# SNA with R-igraph: a tutorial with enron e-mails dataset

Written by [Pedro Concejero](https://www.linkedin.com/in/pedroconcejero) based on previous work by [Carlos Ortega](http://www.qualityexcellence.es/) for [Madrid R users group](http://madrid.r-es.org/). This session was held in Madrid on April 2013, was in spanish and is recorded so you can watch it [here](https://www.intecca.uned.es/portalavip/grabacion.php?ID_Sala=77088&ID_Grabacion=80882&hashData=497fb4164f5080c3a94124de570f6707&amp%3BparamsToCheck=SURfR3JhYmFjaW9uLElEX1NhbGEs)

This document reviews the original 2013 tutorial material, but with appropiate updates (ie links) on august 2015. All code and datasets are available at my [github](https://github.com/pedroconcejero/public-sna).

(check from root <https://github.com/pedroconcejero> in case you fail to find datasets because I must change the structure of github in the near future).

You can contact me here: pedro.concejerocerezo *at* gmail.com. I will be delighted to listen to your comments!!!

This document is done in RStudio using [knitr markdown](http://rmarkdown.rstudio.com/) language and intends to provide a tutorial about the igraph R library for Social Networks Analysis.

Explanations on how to use igraph for producing R graph objects are embedded within the explanations of the data and objectives of the analysis.

# Introduction: The enron scandal

The enron skandal revealed in 2001 and was the most expensive bankruptcy produced till that date (many more expensive ones have happened afterwards). An excellent reference on the enron history can be found in wikipedia:

<http://en.wikipedia.org/wiki/Enron_scandal>

After the company’s collapse a large database of over 600,000 emails generated by 158 employees of the Enron Corporation was acquired by the Federal Energy Regulatory Commission during its investigation after the company’s collapse. A copy of the database was subsequently purchased for $10,000 by Andrew McCallum, a computer scientist at the University of Massachusetts Amherst, who released this copy to researchers as the “Enron corpus”. This analysis is based on this dataset. More about the Enron corpus can be consulted at:

<http://en.wikipedia.org/wiki/Enron_Corpus>

# The enron e-mail database

The dataset object of analysis here is based on a MySQL implementation of all e-mails between the 158 enron employees and all the rest of the world (except private emails that were deleted previously by the database owners). Since this dataset would be rather difficult to use in an educational setting, the dataset was restricted to the emails **between enron employees**, i.e. both sender and receiver are enron employees, thus reducing considerably the dataset size and making it easier the interpretation of links and other practical issues.

This dataset was created from a version of the enron corpus by Jitesh Shetty and Jafar Adibi available as a MySQL database [still available here](http://www.cs.cmu.edu/~./enron/). Please note this database was reviewed in several ocasions after the original tutorial was produced, and authors recommend to download and use the latest version. More in particular, these [deletions](http://www.cs.cmu.edu/~./enron/DELETIONS.txt). These deletions *have not been done* in this august 2015 version but I will try to implement them.

This is the origin of the two dataframes that are required to produce an igraph R graph object: edges (or links, in this case, e-mails), and nodes.

# Setting up the R + igraph environment

Needless to say (or maybe not), you need the igraph library installed in your system. If you have not do so, it is easy, once you have opened R, just introduce this sentence:

install.packages(“igraph”)

First of all we must prepare the R environment, load igraph and other required libraries, and set the working directory. Please change this for your environment!!!

knitr::opts\_chunk$set(echo = TRUE)  
  
knitr::opts\_knit$set(root.dir = '/home/pedro/Escritorio/UTAD\_2021\_22/BAIN 2022/public-sna-master')  
  
library(igraph)

##   
## Attaching package: 'igraph'

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

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

# required to improve plots with function colorpanel  
library(gplots)

##   
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':  
##   
## lowess

You just need two files that contain all data required for these analyses: - [enron.RData file](https://github.com/pedroconcejero/public-sna/blob/master/enron.RData) contains the working space with several R objects that are explained and described below - [edges\_w\_message.RData (please unzip data first!)](https://github.com/pedroconcejero/public-sna/blob/master/edges_w_message.zip) is a file that includes a dataframe containing the links.

For igraph it is essential that two first columns found in the edges dataframe are node id’s -usually first one is sender and second is receiver-. So we have the following information in the edges dataframe: - sender: e-mail address of sender - receiver: e-mail address of receiver - type of e-mail (CC, BCC, TO) - subject: string with the subject of e-mail - body: full text of e-mail message - date

Date format in particular is tricky, you will find some cautions about it in the tutorial.

load("enron.RData")  
load("edges\_w\_message.RData")

We can do a few basic manipulations with these objects:

* Count the number of edges:

nrow(edges.full)

## [1] 61673

* Describe the edges object with str function:

str(edges.full)

## 'data.frame': 61673 obs. of 6 variables:  
## $ sender : chr "mary.hain@enron.com" "mary.hain@enron.com" "mary.hain@enron.com" "cooper.richey@enron.com" ...  
## $ receiver: chr "sean.crandall@enron.com" "mike.swerzbin@enron.com" "robert.badeer@enron.com" "robert.badeer@enron.com" ...  
## $ type : chr "TO" "TO" "TO" "TO" ...  
## $ subject : chr "Enron s transmission/power exchange model for discussion" "Enron s transmission/power exchange model for discussion" "Enron s transmission/power exchange model for discussion" "Change to EnData" ...  
## $ body : chr "---------------------- Forwarded by Mary Hain/HOU/ECT on 08/17/2000 02:15 PM ---------------------------James D"| \_\_truncated\_\_ "---------------------- Forwarded by Mary Hain/HOU/ECT on 08/17/2000 02:15 PM ---------------------------James D"| \_\_truncated\_\_ "---------------------- Forwarded by Mary Hain/HOU/ECT on 08/17/2000 02:15 PM ---------------------------James D"| \_\_truncated\_\_ "The Fundamentals Group is moving Database servers and the existing EnData Excel Add-Inneeds to be changed. If "| \_\_truncated\_\_ ...  
## $ date : chr "2000-08-17 07:11:00" "2000-08-17 07:11:00" "2000-08-17 07:11:00" "2000-08-23 04:39:00" ...

* or we can re-format date string so that we can use dates (ie filter, compute durations) more easily in R.

edges.full$date.R <- as.POSIXct(edges.full$date)  
summary(edges.full$date.R)

## Min. 1st Qu. Median   
## "1998-11-13 04:07:00" "2000-12-12 07:45:00" "2001-06-11 17:17:00"   
## Mean 3rd Qu. Max.   
## "2001-05-08 21:44:27" "2001-10-28 14:41:38" "2002-06-21 13:37:34"

Note that date can be either a string or a native date R-object. We will use both, because external software, in particular gephi, does not understand the exported R-date format.

The other required object to produce an igraph graph object is the nodes dataframe. This contains all the info about the nodes, in our case, the enron employees who were e-mail sender or receivers.

This dataframe contains e-mail address as node id, the lastName as a useful string for labelling, and her/his status in the company (if this info. was available).

# Number of nodes  
nrow(nodes)

## [1] 149

# Description of the nodes object  
str(nodes)

## 'data.frame': 149 obs. of 3 variables:  
## $ Email\_id: chr "marie.heard@enron.com" "mark.e.taylor@enron.com" "lindy.donoho@enron.com" "lisa.gang@enron.com" ...  
## $ lastName: chr "Heard" "Taylor" "Donoho" "Gang" ...  
## $ status : chr "N/A" "Employee" "Employee" "N/A" ...

The rest of this document explains how to handle and what you can do with the igraph SNA object in 8 steps

# R igraph tutorial in 8 steps

## 1- Creating an igraph network graph with graph.data.frame

Just insist on the requirement that: - two first columns of edges object match with node id’s - nodes object must contain all info. from nodes in edges object

When creating the graph we can choose if the network is directed or not. In this case we choose it as directed.

