# Príklady sietí, základné pojmy, základy práce so sieťami v R-ku

Beáta Stehlíková

2-EFM-155 Analýza sociálnych sietí

Fakulta matematiky, fyziky a informatiky, UK v Bratislave, 2019

# Sylabus

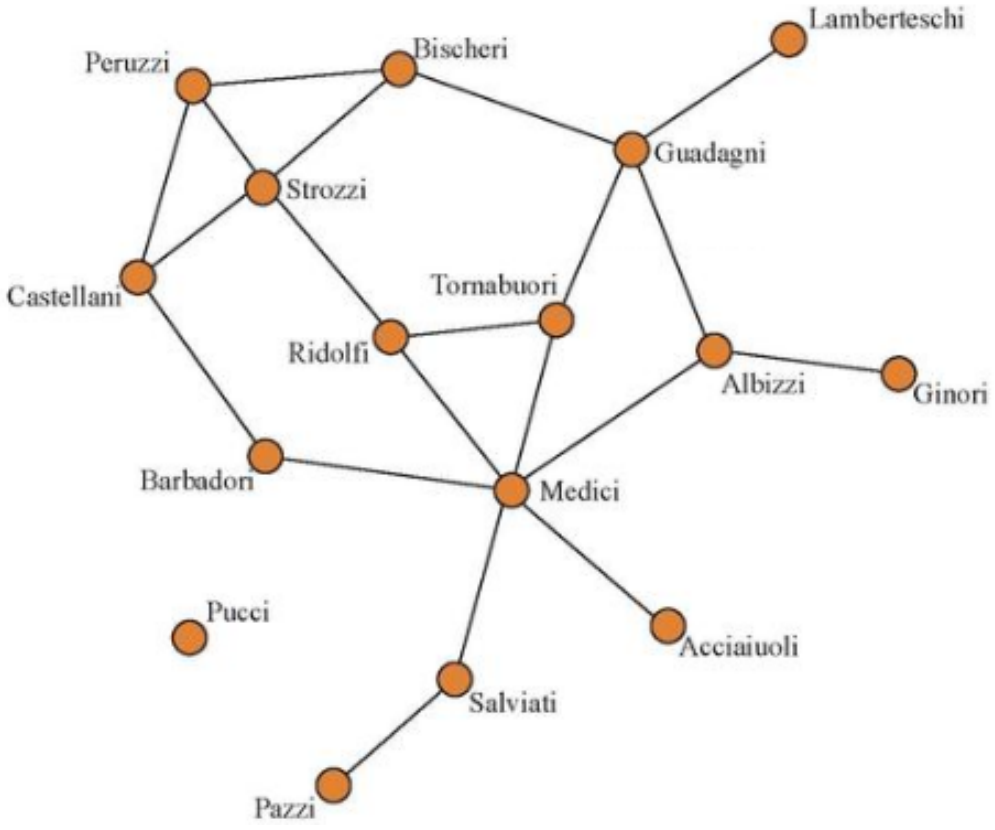
# Sylabus z informačného listu

- Základné pojmy z teórie grafov, príklady grafov/sietí, ich vizualizácia
- Miery centrality vrcholov
- Hľadanie komunít v sieti
- Siete založené na koreláciách
- Náhodné grafy a ich vlastnosti
- Základy štatistických modelov

# Príklady sietí

# Príklad 1: Manželstvá v renesančnej Florencii

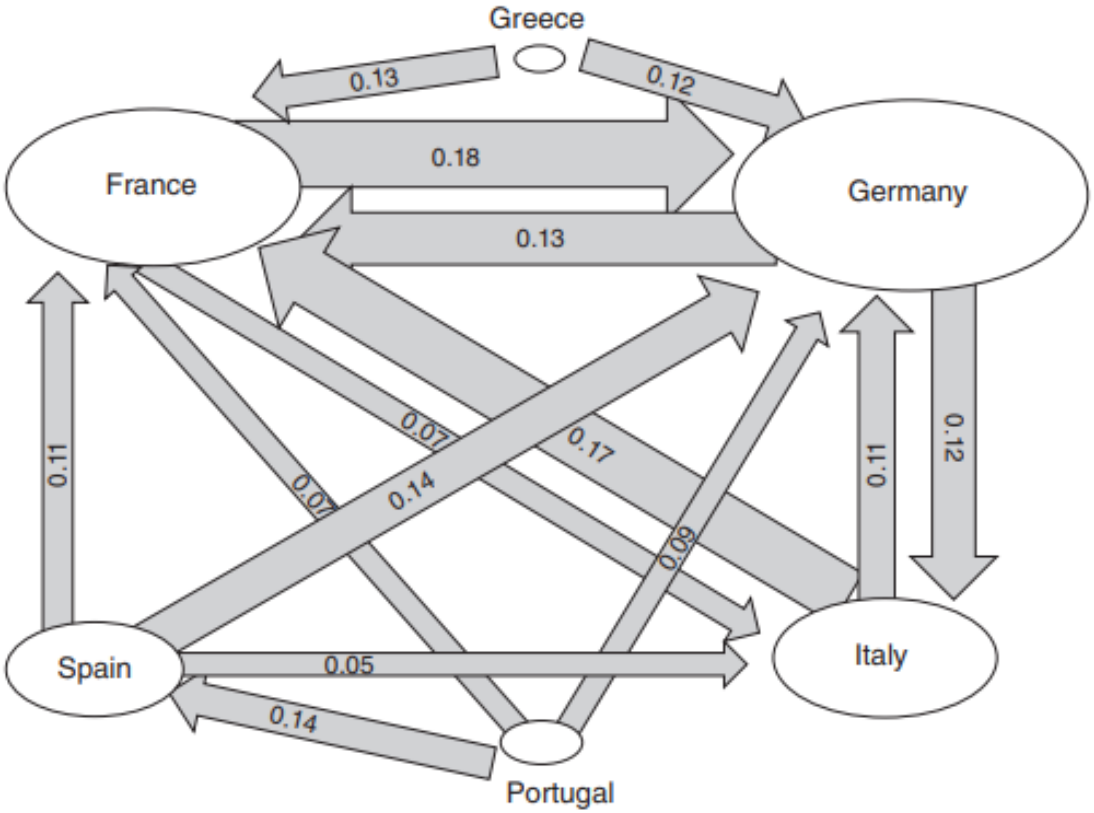
Manželstvá medzi významnými rodinami v renesančnej Florencii (15. storočie)




https://en.wikipedia.org/wiki/File:15th_Century_Florentine_Marriges_Data_from_Padgett_and_Ansell.pdf

# Príklad 2: Finančné siete

## Finančné siete, dôsledky zadĺženosti štátov




The matrix $A$, describing how much each country ultimately depends on the value of others' debt. The widths of the arrows are proportional to the sizes of the dependencies, with dependencies less than 5 percent excluded; the area of the oval for each country is proportional to its underlying asset values

Elliott, M., Golub, B., & Jackson, M. O. (2014). Financial networks and contagion. American Economic Review, 104(10), 3115-53.

# Príklad 2: Finančné siete

Námet na projekt:

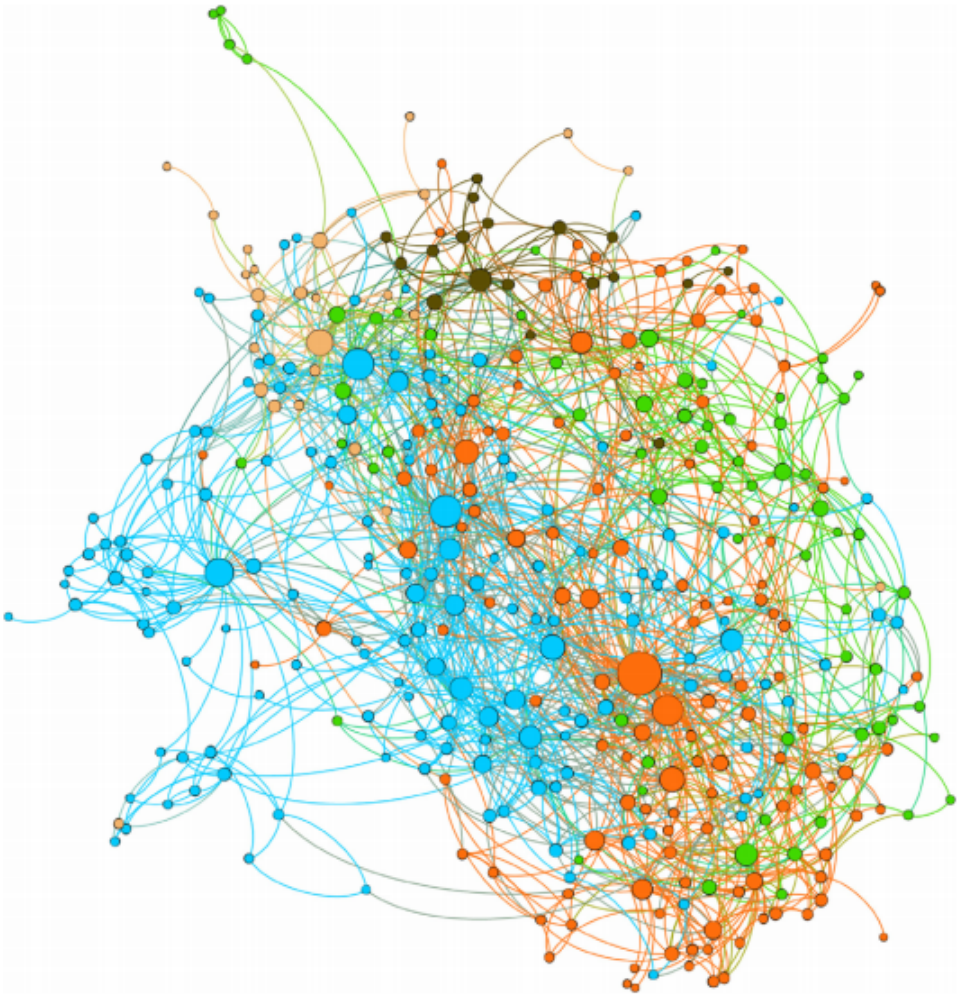
- na základe uvedeného článku vysvetliť, o akú sieť tu ide
- ako sa získa z matice dlhov
- čo a ako sa z nej dá získať
- samostatné zopakovanie výpočtov na základe vstupných dát
- pridať niečo vlastné (treba si premyslieť)

