d) Diameter: 5"
```

```
print(paste("e) Avg. mean path:",igraph::mean_distance(ihg_net)))
```

```
## [1] "e) Avg. mean path: 1.81761006289308"
```

```
print(paste("f) Transitivity:", igraph::transitivity(ihg_net)))
```

```
## [1] "f) Transitivity: 0.659284497444634"
```

3. node features

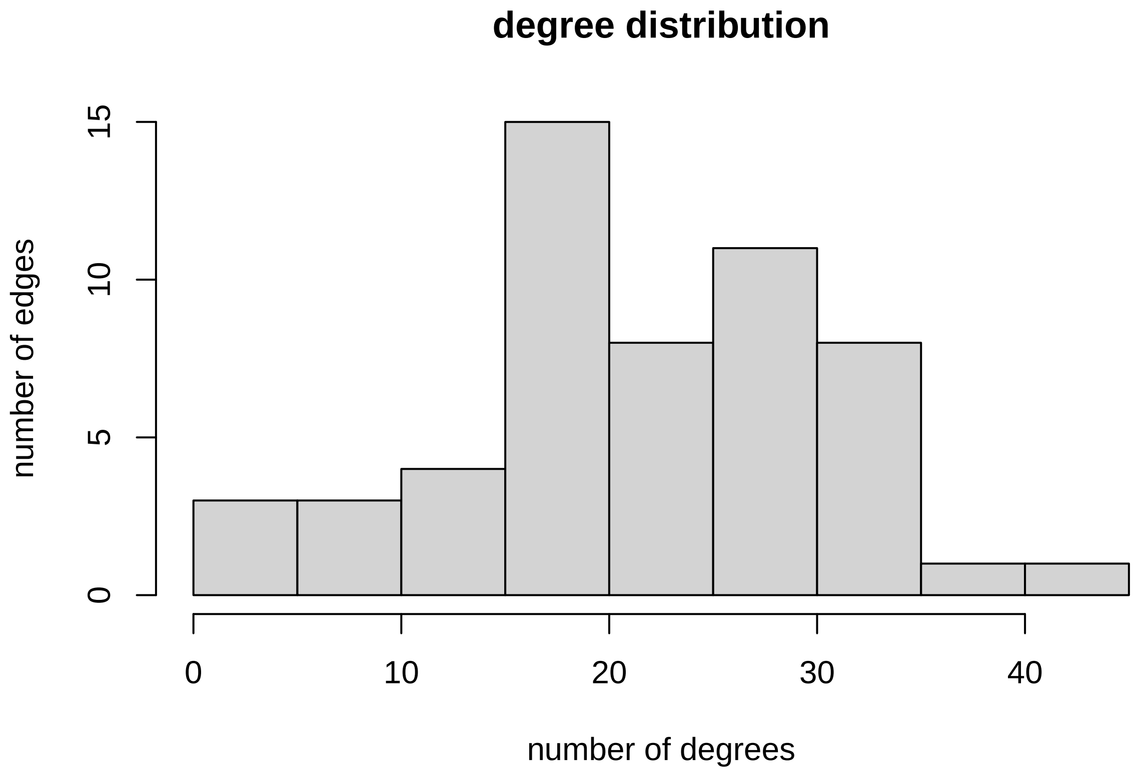
a) mean degree

```
print(mean(sna::degree(hg_net, gmode ="graph")))
```

```
## [1] 21.96296
```

b) degree distribution

```
hist(sna::degree(hg_net, gmode ="graph"),
     main = "degree distribution",
     xlab = "number of degrees",
     ylab = "number of edges")
```



4. Centrality measures

a) table

```
cen_table <- data.frame("id" = as.vector(V(ihg_net)) - 1, # correct ids
                        "degree" = sna::degree(hg_net, gmode = "graph"),
                        "closeness" = sna::closeness(hg_net, gmode = "graph"),
                        "betweenness" = sna::betweenness(hg_net, gmode = "graph"))
```

b) top 3 most central individuals of each centrality measure