Please note that we create an igraph object and summary will tell us the number of nodes and edges. igraph automatically sets as node properties all additional columns in node object (name, lastName, status) and as edge properties all additional columns apart from node id’s (type, date, count)

And note that for igraph V = vertex and E = edge. Please note uppercase. And we filtered out the full text for practical reasons, to make it simpler.

network.full <- graph.data.frame(edges.full[,c("sender",  
 "receiver",  
 "type",  
 "date",  
 "subject")],  
 directed = TRUE,  
 vertices = nodes)  
  
class(network.full);summary(network.full)

## [1] "igraph"

## IGRAPH 7f6cb87 DN-- 149 61673 --   
## + attr: name (v/c), lastName (v/c), status (v/c), type (e/c), date  
## | (e/c), subject (e/c)

## 2- Using the igraph object and V and E components

Best documentation can be found at:

<https://igraph.org/python/doc/tutorial/tutorial.html>

You can access node and edge properties by means of:

V(your\_network\_object)

E(your\_network\_object)

V(network.full)[1:10]

## + 10/149 vertices, named, from 7f6cb87:  
## [1] marie.heard@enron.com mark.e.taylor@enron.com lindy.donoho@enron.com   
## [4] lisa.gang@enron.com jeff.skilling@enron.com lynn.blair@enron.com   
## [7] kim.ward@enron.com kate.symes@enron.com kay.mann@enron.com   
## [10] keith.holst@enron.com

E(network.full)[1:10]

## + 10/61673 edges from 7f6cb87 (vertex names):  
## [1] mary.hain@enron.com ->sean.crandall@enron.com   
## [2] mary.hain@enron.com ->mike.swerzbin@enron.com   
## [3] mary.hain@enron.com ->robert.badeer@enron.com   
## [4] cooper.richey@enron.com->robert.badeer@enron.com   
## [5] mary.hain@enron.com ->m..forney@enron.com   
## [6] mary.hain@enron.com ->robert.badeer@enron.com   
## [7] mary.hain@enron.com ->mike.swerzbin@enron.com   
## [8] jeff.dasovich@enron.com->james.d.steffes@enron.com  
## [9] jeff.dasovich@enron.com->richard.shapiro@enron.com  
## [10] jeff.dasovich@enron.com->james.d.steffes@enron.com

And also its properties via table

table(V(network.full)$status)

##   
## CEO Director Employee In House Lawyer   
## 4 14 41 1   
## Manager Managing Director N/A President   
## 14 3 32 4   
## Trader Vice President   
## 13 23

## 3- Exporting the graph for using it with external software (ie gephi)

Basically we write the graph to graphml format (an XML format special for graph objects). Take care with date format: gephi requires it to be a string.

Gephi tutorial is not part of this document. You can find excellent gephi tutorials just [googling](https://www.google.es/search?q=gephi+tutorial)

write.graph(network,  
 file = "network01.graphml",  
 format = "graphml")

## 4- Computing Individual SNA metrics

With igraph and get.shortest.paths you can obtain the **shortest paths** between two nodes.

get.shortest.paths(from = V(network.full)$lastName == "Pereira",  
 to = V(network.full)$lastName == "Horton",  
 graph = network.full)

## $vpath  
## $vpath[[1]]  
## + 3/149 vertices, named, from 7f6cb87:  
## [1] susan.w.pereira@enron.com kenneth.lay@enron.com   
## [3] stanley.horton@enron.com   
##   
##   
## $epath  
## NULL  
##   
## $predecessors  
## NULL  
##   
## $inbound\_edges  
## NULL

nodes[c(138,11,132),]

## Email\_id lastName status  
## 138 susan.w.pereira@enron.com Pereira Employee  
## 11 kenneth.lay@enron.com Lay CEO  
## 132 stanley.horton@enron.com Horton President

**Diameter** of the graph is the length of the largest distance between nodes

diameter(network.full)

## [1] 5

farthest\_vertices(network.full)

## $vertices  
## + 2/149 vertices, named, from 7f6cb87:  
## [1] joe.quenet@enron.com lisa.gang@enron.com   
##   
## $distance  
## [1] 5

**Centrality measures** are computed and can be added to the node properties table. Basic centrality measure is **degree**, both in\_degree and out\_degree (this is a directed graph), and total\_degree.

nodes$degree\_total <- degree(network.full,   
 v = V(network.full),   
 mode = c("total"))  
nodes$degree\_in <- degree(network.full,   
 v = V(network.full),   
 mode = c("in"))  
nodes$degree\_out <- degree(network.full,   
 v = V(network.full),   
 mode = c("out"))

Let’s see who are the top20 for each measure

head(nodes[order(nodes$degree\_total,  
 decreasing = TRUE),], n = 20L)

## Email\_id lastName status degree\_total  
## 42 jeff.dasovich@enron.com Dasovich Employee 8610  
## 36 james.d.steffes@enron.com Steffes Vice President 5720  
## 141 tana.jones@enron.com Jones N/A 5190  
## 99 mike.grigsby@enron.com Grigsby Manager 4709  
## 125 sara.shackleton@enron.com Shackleton N/A 4708  
## 116 richard.shapiro@enron.com Shapiro Vice President 4327  
## 134 steven.j.kean@enron.com Kean Vice President 4046  
## 2 mark.e.taylor@enron.com Taylor Employee 3477  
## 14 louise.kitchen@enron.com Kitchen President 3241  
## 55 carol.clair@enron.com Clair Vice President 3114  
## 17 kimberly.watson@enron.com Watson N/A 2091  
## 133 stephanie.panus@enron.com Panus Employee 2063  
## 1 marie.heard@enron.com Heard N/A 2048  
## 140 susan.bailey@enron.com Bailey N/A 1918  
## 16 liz.taylor@enron.com Taylor N/A 1890  
## 115 richard.b.sanders@enron.com Sanders Vice President 1813  
## 139 susan.scott@enron.com Scott N/A 1800  
## 98 michelle.lokay@enron.com Lokay Employee 1658  
## 135 steven.harris@enron.com Harris Vice President 1628  
## 93 mary.hain@enron.com Hain N/A 1622  
## degree\_in degree\_out  
## 42 1499 7111  
## 36 2991 2729  
## 141 1633 3557  
## 99 693 4016  
## 125 2211 2497  
## 116 3276 1051  
## 134 2476 1570  
## 2 2422 1055  
## 14 1123 2118  
## 55 937 2177  
## 17 940 1151  
## 133 869 1194  
## 1 1066 982  
## 140 1486 432  
## 16 129 1761  
## 115 1332 481  
## 139 876 924  
## 98 672 986  
## 135 1353 275  
## 93 461 1161

head(nodes[order(nodes$degree\_in,  
 decreasing = TRUE),], n = 20L)

## Email\_id lastName status degree\_total  
## 116 richard.shapiro@enron.com Shapiro Vice President 4327  
## 36 james.d.steffes@enron.com Steffes Vice President 5720  
## 134 steven.j.kean@enron.com Kean Vice President 4046  
## 2 mark.e.taylor@enron.com Taylor Employee 3477  
## 125 sara.shackleton@enron.com Shackleton N/A 4708  
## 141 tana.jones@enron.com Jones N/A 5190  
## 42 jeff.dasovich@enron.com Dasovich Employee 8610  
## 140 susan.bailey@enron.com Bailey N/A 1918  
## 135 steven.harris@enron.com Harris Vice President 1628  
## 115 richard.b.sanders@enron.com Sanders Vice President 1813  
## 49 barry.tycholiz@enron.com Tycholiz Vice President 1494  
## 14 louise.kitchen@enron.com Kitchen President 3241  
## 1 marie.heard@enron.com Heard N/A 2048  
## 17 kimberly.watson@enron.com Watson N/A 2091  
## 55 carol.clair@enron.com Clair Vice President 3114  
## 139 susan.scott@enron.com Scott N/A 1800  
## 133 stephanie.panus@enron.com Panus Employee 2063  
## 75 elizabeth.sager@enron.com Sager Employee 1135  
## 110 phillip.k.ellen@enron.com Allen Manager 1250  
## 96 matthew.lenhart@enron.com Lenhart Employee 1309  
## degree\_in degree\_out  
## 116 3276 1051  
## 36 2991 2729  
## 134 2476 1570  
## 2 2422 1055  
## 125 2211 2497  
## 141 1633 3557  
## 42 1499 7111  
## 140 1486 432  
## 135 1353 275  
## 115 1332 481  
## 49 1181 313  
## 14 1123 2118  
## 1 1066 982  
## 17 940 1151  
## 55 937 2177  
## 139 876 924  
## 133 869 1194  
## 75 816 319  
## 110 785 465  
## 96 775 534

head(nodes[order(nodes$degree\_out,  
 decreasing = TRUE),], n = 20L)