TABLE 1—HIERARCHIES OF CASCADES IN THE BEST-CASE EQUILIBRIUM ALGORITHM, AS A FUNCTION OF THE FAILURE THRESHOLD $\theta$

| Value of $\theta$ | 0.9 | 0.93 | 0.935 | 0.94 |
|---|---|---|---|---|
| First failure | Greece | Greece | Greece | Greece |
| Second failure | | | Portugal | Portugal, Spain |
| Third failure | | | Spain | France, Germany |
| Fourth failure | | | France | Italy |
| Fifth failure | | | Germany, Italy | |

*Source:* Authors' calculations

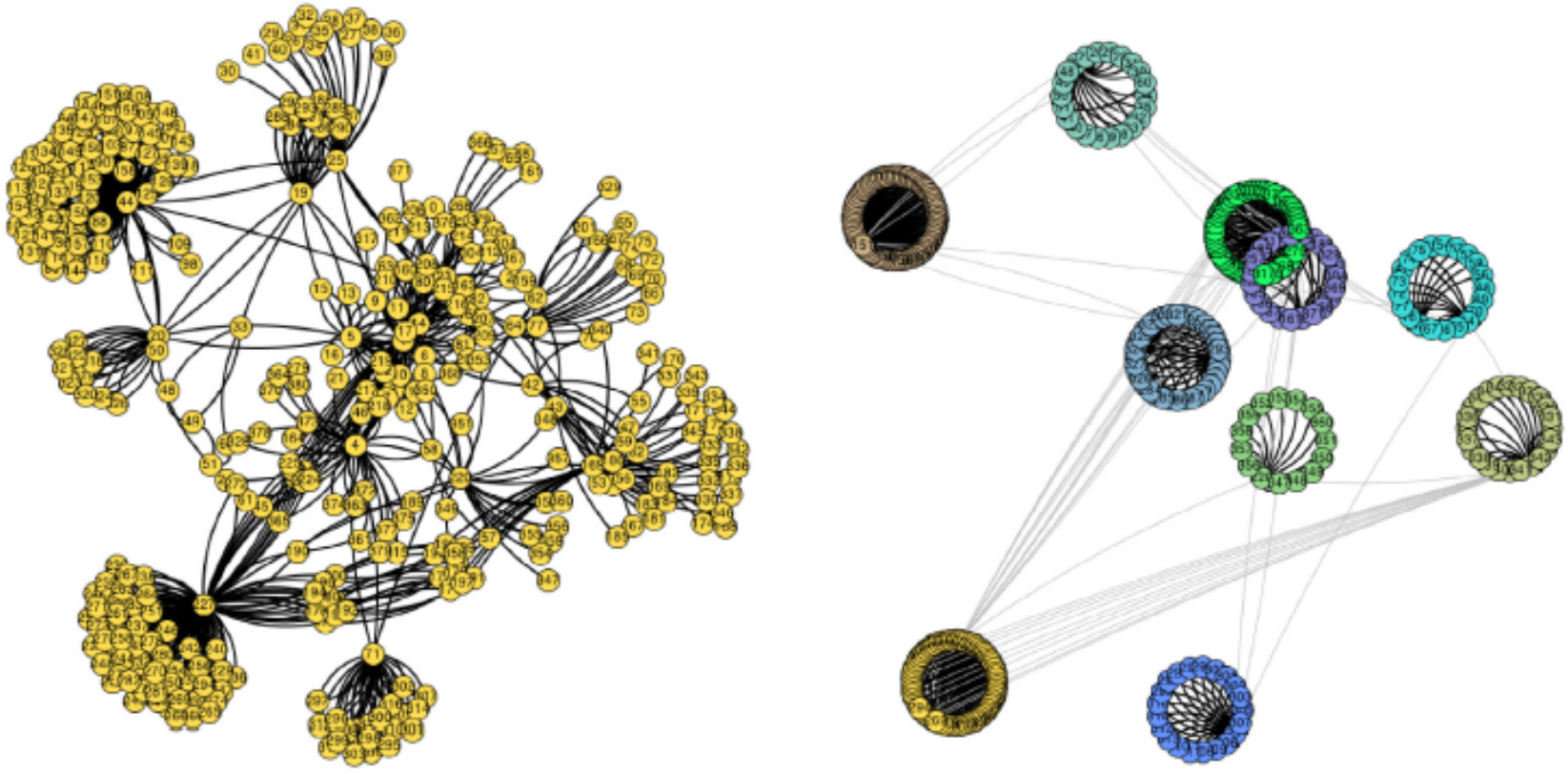
# Príklad 3: Štruktúra zločineckých organizácií



The Five Families of New York City. Note: Nodes are colored according to family membership and sized according to degree.

DellaPosta, D. (2017). Network closure and integration in the mid-20th century American mafia. Social Networks, 51, 148-157.

# Príklad 4: Zobrazenie komunikácie



(a) *Phone call network of 148 nodes and 210 edges.*

(b) *Clusters detected after 46 edges deleted.*

Ferrara, E., De Meo, P., Catanese, S., & Fiumara, G. (2014). Detecting criminal organizations in mobile phone networks. Expert Systems with Applications, 41(13), 5733-5750.

# Základné pojmy

- Graf, sieť (*graph, network*)
- Vrchol (*vertex, node*)
- Hrana (*edge, tie*)
- Hrany môžu byť orientované/neorientované (*oriented/unoriented*), vážené/nevážené(*weighted/unweigted*), ...
- Vrcholy a hrany môžu mať atribúty (*attributes*)

# Siete v sofvéri R: knižnice

- Budeme pracovať s knižnicou `igraph`.
- Niektoré príklady sietí budú z knižnice `igraphdata`.
- Ak treba, nainštalujte knižnice. Potom ich načítame:

```
library(igraph)
library(igraphdata)
```

# Siete v sofvéri R: príklady, vizualizácia a ukážky analýz

- Niekoľko konkrétnych sietí rôznych typov - náhodne generované, načítané zo súboru, už dostupné v R-ku
- Nakreslíme ich a upravíme obrázky pomocou dostupných parametrov
- Spravíme jednoduché analýzy, o ktorých budeme podrobnejšie hovoriť neskôr počas semestra

# Príklad 1: Náhodné grafy Erdősa a Rényiho

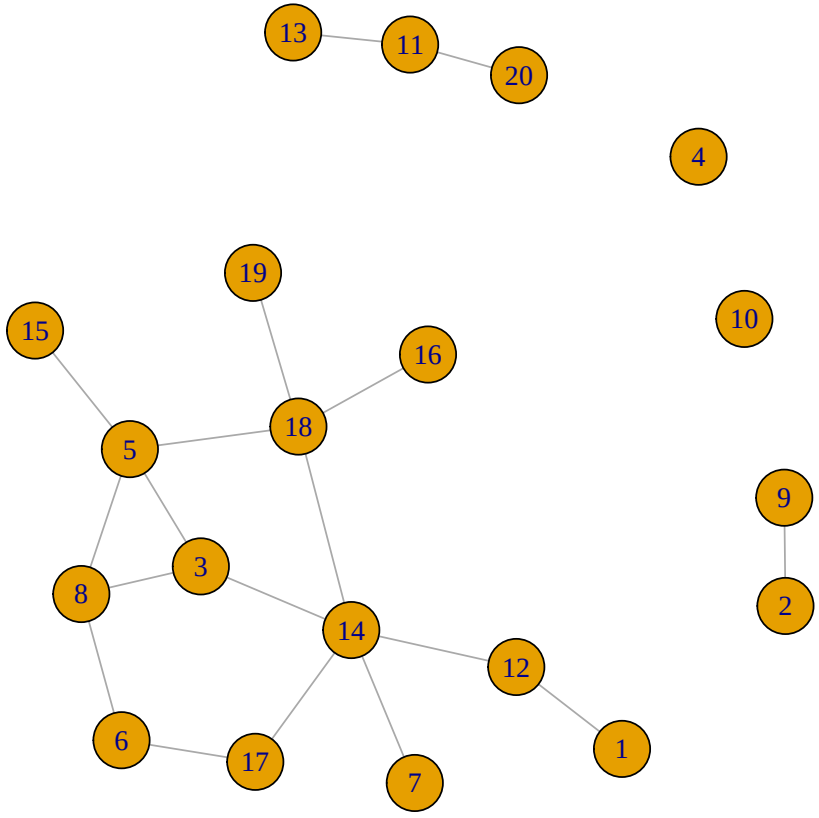
# Definícia a generovanie náhodného grafu v R

- Pametere:
  - $n$ = počet vrcholov
  - $p \in (0, 1)$ = pravdepodobnosť vzniku hrany
- Hrany vznikajú nezávisle na sebe
- V R-ku: funkcia `erdos.renyi.game`

```
set.seed(123)  # kvoli reprodukovatelnosti
g <- erdos.renyi.game(n = 20, p = 0.1)
plot(g)
```

# Definícia a generovanie náhodného grafu v R

*Úpravy 1:* chceli by sme hrany hrubšou čiarou a výraznejšou farbou (napríklad hnedou)



# Parametre funkcie plot

*Základný princíp:*

- parametre týkajúce sa hrán majú tvar `edge. ...`, napr. `edge.color =`
- parametre týkajúce sa hrán majú tvar `vertex. ...`, napr. `vertex.size =`

*Prehľad:*

- http://kateto.net/netscix2016 v kapitole 5.1
- priamo v R-ku pomocou `?igraph.plotting`