```
# sort for degree
print((cen_table %>% arrange(desc(degree)))[1:3,1:2])
```

```
##   id degree
## 1 40     43
## 2 30     39
## 3  2     35
```

```
# sort for closeness
print((cen_table %>% arrange(desc(closeness)))[1:3,c(1, 3)])
```

```
##   id closeness
## 1 40 0.7571429
## 2  2 0.7162162
## 3 30 0.7162162
```

```
# sort for betweenness
print((cen_table %>% arrange(desc(betweenness)))[1:3,c(1, 4)])
```

```
##   id betweenness
## 1 50   150.00000
## 2  2    90.20734
## 3 48    89.41202
```

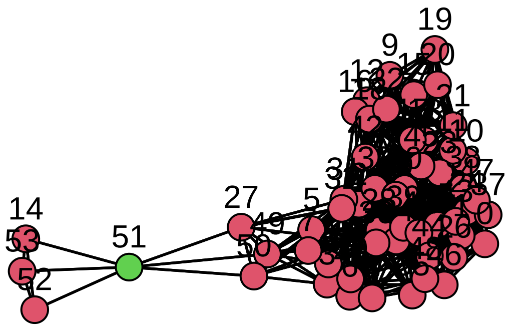
c) discussion

While the measures of centrality and degree agree with the ranking of the top 3 most important individuals agree, betweenness leads to a different conclusion. When looking at the plot of the network, however, this makes sense since number 30, 40 and 2 lie on the the main part of the network, leading to high degrees and close distances to others. The measurement of betweenness doesn't take this into account, but measures how many shortest paths lead over certain nodes and since 50 connects the two subnetworks it has high connectivity. Overall, I'd say that number two is the most central node, since its the only ID that is in the top 3 of all measures.

5. plot cutpoints

```
cuts <- cutpoints(hg_net,mode="graph", return.indicator=T)

gplot(hg_net, gmode="graph", vertex.col = cuts+2, displaylabels=T,
      # color schemes vary with adding different integers
      label.pos=3, vertex.cex=2, label.cex = 1)
```



6. Coreness

a) min and max coreness

```
coreness_hg <- igraph::coreness(ihg_net)
print(paste("max coreness:", max(coreness_hg)))
```

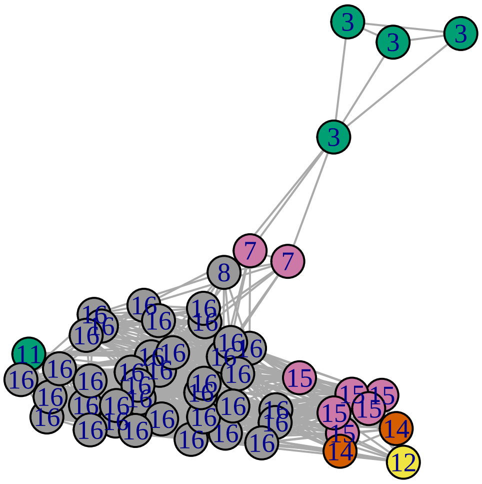
```
## [1] "max coreness: 16"
```