## Email\_id lastName status degree\_total degree\_in  
## 42 jeff.dasovich@enron.com Dasovich Employee 8610 1499  
## 99 mike.grigsby@enron.com Grigsby Manager 4709 693  
## 141 tana.jones@enron.com Jones N/A 5190 1633  
## 36 james.d.steffes@enron.com Steffes Vice President 5720 2991  
## 125 sara.shackleton@enron.com Shackleton N/A 4708 2211  
## 55 carol.clair@enron.com Clair Vice President 3114 937  
## 14 louise.kitchen@enron.com Kitchen President 3241 1123  
## 16 liz.taylor@enron.com Taylor N/A 1890 129  
## 134 steven.j.kean@enron.com Kean Vice President 4046 2476  
## 133 stephanie.panus@enron.com Panus Employee 2063 869  
## 93 mary.hain@enron.com Hain N/A 1622 461  
## 17 kimberly.watson@enron.com Watson N/A 2091 940  
## 123 sally.beck@enron.com Beck Employee 1313 252  
## 2 mark.e.taylor@enron.com Taylor Employee 3477 2422  
## 116 richard.shapiro@enron.com Shapiro Vice President 4327 3276  
## 73 drew.fossum@enron.com Fossum Vice President 1331 320  
## 98 michelle.lokay@enron.com Lokay Employee 1658 672  
## 1 marie.heard@enron.com Heard N/A 2048 1066  
## 57 chris.germany@enron.com Germany Employee 1086 131  
## 139 susan.scott@enron.com Scott N/A 1800 876  
## degree\_out  
## 42 7111  
## 99 4016  
## 141 3557  
## 36 2729  
## 125 2497  
## 55 2177  
## 14 2118  
## 16 1761  
## 134 1570  
## 133 1194  
## 93 1161  
## 17 1151  
## 123 1061  
## 2 1055  
## 116 1051  
## 73 1011  
## 98 986  
## 1 982  
## 57 955  
## 139 924

**Reach** is another measure, also known as **neighborhood.size**. You must specify a specific order (an integer), meaning the total number of people you can reach with that number of steps. We can observe how this metric is very much linked to actual connectivity.

nodes$reach\_2\_step <-   
 neighborhood.size(network.full,   
 order = 2,  
 nodes = V(network.full),   
 mode = c("all"))  
  
head(nodes[order(nodes$reach\_2\_step,  
 decreasing = TRUE),], n = 30L)

## Email\_id lastName status degree\_total  
## 15 kevin.m.presto@enron.com Presto Vice President 1146  
## 16 liz.taylor@enron.com Taylor N/A 1890  
## 11 kenneth.lay@enron.com Lay CEO 597  
## 26 lavorato@enron.com Lavorato CEO 377  
## 123 sally.beck@enron.com Beck Employee 1313  
## 14 louise.kitchen@enron.com Kitchen President 3241  
## 36 james.d.steffes@enron.com Steffes Vice President 5720  
## 68 david.w.delainey@enron.com Delainey CEO 1078  
## 110 phillip.k.ellen@enron.com Allen Manager 1250  
## 134 steven.j.kean@enron.com Kean Vice President 4046  
## 24 m..forney@enron.com Forney Manager 289  
## 49 barry.tycholiz@enron.com Tycholiz Vice President 1494  
## 88 e..haedicke@enron.com Haedicke Managing Director 1176  
## 117 rick.buy@enron.com Buy Manager 439  
## 5 jeff.skilling@enron.com Skilling CEO 242  
## 63 dana.davis@enron.com Davis Vice President 261  
## 85 greg.whalley@enron.com Whalley President 833  
## 93 mary.hain@enron.com Hain N/A 1622  
## 99 mike.grigsby@enron.com Grigsby Manager 4709  
## 116 richard.shapiro@enron.com Shapiro Vice President 4327  
## 139 susan.scott@enron.com Scott N/A 1800  
## 10 keith.holst@enron.com Holst Director 638  
## 25 john.arnold@enron.com Arnold Manager 969  
## 75 elizabeth.sager@enron.com Sager Employee 1135  
## 80 fletcher.j.sturm@enron.com Sturm Vice President 389  
## 141 tana.jones@enron.com Jones N/A 5190  
## 148 j.kaminski@enron.com Kaminski Manager 451  
## 2 mark.e.taylor@enron.com Taylor Employee 3477  
## 97 michelle.cash@enron.com Cash Employee 245  
## 115 richard.b.sanders@enron.com Sanders Vice President 1813  
## degree\_in degree\_out reach\_2\_step  
## 15 459 687 146  
## 16 129 1761 145  
## 11 210 387 144  
## 26 6 371 144  
## 123 252 1061 143  
## 14 1123 2118 142  
## 36 2991 2729 142  
## 68 556 522 142  
## 110 785 465 142  
## 134 2476 1570 142  
## 24 106 183 141  
## 49 1181 313 141  
## 88 695 481 141  
## 117 328 111 141  
## 5 141 101 140  
## 63 244 17 140  
## 85 769 64 140  
## 93 461 1161 140  
## 99 693 4016 140  
## 116 3276 1051 140  
## 139 876 924 140  
## 10 614 24 139  
## 25 495 474 139  
## 75 816 319 139  
## 80 256 133 139  
## 141 1633 3557 139  
## 148 104 347 139  
## 2 2422 1055 138  
## 97 130 115 138  
## 115 1332 481 138

There is a lot of info. about enron employees that can guide you for further exploring the enron social network, ie <http://www.inf.ed.ac.uk/teaching/courses/tts/assessed/roles.txt>

Other interesting measures are **clustering coefficient** and **transitivity** [wikipedia on transitivity measures](http://en.wikipedia.org/wiki/Clustering_coefficient). For instance: “The clustering coefficient places more weight on the low degree nodes, while the transitivity ratio places more weight on the high degree nodes”.

nodes$transitivity\_ratio <-   
 transitivity(network.full,   
 vids = V(network.full),   
 type = "local")  
  
head(nodes[order(nodes$transitivity\_ratio,  
 decreasing = FALSE),], n = 20L)

## Email\_id lastName status degree\_total  
## 139 susan.scott@enron.com Scott N/A 1800  
## 16 liz.taylor@enron.com Taylor N/A 1890  
## 26 lavorato@enron.com Lavorato CEO 377  
## 123 sally.beck@enron.com Beck Employee 1313  
## 57 chris.germany@enron.com Germany Employee 1086  
## 11 kenneth.lay@enron.com Lay CEO 597  
## 7 kim.ward@enron.com Ward N/A 1000  
## 65 daren.j.farmer@enron.com Farmer Manager 105  
## 24 m..forney@enron.com Forney Manager 289  
## 52 bill.williams@enron.com Williams N/A 381  
## 14 louise.kitchen@enron.com Kitchen President 3241  
## 22 kam.keiser@enron.com Keiser Employee 1081  
## 84 gerald.nemec@enron.com Nemec N/A 1294  
## 56 charles.weldon@enron.com Weldon N/A 99  
## 93 mary.hain@enron.com Hain N/A 1622  
## 42 jeff.dasovich@enron.com Dasovich Employee 8610  
## 62 dan.hyvl@enron.com Hyvl Employee 531  
## 15 kevin.m.presto@enron.com Presto Vice President 1146  
## 69 debra.perlingiere@enron.com Perlingiere Employee 972  
## 99 mike.grigsby@enron.com Grigsby Manager 4709  
## degree\_in degree\_out reach\_2\_step transitivity\_ratio  
## 139 876 924 140 0.2115722  
## 16 129 1761 145 0.2277193  
## 26 6 371 144 0.2343505  
## 123 252 1061 143 0.2595335  
## 57 131 955 130 0.2666667  
## 11 210 387 144 0.2956108  
## 7 450 550 135 0.3084677  
## 65 82 23 123 0.3142857  
## 24 106 183 141 0.3233333  
## 52 105 276 117 0.3238095  
## 14 1123 2118 142 0.3294553  
## 22 289 792 126 0.3298097  
## 84 593 701 134 0.3484848  
## 56 63 36 126 0.3508772  
## 93 461 1161 140 0.3655172  
## 42 1499 7111 136 0.3684211  
## 62 235 296 106 0.3766234  
## 15 459 687 146 0.3860544  
## 69 284 688 110 0.3874644  
## 99 693 4016 140 0.3931545

V(network.full)$outdegree <- degree(network.full, mode = "out")  
V(network.full)$indegree <- degree(network.full, mode = "in")  
V(network.full)$degree <- degree(network.full, mode = "all")  
V(network.full)$reach\_2\_step <- neighborhood.size(network.full,   
 order = 2,  
 nodes = V(network.full),   
 mode = c("all"))  
V(network.full)$transitivity\_ratio <- transitivity(network.full,   
 vids = V(network.full),   
 type = "local")

## 5- Extracting subgraphs

Extracting parts of a graph using igraph is very easy. You just need to know two functions:

induced.subgraph

subgraph.edges

For instance, to extract subgraphs of the most relevant people when enron came into bankruptcy. Many details about this [here](http://es.wikipedia.org/wiki/Enron" \l "Ca.C3.ADda_de_la_empresa) (CAVEAT: in spanish, as I have not been able to find this information with such detail in english)

edges.full$day <- strftime(edges.full$date.R, "%Y-%m-%d")  
  
network.august <- subgraph.edges(network.full,  
 which(as.Date(E(network.full)$date) > "2001-02-12 00:00:00"),  
 delete.vertices = TRUE)

We actually produce a new graph

summary(network.august)

## IGRAPH 98e8cee DN-- 146 42126 --   
## + attr: name (v/c), lastName (v/c), status (v/c), outdegree (v/n),  
## | indegree (v/n), degree (v/n), reach\_2\_step (v/n), transitivity\_ratio  
## | (v/n), type (e/c), date (e/c), subject (e/c)

For instance let’s see messages from president Kenneth Lay

mails.lay <- edges.full[(edges.full$sender == "kenneth.lay@enron.com" &  
 as.Date(edges.full$date.R) > "2001-07-01 00:00:00") |  
 (edges.full$receiver == "kenneth.lay@enron.com" &  
 as.Date(edges.full$date.R) > "2001-07-01 00:00:00")   
 ,]  
mails.lay <- mails.lay[order(as.Date(mails.lay$date.R)),]  
nrow(mails.lay)

## [1] 506

See how employees were not aware until last minute of what was going on, in spite of all the stakes they had in the company performance. But of course all this depended on the position you had in the company.