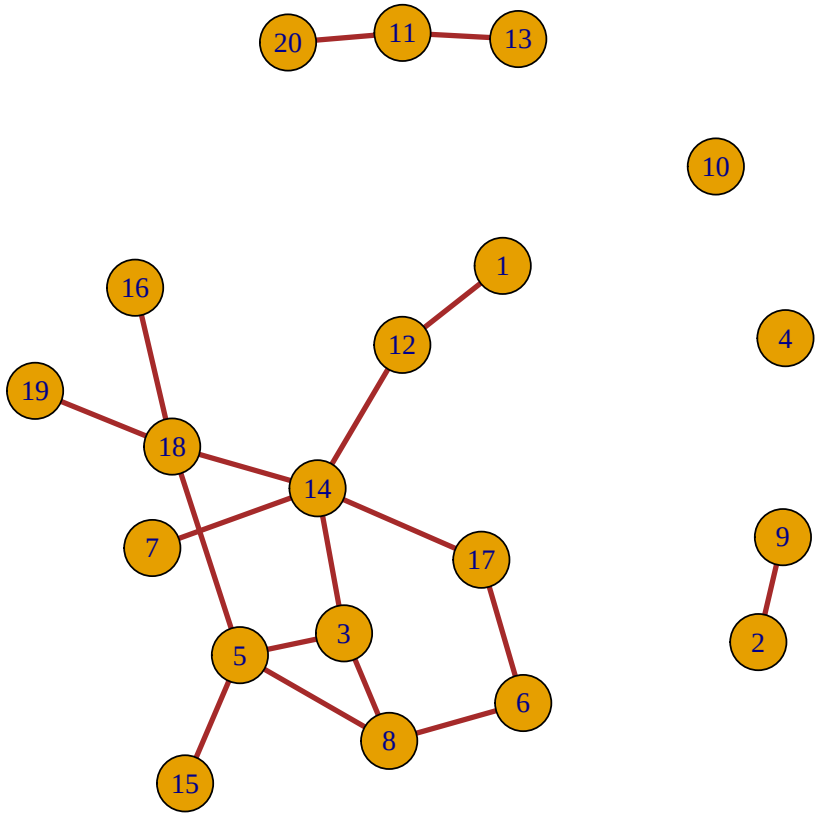
*V našom prípade:*

- chceme hrubšiu čiaru znázorňujúcu hranu: nastavíme parameter `edge.width` (prednastavená hodnota je 1, vyskúšame vyššie)
- chceme hnedú čiaru: nastavíme `edge.color` na `"brown"`

```
plot(g, edge.width = ..., edge.color = "brown")
```

# Parametre funkcie plot

Výstup môže byť napríklad:



# Parametre funkcie plot

*Úpravy 2:* Všimnime si, že pri opätovnom kreslení nemusia byť vrcholy rozmiestnené rovnako. Stabilizujme preto rozmiestnenie vrcholov.

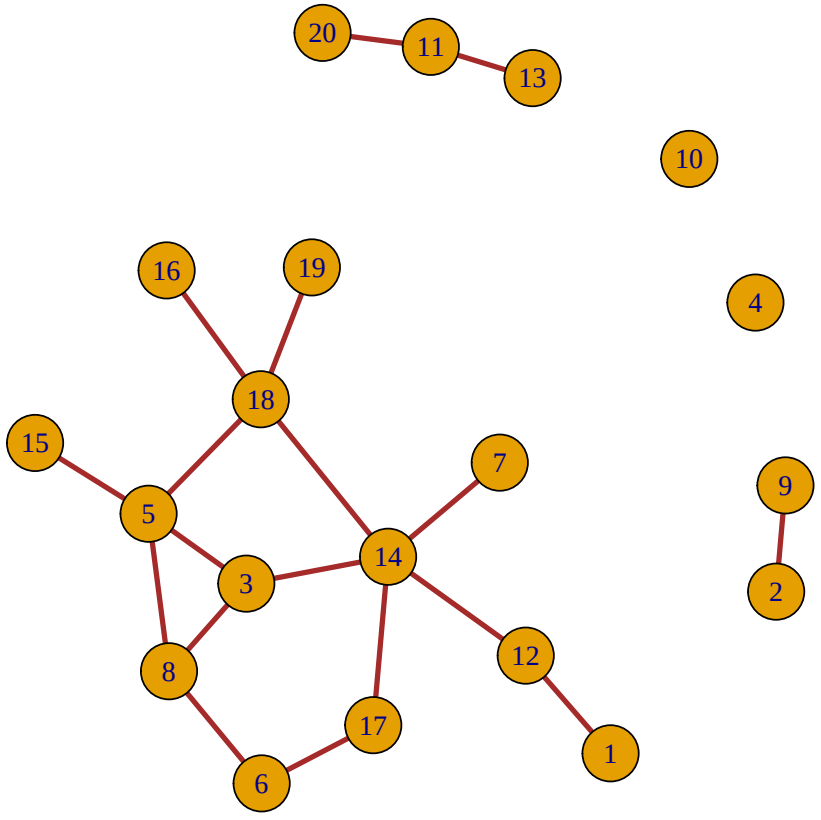
*Postup:*

- Tento fakt je dôsledkom náhodnosti algoritmu, ktorý počíta polohu vrcholov.
- Zvolíme konkrétny algoritmus a voľbu náhodných čísel pomocou `set.seed`
- Budeme potrebovať parameter `layout`, zvolíme napríklad metódu `layout_with_graphopt` (oplatí sa vyskúšať ich niekoľko a vybrať tú, pri ktorej sa nám výstup najviac páči) alebo to necháme na R-ko voľbou `layout_nicely`

```
set.seed(2019)  # kvoli nahodnosti algoritmu
plot(g, edge.width = ..., edge.color = "brown",
     layout = ...)
```

# Parametre funkcie plot

Napríklad:



# Parametre funkcie plot

*Úpravy 3:* Vidíme, že graf sa rozpadá na niekoľko súvislých podgrafov, tzv. komponentov súvislosti. Chceli by sme ich farebne odlíšiť.

*Riešenie:*

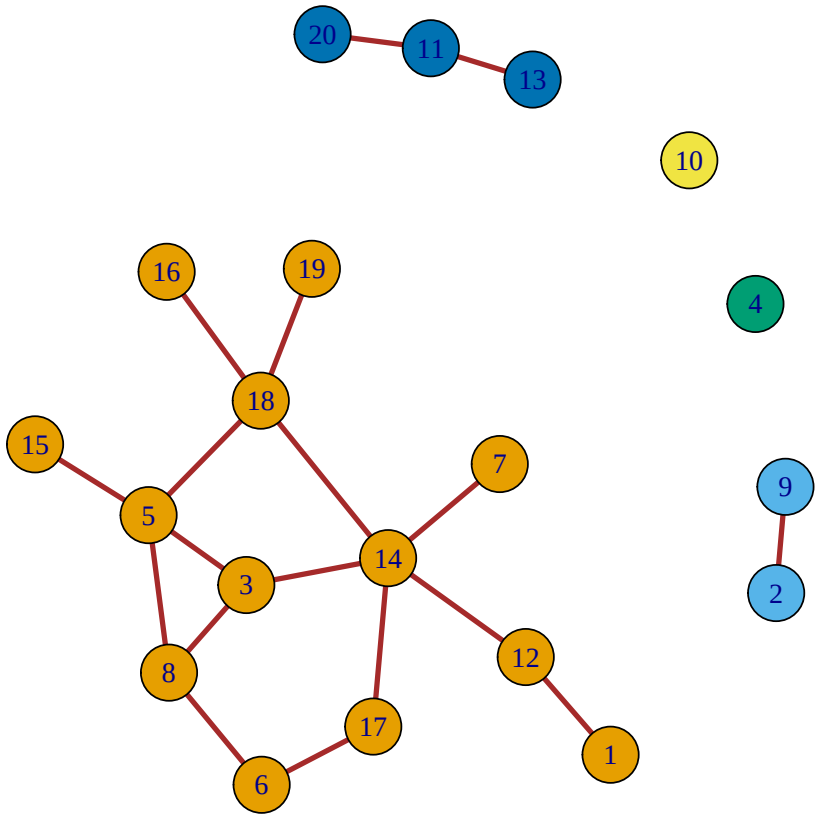
- potrebujeme zmeniť hodnotu `vertex.color`
- užitočná informácia je, že sa pripúšťa aj číselná hodnota, skúste napríklad `vertex.color = 5`
- na určenie komponentov použijeme funkciu `componenets`, ktorá má ako vstupný parameter študovanú sieť

```
components(g)
```

```
## $membership
##  [1] 1 2 1 3 1 1 1 1 2 4 5 1 5 1 1 1 1 1 1 5
##
## $csize
## [1] 13  2  1  1  3
```

# Parametre funkcie plot

- Ako hodnotu `vertex.color` teda môžeme zobrať `components(g)$membership`



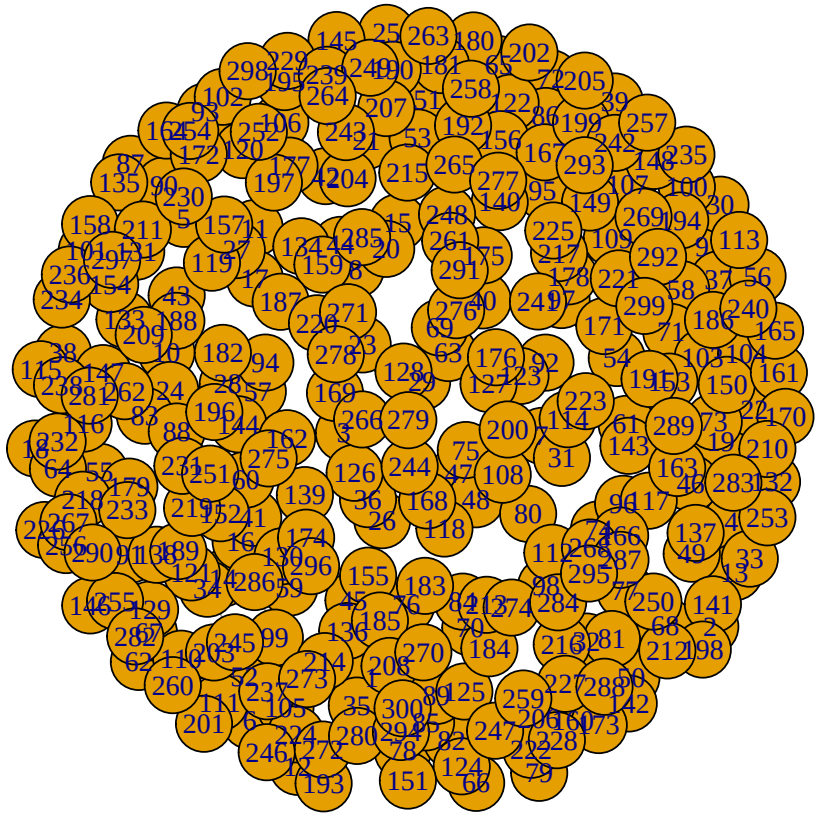
# Cvičenie

Pri veľkom počte vrcholov nie je zobrazenie grafu s očíslovanými vrcholmi prehľadné. Zrušte preto pomocou `vertex.label = NA` označenie vrcholov a spravte podľa vlastného uváženia ďalšie úpravy v zobrazení nasledujúcej siete:

```
set.seed(123)
g <- erdos.renyi.game(n = 300, p = 0.005)
plot(g)
```