```
print(paste("min coreness:", min(coreness_hg)))
```

```
## [1] "min coreness: 3"
```

b) k-coreplot

```
V(ihg_net)$colour <- coreness_hg
plot(ihg_net,vertex.label.cex=0.8, vertex.color=coreness_hg,
     vertex.label=coreness_hg, gmode="graph")
```

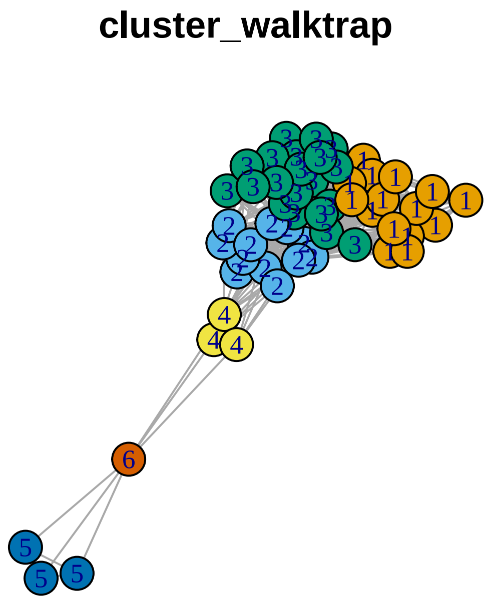


7. community detection

a) detect using algorithms

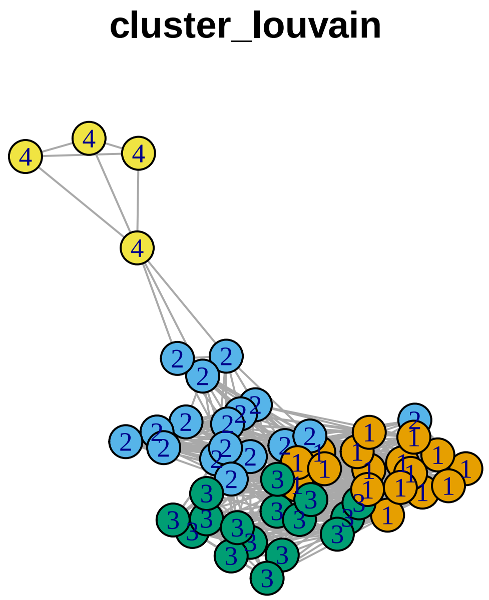
walktrap

```
walktrap_members <- igraph::membership(igraph::cluster_walktrap(ihg_net))
plot(ihg_net,vertex.label.cex=0.8, vertex.color=walktrap_members,
     vertex.label=walktrap_members, gmode="graph", main ="cluster_walktrap")
```



cluster_louvain

```
louvain_members <- igraph::membership(igraph::cluster_louvain(ihg_net))
plot(ihg_net,vertex.label.cex=0.8, vertex.color=louvain_members,
     vertex.label=louvain_members, gmode="graph", main ="cluster_louvain")
```



b) identifical classification by algorithms

```
# create df with results for ids
cluster_df <- data.frame("id" = as.vector(V(ihg_net) - 1),
                         "walktrap" = as.vector(walktrap_members),
                         "louvain" = as.vector(louvain_members))
# slice the ids that are NOT classified the same
print(cluster_df[walktrap_members != louvain_members,])
```

```
##    id walktrap louvain
## 1   0        3       1
## 2   1        3       1
## 10  9        1       3
## 12 11        1       3
## 13 12        1       3
## 14 13        1       3
## 15 14        5       4
## 16 15        1       3
## 17 16        1       3
## 18 17        1       3
## 19 18        1       3
## 20 19        1       3
## 21 20        1       3
## 22 21        1       3
## 23 22        1       3
## 24 23        1       3
## 25 24        3       1
## 26 25        3       1
## 27 26        3       1
## 28 27        4       2
## 31 30        3       1
## 34 33        3       2
## 35 34        3       1
## 36 35        3       2
## 38 37        3       1
## 39 38        3       1
## 40 39        3       1
## 42 41        3       1
## 44 43        3       1
## 45 44        3       1
## 46 45        1       3
## 47 46        3       1
## 48 47        3       1
## 49 48        4       2
## 50 49        4       2
## 51 50        6       4
## 52 51        5       4
## 53 52        3       1
## 54 53        5       4
```

```
print(paste("Different classifications: ", sum(walktrap_members != louvain_members), "of 53 nodes"))
```

```
## [1] "Different classifications:  39 of 53 nodes"
```

8. Model networks

a) random and small world network

ask whether type is right