This is the email to Kenneth Lay from a young employee who lost money after believing in enron executives communications (though they were doing the opposite):

mails.lay[rownames(mails.lay) == 3473,]

## sender receiver type  
## 3473 susan.w.pereira@enron.com kenneth.lay@enron.com TO  
## subject  
## 3473 ENRON - A Study in How So Few Could Screw So Many  
## body  
## 3473 Mr. Lay-After reading the news of your $60,000,000 to $80,000,000 payout, I am disg=usted and appalled all over again. Unfortunately, that s a daily occurrenc=e.=20I have been employed with Enron since May of 1993, when Enron purchased LRC=. I was very skeptical of Enron back then and was unsure of my future with= the company. As time went on, I got more comfortable with the Enron way a=nd learned how to survive and even be a little successful here. I became a= believer in the things Enron could accomplish, a champion of our hard-nose=d, driven executives (at least some of them). I bought Enron stock and hel=d on to those valuable options.Over the last 18 months, my coworkers and I have viewed the weekly, sometim=es daily, selling of Enron stock and exercising of options by our top execu=tives (past and present and including you). We came up with all kinds of r=easons that the executives would be doing this -- they re so overly compens=ated that they have to cash out some every now and then, divorce settlement=s, mistress settlements, buying a new home in Aspen, buying an island in th=e Caribbean, etc., etc. etc.... We never wanted to admit that they knew so=mething we didn t. Things were great, weren t they? Jeff Skilling told us= that Enron was going to be the "World s Leading Company." He even put the= goofy acronyms on his car license plate. He told us gas traders in Februa=ry how the stock was going to $126 by the end of the year. Now, being the c=ynic that I am, I didn t believe the $126, although I have to admit I was h=opeful; I figured that if Jeff had the audacity to throw out a number that =high, then it was reasonable to expect the stock to be fairly stable, i.e.,= +/- 20%. =20Needless to say, I didn t sell my stock and didn t exercise any more option=s. In fact, I bought more stock when it first started going down. I m afr=aid that there are many more just like me. I m fortunate in that I have ma=ny working years ahead of me (where, I don t know) to try to build up my sa=vings again. Many others are not that fortunate. So many have spent their= entire careers here, helping to build this company up. They were looking =forward to retiring soon and enjoying the fruits of their labor. And let m=e remind you that their retirement accounts were in many cases a lot less t=han a month s compensation for you. Now even that is essentially gone. Ot=her employees were just ready to cash in some of their options to pay for t=heir children s college tuition. The stories are too numerous to list, and= the more I think about it, the more sickened I become.It is painfully obvious to me and my coworkers, as well as the rest of the =industry and Houston, that Enron s executives knew that there were skeleton=s in the closet and began cashing in ahead of this freefall. The employees= and the rest of the world were fed a bunch of half-truths and mystery mumb=o-jumbo. There should be an accounting for this behavior. You and your co=horts have ruined so many lives. Think about that while you re spending yo=ur millions.....Susan PereiraENA Gas Trader  
## date date.R day  
## 3473 2001-11-13 11:38:01 2001-11-13 11:38:01 2001-11-13

And this is the message of an executive to the same. Tone or words cannot be more different.

mails.lay[rownames(mails.lay) == 60469,]

## sender receiver type subject  
## 60469 stanley.horton@enron.com kenneth.lay@enron.com TO Difficult times  
## body  
## 60469 I just wanted to let you know that if there is anything I can do to help I am more than willing to do it. These are difficult times and I am doing alot of floor meetings and table talks with the employees. They clearly do not understand how we got into this situation and just want some face time with Management. With the exception of NEPCO our fourth quarter looks good. I have uncovered alot of issues with NEPCO that I do not think anyone knew existed. I ll know more Friday after a businnes/budget review session.Please let me know if I can help.Stan  
## date date.R day  
## 60469 2001-10-31 05:57:26 2001-10-31 05:57:26 2001-10-31

Another way of extracting a subgraph, all nodes who had contact with Kenneth Lay

nodes.with.lay <- unique(c(mails.lay$sender,  
 mails.lay$receiver))  
  
network.kenneth.lay <- graph.data.frame(mails.lay[,c("sender",  
 "receiver",  
 "type",  
 "date",  
 "subject")],  
 directed = TRUE)  
  
summary(network.kenneth.lay)

## IGRAPH 811ec6c DN-- 61 506 --   
## + attr: name (v/c), type (e/c), date (e/c), subject (e/c)

And now see how many people were in Lay’s neighbourhood. This was the CEO so it was extremely easy for him to reach the whole company in only two steps.

neighborhood.size(network.full,   
 1,   
 V(network.full)$lastName == "Lay")

## [1] 63

neighborhood.size(network.full,   
 2,   
 V(network.full)$lastName == "Lay")

## [1] 144

## 6- Reciprocity and Dyads - Creating the social graph

The reciprocity function gives this measure, which is the basis for computing dyads (“couples”)

reciprocity(network.full)

## [1] 0.4291992

And you can also obtain the dyads. This gives back A named numeric vector with three elements:

mut The number of pairs with mutual connections.

asym The number of pairs with non-mutual connections.

null The number of pairs with no connection between them.

Similar thing can be done with triplets (not done here). See <https://igraph.org/r/doc/triad_census.html>

dyad.census(network.full)

## $mut  
## [1] 761  
##   
## $asym  
## [1] 968  
##   
## $null  
## [1] 9297

Social graph should contain reciprocal pairs. This is, our directed pairs A->B would require that we also have a relationship B->A. This way we would have a reciprocal relationship A<->B. This way we convert a **communications graph** into a **social graph**

For that purpose we need a preliminary step: computing the weight of the link between two nodes. The simplest measure is the number of communications without distinction by type (to, cc, bcc).

First we extract unique pairs and we order them:

pairs <- as.data.frame(unique(edges.full[c(1,2)]))  
pairs <- pairs[order(pairs$sender, pairs$receiver),]  
  
edges.ordered <- edges.full[order(edges.full$sender, edges.full$receiver),]  
  
weight <- aggregate(edges.ordered[,3],  
 by = list(edges.ordered[,1],  
 edges.ordered[,2]),  
 length)   
  
weight <- weight[order(weight$Group.1, weight$Group.2),]

Let’s verify with head and tail

head(pairs, n = 5L)

## sender receiver  
## 51982 albert.meyers@enron.com bill.williams@enron.com  
## 51981 albert.meyers@enron.com ryan.slinger@enron.com  
## 50480 andrea.ring@enron.com brad.mckay@enron.com  
## 50482 andrea.ring@enron.com chris.germany@enron.com  
## 50508 andrea.ring@enron.com gerald.nemec@enron.com

head(weight, n = 5L)

## Group.1 Group.2 x  
## 96 albert.meyers@enron.com bill.williams@enron.com 5  
## 2039 albert.meyers@enron.com ryan.slinger@enron.com 2  
## 111 andrea.ring@enron.com brad.mckay@enron.com 1  
## 187 andrea.ring@enron.com chris.germany@enron.com 1  
## 622 andrea.ring@enron.com gerald.nemec@enron.com 1

tail(pairs, n = 5L)

## sender receiver  
## 51179 vladi.pimenov@enron.com geoff.storey@enron.com  
## 51183 vladi.pimenov@enron.com jane.tholt@enron.com  
## 50723 vladi.pimenov@enron.com john.griffith@enron.com  
## 51174 vladi.pimenov@enron.com jonathan.mckay@enron.com  
## 51176 vladi.pimenov@enron.com sandra.f.brawner@enron.com

tail(weight, n = 5L)