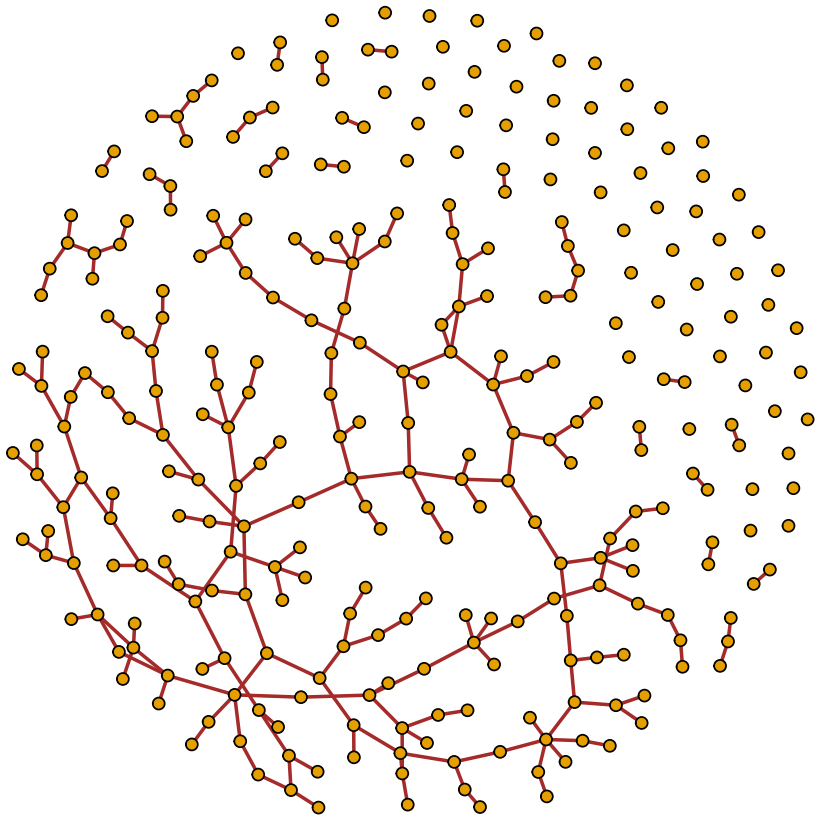
# Cvičenie

Pôvodné zobrazenie:



# Cvičenie

Ukážka upraveného obrázku (váš môže byť iný):



# Príklad 2: Kradnuté autá

# Dáta

Budeme pracovať s dátami zo stránky https://sites.google.com/site/ucinetsoftware/datasets/covert-networks/togo (zo školskej siete je dostupná aj kniha, v ktorej boli uverejnené)

Základná informácia zo stránky:

*Project Togo began in February 1998 when a Toronto-based ringing operation was dismantled and one of its participants informed the police that he was previously employed by a Montreal businessman who was also active in the resale of stolen vehicles. This initial tip was corroborated soon after by a thief who had been arrested while driving a stolen vehicle. By December 1998, the Togo investigation was under way. It spanned into February 1999 and 20 cars that were destined for France, Ghana, and local buyers in southern Quebec were retrieved.*

# Dáta

Popis dát zo stránky:

- *1-mode matrix 33 x 33 person by person. Undirected ties.*
- *Ties are communication exchanges between criminals.*
- *Data comes from police wiretapping.*

Jeden z dostupných formátor je CSV, ten vieme načítať do R-ka:

```
data <-  read.csv("TOGO.csv",
          header = TRUE, # prvy riadok je hlavicka
          check.names = FALSE, # nazvy stlpcov
                               # zostanu 1, 2, 3,...
                               # inak by bolo X1, X2, ...
          row.names = 1) # prvy stlpec obsahuje
                         # nazvy riadkov
```

*Otázka:* Ako z toho spraviť sieť?

# Matica susednosti

**Matica susednosti** (*adjacency matrix*) pre nevážený neorientovaný graf - má v i-tom riadku a j-tom stĺpci

- hodnotu 1, ak sú vrcholy i, j spojené hranou
- inak má hodnotu 0

Pre iné grafy:

- Ak je graf vážený, namiesto hodnoty 1 je v matici váha príslušnej hrany.
- Ak je graf orientovaný, $A_{ij} = 1$, ak existuje hrana z vrcholu i do vrcholu j; analogicky vážené grafy

# Vytvorenie siete z matice susednosti

**V našom prípade:**

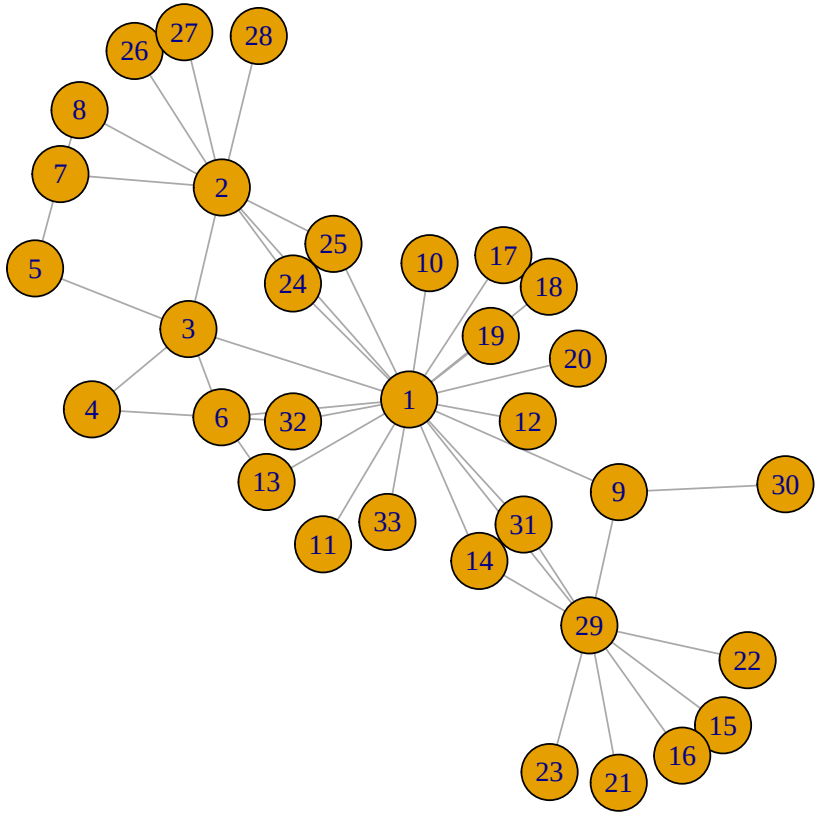
- R-ko má funkciu `graph_from_adjacency_matrix`
- Z dát uložených v premennej `data` spravíme maticu
- Špecifikujeme, že má vzniknúť neorientovaný nevážený graf
- Mená vrcholov sa automaticky zoberú z mien stĺpcov matice `A`

```
A <- as.matrix(data)

g <- graph_from_adjacency_matrix(A,
       mode = "undirected",   # neorientovany
       weighted = NULL        # nevazeny
)

plot(g)
```

# Vytvorenie siete z matice susednosti



# Centralita vrcholov

**Aká je centralita (dôležitosť) vrcholov siete?** Teda: **Aká je centralita (dôležitosť) ľudí, ktorých predstavujú?**

Rôzne pohľady na to, čo znamená centralita:

- S koľkými vrcholmi je daný vrchol spojený?
- "Ako rýchlo" sa informácia od neho dostane k ostatným vrcholom (resp. naopak - od ostatných k nemu)?
- Ako často sa vyskytuje v najkratších cestách, ktoré spájajú dva vrcholy?

Teraz len základné myšlienky pre neorientované nevážené grafy, podrobnosti a ďalšie miery centrality neskôr.

# Centralita vrcholov

**Námet na projekt:** Centralita v bipartitných grafoch + zaujímavé aplikácie

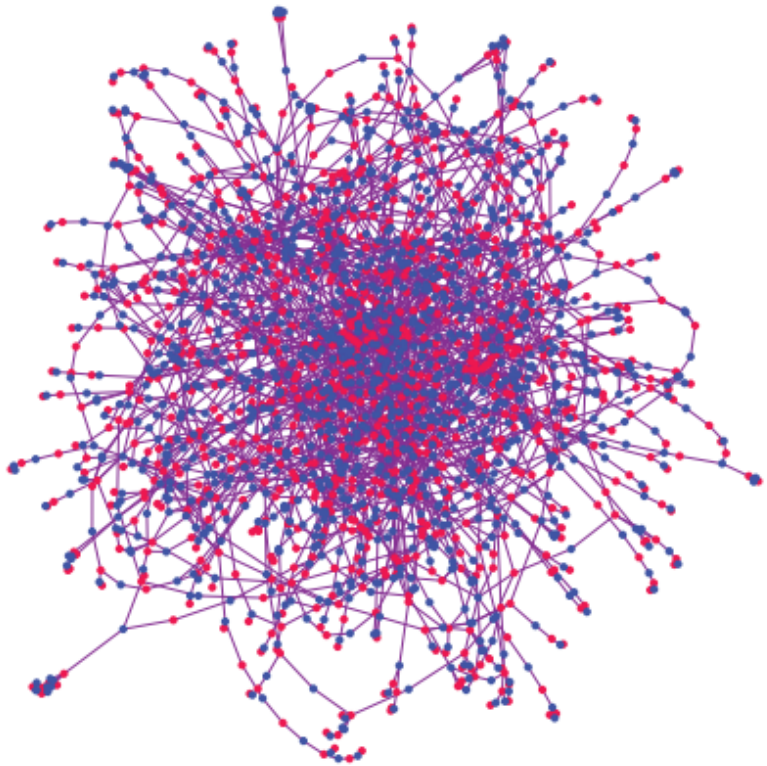


**Fig. 1.** The giant component of the affiliation network after removing director nodes with degree one, in which the red filled circle represents company and blue circle represents director nodes. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of the article.)

Sankar, C. P., Asokan, K., & Kumar, K. S. (2015). Exploratory social network analysis of affiliation networks of Indian listed companies. Social Networks, 43, 113-120.