```
hg_rnd <- erdos.renyi.game(n = length(V(ihg_net)),  # n = number of edges?
                           p = sna::gden(hg_net),   # p probabikity?
                           type = "gnp")            # type ?

hg_swn <- watts.strogatz.game(dim=1,
                              size=length(V(ihg_net)),
                              nei=mean(sna::degree(hg_net, gmode ="graph")/2), # neighbours
                              p=0.05) # randomness
```

b) comparison table of networks

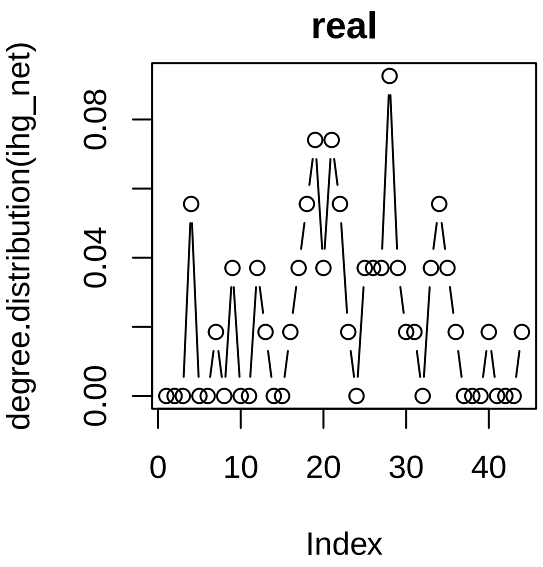
```
compare_net <- data.frame(network = c("real", "random", "SWM"),
                         diameter = c(diameter(ihg_net),
                                      diameter(hg_rnd),
                                      diameter(hg_swn)),
                         mean_path = c(mean_distance(ihg_net),
                                       mean_distance(hg_rnd),
                                       mean_distance(hg_swn)),
                         transitivity = c(transitivity(ihg_net),
                                          transitivity(hg_rnd),
                                          transitivity(hg_swn)))
print(compare_net)
```

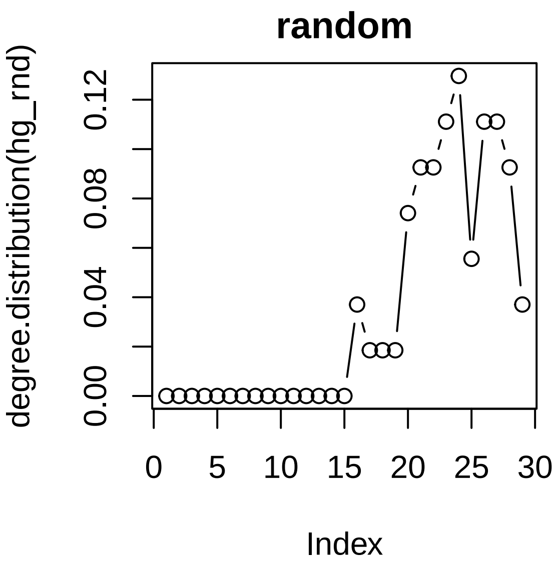
```
##   network diameter mean_path transitivity
## 1    real        5  1.817610    0.6592845
## 2  random        2  1.572327    0.4280021
## 3     SWM        3  1.653389    0.6004082
```

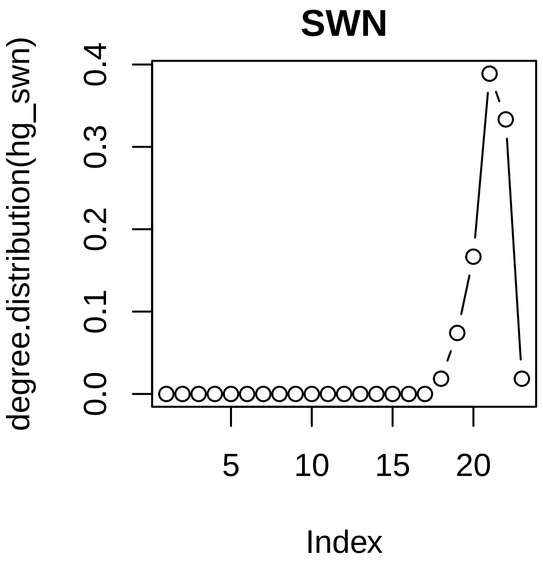
c) distribution of networks

degree distributions