## Group.1 Group.2 x  
## 621 vladi.pimenov@enron.com geoff.storey@enron.com 1  
## 835 vladi.pimenov@enron.com jane.tholt@enron.com 1  
## 1053 vladi.pimenov@enron.com john.griffith@enron.com 4  
## 1099 vladi.pimenov@enron.com jonathan.mckay@enron.com 4  
## 2084 vladi.pimenov@enron.com sandra.f.brawner@enron.com 2

pairs[seq(236:248),]

## sender receiver  
## 51982 albert.meyers@enron.com bill.williams@enron.com  
## 51981 albert.meyers@enron.com ryan.slinger@enron.com  
## 50480 andrea.ring@enron.com brad.mckay@enron.com  
## 50482 andrea.ring@enron.com chris.germany@enron.com  
## 50508 andrea.ring@enron.com gerald.nemec@enron.com  
## 50483 andrea.ring@enron.com judy.townsend@enron.com  
## 50484 andrea.ring@enron.com peter.keavey@enron.com  
## 50467 andrea.ring@enron.com richard.ring@enron.com  
## 50470 andrea.ring@enron.com sandra.f.brawner@enron.com  
## 50481 andrea.ring@enron.com scott.hendrickson@enron.com  
## 50468 andrea.ring@enron.com scott.neal@enron.com  
## 51805 andy.zipper@enron.com barry.tycholiz@enron.com  
## 16708 andy.zipper@enron.com brad.mckay@enron.com

weight[seq(236:248),]

## Group.1 Group.2 x  
## 96 albert.meyers@enron.com bill.williams@enron.com 5  
## 2039 albert.meyers@enron.com ryan.slinger@enron.com 2  
## 111 andrea.ring@enron.com brad.mckay@enron.com 1  
## 187 andrea.ring@enron.com chris.germany@enron.com 1  
## 622 andrea.ring@enron.com gerald.nemec@enron.com 1  
## 1109 andrea.ring@enron.com judy.townsend@enron.com 1  
## 1825 andrea.ring@enron.com peter.keavey@enron.com 1  
## 1928 andrea.ring@enron.com richard.ring@enron.com 22  
## 2066 andrea.ring@enron.com sandra.f.brawner@enron.com 16  
## 2101 andrea.ring@enron.com scott.hendrickson@enron.com 1  
## 2108 andrea.ring@enron.com scott.neal@enron.com 1  
## 39 andy.zipper@enron.com barry.tycholiz@enron.com 3  
## 112 andy.zipper@enron.com brad.mckay@enron.com 3

Then we mix pairs and weight

pairs$weight <- weight$x  
head(pairs)

## sender receiver weight  
## 51982 albert.meyers@enron.com bill.williams@enron.com 5  
## 51981 albert.meyers@enron.com ryan.slinger@enron.com 2  
## 50480 andrea.ring@enron.com brad.mckay@enron.com 1  
## 50482 andrea.ring@enron.com chris.germany@enron.com 1  
## 50508 andrea.ring@enron.com gerald.nemec@enron.com 1  
## 50483 andrea.ring@enron.com judy.townsend@enron.com 1

Now we substitute the mails table by a links table and we produce a new graph using this as the edge table

network.sna <- graph.data.frame(pairs,  
 directed = TRUE,  
 vertices = nodes)  
  
summary(network.sna)

## IGRAPH 193e3db DNW- 149 2490 --   
## + attr: name (v/c), lastName (v/c), status (v/c), degree\_total (v/n),  
## | degree\_in (v/n), degree\_out (v/n), reach\_2\_step (v/n),  
## | transitivity\_ratio (v/n), weight (e/n)

Let’s check with reciprocity and dyad.census

reciprocity(network.sna)

## [1] 0.611245

dyad.census(network.sna)

## $mut  
## [1] 761  
##   
## $asym  
## [1] 968  
##   
## $null  
## [1] 9297

Now we can impose the requirement that a link must be reciprocal to have a **social relationship**

Thanks to [Carlos Gil Bellosta](http://www.datanalytics.com/) for suggesting:

<http://stackoverflow.com/questions/13006656/igraph-nonreciprocal-edges-after-converting-to-undirected-graph-using-mutual>

<https://igraph.org/r/html/latest/as.directed.html>

network.social <- as.undirected(network.sna,   
 mode = "collapse",   
 edge.attr.comb = "sum")

## 7- Computing Communities

You should know there are recommendations about availability of [algorithms for computing communities](http://igraph.wikidot.com/community-detection-in-r) depending on the type of your graph (directed vs. non-directed).

A classical algorithm is the one by Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, Etienne Lefebvre, “Fast unfolding of communities in large networks”, in Journal of Statistical Mechanics: Theory and Experiment 2008 (10), P1000, and is part of gephi in the function [multilevel.community](https://igraph.org/r/html/latest/cluster_louvain.html)

communities <- multilevel.community(network.social)  
  
comms.df <- data.frame(row.names = seq(1:149))  
comms.df$Email\_id <- communities$names  
comms.df$community <- communities$membership

Then you can add each node’s community to the nodes table

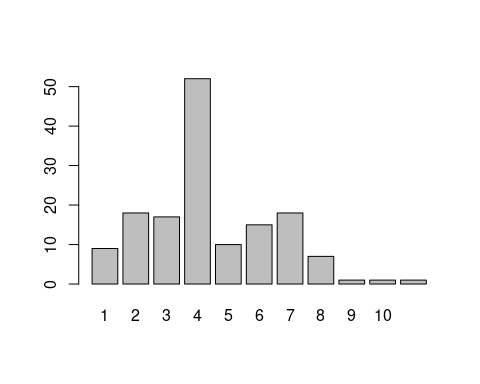
nodes.def <- merge(nodes, comms.df,   
 by.x = "Email\_id",  
 by.y = "Email\_id")  
  
str(nodes.def)

## 'data.frame': 149 obs. of 9 variables:  
## $ Email\_id : chr "albert.meyers@enron.com" "andrea.ring@enron.com" "andy.zipper@enron.com" "barry.tycholiz@enron.com" ...  
## $ lastName : chr "Meyers" "Ring" "Zipper" "Tycholiz" ...  
## $ status : chr "Employee" "N/A" "Vice President" "Vice President" ...  
## $ degree\_total : num 38 142 529 1494 96 ...  
## $ degree\_in : num 31 97 327 1181 67 ...  
## $ degree\_out : num 7 45 202 313 29 283 276 21 0 325 ...  
## $ reach\_2\_step : num 89 125 136 141 118 74 117 120 1 63 ...  
## $ transitivity\_ratio: num 0.833 0.463 0.511 0.41 0.533 ...  
## $ community : num 3 6 4 5 4 2 3 6 10 3 ...

head(nodes.def)

## Email\_id lastName status degree\_total degree\_in  
## 1 albert.meyers@enron.com Meyers Employee 38 31  
## 2 andrea.ring@enron.com Ring N/A 142 97  
## 3 andy.zipper@enron.com Zipper Vice President 529 327  
## 4 barry.tycholiz@enron.com Tycholiz Vice President 1494 1181  
## 5 benjamin.rogers@enron.com Rogers Employee 96 67  
## 6 bill.rapp@enron.com Rapp N/A 434 151  
## degree\_out reach\_2\_step transitivity\_ratio community  
## 1 7 89 0.8333333 3  
## 2 45 125 0.4632353 6  
## 3 202 136 0.5111111 4  
## 4 313 141 0.4102564 5  
## 5 29 118 0.5333333 4  
## 6 283 74 0.7333333 2

barplot(table(nodes.def$community))