# Centralita stupňa

**Stupeň vrchola** (*degree*) - počet hrán, ktoré vychádzajú z vrchola (pri orientovaných sa rozlišuje počet hrán, ktoré vchádzajú a ktoré vychádzajú)

Funkcia `degree`:

- ako vstup dostane graf
- výstupom je vektor s hodnotami stupňov jednotlivých vrcholov

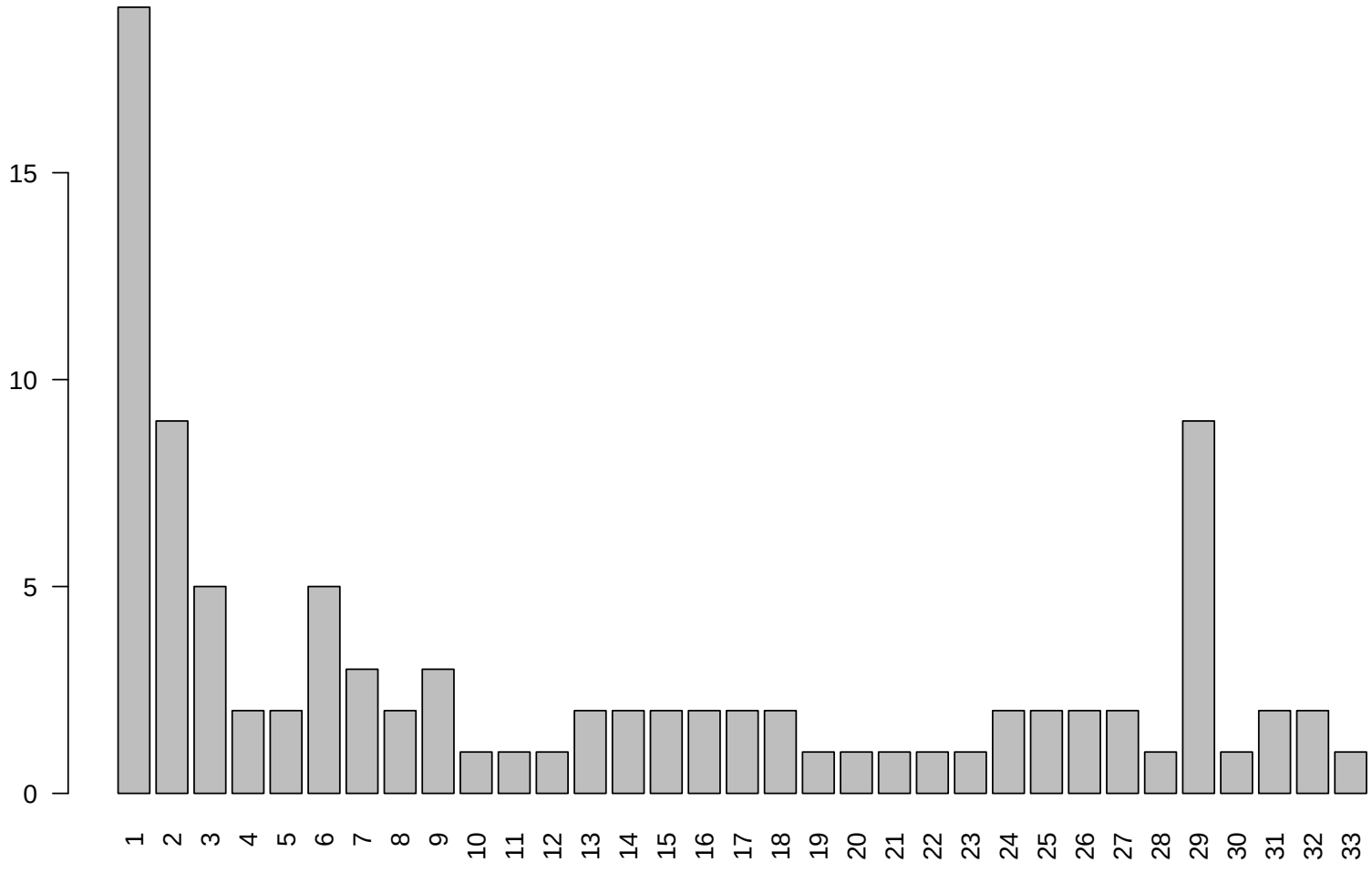
```
degree(g)
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19
## 19  9  5  2  2  5  3  2  3  1  1  1  2  2  2  2  2  2  1
## 26 27 28 29 30 31 32 33
##  2  2  1  9  1  2  2  1
```

V prvom riadku je názov vrchola, v druhom riadku príslušný stupeň

# Centralita stupňa

```
barplot(degree(g), las=2)
```



# Centralita blízkosti a medzipolohy

**Centralita blízkosti (closeness)**

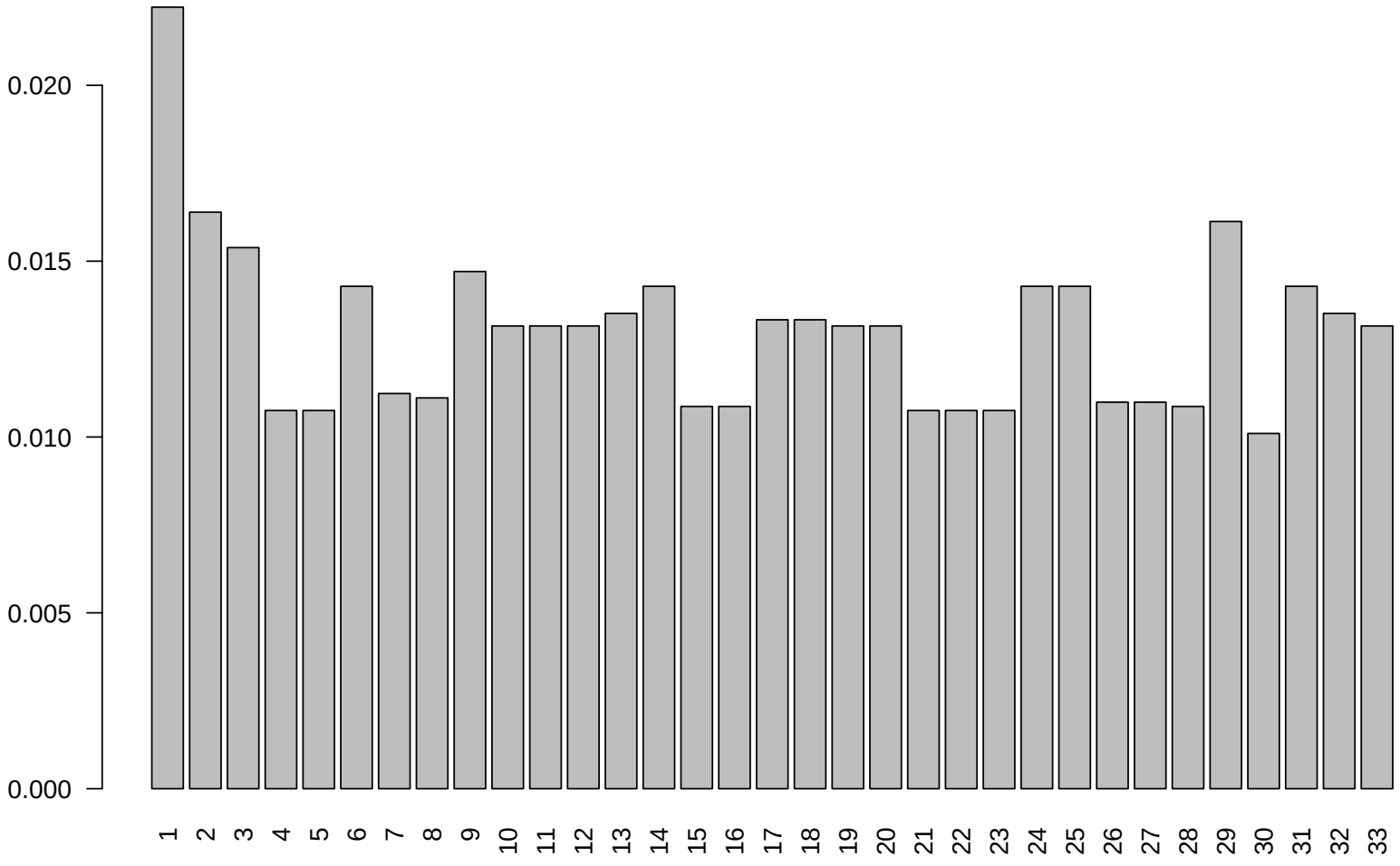
- Vzdialenosť vrcholov $i$ a $j$ definujeme ako dĺžku najkratšej cesty (počet hrán v ceste), ktorá ich spája, ozn. $d(i,j)$
- Centralita blízkosti vrchola $i$ je nepriamo úmerná $\sum_{j \neq i} d(i,j)$
- V R-ku funkcia `closeness`

**Centralita medzipolohy (betweenness)**

- $P(i,j) =$ počet najkratších ciest medzi $i$ a $j$
- $P_k(i,j) =$ počet najkratších ciest medzi $i$ a $j$, ktoré obsahujú vrchol $k$
- Centralita medzipolohy brchola $k$ je priamo úmerná $\sum_{i,j \neq k} P_k(i,j)/P(i,j)$
- V R-ku funkcia `betweenness`