```
par(mar = c(4, 4, 2, 1))
plot(degree.distribution(ihg_net), type="b", main="real")
plot(degree.distribution(hg_rnd), type="b", main="random")
plot(degree.distribution(hg_swn), type="b", main="SWN")
```

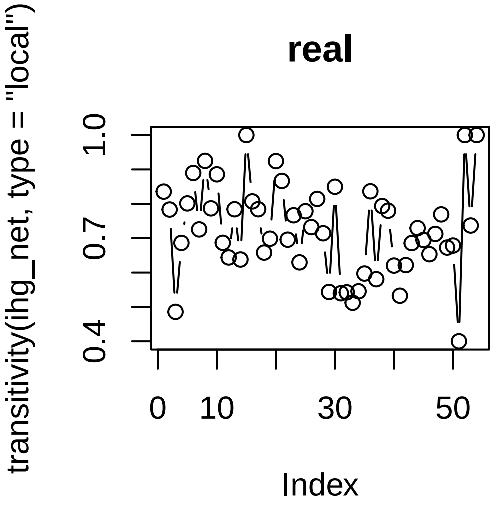


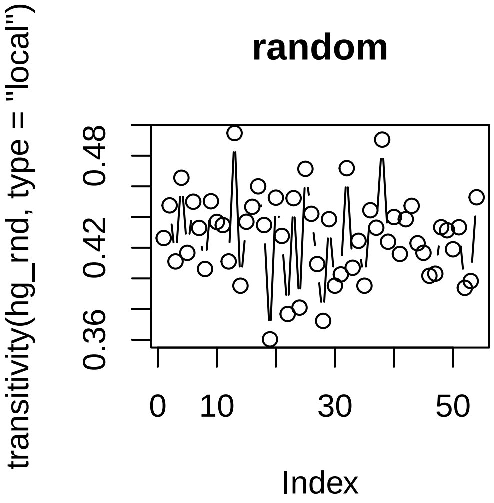


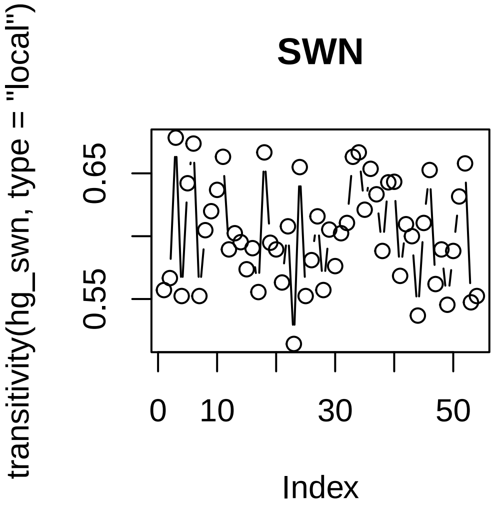


### local transitivity distributions

```
plot(transitivity(ihg_net, type = "local"), type="b", main="real")
plot(transitivity(hg_rnd, type = "local"), type="b", main="random")
plot(transitivity(hg_swn, type = "local"), type="b", main="SWN")
```







d) Conclusion

The comparison table makes clear that the SWN has a more similar mean path length and transitivity than the random network to the real one. The diameter, however, is not helpful in this comparison. The distribution comparisons,however, confirm that the SWN is more similar as the ranges of the probabilities of thetransitivity distribution is more similar to the one of the real one. The distributions of the degrees do not lead to a clear conclusion. In closing, the SWN is overall more similar to the real network and more suited to model it.