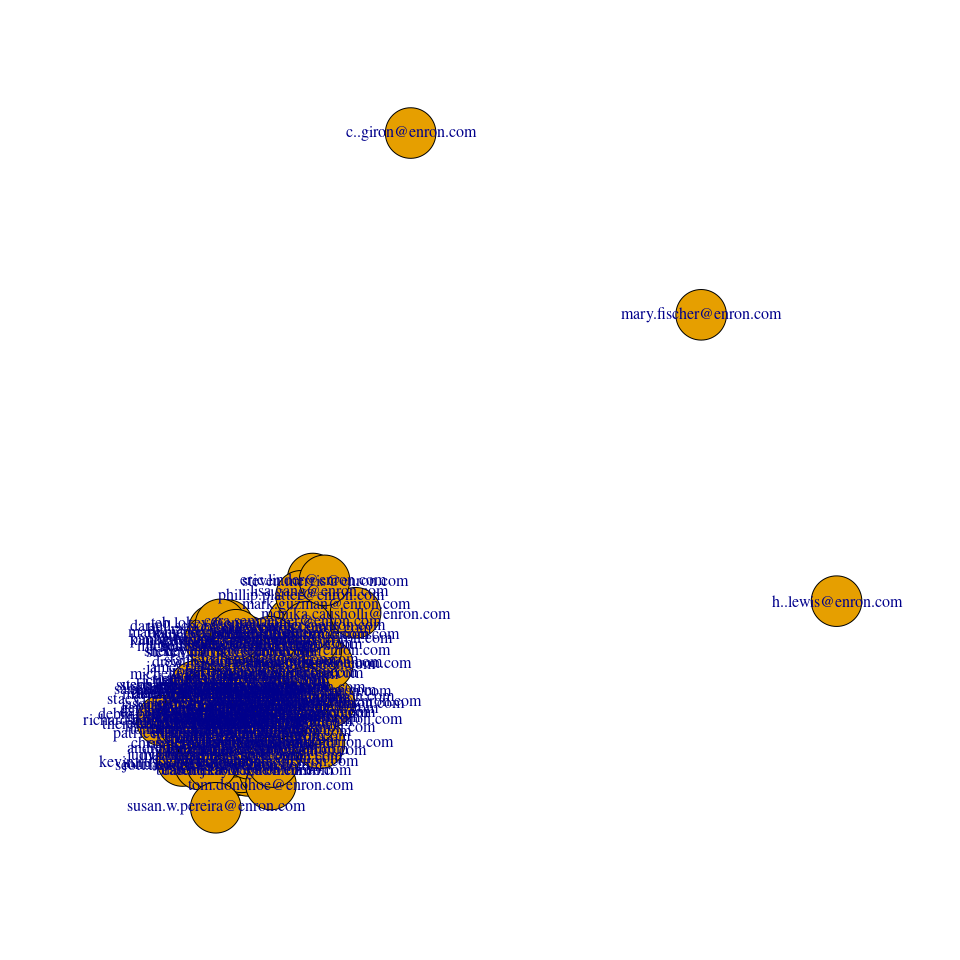


V(network.social)$community <- communities$membership

## 8- Graph visualization (plotting) with igraph

Let’s try plot quick and dirty:

plot(network.social)



First recommendation:

**plotting a large graph with igraph -and the enron graph is relatively small- is useless**

Gephi is an excellent alternative for high quality plots of relatively large graphs, but anyway the smaller, more compact and especially more relevant the better.

So the first thing we shall do is extracting a relevant subgraph. Let’s extract the “CEO’s COMMUNITIES”:

str(nodes.def)

## 'data.frame': 149 obs. of 9 variables:  
## $ Email\_id : chr "albert.meyers@enron.com" "andrea.ring@enron.com" "andy.zipper@enron.com" "barry.tycholiz@enron.com" ...  
## $ lastName : chr "Meyers" "Ring" "Zipper" "Tycholiz" ...  
## $ status : chr "Employee" "N/A" "Vice President" "Vice President" ...  
## $ degree\_total : num 38 142 529 1494 96 ...  
## $ degree\_in : num 31 97 327 1181 67 ...  
## $ degree\_out : num 7 45 202 313 29 283 276 21 0 325 ...  
## $ reach\_2\_step : num 89 125 136 141 118 74 117 120 1 63 ...  
## $ transitivity\_ratio: num 0.833 0.463 0.511 0.41 0.533 ...  
## $ community : num 3 6 4 5 4 2 3 6 10 3 ...

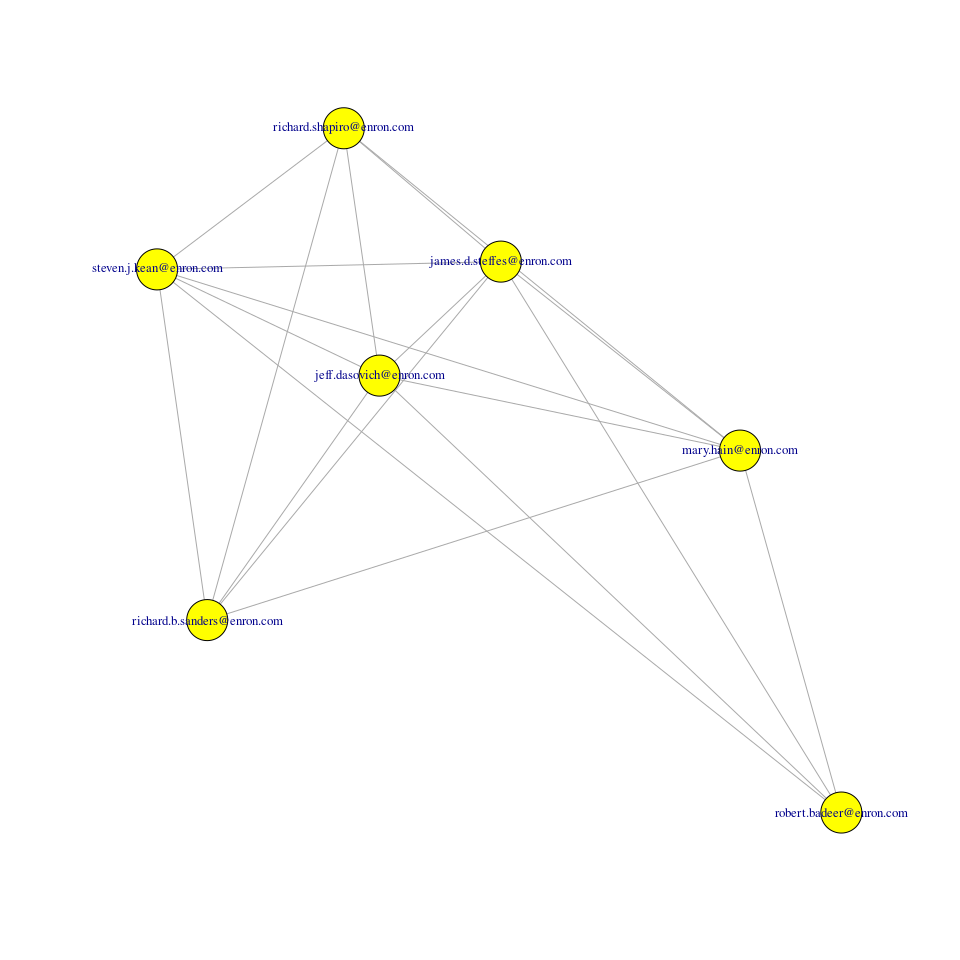
nodes.def[nodes.def$lastName == "Lay",]

## Email\_id lastName status degree\_total degree\_in degree\_out  
## 72 kenneth.lay@enron.com Lay CEO 597 210 387  
## reach\_2\_step transitivity\_ratio community  
## 72 144 0.2956108 4

# uncomment if you want to see all details of community number 8  
#nodes.def[nodes.def$community == 8, c(2:9)]  
  
com.ceos <- induced.subgraph(network.social,  
 V(network.social)$community == 8,  
 impl = "auto") # Ver ayuda  
  
summary(com.ceos)

## IGRAPH 1c4bdd4 UNW- 7 19 --   
## + attr: name (v/c), lastName (v/c), status (v/c), degree\_total (v/n),  
## | degree\_in (v/n), degree\_out (v/n), reach\_2\_step (v/n),  
## | transitivity\_ratio (v/n), community (v/n), weight (e/n)

g <- com.ceos  
  
plot(g,  
 vertex.label.font = 1,  
 vertex.label.cex = 0.8,   
 edge.arrow.size = 0.3,  
 vertex.size = 12,  
 vertex.color = "yellow"  
)



You can see another particularity: By default (no layout), nodes are projected on *random co-ordinates*, with automatic labels starting by 0, correlative numbers afterwards. Therefore every time you plot the graph, the result will be different. If you want to keep stable a plot you must fix the layout, or you can use some pre-computed layouts with fancy names, as “Reingold-Tilford” (after its authors), which tell you nothing about what they do (so you just have to try and see) or some others who are more clearly labelled, as the layout.circle.

Let’s see some of them. Please note this was prepared in 2013 and layouts have changed in igraph. Always check latest documentation to see this! I have updated the original tutorial with this info, but it can still change.

### How to fix a layout in a plot

First the definition of layout: Layout - Either a function or a numeric matrix. It specifies how the vertices will be placed on the plot. If it is a numeric matrix, then the matrix has to have one line for each vertex, specifying its coordinates. The matrix should have at least two columns, for the x and y coordinates, and it can also have third column, this will be the z coordinate for 3D plots and it is ignored for 2D plots.