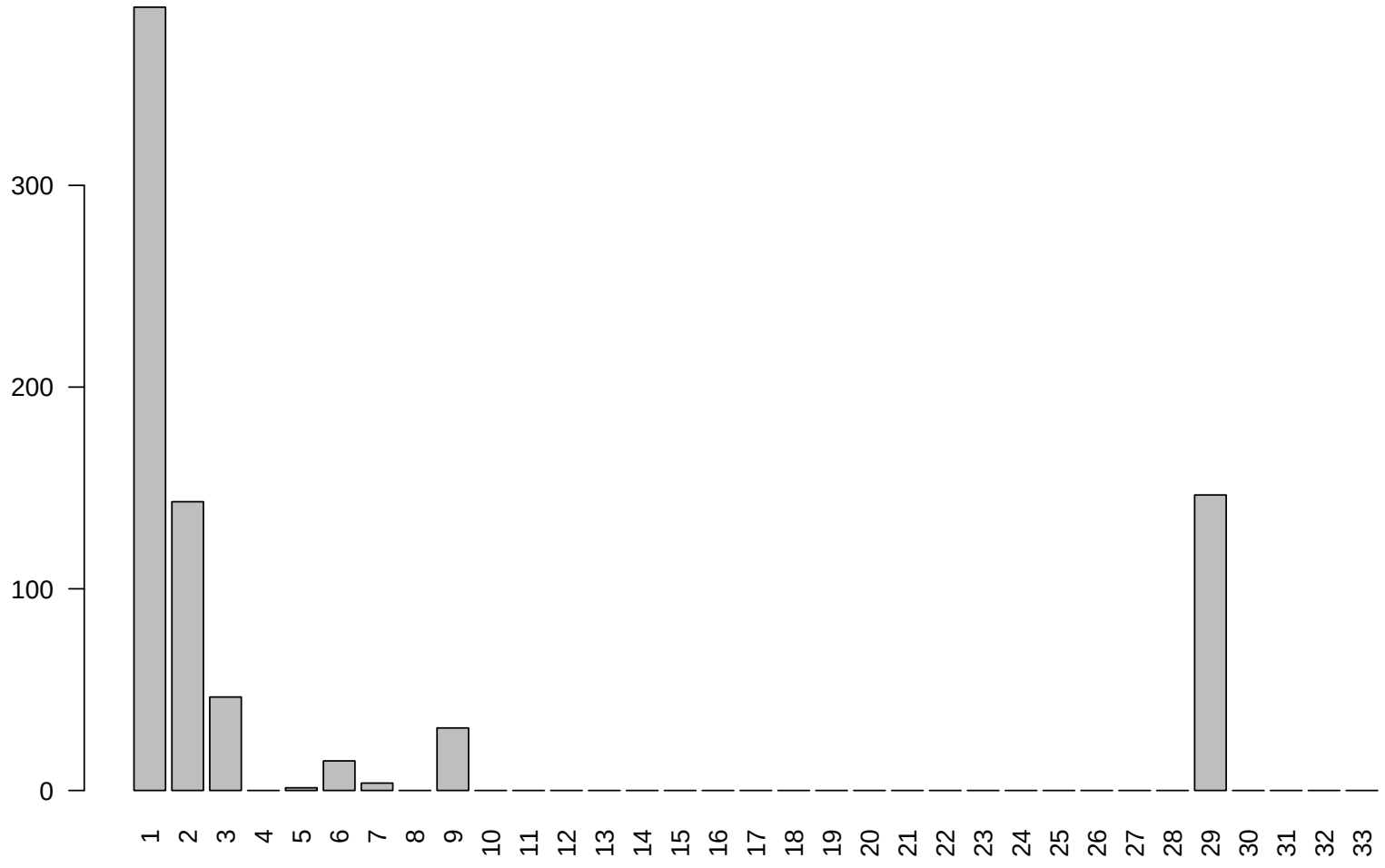
# Centralita blízkosti

```
barplot(closeness(g), las=2)
```



# Centralita medzipolohy

```
barplot(betweenness(g), las=2)
```



# Príklad 3: Zacharyho karate klub

# Dáta

Pozrieme sa na sieť Zacharyho karate klubu pomocou knižnice `igraphdata`

```
data(karate)  # nacitanie dat, t.j. siete
g <- karate   # graf vlozime do premennej `g`
```

# Dáta

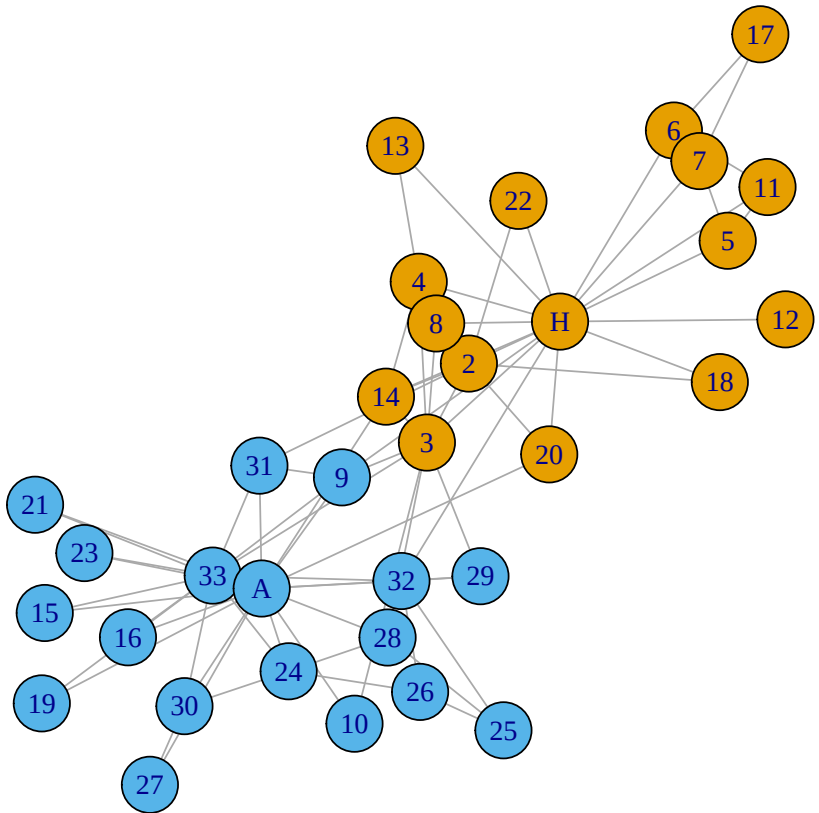
Informáciu o dátach zobrazíme pomocou `?karate`

- *Social network between members of a university karate club, led by president John A. and karate instructor Mr. Hi (pseudonyms).*
- *The edge weights are the number of common activities the club members took part of.*
- *Zachary studied conflict and fission in this network, as the karate club was split into two separate clubs, after long disputes between two factions of the club, one led by John A., the other by Mr. Hi.*
- *The* `Faction` *vertex attribute gives the faction memberships of the actors.*

# Grafické zobrazenie

Nakreslíme graf (bez špecifikovania parametrov, použijú sa defaultne alebo už definované v grafe):

```
plot(g)
```



# Vrcholy a hrany

Pozrieme sa na vrcholy (*vertices*, preto `V`) a hrany (*edges*, preto `E`) nášho grafu:

```
V(g)
```

```
## + 34/34 vertices, named, from 4b458a1:
##  [1] Mr Hi    Actor 2  Actor 3  Actor 4  Actor 5  Actor
##  [8] Actor 8  Actor 9  Actor 10 Actor 11 Actor 12 Actor
## [15] Actor 15 Actor 16 Actor 17 Actor 18 Actor 19 Actor
## [22] Actor 22 Actor 23 Actor 24 Actor 25 Actor 26 Actor
## [29] Actor 29 Actor 30 Actor 31 Actor 32 Actor 33 John A
```

```
E(g)
```

```
## + 78/78 edges from 4b458a1 (vertex names):
##  [1] Mr Hi  --Actor 2  Mr Hi  --Actor 3  Mr Hi  --Actor
##  [4] Mr Hi  --Actor 5  Mr Hi  --Actor 6  Mr Hi  --Actor
##  [7] Mr Hi  --Actor 8  Mr Hi  --Actor 9  Mr Hi  --Actor
```

# Informácia o sieti

```
summary(g)
```

```
## IGRAPH 4b458a1 UNW- 34 78 -- Zachary's karate club netwo
## + attr: name (g/c), Citation (g/c), Author (g/c), Factio
## | name (v/c), label (v/c), color (v/n), weight (e/n)
```

4 znaky charakterizujú graf - v našom prípade **UNW-**

1. **D** - *directed*, **U** - undirected
2. **N** - *named*, ak majú vrcholy definovaný atribút `name`
3. **W** - *weighted*, ak majú hrany definovaný atribút `weight`
4. **B** - *bipartite*, vrcholy majú definovaný atribút `type`, ide o tzv. bipartitný graf

Nasleduje počet vrcholov a hrán, názov grafu (ak ho graf má) a informácia o atribútoch

# Atribúty

Atribúty - čoho sa týkajú:

- grafu (**g** - *graph*)
- vrcholov (**v** - *vertex*)
- hrán (**e** - *edge*)

a akého sú typu:

- **c** - *character*
- **n** - *numeric*
- **l** - *logical*
- **x** - iné

Napríklad `weight` je atribút hrany (**e**) a je to číslo (**n**).

Pozrite si konkrétne hodnoty atribútov:

```
graph.attributes(g)
vertex.attributes(g)
edge.attributes(g)
```

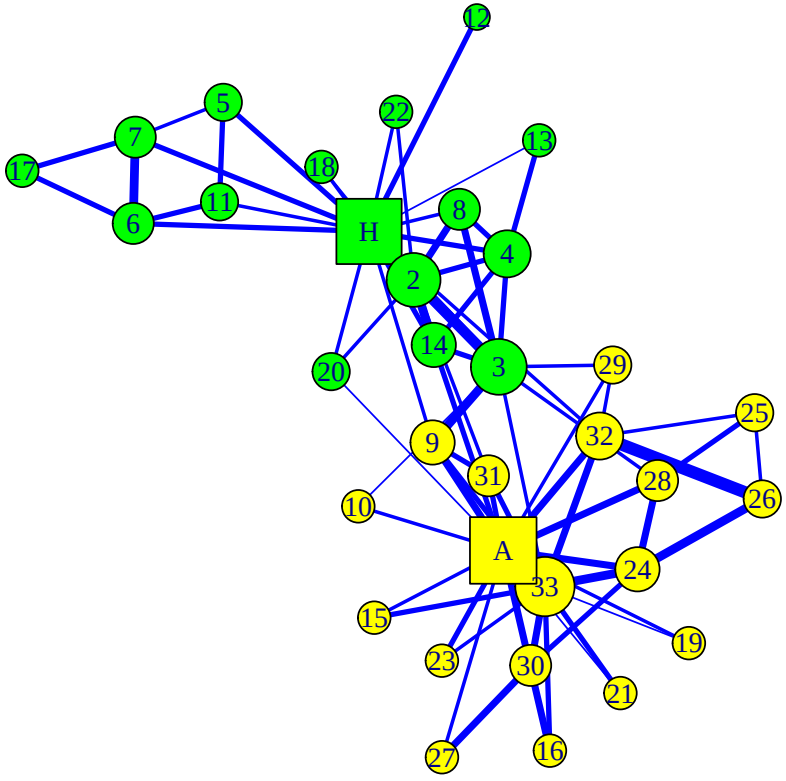
# Cvičenie

Upravme obrázok so sieťou nasledovne:

- Hrany budú mať modrú farbu a hrúbka hrán bude úmerná váhe
- Zmeňme farbu vrcholov na zelenú a žltú
- Vrcholy *Mr. Hi* a *John A* budú mať tvar štvorca
- Veľkosť vrchola bude závisieť od počtu hrán, ktoré z neho vychádzajú (viac hrán $\rightarrow$ väčší vrchol grafu)

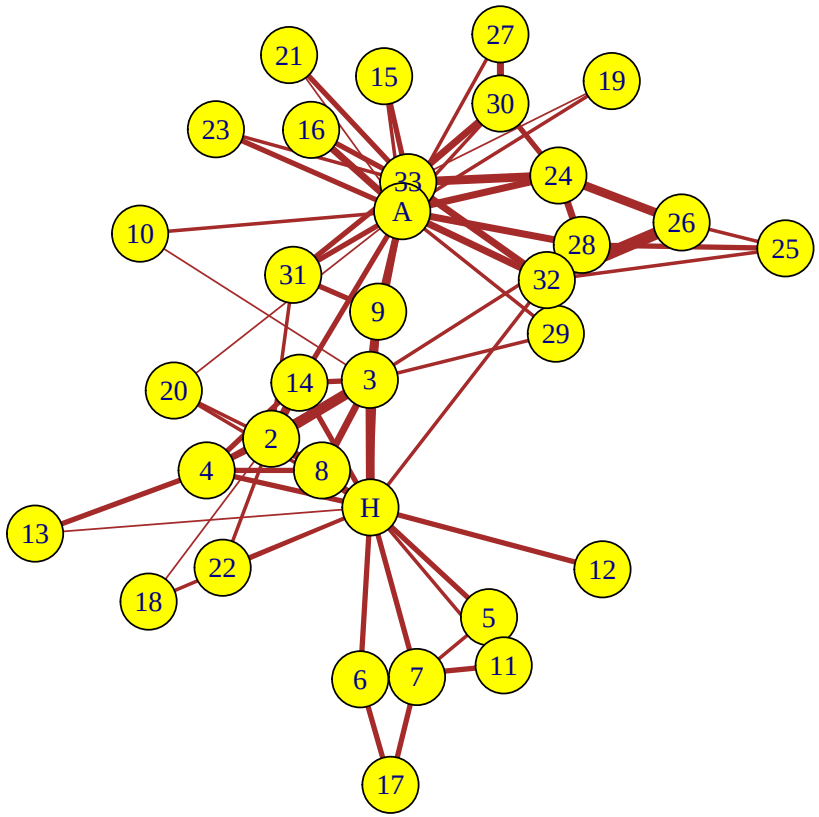
# Cvičenie

Ukážka možného výstupu:



# Hľadanie komunít (zhlukovanie) v sieťach

- Zobrazme si sieť vzťahov v klube bez informácie o tom, ako sa nakoniec klub rozdelil, pričom zobrazíme silu kontaktov
- *Dalo by sa rozdelenie klubu predpovedať?*

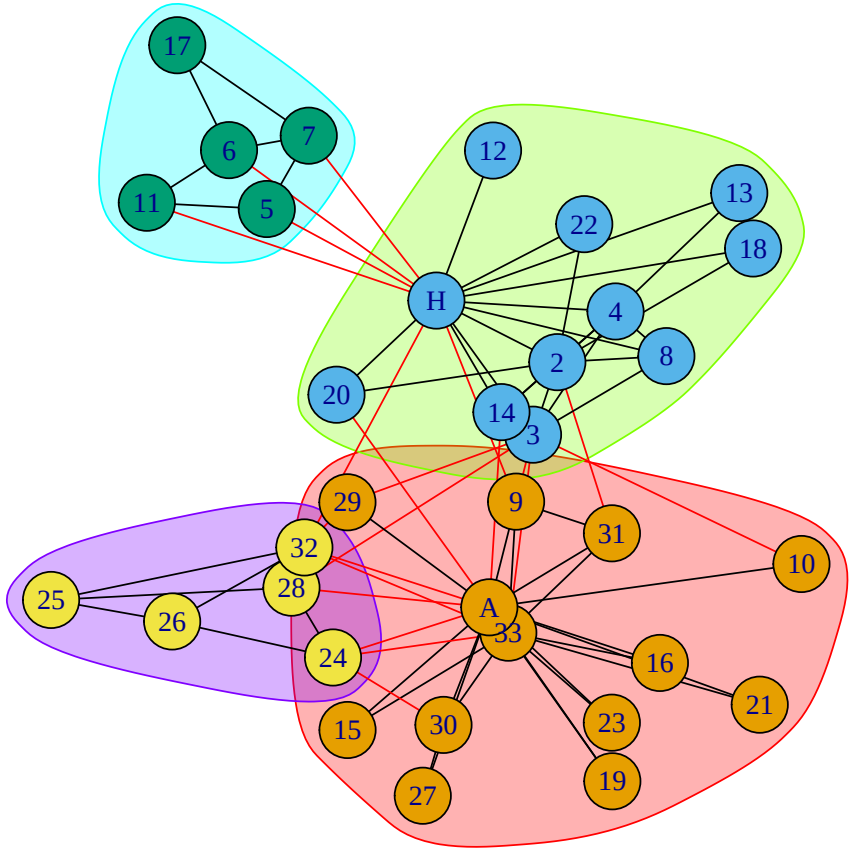


# “Walktrap” algoritmus

- Existuje veľa algoritmov na hľadanie komunít, resp. zhlukov v sieťach - budeme sa nimi zaoberať
- Na ukážku: funkcia `cluster_walktrap`
- Základná myšlienka algoritmu: pri krátkej náhodnej prechádzke po hranách grafu sa dá očakávať, že zostaneme v tej istj komunite (v tom istom zhluku)

```
zhlukovanie <- cluster_walktrap(g)
plot(zhlukovanie, g)
```

# “Walktrap” algoritmus



# "Walktrap" algoritmus: porovnanie s realitou

Porovnajme výsledky zhlukovania s rozpadom klubu.

Budeme potrebovať informáciu o tom, do ktorého zhluku patria jednotlivé vrcholy siete:

```
zhlukovanie$membership
```

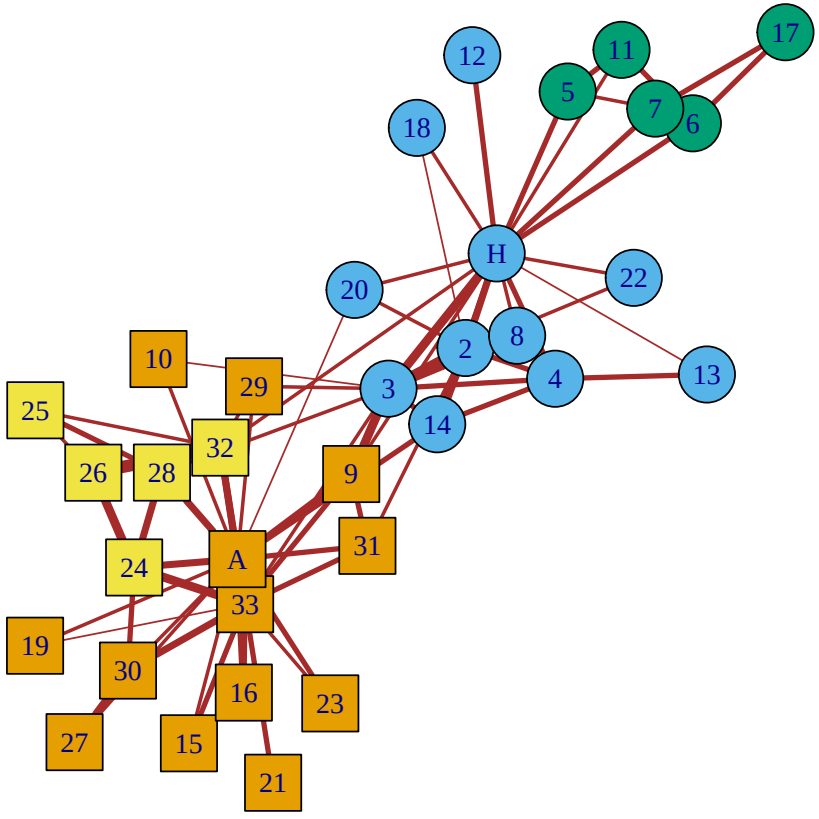
```
##  [1] 2 2 2 2 3 3 3 2 1 1 3 2 2 2 1 1 3 2 1 2 1 2 1 4 4 4
```

Teraz spravíme grafické porovnanie:

- Farbami vrcholov odlíšime jednotlivé zhluky
- Tvarom odlíšime skupiny, na ktoré sa klub rozpadol