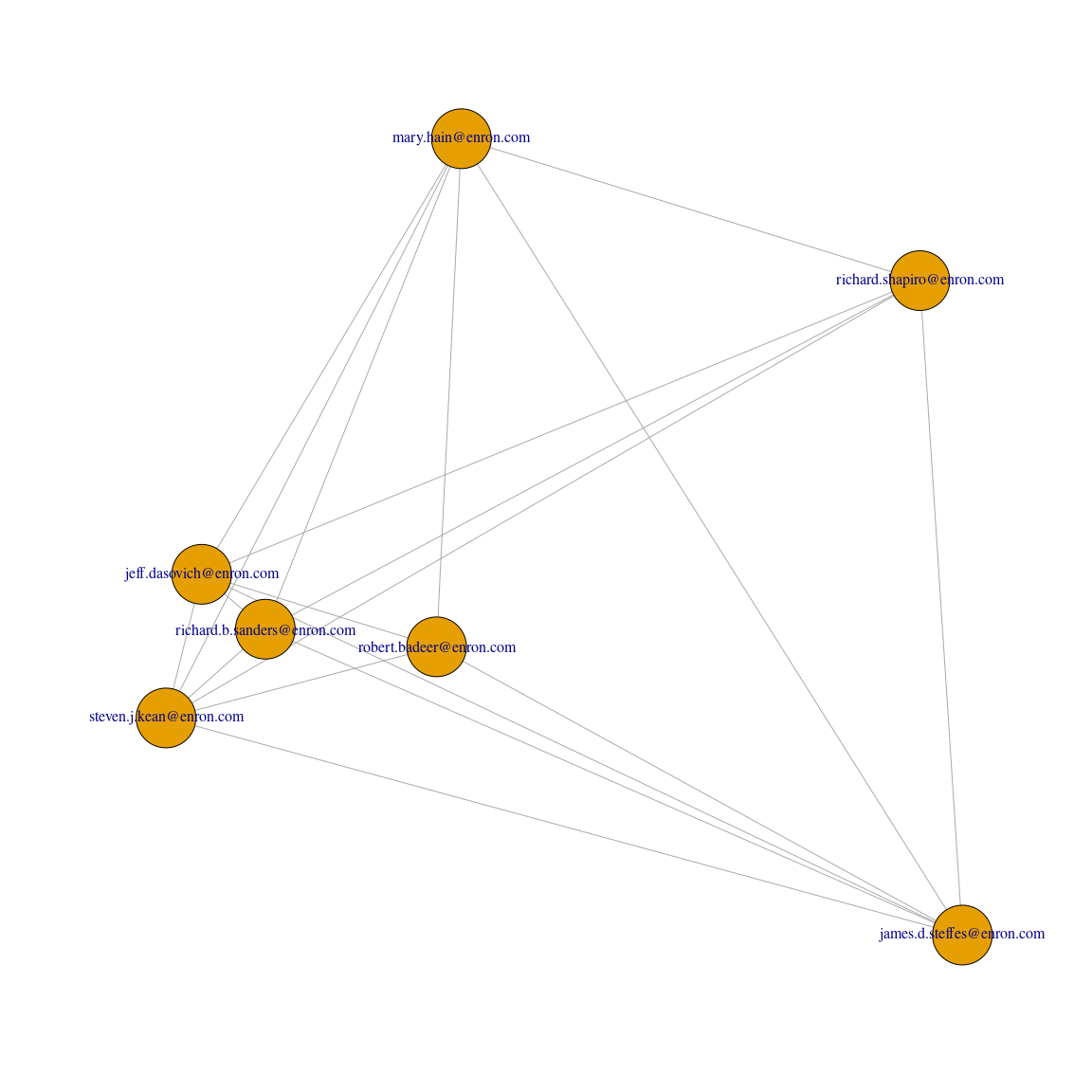
If a two column matrix is given for the 3D plotting function rglplot then the third column is assumed to be 1 for each vertex.

If layout is a function, this function will be called with the graph as the single parameter to determine the actual coordinates. The function should return a matrix with two or three columns.

For the 2D plots the third column is ignored.

Let’s fix a random layout. Note: the layout is still random, what you do with this code is to produce the same result every time you use the plot function.

l <- layout.random(g)  
plot(g,layout = l)



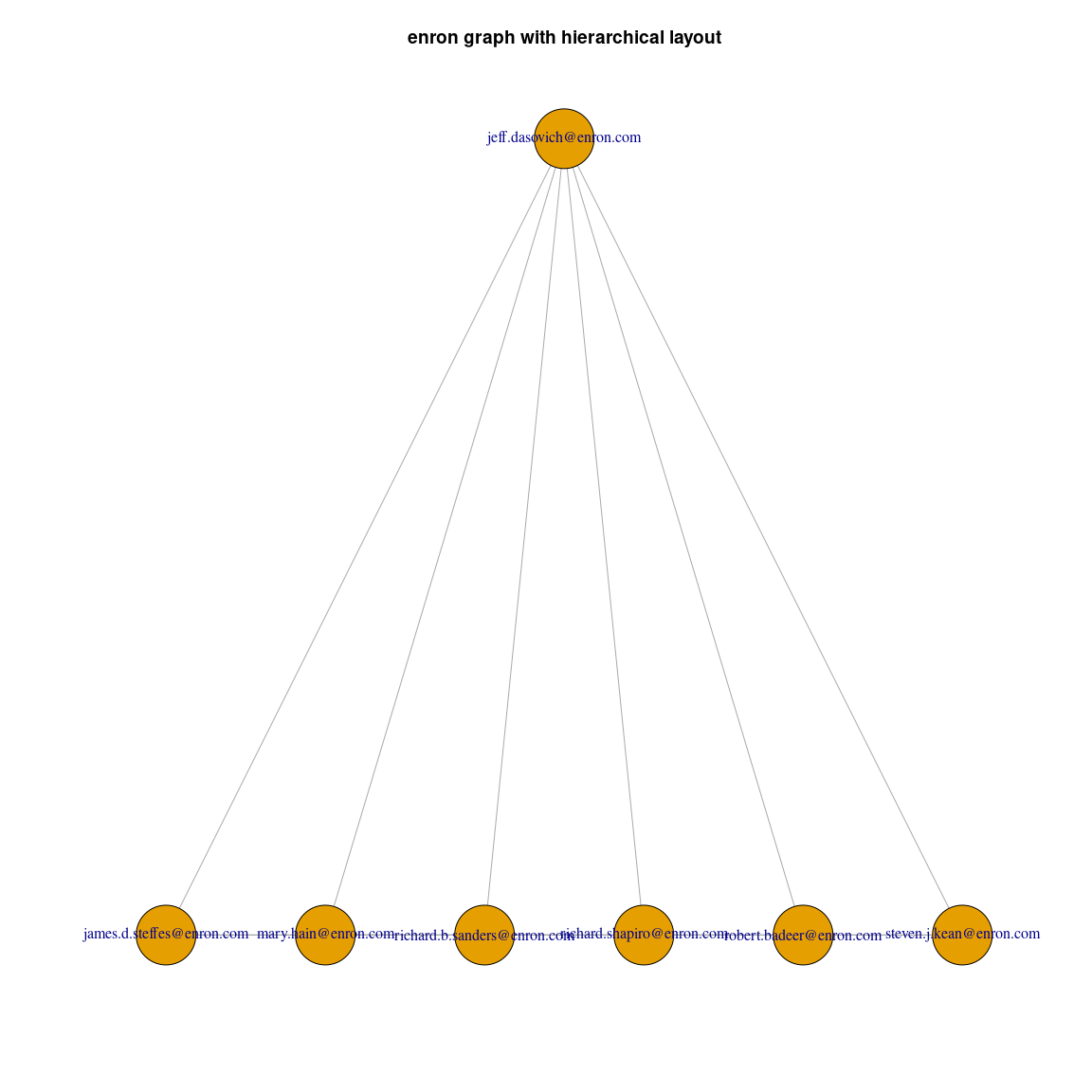
A preliminary conclusion: even for small graphs a random layout might not be the best representation.

Let’s check some other layouts.

### Reingold-Tilford layout

Reingold.tilford produces a hierarchical graph. Note! This function is deprecated, it might issue a warning.