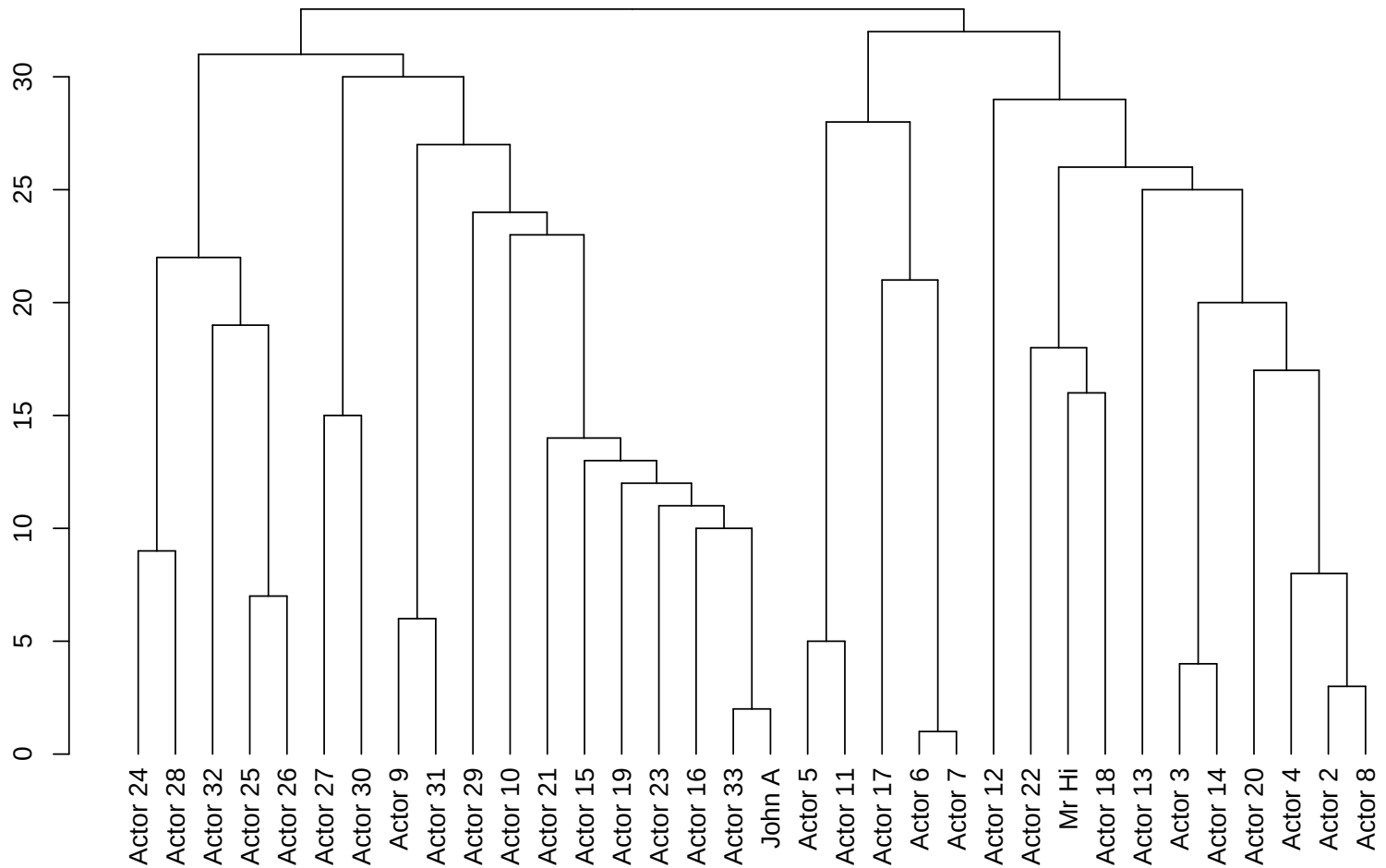
```
tvary <- c("circle", "square")
plot(g,
     vertex.color = ...
     vertex.shape = ...)
```

# “Walktrap” algoritmus: porovnanie s realitou



# “Walktrap” algoritmus: vlastný počet zhlukov

```
plot(as.dendrogram(zhlukovanie))
```



# "Walktrap" algoritmus: vlastný počet zhlukov

- Algoritmus určil počet zhlukov na zákade určitého kritéria.
- My ale môžeme algoritmu zadať vlastný počet zhlukov
- Ide o to, kde odrežeme dendrogram
- Funkcia v R-ku: `cut_at`

Vytvorme dva zhluky a porovnajme ich s rozdelením klubu:

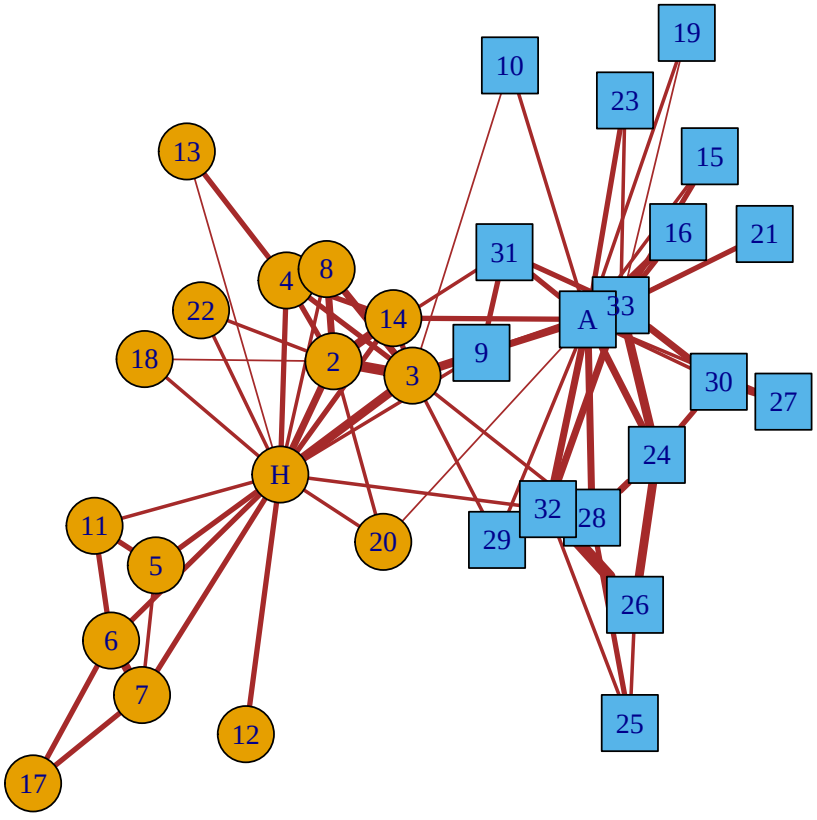
```
zhlukovanie2 <- cut_at(zhlukovanie, n = 2)
zhlukovanie2
```

```
##  [1] 1 1 1 1 1 1 1 1 2 2 1 1 1 1 2 2 1 1 2 1 2 1 2 2 2 2
```

Spravte teraz grafické porovnanie ako v predchádzajúcom prípade

# “Walktrap” algoritmus: vlastný počet zhlukov

Výstup:



# Príklad 4: Futbalový zápas

# Zadanie

Vhodným spôsobom zobrazte sieť (je orientovaná a vážená) danú nasledovnou tabuľkou:

Table 1. Passing pattern of Arsenal against Aston Villa; Saturday, August 19, 2006, Emirates Stadium.

| | Fabregas | Silva | Hleb | Toure | Djourou | Henry | Eboue | Hoyte |
|---|---|---|---|---|---|---|---|---|
| Fabregas | – | 9 | 24 | 5 | 2 | 12 | 10 | 3 |
| Silva | 17 | – | 15 | 11 | 5 | 8 | 3 | 11 |
| Hleb | 17 | 8 | – | 3 | 1 | 15 | 7 | – |
| Toure | 8 | 9 | 14 | – | 13 | 4 | 10 | 1 |
| Djourou | 5 | 13 | 2 | 17 | – | 1 | – | 6 |
| Henry | 4 | 5 | 10 | 3 | 2 | – | 3 | 1 |
| Eboue | 12 | 9 | 7 | 12 | 2 | 2 | – | 1 |
| Hoyte | 12 | 12 | 2 | – | 9 | 2 | 3 | – |

*Note*: Values indicate the number of passes from row to column player. Only information for the 8 most active players are shown. Ljungberg, Adebayor and Hoyte were substituted. Lehman was the goalkeeper.

Grund, T. U. (2012). Network structure and team performance: The case of English Premier League soccer teams. Social Networks, 34(4), 682-690.

Dostupné zo školskej siete: https://www.sciencedirect.com/science/article/pii/S0378873312000500

Čo treba určite spraviť:

- Pri prvom pohľade na orientovanú sieť vidieť, že treba zmenšiť šípky, ktoré ukazujú orientáciu hrán
- Hrany musia byť oblé, aby sa dali rozlíšiť hrany typu `A -> B` a `B -> A`

Ostatné je na vás, chceme, aby bol obrázok pekný, prehľadný a výstižný :)

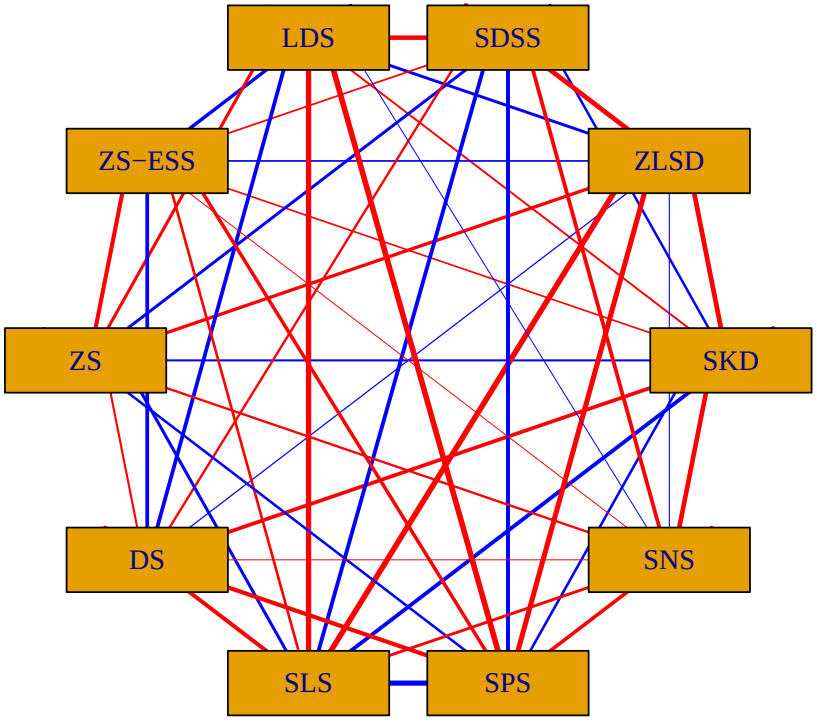
# Príklad 5: Politické strany v Slovinsku

# Dáta

- Dáta a ich popis na stránke http://vlado.fmf.uni-lj.si/pub/networks/data/soc/Samo/Stranke94.htm
- Vyjadrujú podobnosť politických strán, hodnoty sú priradené na základe dotazníkov
- Váha hrany v sieti je mierou podobnosti strán
- Samostatne zostrojte obrázky na nasledujúcich stranách, resp. spravte vlastnú vizualizáciu tejto siete

# Vizualizácia

Červenou farbou záporné váhy, modrou kladné, hrúbka čiary je úmerná absolútnej hodnote váhy



# Vizualizácia

Pre lepšiu prehľadnosť vynecháme v predchádzajúcom grafe hrany s absolútnou hodnotou menšou ako 150