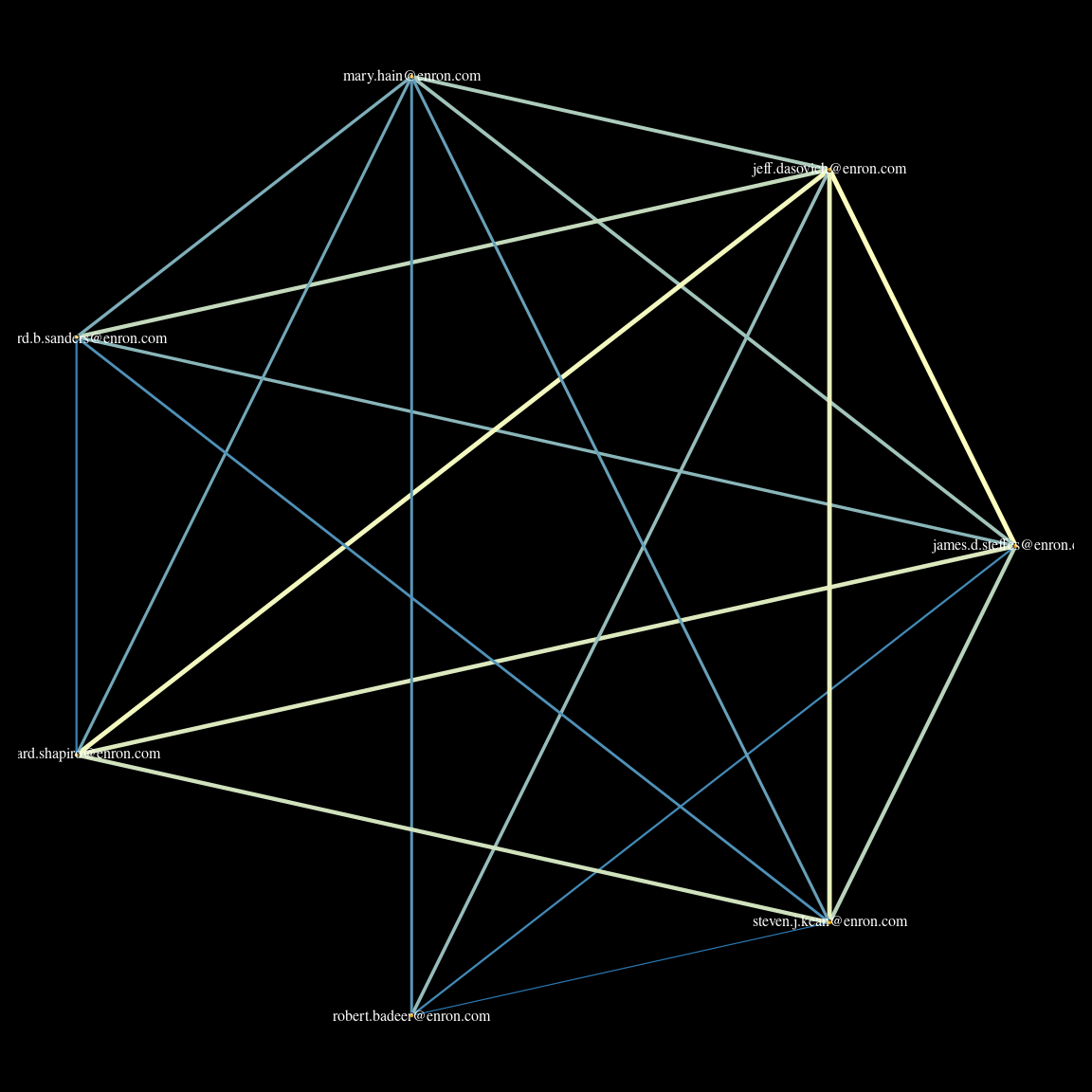
plot(g,   
 main = "enron graph with hierarchical layout",   
 layout = layout.reingold.tilford,  
 vertex.label = V(g)$label)



### Circle layout (Chord plot)

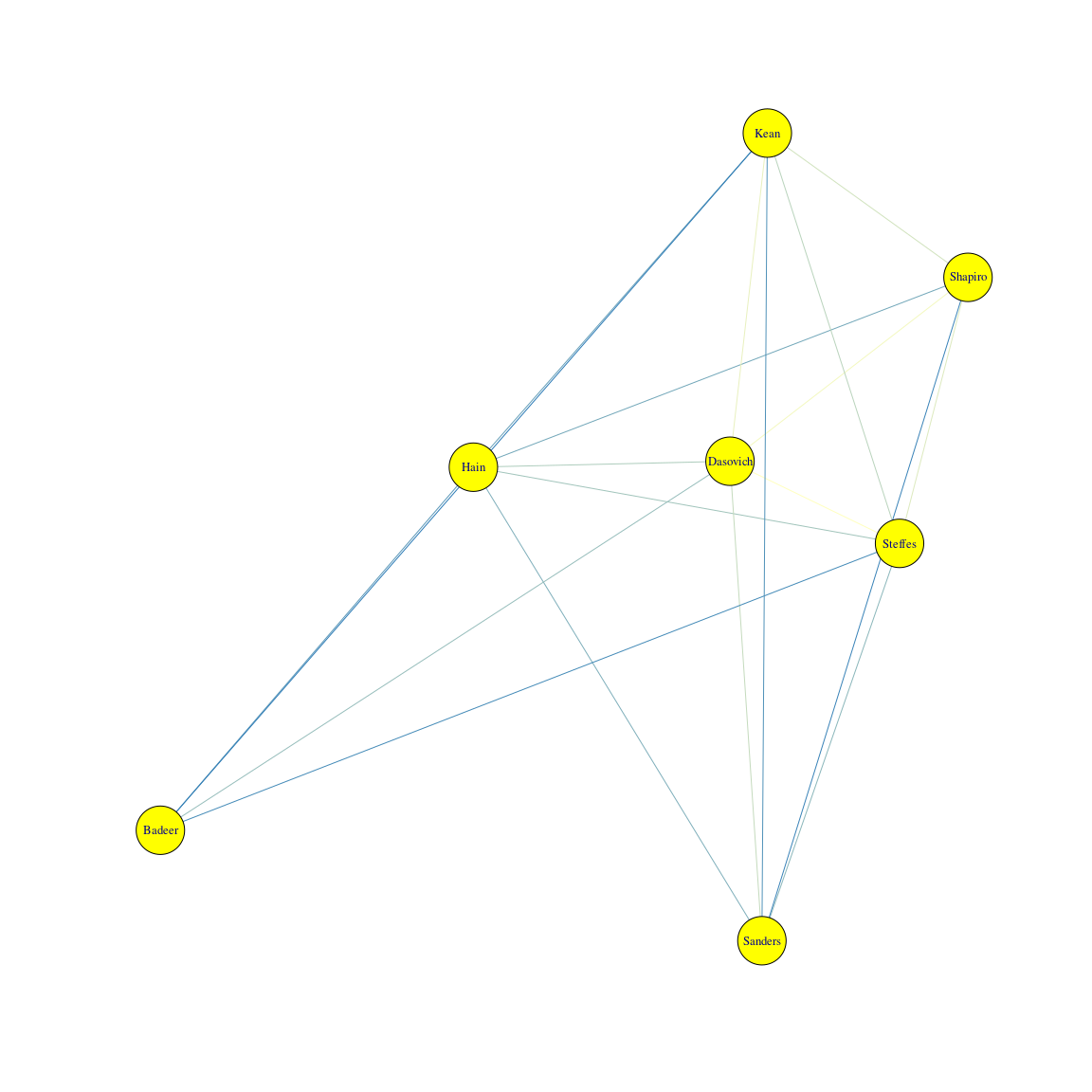
layout.circle produces circle or also known as chord plots. Note! This function is deprecated, it might issue a warning.

l <- layout.circle(g)  
  
# use colour functions  
par(bg = "#000000", mar = c(1,1,1,1), oma= c(1,1,1,1))  
  
edge\_col <- colorpanel(length(table(E(g)$weight)),   
 low = "#2C7BB6", high = "#FFFFBF")   
E(g)$color <- edge\_col[factor(E(g)$weight)]  
  
plot(g,   
 layout=l,  
 vertex.label = V(g)$label,  
 vertex.size=1,  
 vertex.label.color = "white",  
 edge.width = 1.5\*log10(E(g)$weight),   
 edge.curved = F,   
 edge.color = E(g)$color)



### Other layouts: fruchterman.reingold and kamada.kawai

# Let't try to improve the position of objets in plain  
# Let's put name as the node label  
  
V(g)$label <- V(g)$lastName  
  
# This is fruchterman reingold  
  
plot(g,   
 layout = layout.fruchterman.reingold,   
 vertex.label.font = 1,  
 vertex.label.cex = 0.8,   
 edge.arrow.size = 0.3,  
 vertex.size = 12,  
 vertex.color = "yellow")



# This is kamada.kawai  
  
plot(g,  
 layout=layout.kamada.kawai)

